universität wien

MASTERARBEIT / MASTER'S THESIS

Titel der Masterarbeit / Title of the Master's Thesis

The impact of recurrent origins and gene flow on the genetic structure of allopolyploid marsh orchids (*Dactylorhiza*, Orchidaceae)

verfasst von / submitted by

Anna-Sophie Hawranek, BSc

angestrebter akademischer Grad / in partial fulfilment of the requirements for the degree of

Master of Science (MSc)

Wien, 2021 / Vienna, 2021

Studienkennzahl lt. Studienblatt / degree programme code as it appears on the student record sheet:	UA 066 832
Studienrichtung lt. Studienblatt / degree programme as it appears on the student record sheet:	Master's degree programme Botany
Betreut von / Supervisor:	Assoz. Prof. Dipl.-Ing. Dr. Ovidiu Paun

Abstract

Recurrently emerging polyploids are known from various organisms, and are especially frequent in plants. For early generation polyploids a relatively quick recovery from the genetic bottleneck associated with their origin is crucial. Their genetic variation can be enriched via multiple origins from distinct parental populations, or by subsequent introgression from relatives. Inferences of polyploid population genetics and evolutionary history are hampered by complex inheritance, difficult allele dosage assessment, and lacking statistical tools. This thesis focusses on the genetic structure within the allotetraploid European complex *Dactylorhiza majalis s.l.* (Orchidaceae), which originated through recurrent unidirectional crosses of the same diploid parents. Starting from a RADseq data set comprising hundreds of accessions of three species across their distribution area, and taking as reference allelic frequencies from about one hundred representatives of each parental taxon, a genotype likelihoods-based method was chosen to separate paternal from maternal homoeologs. The consistent signal obtained from each polyploid subgenome unveils a complex genetic structure and demographic history shaped by polytopic independent origins, isolation by distance, and regional gene flow between allopolyploids, and with their parents. After range expansion each primary allopolyploid lineage segregated further due to genetic drift in allopatry, the allopolyploids came into contact in different areas. The analyses revealed frequent introgression from diploids to tetraploids, likely facilitated by the absence of an endosperm in these plants' seeds. This study elucidates main phases, and major contributors during the evolution of allopolyploid marsh orchids, and our analytical approach should prove useful for other studies involving non-model allopolyploids.

Zusammenfassung

Wiederkehrende Polyploidisierung kommt besonders häufig bei Pflanzen vor, ist aber von verschiedenen Organismengruppen bekannt. Für die ersten polyploiden Generationen ist eine relative schnelle Erholungsphase des durch ihren Ursprung bedingten genetischen Flaschenhalses wesentlich. Des Weiteren kann ihre genetische Variation durch multiple Ursprünge, welche auf verschiedene Elternpopulationen zurückzuführen wären, oder durch nachfolgende Introgression von verwandten Gruppen, erweitert werden. Rückschlüsse auf die Populationsgenetik und evolutionäre Geschichte sind aufgrund der komplexen Vererbungsmuster, Bestimmung der Alleldosis, und fehlende statistische Programme erschwert. Diese Arbeit handelt von der genetischen Struktur des allotetraploiden Europäischen Komplexes *Dactylorhiza majalis s.l.* (Orchidaceae), der aus mehrmaligen unidirektionellen Kreuzungen desselben diploiden Elternpaares hervorgegangen ist. Für die Analyse eines RADseq-Datensatzes bestehend aus hunderten Individuen, welche das gesamte Verbreitungsgebiet abdecken, wurde eine Methode gewählt, welche auf Genotypenwahrscheinlichkeiten beruht. Diese separiert paternale von maternalen Homeologs unter Zuhilfenahme der Referenzallelfrequenzen von etwa einhundert Vertretern beider Elterntaxa. Das einheitliche Signal, welches von jedem der polyploiden Subgenome kommt, offenbart eine komplexe genetische Struktur und demographische Geschichte geformt von polytopen unabhängigen Ursprüngen, Isolation durch geographische Distanz, und regionalen Genfluss zwischen den Allopolyploiden und mit ihren Elterntaxa. Nach der Ausweitung des Verbreitungsgebietes jeder primären allopolyploiden Abstammungslinie kam es zur Segregation durch genetische Drift in Allopatrie, in sympatrischen Gebieten kamen die Allopolyploiden in Kontakt. Die Analysen zeigen häufigere Introgression von Diploiden zu Tetraploiden, die wahrscheinlich durch die Abwesenheit eines Endosperms in den Samen dieser Pflanzen erleichtert wurde. Diese Studie deckt wesentliche Phasen und Komponenten während der Evolution polyploider Knabenkräuter auf, der gewählte analytische Ansatz scheint nützlich für andere Studien an allopolyploiden Nicht-Modellorganismen.

Table of Contents

Introduction

Whole genome duplication or polyploidization is seen as a major force driving evolution, as reviewed by Soltis, Visger, and Soltis (2014), and was shown to increase adaptability to novel environments (Leitch and Leitch 2008, Soltis and Soltis 1993) after the newly formed polyploids have overcome initial genetic and functional challenges (McClintock 1983, 1984).

Polyploidization, which describes the multiplication of the genome either through intraspecific genome doubling (i.e., autopolyploidy) or hybridisation while the entire parental genomes are retained in the descending lineage (i.e., allopolyploidy), is often regarded as a major force driving speciation and diversification especially in plants, but also in other organisms (Paun et al. 2007, Ramsey and Schemske 2002, Weiss-Schneeweiss et al. 2013). Accordingly to estimations by Otto and Whitton (2000) between 2 and 4 % of plants speciated through polyploidization. However, taking advantage of the potential which comes along with the duplicated genome size (as in tetraploids) and therefore increased diversity (Meirmans and Van Tienderen 2013) was found to often require extensive genomic alterations to circumvent hindrances to evolutionary success. For instance, issues like failures during meiotic pairing of homoeologous chromosomes (Paun et al. 2007), i.e., the corresponding set of chromosomes from each parent (Glover, Redestig, and Dessimoz 2016), could happen due to the varying divergence along the genome between parental taxa. Those issues were collectively identified and termed “genomic shock” by Barbara McClintock (1983) (Comai et al. 2003).

Newly formed polyploids suffer from the sudden reduction of genetic diversity (i.e., genetic bottleneck), which is then increased through backcrossing with parental lineages, hybridisation with related taxa and also recurrent polyploidization (Brandrud 2019), followed by gene flow and recombination (Paun et al. 2007, Soltis and Soltis 1999).

Nevertheless, tracing back the evolutionary history of polyploid taxa by applying population genomic approaches is still challenging as bioinformatic and population genetic tools are generally developed for diploid data (Meirmans, Liu, and van Tienderen 2018). Applying these tools then to polyploid data could mislead through differences in meiotic segregation and the difficulty of assessing allele dosage in order to differ between partial heterozygotes (Meirmans, Liu, and van Tienderen 2018). In polyploids segregation patterns can be partly disomic and polysomic (Ramsey and Schemske 2002), these so-called “segmental allopolyploids” (Stebbins Jr 1947) were, for example, partially re-diploidised and may lead to a bias in estimating summary statistics due to the varying frequency of polysomy (Meirmans and Van Tienderen 2013). Double reduction is another potential cause for erroneous inferences: here two copies from one chromatid segment are located in the same gamete after segregation, increasing homozygosity (Butruille and Boiteux 2000, Meirmans, Liu, and van Tienderen 2018, Xiong et al. 2021). Also, some species complexes include multiple ploidy levels,

which are again difficult to treat (Allendorf et al. 2015). Recently, Blischak, Kubatko, and Wolfe (2018) developed a method which allows *in-silico* separation of a polyploid genome into its subgenomes based on the allelic frequencies of extant individuals from the parental lineages opening paths to more sophisticated analyses of each subgenome separately. Analysing the estimated diploid components of polyploid (here tetraploid) genotypes allows for more accurate estimations of population genetic parameters and inferences of evolutionary history (Blischak, Kubatko, and Wolfe 2018, Meirmans, Liu, and van Tienderen 2018).

Recurrently emerging polyploids are well known from various organisms: For example several plant groups (e.g. *Tragopogon mirus*, *Triticum* spp.) underwent multiple events of polyploidization, as reviewed by Soltis and Soltis (1993). Recurrent origins may lead to distinct genotypes with different morphologies, ecologies, genetics, and physiologies, and thus further enrich the initially poor polyploid gene pool (Brandrud 2019). Also in some vertebrate groups polyploidization is hypothesised to have occurred more than once, e. g. in the siluriform subfamily Corydoradinae (Oliveira et al. (1992), Oliveira et al. (1993), as reviewed by Comber and Smith (2004)).

Extant species complexes that include polyploids have emerged at different times in the past: for example, the origin of the oldest *Nicotiana* (Solanaceae) polyploids were estimated 4.5 million years ago (Leitch et al. 2008); one of the very recent complexes is found within the asterid genus *Tragopogon* and formed about 80 years ago (Soltis et al. 2004). The *Dactylorhiza majalis s.l.* allopolyploid complex studied herein is expected to have emerged in between the older *Nicotiana* and the younger *Tragopogon* polyploids (Brandrud 2019, Brandrud et al. 2020). Therefore, *D. majalis s.l.* had already time to overcome some of the initial challenges of polyploidization, but still did not evolve further over millions of years (Brandrud 2019, Pillon et al. 2007). These characteristics allow for tracking processes of recurrent polyploid formation, therefore marsh orchids of the genus *Dactylorhiza* serve as a good model system for studies on polyploid evolution.

Dactylorhiza includes two recognised major clades (Brandrud et al. 2020): the *D. fuchsii-maculata* clade and the *D. incarnata-euxina* clade. While the former contains diploids and one autotetraploid taxon (i.e., *D. maculata*), the latter is entirely diploid. Between those two clades hybridisation led to reticulations resulting in the emergence of allotetraploids (Brandrud et al. 2020), some of which are thought to have survived the last ice age within Alpine (Ståhlberg 2007), Balkan (Hedrén et al. 2007) and partly central Russian (Ståhlberg 2007) refugia (Nordström and Hedrén 2009) or emerged within the last 11,500 years after the last periglacial period (Bateman 2011).

The time of the divergence event leading to the two clades described above was dated to approximately six million years ago (Brandrud et al. 2020). *Dactylorhiza incarnata* exhibits low levels

of genetic diversity, likely due to a strong genetic bottleneck that happened either while migrating from Asia to Europe (Hedrén and Nordström 2009), due to survival in southern European Pleistocene refugia and/or while recolonising after glaciation (Balao et al. 2016, Balao et al. 2017). In stark contrast, *D. fuchsii* shows higher genetic diversity and a lack of geographic structure (Brandrud et al. 2020).

Brandrud et al. (2020) confirmed the putative parental lineages for polyploids of the Eurasian orchid genus *Dactylorhiza* based on genotype likelihoods, three of those polyploids are going to be the focus of this thesis: *Dactylorhiza majalis*, *D. traunsteineri* and *D. purpurella*. By having similar parents, the three differ in age and thus in time they had to rearrange their newly formed allotetraploid genomes assembled from *D. incarnata* on the paternal and *D. fuchsii* on the maternal side (Brandrud et al. 2020).

Previous analyses based on private alleles derived from RADseq data (Brandrud et al. 2020) proposed the following succession of allopolyploid formation for species of the *D. majalis s.l.* allotetraploid complex: *D. majalis*, *D. elatior*, *D. traunsteineri*, *D. praetermissa*, *D. baltica* and *D. purpurella* as the youngest allopolyploid (Brandrud et al. 2020). Those allotetraploids differ in their morphologies, their ecological preferences and spatial distribution patterns (Pillon et al. 2007). *Dactylorhiza majalis*, *D. traunsteineri* and *D. purpurella* are going to be introduced in the following, their different current distribution ranges are displayed in Figure 1.

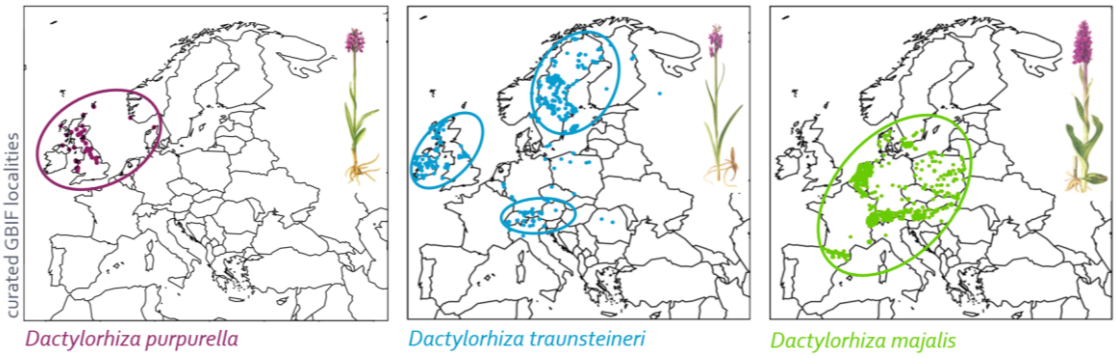

Figure 1: Distribution patterns of the three allotetraploids in this thesis' focus, based on curated localities initially extracted from the Global Biodiversity Information Facility (GBIF, accessed February 2018). Plant drawings by Erich Nelson (Nelson 1976).

Growing in damp meadows and marshlands, *D. majalis*, a continental European allotetraploid is characterised by a comparatively wide ecological niche in regard of soil moisture and elemental chemistry (Paun et al. 2010, Wolfe et al. 2021). This taxon shows rather little genetic variation throughout its distribution range which might be explained by its likely pre-glacial origin and subsequent glacially induced genetic bottlenecks happening in one to few refugia in Southern Europe

(Pillon et al. 2007). Unique plastid haplotypes not found in present day representatives of its maternal lineage might also indicate a rather old age and thus time for divergence (Paun et al. 2010). Its relatively high amount of private alleles deem it as the oldest surviving allopolyploid from this sibling series (Brandrud et al. 2020).

Dactylorhiza traunsteineri is found to be morphologically (Bateman and Denholm 1983, Bateman 2011) as well as genetically (Pillon et al. 2007) more heterogeneous than *D. majalis*, but the most locally specialised out of the allotetraploids (Bateman 2011, Paun et al. 2010). It occurs in three major disjunct regions: in the Alps, in Scandinavia and in the Northern half of British Isles with slightly different morphologies and genetic signatures; however, a common origin is so far considered as the most likely scenario (Bateman 2011, Brandrud 2019). Bateman (2011) further draws similarities in the distribution pattern of recent *D. traunsteineri* across Europe and the disjunct appearance of the Weichselian (i.e., Devensian) ice sheet raising questions on whether *D. traunsteineri* was formed once in one of the three regions followed by long-distance dispersal via their characteristic wind-dispersed dust-like seeds (Bateman 2006) or whether parallel evolution through adaptation to similar environmental conditions took place in each region. Its ecological tolerance is quite narrow: these plants are found in calcareous fens, marshes and moist post-glacial depressions (Bateman 2011, Paun et al. 2010), characterised by a low availability of macro- and micronutrients in the soil (Wolfe et al. 2021).

The distribution of *D. purpurella* covers Scotland, Northern England, Northern Ireland and western parts of Scandinavia (Bateman 2011). It was found to share haplotypes with both parental taxa suggesting bidirectional hybridisation additionally to polyploidization or backcrossing to its parents after polyploidization yielding additional plastid haplotypes (Hedrén, Nordström, and Bateman 2011, Pillon et al. 2007). Analysis of plastid haplotype data led to the hypothesis of a single post-glacial origin followed by an expansion of its distribution area (Hedrén, Nordström, and Bateman 2011), whereas the low amount of private alleles render this allotetraploid as the youngest among its siblings (Brandrud et al. 2020).

The present thesis aims at identifying the number of origins within *D. majalis*, *D. traunsteineri* and *D. purpurella* as well as assessing gene flow within and across (i.e., backcrosses with the parental lineages) ploidy levels. Additionally, divergence times are going to be approximated by demographic inference modelling. By studying a RADseq data set comprised of both, allotetraploid genotypes and their two diploid parental taxa, three hypotheses are going to be tested: (1) each of the three polyploids has a distinct origin in contrast to a common formation event followed by diversification at the tetraploid level; (2) following previous conclusions from private allele data (Brandrud et al. 2020) and haplotype analyses (Hedrén, Nordström, and Bateman 2011, Nordström and Hedrén 2008,

2009, Paun et al. 2010) at least some of the allotetraploids are expected to be older than the last glacial maximum, further, the parental taxa met in Europe approximately 1.5 million years ago, thus at least some (if not all) of the allotetraploids are expected to be younger than 1 million years; and (3) as previously demonstrated, gene flow takes place between polyploids (Balao et al. 2016) – it is hypothesised that stability and successful adaptation increases with age; therefore, introgressive hybridisation is expected to be lower in older polyploids.

Material and methods

Sampling

The sampling for this study comprised of 104 diploid individuals (48 *D. fuchsii* and 56 *D. incarnata*) and 241 allotetraploids (86 *D. majalis*, 10 *D. purpurella* and 145 *D. traunsteineri*). The analysed RADseq data set covers 63 localities across 18 European countries, belonging to diploid *Dactylorhiza fuchsii* and *D. incarnata* as well as allotetraploid *D. majalis*, *D. purpurella* and *D. traunsteineri* as described in Brandrud (2019). Both, allopatric and sympatric areas are covered as reported in Figure 2. Plotting of sampling localities onto a geographical map was done using "marmap" (Pante and Simon-Bouhet 2013), "rworldmap" (South 2011) and "rworldxtra" (South 2012) packages in R 4.0.2 (R Core Team 2020).

Excluding introgressed diploid individuals

Starting from the dataset from Brandrud et al. (2020), the heatmap of pairwise relatedness of diploid individuals was recalculated. After mapping, genotype likelihoods were inferred for all available individuals by using ANGSD v. 0.928 (Korneliussen, Albrechtsen, and Nielsen 2014) with GATK model (--GL 2), --minInd option was set to half the number of individuals and --minMaf was defined as the frequency of 2 individuals. The covariance matrix was then calculated with *pcangsd.py* script from Meisner and Albrechtsen (2018) by keeping the same --minMaf value as before. This covariance matrix was then used to create a heatmap of pairwise relatedness with *heatmap.2()* function from R package "gplots" (Warnes et al. 2019) running in R v. 3.4.4 (R Core Team 2018). In LibreOffice Calculator the sum of rows and columns was calculated, all individuals with this summed value being smaller than and arbitrarily set threshold of -5 were excluded from further analysis since low values indicate that individuals descend from different populations, thus it is likely that introgression results in negative values. This procedure led to an exclusion of 36 of 84 individuals of *Dactylorhiza fuchsii* and of 21 of 77 *D. incarnata*. The total number of sites analysed dropped from 914,463 to 886,663; the number of sites retained after filtering from 8,069 to 6,334.

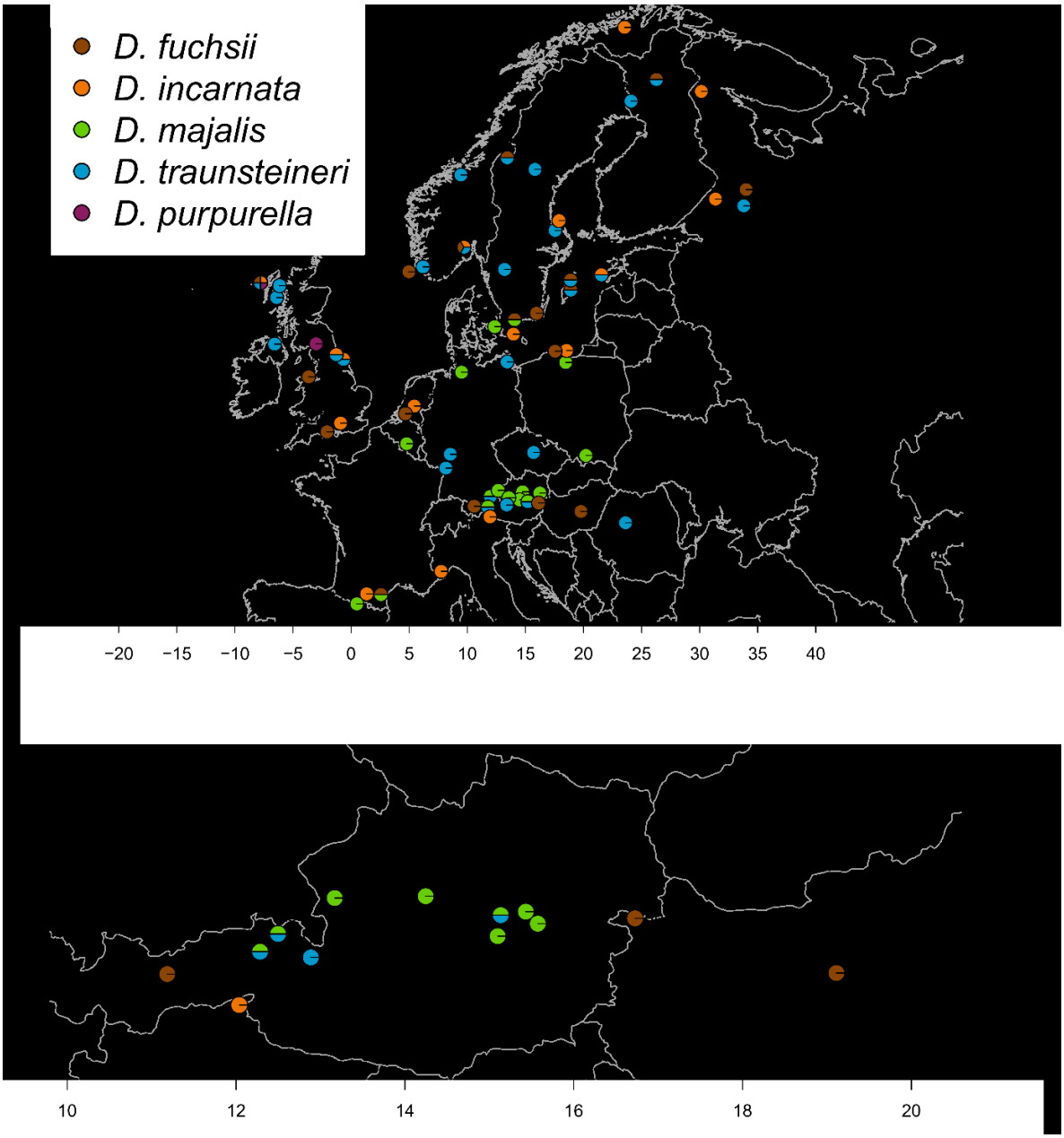

Figure 2: Sampling localities of the two diploid and the three tetraploid taxa. The displayed ratios do not represent abundance data, they only document sympatric localities. The Alpine region is plotted separately for better readability. Collection details to be found in Table S 1.

Read Mapping

After a quality check using FastQC v. 0.11.5 (Andrews 2018), paired and single end reads were mapped to the *Dactylorhiza incarnata* reference genome v. 1.0 (Wolfe et al. 2021) with bwa mem v. 0.7.12 (Li 2013). After mapping read groups were added with picard v. 2.1.0 (available from http://broadinstitute.github.io/picard/, Broad Institute 2018), reads were sorted by coordinate with picard v. 2.20.6 (available from http://broadinstitute.github.io/picard/, Broad Institute 2018). With samtools v. 1.3 (available from http://www.htslib.org/, Li et al. 2009) bam index files were created

for every accession. GATK v. 3.8 (McKenna et al. 2010) was used to realign indels in two steps: first the genomic sites were targeted with RealignerTargetCreator command, then indels were realigned with IndelRealigner command with a maximum of 100,000 reads.

Genotyping

A .vcf file containing all individuals was created for each ploidy level with UnifiedGenotyper from GATK v. 3.8 (McKenna et al. 2010) and filtered with the *filter-vcf.R* script provided by the ebg (Empirical Bayesian Genotyping) package (available from https://pblischak.github.io/polyploid-genotyping/, Blischak, Kubatko, and Wolfe 2018) to only retain biallelic SNPs present in at least half of the individuals with a minimum quality of 100, the read depth minimum was set to five. The *intersect-vcf.R* helper script provided by ebg was applied to find shared variants in the previously filtered .vcf files per ploidy level (45,186 loci found). Then, individuals of alternative ploidy were excluded from each file.

Total and alternative read counts were calculated with the *gt-from-vcf.R* helper script from ebg. From the .bam files a .pileup file for the shared variants was generated with samtools v. 1.3 (available from http://www.htslib.org/, Li et al. 2009) and the per locus error rate was calculated with the respective helper python script provided by ebg. Ebg was then first run in gatk mode to obtain tetraploid coded data (i.e., four alleles per locus) for the polyploids.

Genome separation and accuracy testing of ebg alloSNP

By running ebg in alloSNP mode, SNPs in the allotetraploid genomes can be assigned to their putative parental origin by providing as reference allele frequencies of one parental lineage. Due to the availability of present-day genomes of individuals from both parental lineages, the allotetraploid genomes were alternatively separated on the basis of either parental allele frequencies. As the two parental species differ in their population genetic characteristics (see introduction), the accuracy of separating the subgenomes based on the information from either parent was tested first. To this aim *in-silico* allopolyploids were generated from reads from *D. fuchsii* and *D. incarnata* individuals (used data given in Table S 2) in a 2.89:3.55 ratio according to the difference in genome size between the diploid parents (Aagaard et al. 2005). Ten replicates were run through the pipeline alike the real data, but since the real affiliation of each read is known in the case of the *in-silico* data set, it was possible to trace back in which homeolog file each read is ending up. Thus, the accuracy of ebg alloSNP was quantifiable.

Inferring population structure

Pairwise relatedness of allotetraploids

The relatedness between the 241 allotetraploid accessions genotyped by ebg gatk (i.e., tetraploid coded data set), and filtered by a minor allele frequency of 0.02 was calculated with the *Method of Moments* implemented in polyrelatedness v. 1.8 (Huang et al. 2014). The resulting matrix of pairwise relatedness was then visualised as a heatmap similarly as explained for the diploid analyses (see above).

Genotype clustering

To assess population structure, STRUCTURE v. 2.3.4 (Pritchard, Stephens, and Donnelly 2000) was run on LiSC (https://cube.univie.ac.at/lisc) with a Markov chain Monte Carlo (MCMC) for one million generations plus 10% burn-in (for other options the default was kept) from $K = 1$ to $K = 9$. The analyses were run for (1) the data set containing 241 tetraploid genomes, filtered by a minor allele frequency of 0.02, and one SNP every 10 kB (to account for linkage disequilibrium); (2) 241 diploid *fuchsii* subgenomes were run together with 48 diploid *D. fuchsii* genomes (filtered as for (1)); and (3) 241 diploid *incarnata* subgenomes together with 56 diploid *D. incarnata* genomes (filtered as for (1)). The ten replicates per data set were further processed with CLUMPP v. 1.1.2 (available from https://rosenberglab.stanford.edu/clumpp.html, Jakobsson and Rosenberg 2007) by using the "Greedy" algorithm. Likelihood values were calculated with STRUCTURE HARVESTER Web v. 0.6.94 (accessed via http://taylor0.biology.ucla.edu/structureHarvester/, Earl and vonHoldt 2012).

Constructing folded joint allelic site frequency spectra (jSFS)

To prepare for demographic inference, folded jSFS were built using two custom R scripts run in R v. 4.0.2 (R Core Team 2020). To exclude individuals potentially affected by recent interspecific gene flow, we selected for jSFS construction individuals with q-values > 0.8 in at least one of the subgenomes and at least 0.75 in the other based on STRUCTURE analyses assuming $K = 6$. For each group one file containing the genotypes was generated.

In the first R script "*jSFSfiltering_final_v2.R*" the filtering procedure was conducted. All major alleles were converted to minor at the beginning (function *convert2minor()*). Then, sites missing in more than the desired final deme size were excluded (*missingData()*), these data were subsampled randomly (*preSubsetCombined()*, *randomSubset()*). Monomorphic loci were excluded (*monoPoly()*, *getPolymorphicData()*). To account for linkage disequilibrium only one SNPs every 10 kB was kept (*filter1SNPevXbp()*, *keep1SNPevXbp()*). Finally, all those functions were run together with the wrapper script *filteredData()*.

The second R script "*jSFS_matrix24-24.R*" (note: several scripts have been developed depending on the final matrix size) was used to calculate the allele sum per deme *allelesPerDeme()* which were then utilised to build the jSFS with *jSFS()*.

Plotting of jSFS was done with the *heatmap.2()* function from R package "gplots" (Warnes et al. 2019), and the "viridis" colour scheme (Garnier 2018) in R v. 4.0.2 (R Core Team 2020).

Inferring demographic history

Several demographic scenarios for the three allopolyploid species have been evaluated with fastSimcoal2 v. 2.6.0.3 (Excoffier et al. 2013). While focussing on each subgenome separately, divergence patterns between four groups: *D. majalis*, *D. purpurella*, *D. traunsteineri* (i.e., the 'continental' group – the light blue gene pool indicated for $K = 5$ in Figure 4), plus the respective diploid parent have been analysed. The divergence patterns within the different groups of *D. traunsteineri* were considered outside the scope of this MSc thesis, as the numbers of models to be tested would be significantly large. Further, to minimise the number of models tested as much as possible, previous knowledge of the relative age of the allopolyploids, setting *D. majalis* as the oldest of the three, and *D. purpurella* as the youngest (Brandrud et al. 2020), was taken into account for model design. Given the rest of our results and previous knowledge (e.g., Balao et al. 2016, Brandrud et al. 2020), only models of isolation with migration, i.e., including a continuous gene flow between the demes, were tested. These considerations resulted in testing the following scenarios for each subgenome separately: (1) one origin for the allopolyploids with *D. majalis* splitting first from the diploid group, *D. traunsteineri* splitting later from *D. majalis*, and *D. purpurella* splitting last from *D. traunsteineri*; (2) two origins scenario "MT", with the allopolyploids originating twice, once to form *D. majalis*, with *D. traunsteineri* splitting later from it, and an independent origin for *D. purpurella* from the diploids; (3) two origins scenario "TP", with an earlier independent origin of *D. majalis*, and an independent origin for *D. traunsteineri*, with *D. purpurella* splitting later from it; and (4) three independent origins for the allopolyploids, in the order expected given the previous information (Brandrud et al. 2020).

The analyses used as summary statistics the constructed two-dimensional jSFS for all six combinations of populations (see above). For all models the algorithm was allowed to estimate the effective population size (N), the mutation rate (μ), the time of each split ($T1$, $T2$, and $T3$), and the individual migration rates between all pairs of demes and directions. The priors of the simulations were set for the effective population size between 10K and 500K (diploid populations) and between 50K and 10K (allopolyploid demes), to 1e-08 and 1e-10 for the mutation rate, for the time of each split between 100 and 50,000 generations ago, and for the migration rates (m) to between 1e-07 and one.

By excluding monomorphic sites (-0 option) and running 50 optimisation cycles, 200,000 simulations were performed. To find the global optimum of the best combination of parameter estimates, 60 replicates of simulation runs for each model were done. For each model *ΔL*, i.e., the difference between the maximum estimated likelihood across all replicate runs (i.e., MaxEstLhood), and the maximum possible value for the likelihood if there was a perfect fit of the expected to the observed jSFS (i.e., MaxObsLhood) was reported, and the model which minimises *ΔL* was accepted as the best model. Finally, the parameter estimations of the best ten runs of the best model per subgenome were retained as confidence intervals for each parameter.

Results

Accuracy of ebg alloSNP

The ebg alloSNP estimates of the genotypes of the *fuchsii* tetraploid subgenomes showed lower error rates when inferred based on the *D. incarnata* reference allele frequencies as compared to the inference based on its own diploid frequencies. Similarly, inferring the *incarnata* subgenome yielded a higher accuracy when based on the *D. fuchsii* reference allele frequencies (Table 1).

Table 1: Overview showing the average accuracy of ebg alloSNP in assigning the reads of ten *in-silico* allotetraploid genomes to their respective origin.

Allelic reference frequencies from:	***D. fuchsii***	***D. incarnata***
***fuchsii* subgenome**	0.70	0.84
***incarnata* subgenome**	0.88	0.50

Filtering the data set

After excluding likely introgressed diploid accessions (a heatmap of pairwise relatedness of all retained diploids can be found in Figure S 1), the diploid and the allotetraploid files were filtered with *filter-vcf.R* (part of the ebg package) for biallelic sites only, a minimum quality score of 100 per site (default), a minimum read depth of five (default) and sites with at least half the number of individuals present (173). After that 47,057 and 54,334 SNPs were retained in the diploid and the allotetraploid files respectively. The *intersect-vcf.R* script (also part of the ebg package) found 45,318 shared variants which built the basis for the following analyses.

Genotyping with ebg gatk identified 9.3 % in the allotetraploid file. Further filtering of the data set retained 18,879 SNPs for estimating pairwise relatedness, and 2,466 'unlinked' SNPs (i.e., one every 10 kB) for the STRUCTURE analyses.

Splitting the allotetraploid genomes required the removal of 132 SNPs that had encountered failures during the genotyping or filtering process in the diploid files (i.e., they appeared as “nan” when estimating the allelic frequencies). Those were also removed in the allotetraploid files for consistency, leaving 45,186 SNPs for the analyses at he subgenome level. Accessions of *D. fuchsii* were filtered together with the maternal part of the allotetraploid genomes, and *D. incarnata* together with the paternal side resulting in 2,390 and 2,355 SNPs respectively, when filtering the data for use with STRUCTURE.

Ten jSFS per parental lineage were constructed, 20 in total (Figure S 9). Each one is based on between 1,495 and 2,292 loci (*mean* = 1,916.0) on the maternal side and 1,565 to 2,314 loci (*mean* = 1,928.6) on the paternal side.

Allotetraploid population structure

Analyses of the tetraploid data set retrieved via ebg gatk revealed distinct patterns of pairwise relatedness for each of the three taxa (Figure 3). *Dactylorhiza majalis* appeared relatively homogeneous across sampling sites with low levels of geographic isolation. There is no clear genetic distinction detectable between French Pyrenean (fr1, fr2), North-central European (se1, se2, be3, de1, pl1), Southern Polish (pl5) and Austrian Alpine (at1-at6, at8, at10) sites.

This is in stark contrast to the observation made in the disjunctly distributed *D. traunsteineri*, which showed high pairwise relatedness within sampling sites and lower levels of similarity to other geographic localities. Some genotypes of *D. traunsteineri* from the Alpine area around Kitzbühel and Zell am See (at7, at8) showed closer similarities with sympatric *D. majalis* individuals than with other individuals of its own kind (see also Figure 4), while other Alpine localities (at3, at10) are clearly distinct from *D. majalis*. *Dactylorhiza traunsteineri* from the Estonian Saaremaa Island showed highest pairwise relatedness to the North-western Romanian site (ro5), while Estonian genotypes were shown to be very dissimilar to individuals from Gotland (se3, se4) and South-eastern Sweden (se5), despite their geographic proximity. This analysis revealed no genetic differentiation between the two aforementioned localities on Gotland, but some to sites on mainland Sweden, which cluster in three closely related groups: South-eastern Sweden (se5), North-eastern Sweden (se6), and the central Swedish (se7, se13) together with the Southern Swedish (se15) sites. The latter composed group showed some genetic similarities with the West Russian site (ru4), genotypes from Northern Finland (fi1), North-western Romania (ro5), central Czechia (cz2), North-eastern Germany (de4), South-eastern Norway (no2) and South-central Norway (no6); nevertheless, each of those sites was found to be locally isolated. Both of the South-western German sites (de3, de6) show average levels of similarity to any other *D. traunsteineri* genotype, but some relatedness between the two. The

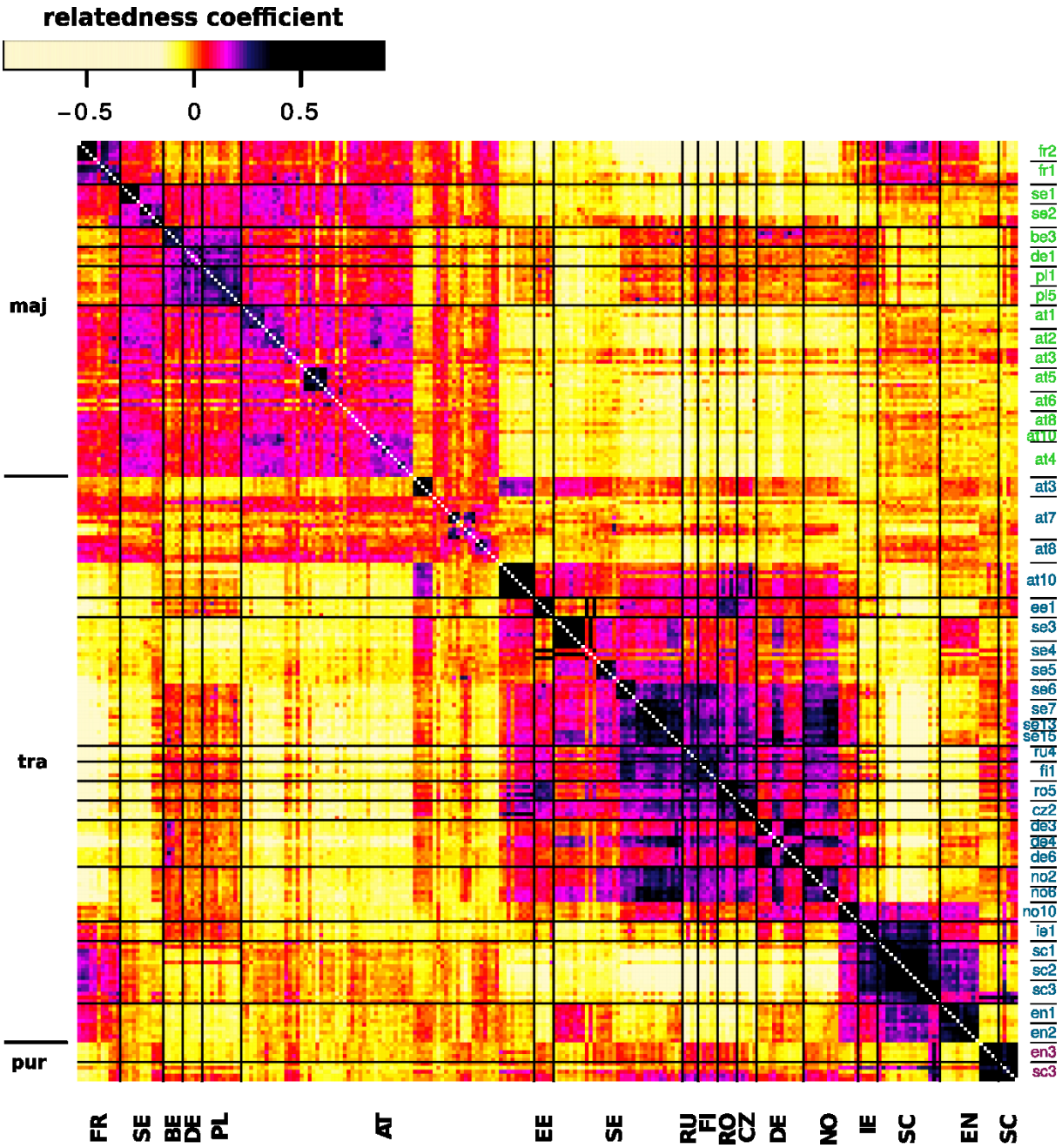

Figure 3: Heatmap of pairwise relatedness generated with a Method of Moments (Huang et al. 2014) between 241 allotetraploid *Dactylorhiza* accessions. Colours indicate degrees of relatedness according to the legend. The estimates on the diagonal have been excluded to improve for colour resolution. On the lower x-axis country codes are given, on the left side species are indicated: maj, *Dactylorhiza majalis*; pur, *D. purpurella*; tra, *D. traunsteineri*; on the right locality acronyms are given. For individual accession numbers please refer to Figure S 2.

third Norwegian site from the South-eastern costal area (no10) showed similarities with *D. traunsteineri* from the British Isles (see also Figure 4). Apart from this observation, the accessions from Northern Ireland (ie1) together with the Scottish sites (sc1-sc3) showed very high degrees of isolation from any other locality. The two sites from North-eastern England (en1, en2) were found to

be again isolated, but still, some genetic similarity to the (South-eastern Norway-)Northern Ireland-Scotland cluster was observable.

Genotypes of the third taxon, *D. purpurella*, displayed relatively higher degrees of genetic isolation against its two sibling allotetraploids, However, notably high levels of coancestry are obvious between the *D. purpurella* accessions from the Hebridean island North Uist (sc3) and some of the *D. traunsteineri* accessions from the same locality, indicating a high level of gene flow at sympatric localities.

Results of the STRUCTURE analysis of the tetraploid data set have been displayed as bar plots (Figure 4). Following Evanno's method (Evanno, Regnaut, and Goudet 2005) *K* = 2 was found to be the best

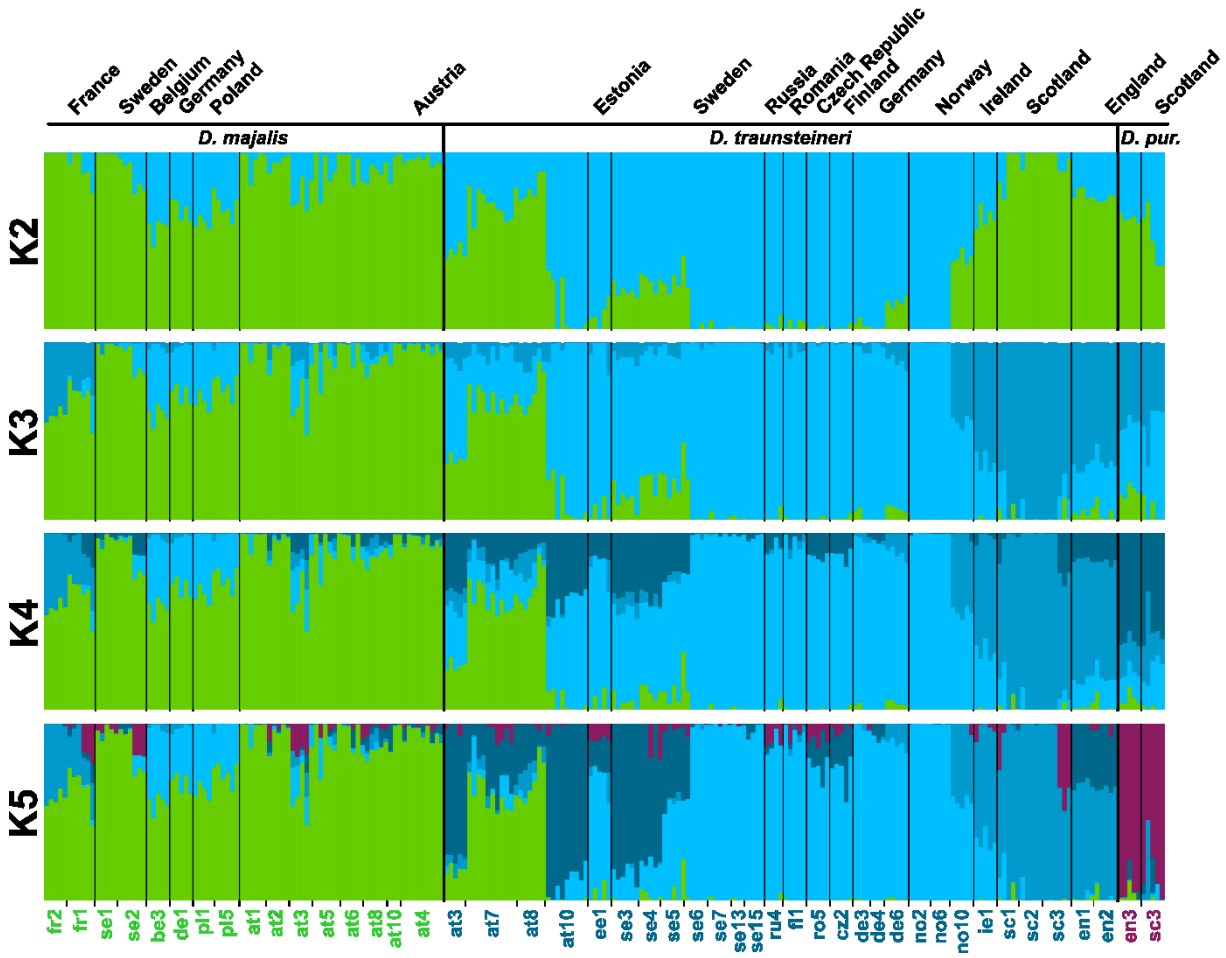

Figure 4: STRUCTURE results for the tetraploid-encoded data set (i.e., both homeologs together) of 241 individuals for *K* = 2 to *K* = 5. Colours represent different gene pools. *D. pur.* abbreviates *Dactylorhiza purpurella*. Individual accession numbers to be found in Figure S 3, likelihood values in Figure S 4.

supported number of clusters, followed by *K* = 3 and *K* = 5 (Figure S 4). *K* = 2 showed a distinction between *D. majalis* and continental *D. traunsteineri* (except some notable admixture in Austrian Alpine sites, see above). *D. purpurella* appeared to be constituted of about half the 'majalis'

genotype and half the 'traunsteineri' genotype in *K* = 2. *K* = 3 leaves *D. majalis* as a panmictic population showing again some admixture with Austrian Alpine *D. traunsteineri*. The rest of the *D. traunsteineri* accessions felt into two groups: continental European and British Isles. *Dactylorhiza purpurella* again was recovered as admixed, this time between the two *D. traunsteineri* genotypes. In the case of *K* = 4 a gene pool appeared (dark blue) that has no individual completely included in it. Increasing the number of inferred clusters to *K* = 5 led to a distinct Austrian Alpine *D. traunsteineri* cluster (at10) and a fifth cluster almost unique to *D. purpurella*.

Population structure of the diploid subgenomes

The STRUCTURE analysis of splitted (now diploid) subgenomes was conducted twice: (1) recent accessions of the maternal lineage (*D. fuchsii*) were analysed together with the maternal part of the allotetraploid genotypes, inferred from allelic frequencies of recent paternal (*D. incarnata*) accessions (Figure 5); (2) recent accessions of the paternal lineage (*D. incarnata*) together with the paternal part of the allotetraploid genotypes, inferred from allelic frequencies of recent maternal (*D. fuchsii*) accessions (Figure 6).

In both parts of the genome *K* = 2 revealed a clustering into the respective parental lineages and all the allotetraploids with some degree of admixture across the entire group. However, slightly more introgression was found on the paternal as compared to the maternal side (see also Figure 7). *K* = 3 reconstructed an almost similar population structure as seen in the allotetraploid analysis (Figure 4). A difference can be observed in *D. majalis*; it appeared (except of French and Swedish localities) more admixed with the blue 'traunsteineri' cluster in both subgenome, but more pronounced in the paternal one. Again, with the exception of the Easternmost Austrian population (at10), *D. traunsteineri* in the Alpine region was inferred to be well admixed with *D. majalis*. When assuming *K* = 4 *D. traunsteineri* was separated into a continental European group and another on the British Isles, with the purest individuals found in Scotland as the allotetraploid analyses already showed (Figure 4). The greatest difference to the allotetraploid analyses as well as between the subgenomes was observed in the case of *K* = 5: On the maternal side this fifth population accounted for the *D. purpurella* genotype with admixture across almost all localities of *D. majalis* and *D. traunsteineri* and turned out to be almost unique to *D. purpurella* when allowing for one more population; on the paternal side *K* = 5 resulted in a differentiation into heavily admixed *D. traunsteineri* from Western Austria, *D. traunsteineri* from Eastern Austria (at10) and Gotland (se3, se4) formed the newly introduced cluster (dark blue) with $q > 70\%$, when introducing a sixth potential population, the Gotland sites appeared more similar to other continental European *D. traunsteineri* again. As mentioned before, the corresponding *K* =4 from the allotetraploid analysis recovered a ghost cluster. *K* = 6 on the maternal side led to a clearly seen 'purpurella' genotype showing some introgression

into sympatric Hebridean *D. traunsteineri* (sc3). This admixture pattern was found to be a little weaker in the paternal part of the subgenome.

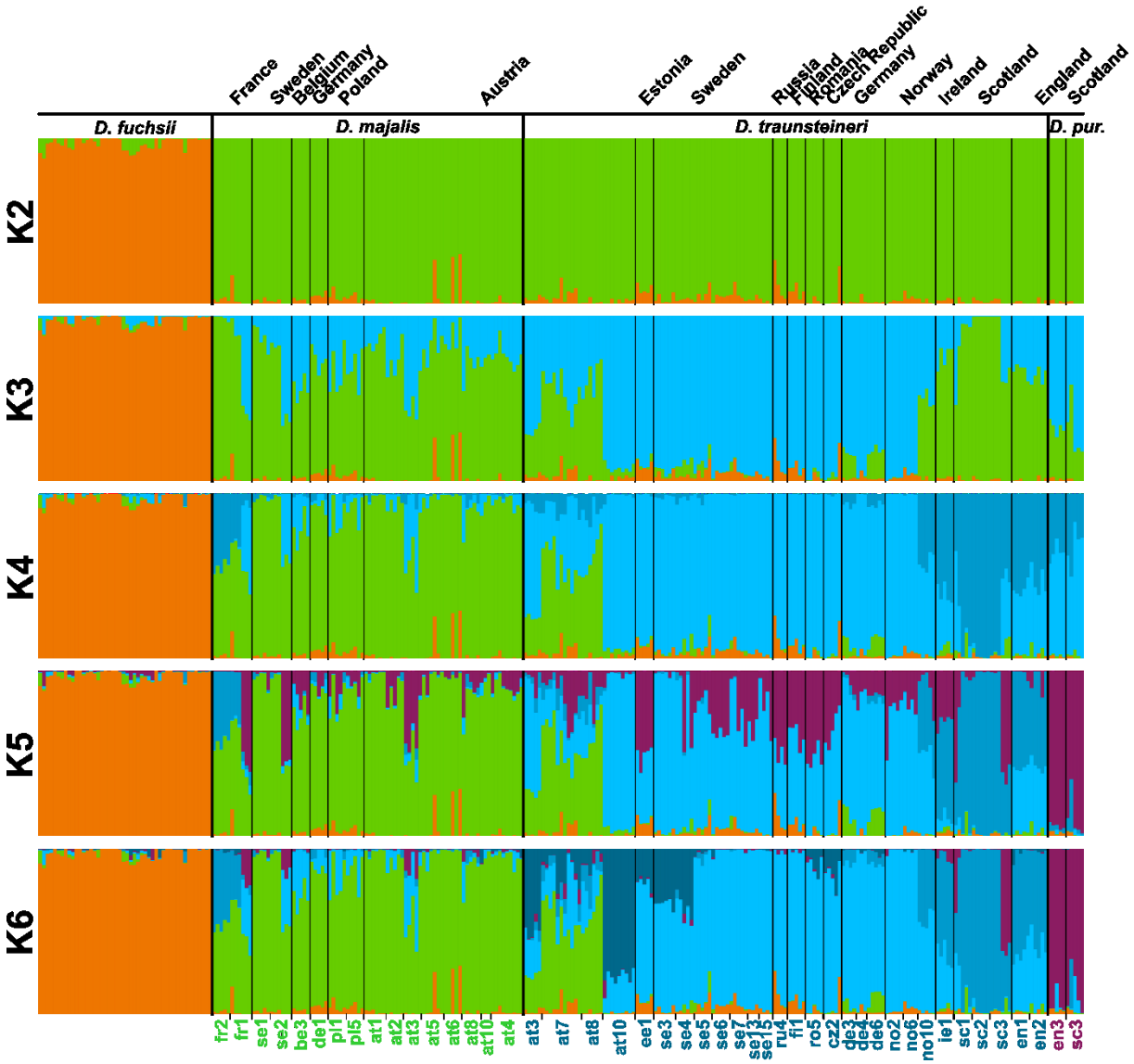

Figure 5: STRUCTURE results for 48 diploid *Dactylorhiza fuchsii* accessions (maternal lineage), along with the maternal part of allotetraploid accessions of 241 individuals for *K* = 2 to *K* = 6. Colours represent different gene pools. *D. pur.* abbreviates *Dactylorhiza purpurella*. Individual accession numbers to be found in Figure S 5, likelihood values in Figure S 6.

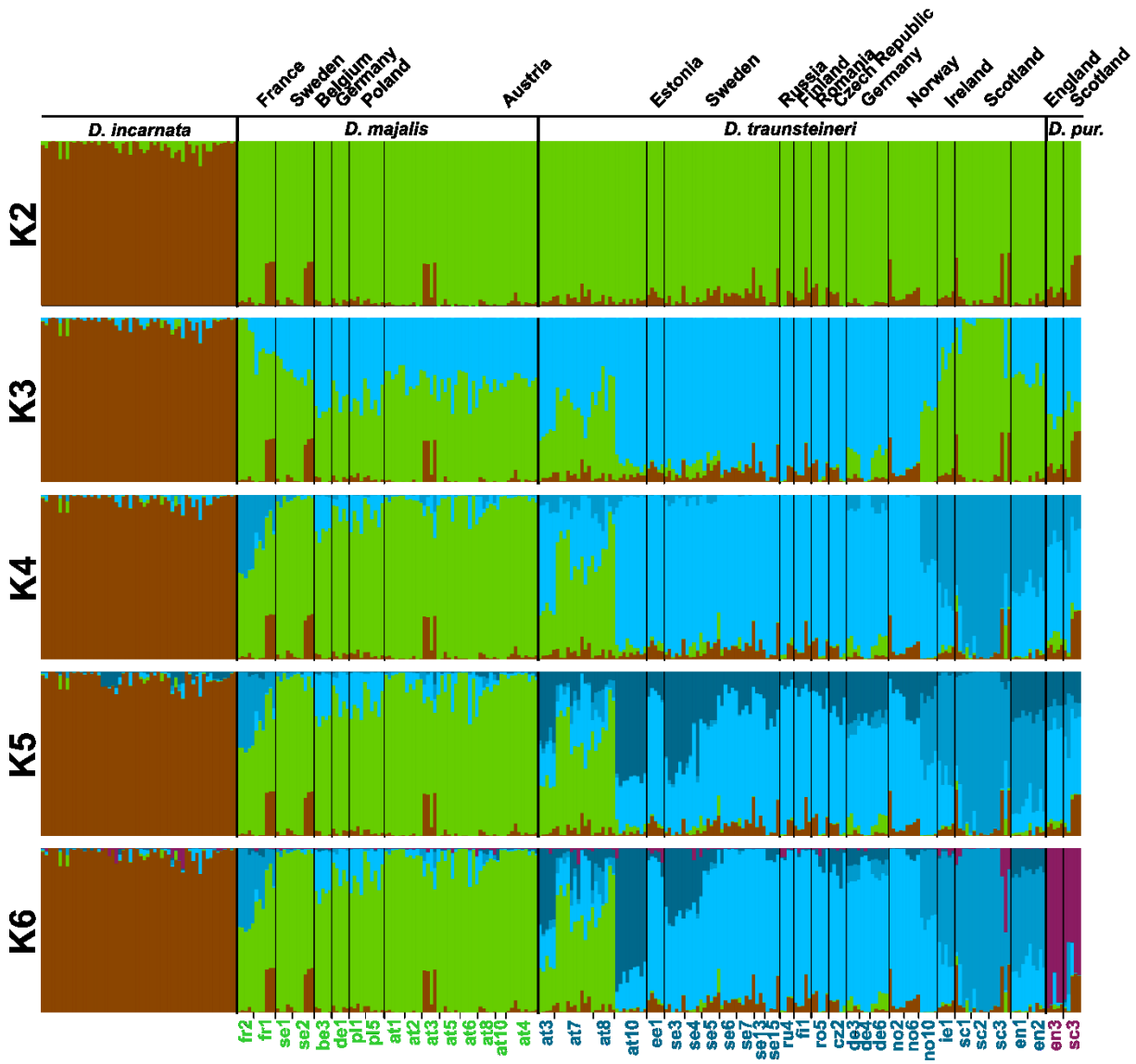

Figure 6: STRUCTURE results for 56 diploid *Dactylorhiza incarnata* accessions (paternal lineage), along with the paternal part of allotetraploid accessions of 241 individuals for *K* = 2 to *K* = 6. Colours represent different gene pools. *D. pur.* abbreviates *Dactylorhiza purpurella*. Individual accession numbers to be found in Figure S 7, likelihood values in Figure S 8.

As already indicated previously, the present analyses revealed overall slightly higher levels of admixture between *D. incarnata* and the paternal part of the allotetraploid genomes than on the maternal side (Figure 7). Higher introgression from the diploid to the allotetraploid level was observed in all three sibling taxa. In *D. majalis* and *D. purpurella* the difference in the direction of introgression was tested significant to highly significant, while no significant difference was detectable in *D. traunsteineri*.

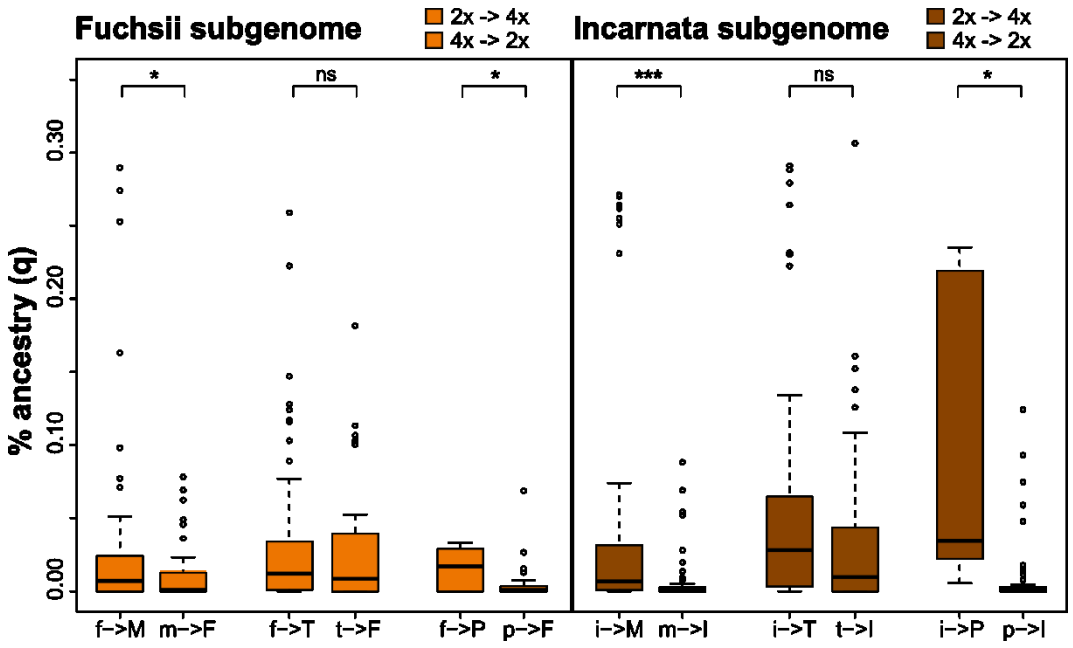

Figure 7: Boxplot diagrams showing relative introgression (as admixture proportions – *q* values) inferred with STRUCTURE. Significance of differences in distribution is indicated towards the upper side of the plots: *, $p < 0.05$; ***, $p < 0.001$; ns, not significant.

Coalescent demographic inference

The coalescent demographic analyses consistently showed three distinct polyploidization events (Table 2): the formation of *D. majalis* was followed by *D. traunsteineri* and later by *D. purpurella* with all three taxa branching off *D. fuchsii* and *D. incarnata*. Taking into account an average generation time for *Dactylorhiza* of 5.8 years (Øien and Moen 2002), the two oldest taxa (*D. majalis* and *D. traunsteineri*) were inferred to have formed in the Pleistocene likely during interglacials, the most recent allopolyploid (*D. purpurella*) emerged later during the Holocene. The effective population sizes (N_e) were calculated for each diploid taxon and each diploid subgenome. N_e of *D. fuchsii* is inferred to be almost three times larger than N_e of the paternal parent *D. incarnata*, and both are inferred to be much larger than those found for the allopolyploids. The two older allotetraploids showed a much larger N_e than the youngest, *D. purpurella*. For the paternal subgenome, gene flow (migration rates) from *D. incarnata* to tetraploids was found to be in the range of 3.0e-03 to 3.5e-06, from tetraploids to *D. incarnata* 1.4e-04 to 1.2e-07, and within tetraploids 8.1e-03 to 3.3e-07. For the maternal subgenome, migration rates were inferred in the ranges 7.6e-04 to 5.3e-06 for *D. fuchsii* to allopolyploids, 1.3e-04 to 1.5e-07 for allopolyploids to *D. fuchsii*, and 8.9e-03 to 5.5e-07 between the allopolyploids. The results coming from the two subgenomes show quite consistent patterns in all researched aspects.

Table 2: Values of *ΔL*, representing the difference between MaxEstLhood and MaxObsLhood. The best model is 3-origins that minimises the value of *ΔL*.

	Subgenome	
Model	***D. fuchsii***	***D. incarnata***
1-origin	849.4	986.9
2-origins 'MT'	841.5	986.5
2-origins 'TP'	888.8	1016.4
3-origins	**813.3**	**975.3**

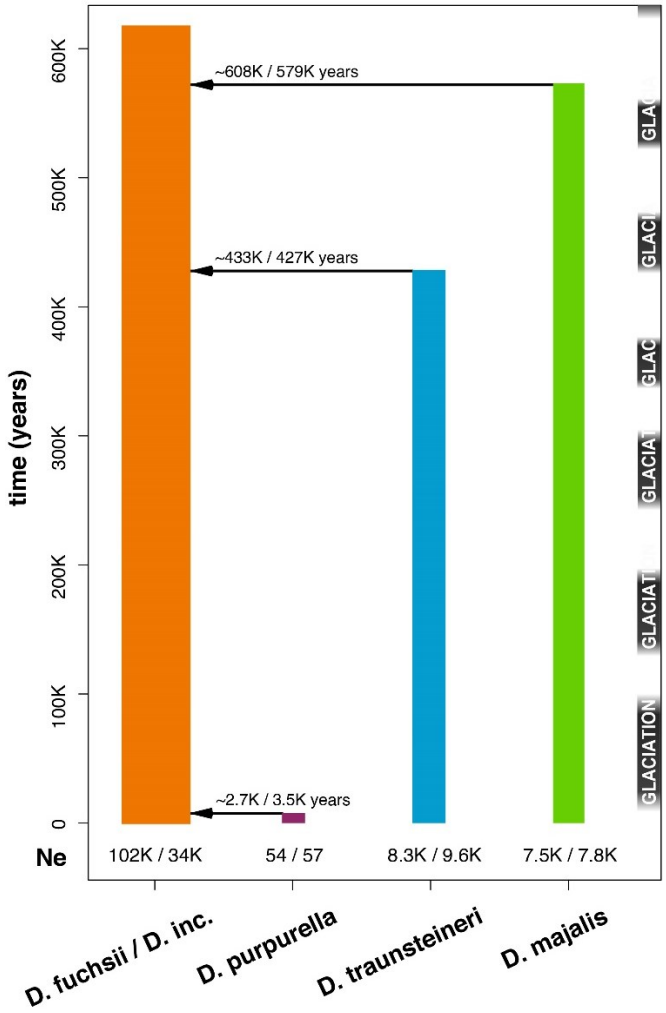

Figure 8: Summary representation of the best model for each subgenome, indicating the effective population size (N_e), and the time of split for each of the allopolyploid species. The first value is the average over best ten runs for the maternal subgenome, and the right values for the paternal subgenome.

Discussion

Data set

After assessing the accuracy of ebg alloSNP by generating and analysing *in-silico* allotetraploids it was found that the paternal part of the subgenome had lower error rates when taking the maternal allelic reference frequencies as the basis for splitting the allotetraploid genotypes; the same hold true for the maternal side. This is probably linked to our lack of knowledge of the exact diploid populations that produced the allopolyploids, and also as a result of further microevolution (i.e., changes in allelic frequencies) within the diploid species themselves, for example during the last glacial cycles. Considering how different the genomes of the two parental lineages are, with maternal *Dactylorhiza fuchsii* being genetically much more heterogeneous than paternal *D. incarnata* (Hedrén and Nordström 2009), chances that certain genotypes are missed by sampling is higher for *D. fuchsii*. Subsequently, estimating the maternal part of the tetraploid genotypes is more difficult than the paternal part. Together with ebg's programmer and co-author Paul Blischak we agreed to use these two data sets for further analyses, therefore yielding higher accuracy without expecting any bias from that specific data selection.

More recent genomic history: Gene flow and introgression

At tetraploid level

When comparing the present results based on genomic data for *Dactylorhiza majalis* and *D. traunsteineri* one can see a very consistent pattern as found in the preceding study using microsatellite loci by Balao et al. (2016). Strong signals of admixture were especially detected in the Alpine region, where the continentally European distributed *D. majalis* meets the Alpine *D. traunsteineri*. Here, these two taxa are still ecologically distinct (Wolfe et al. 2021), nevertheless their preferred habitat types are often found to be spatially very proximate. Those sympatric localities were also covered by this study (i.e., localities at3, at8, at10). *Dactylorhiza traunsteineri* shows at at8 strong admixture despite being the type locality for this species, whereas less so at at3 where *D. traunsteineri* and *D. majalis* are separated by a clump of trees. Finally, at locality at10 the two taxa remain well separated despite growing intermingled, most likely due to a temporal isolating barrier as *D. majalis* flowers approximately three weeks earlier at this site than *D. traunsteineri* (Paun O., personal communication).

The extant disjunct distribution pattern of *D. traunsteineri* is well reflected in its genomic population structure revealing three corresponding gene pools at the genome, and the subgenome levels.

The separation of *D. purpurella* from *D. traunsteineri* in all conducted analyses supports its species status, despite clear signs of introgression with *D. traunsteineri* at sc3.

Across ploidy levels

Orchid seeds are in general small, and travel far by wind ('dust-like'), thus facilitating migration (Bateman 2011, 2006, Rasmussen 1995). Also, pollinators (bees) carry pollinia from one taxon to the other when they are close by, due to the absence of different pollination syndromes within *Dactylorhiza*, and at least partly overlapping phenologies (Bateman and Denholm 1983, Summerhayes 1951). Introgression from the diploid to the tetraploid level was observed to be significantly stronger than vice versa, accounting for additional enrichment of the initially narrow polyploid gene pool.

Apart from allowing introgression to some degree ('porous genomes') in species concepts around the 'genic view of speciation' going back to Wu (2001), hybridisation barriers enable species isolation with endosperm formation as one of the main post-zygotic isolation mechanisms in flowering plants (Köhler, Dziasek, and Del Toro-De León 2021). Endosperm formation is crucial for angiosperm development as it provides nutrients to the growing embryo. Different hypotheses on how it evolved are found in the literature, see Baroux, Spillane, and Grossniklaus (2002). Ontogenetically it emerges out of secondary fertilisation ('double fertilisation'), the fusion of a haploid sperm nucleus with the diploid central cell nucleus of the embryo sac (Linkies et al. 2010) leading to a triploid tissue with a two maternal to one paternal ratio in most species (Lafon-Placette and Köhler 2016). When two plants of different ploidy levels cross, this specific developmental trajectory is disturbed and hence endosperm development fails resulting in embryo abortion (Lafon-Placette and Köhler 2016). This interploidy post-zygotic hybridisation barrier was described as the 'triploid block' after Marks (1966) (Lafon-Placette and Köhler 2016). For orchids, this system does not hold true – they lack an endosperm (Yeung 2017). Orchids are mycoheterotrophic with seeds containing very low amounts of reserve nutrients, and therefore rely on a symbiotic relationship to nutrients-providing fungi (Rasmussen 1995). Inoculation starts with the onset of germination. While the exact inoculation time can vary between taxa, Rasmussen (1995) also mentioned that *Dactylorhiza* seeds are even capable of germination without the immediate presence of a fungal symbiont, but need to get infected within some days or weeks enabling further seedling development. This dependency on a suitable symbiont however eliminates the 'triploid block' and facilitates interploidy hybridisation followed by successful fertilisation.

Additionally, diploids can produce unreduced gametes, which fuse with reduced tetraploid gametes resulting in diploid-to-tetraploid introgression (Husband 2004, Schinkel et al. 2017). However, the role this process might play in that particular system of marsh orchids is not yet fully researched.

Going further back in time: Effective population sizes of diploids and the allotetraploids' origin(s)

Effective population size N_e

First of all, a clear difference in the effective population size (N_e) of the two parental taxa was detected. As already mentioned before, one of them, *D. incarnata*, likely went through a genetic bottleneck when migrating from Asia to Europe (Hedrén and Nordström 2009). A loss of genetic diversity (as here induced by migration) is coupled with a decrease in N_e (Nei, Maruyama, and Chakraborty 1975), well reflected by the results from the presented demographic modelling. In addition, *D. incarnata* stands are usually encountered around wet habitats that are limited in frequency and extent. Hence, potentially limiting effective population size compared with *D. fuchsii* that grows at forest margins and in meadows.

Effective population sizes of the three allotetraploid siblings in general increase with age. However, N_e of the (in terms of age) intermediate *D. traunsteineri* was found to be slightly higher than N_e of the oldest allotetraploid *D. majalis*. This might be explained by the current disjunct distribution and gene flow coming especially from *D. majalis* in the Alpine region, enhancing isolation by distance, and thus maintaining stronger separated gene pools than in the continuously distributed *D. majalis*; reoccurring introgression further fuels the genetic diversity of *D. traunsteineri*. This explanation is supported by the analysis of pairwise relatedness (Figure 3), and also by the STRUCTURE analyses (Figure 4, Figure 5, Figure 6). Furthermore, these results correspond well to the previously estimated numbers of private alleles, and studies on the relatedness to parental lineages by Brandrud et al. (2020).

Age of allotetraploids

Previous analyses on haplotype data suggested a very recent origin after the Last Glacial Maximum (LGM) of *D. purpurella* (Hedrén, Nordström, and Bateman 2011), which can be confirmed by the present study. It is characterised by a dominance of haplotype 60, which is just one mutation away from a frequent haplotype in British *D. fuchsii* (Hedrén, Nordström, and Bateman 2011), and also appears well isolated from any other taxon from the genus, as discussed above.

Dactylorhiza traunsteineri and *D. majalis* were considered to be older, with a decrease in age towards the North (Pillon et al. 2007). However, their inferred age implies survival of several glacial periods, followed by range expansion.

Haplotypes of British *D. traunsteineri* (*D. traunsteinerioides*) were found to be shared with continental European *D. traunsteineri*, and *D. majalis*, indicating a continental origin of *D. traunsteineri*, followed by northward migration (Hedrén, Nordström, and Bateman 2011, Pillon et al. 2007). These results are supported by demographic modelling presented herein.

Past climatic circumstances around polyploid formation events

Given the modelled allotetraploid ages, one can attempt to assign their origins to standardised geological stages, the so-called 'Marine Isotope Stages' (MIS) introduced by Cesare Emiliani in 1955 on the basis of palaeotemperatures inferred from oxygen isotope ratios of pelagic Foraminifera in marine sediment cores (Emiliani 1955, Shackleton and West 1969). Each allotetraploid was found to have originated in distinct MIS stages: *D. majalis* during MIS 15, *D. traunsteineri* during MIS 11 (also known as the 'Hoxnian stage') and *D. purpurella* in the ongoing MIS 1 stage. All three stages are interglacial periods, a term vaguely defined, but in general interglacials happen globally, and are characterised by a sea level rise, accompanied by a decrease in marine oxygen isotope ratios, increased sea surface, and air temperatures, increased concentration of Greenhouse Gases, especially CO_2 and methane (Past Interglacials Working Group of PAGES 2016).

Interestingly, MIS 15, the stage where *D. majalis* emerged accordingly to the presented analyses, was found to be the wettest, and strongest vegetated interglacial of the last 800K years; maximum temperatures range from 14.1 to 14.4 °C at 57.51°N in the Atlantic region, which corresponds approximately to the centre of *D. majalis'* current distribution range, the maximum temperature of MIS 11 and MIS 1 was/is about 1 °C higher (Past Interglacials Working Group of PAGES 2016). MIS 11 is often referred to as an analogue to the ongoing MIS 1, therefore used to forecast future climate scenarios (Tzedakis 2010). Due to a lack of detailed knowledge on past climates, and geophysical properties, the Past Interglacials Working Group of PAGES (2016) questions this status. Nevertheless, it is notable that there is some climatic similarity between those two stages in which *D. traunsteineri*, and *D. purpurella* emerged accordingly to demographic modelling.

Conclusions and outlook

Testing the accuracy of the genotype-likelihoods based ebg alloSNP tool revealed a low error rate, and thus allows an accurate separation of allopolyploid homeologs. These two diploid data sets then enable more sophisticated analyses of each subgenome in isolation, including STRUCTURE analyses along with recent parental accessions, and demographic inference based on a composite likelihood framework with FastSimCoal2 using joint Site Frequency Spectra. The presented results from RADseq data well support previous studies with microsatellite and haplotype data. Frequent gene flow between sibling allopolyploids was detected. More introgression was found in the direction from the diploid taxa to the tetraploids rather than vice versa. Further, each of the three studied allotetraploids was found to have emerged in different interglacials independently of each other.

One of the next steps would be to have a closer look on the evolution of *D. traunsteineri* via running coalescent demographic modelling for this taxon separately.

Acknowledgements

I thank Ovidiu Paun for his patience with supervising this almost never-ending Master's project, all his advice, and help even during some very difficult times. Thanks to Thibault Leroy, who became an integral part of our meetings during the second half of this project, Marie Brandrud for her support in the beginning and Paul Blischak for helping out with questions related to ebg. A big thank you also to the whole group of Ecological Genomics, especially Mimmi Eriksson, Aglaia Szukala and Christina Hedderich, our current and previous lab technicians Dana Paun, Marie Huber and Juliane Baar, as well as our collaborators Richard Bateman, Mark Chase and Mikael Hedrén. Funding was provided by Austrian Science Fund (FWF), project number Y661-B16 to O.P.

References

Aagaard, S. M. D., S. M. Såstad, J. Greilhuber, and A. Moen. 2005. "A secondary hybrid zone between diploid *Dactylorhiza incarnata* ssp. *cruenta* and allotetraploid *D. lapponica* (Orchidaceae)." *Heredity* 94 (5):488-496. doi: 10.1038/sj.hdy.6800643.

Allendorf, Fred W., Susan Bassham, William A. Cresko, Morten T. Limborg, Lisa W. Seeb, and James E. Seeb. 2015. "Effects of crossovers between homeologs on inheritance and population genomics in polyploid-derived salmonid fishes." *Journal of Heredity* 106 (3):217-227.

Andrews, Simon. 2018. "FastQC: A quality control tool for high throughput sequence data."

Balao, F., M. Tannhäuser, Maria T. Lorenzo, Mikael Hedrén, and Ovidiu Paun. 2016. "Genetic differentiation and admixture between sibling allopolyploids in the *Dactylorhiza majalis* complex." *Heredity* 116:351-361. doi: 10.1038/hdy.2015.98.

Balao, F., E. Trucchi, T. M. Wolfe, B. H. Hao, M. T. Lorenzo, J. Baar, L. Sedman, C. Kosiol, F. Amman, M. W. Chase, M. Hedren, and O. Paun. 2017. "Adaptive sequence evolution is driven by biotic stress in a pair of orchid species (*Dactylorhiza*) with distinct ecological optima." *Molecular Ecology* 26 (14):3649-3662. doi: 10.1111/mec.14123.

Baroux, Célia, Charles Spillane, and Ueli Grossniklaus. 2002. "Evolutionary origins of the endosperm in flowering plants." *Genome Biology* 3 (9):reviews1026.1. doi: 10.1186/gb-2002-3-9-reviews1026.

Bateman, R. M., and I. Denholm. 1983. "A reappraisal of the British and Irish dactylorchids, 1. The tetraploid marsh-orchids." *Watsonia* 14:347-376.

Bateman, Richard M. 2006. "How many orchid species are currently native to the British Isles?" In *Current Taxonomic Research on the British and European Flora*, edited by J. P. Bailey and R. G. Ellis, 89–110. London: Botanical Society of the British Isles.

Bateman, Richard M. 2011. "Glacial progress: do we finally understand the narrow-leaved marsh-orchids?" *New Journal of Botany* 1 (1):2-15.

Blischak, Paul D., Laura S. Kubatko, and Andrea D. Wolfe. 2018. "SNP genotyping and parameter estimation in polyploids using low-coverage sequencing data." *Bioinformatics* 34 (3):407-415.

Brandrud, Marie K., Juliane Baar, Maria T. Lorenzo, Alexander Athanasiadis, Richard M. Bateman, Mark W. Chase, Mikael Hedrén, and Ovidiu Paun. 2020. "Phylogenomic Relationships of Diploids and the Origins of Allotetraploids in *Dactylorhiza* (Orchidaceae)." *Systematic Biology*. doi: 10.1093/sysbio/syz035.

Brandrud, Marie Kristine. 2019. "Chapter 2: The impact of recurrent origins, genetic drift and gene flow on the genetic structure of allopolyploid marsh orchids." In *The use of phylogenomic tools to investigate diploid and polyploid evolution in Dactylorhiza and other orchids*. University of Vienna: PhD thesis, Department of Botany and Biodiversity Research.

Broad Institute. 2018. "Picard toolkit." *Broad Institute, GitHub repository*.

Butruille, D. V., and L. S. Boiteux. 2000. "Selection–mutation balance in polysomic tetraploids: impact of double reduction and gametophytic selection on the frequency and subchromosomal localization of deleterious mutations." *Proceedings of the National Academy of Sciences* 97 (12):6608-6613.

Comai, Luca, Andreas Madlung, Caroline Josefsson, and Anand Tyagi. 2003. "Do the different parental 'heteromes' cause genomic shock in newly formed allopolyploids?" *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences* 358 (1434):1149-1155.

Comber, Steven C. L. E., and Carl Smith. 2004. "Polyploidy in fishes: patterns and processes." *Biological Journal of the Linnean Society* 82 (4):431-442. doi: 10.1111/j.1095-8312.2004.00330.x.

Earl, Dent A., and Bridgett M. vonHoldt. 2012. "STRUCTURE HARVESTER: a website and program for visualizing STRUCTURE output and implementing the Evanno method." *Conservation Genetics Resources* 4 (2):359-361. doi: 10.1007/s12686-011-9548-7.

Emiliani, Cesare. 1955. "Pleistocene temperatures." *The Journal of Geology* 63 (6):538-578.

Evanno, G., S. Regnaut, and J. Goudet. 2005. "Detecting the number of clusters of individuals using the software structure: a simulation study." *Molecular Ecology* 14 (8):2611-2620. doi: 10.1111/j.1365-294X.2005.02553.x.

Excoffier, Laurent, Isabelle Dupanloup, Emilia Huerta-Sánchez, Vitor C. Sousa, and Matthieu Foll. 2013. "Robust Demographic Inference from Genomic and SNP Data." *PLoS Genetics* 9 (10):e1003905. doi: 10.1371/journal.pgen.1003905.

Garnier, Simon. 2018. "viridis: Default Color Maps from 'matplotlib'."

Glover, Natasha M., Henning Redestig, and Christophe Dessimoz. 2016. "Homoeologs: What Are They and How Do We Infer Them?" *Trends in Plant Science* 21 (7):609-621. doi: 10.1016/j.tplants.2016.02.005.

Hedrén, Mikael, and Sofie Nordström. 2009. "Polymorphic populations of *Dactylorhiza incarnata s.l.* (Orchidaceae) on the Baltic island of Gotland: morphology, habitat preference and genetic differentiation." *Annals of Botany* 104 (3):527-542. doi: 10.1093/aob/mcp102.

Hedrén, Mikael, Sofie Nordström, and Richard M. Bateman. 2011. "Plastid and nuclear DNA marker data support the recognition of four tetraploid marsh orchids (*Dactylorhiza majalis s.l.*, Orchidaceae) in Britain and Ireland, but require their recircumscription." *Biological Journal of the Linnean Society* 104 (1):107-128. doi: 10.1111/j.1095-8312.2011.01708.x.

Hedrén, Mikael, Sofie Nordström, Helena A. Persson Hovmalm, Henrik Ærenlund Pedersen, and Sven Hansson. 2007. "Patterns of polyploid evolution in Greek marsh orchids (*Dactylorhiza*; Orchidaceae) as revealed by allozymes, AFLPs, and plastid DNA data." *American Journal of Botany* 94 (7):1205-1218.

Huang, K., K. Ritland, S. Guo, M. Shattuck, and B. Li. 2014. "A pairwise relatedness estimator for polyploids." *Molecular Ecology Resources* 14 (4):734-44. doi: 10.1111/1755-0998.12217.

Husband, Brian C. 2004. "The role of triploid hybrids in the evolutionary dynamics of mixed-ploidy populations." *Biological Journal of the Linnean Society* 82 (4):537-546.

Jakobsson, Mattias, and Noah A. Rosenberg. 2007. "CLUMPP: a cluster matching and permutation program for dealing with label switching and multimodality in analysis of population structure." *Bioinformatics* 23 (14):1801-1806. doi: 10.1093/bioinformatics/btm233.

Köhler, Claudia, Katarzyna Dziasek, and Gerardo Del Toro-De León. 2021. "Postzygotic reproductive isolation established in the endosperm: mechanisms, drivers and relevance." *Philosophical Transactions of the Royal Society B: Biological Sciences* 376 (1826):20200118. doi: 10.1098/rstb.2020.0118.

Korneliussen, Thorfinn Sand, Anders Albrechtsen, and Rasmus Nielsen. 2014. "ANGSD: Analysis of Next Generation Sequencing Data." *BMC Bioinformatics* 15 (1):356. doi: 10.1186/s12859-014-0356-4.

Lafon-Placette, Clément, and Claudia Köhler. 2016. "Endosperm-based postzygotic hybridization barriers: developmental mechanisms and evolutionary drivers." *Molecular Ecology* 25 (11):2620-2629. doi: 10.1111/mec.13552.

Leitch, A. R., and I. J. Leitch. 2008. "Genomic Plasticity and the Diversity of Polyploid Plants." *Science* 320 (5875):481. doi: 10.1126/science.1153585.

Leitch, I. J., L. Hanson, K. Y. Lim, A. Kovarik, M. W. Chase, J. J. Clarkson, and A. R. Leitch. 2008. "The ups and downs of genome size evolution in polyploid species of Nicotiana (Solanaceae)." *Annals of Botany* 101 (6):805-814.

Li, Heng. 2013. "Aligning sequence reads, clone sequences and assembly contigs with BWA-MEM." *arXiv preprint arXiv:1303.3997*.

Li, Heng, Bob Handsaker, Alec Wysoker, Tim Fennell, Jue Ruan, Nils Homer, Gabor Marth, Goncalo Abecasis, Richard Durbin, and Subgroup Genome Project Data Processing. 2009. "The Sequence Alignment/Map format and SAMtools." *Bioinformatics* 25 (16):2078-2079. doi: 10.1093/bioinformatics/btp352.

Linkies, Ada, Kai Graeber, Charles Knight, and Gerhard Leubner-Metzger. 2010. "The evolution of seeds." *New Phytologist* 186 (4):817-831. doi: 10.1111/j.1469-8137.2010.03249.x.

Marks, G. E. 1966. "The origin and significance of intraspecific polyploidy: experimental evidence from *Solanum chacoense*." *Evolution*:552-557.

McClintock, Barbara. 1983. "The significance of responses of the genome to challenge." *Nobel Lecture*.

McClintock, Barbara. 1984. "The significance of responses of the genome to challenge." *Science* 266:792-801.

McKenna, Aaron, Matthew Hanna, Eric Banks, Andrey Sivachenko, Kristian Cibulskis, Andrew Kernytsky, Kiran Garimella, David Altshuler, Stacey Gabriel, and Mark Daly. 2010. "The Genome Analysis Toolkit: a

MapReduce framework for analyzing next-generation DNA sequencing data." *Genome research* 20 (9):1297-1303.

Meirmans, P. G., and P. H. Van Tienderen. 2013. "The effects of inheritance in tetraploids on genetic diversity and population divergence." *Heredity* 110 (2):131-137.

Meirmans, Patrick G., Shenglin Liu, and Peter H. van Tienderen. 2018. "The Analysis of Polyploid Genetic Data." *Journal of Heredity* 109 (3):283-296. doi: 10.1093/jhered/esy006.

Meisner, Jonas, and Anders Albrechtsen. 2018. "Inferring Population Structure and Admixture Proportions in Low-Depth NGS Data." *Genetics* 210:719-731. doi: 10.1534/genetics.118.301336.

Nei, Masatoshi, Takeo Maruyama, and Ranajit Chakraborty. 1975. "The bottleneck effect and genetic variability in populations." *Evolution*:1-10.

Nelson, Erich. 1976. *Monographie und Ikonographie der Orchideaceen-Gattung Dactylorhiza*. Zürich: Speich.

Nordström, Sofie, and Mikael Hedrén. 2008. "Genetic differentiation and postglacial migration of the *Dactylorhiza majalis* ssp. *traunsteineri/lapponica* complex into Fennoscandia." *Plant Systematics and Evolution* 276 (1-2):73-87.

Nordström, Sofie, and Mikael Hedrén. 2009. "Genetic diversity and differentiation of allopolyploid *Dactylorhiza* (Orchidaceae) with particular focus on the *Dactylorhiza majalis* ssp. *traunsteineri/lapponica* complex." *Biological Journal of the Linnean Society* 97 (1):52-67. doi: 10.1111/j.1095-8312.2008.01183.x.

Øien, D.-I., and A. Moen. 2002. "Flowering and survival of *Dactylorhiza lapponica* and *Gymnadenia conopsea* in the Sølendet Nature Reserve, Central Norway." In *Trends and fluctuations and underlying mechanisms in terrestrial orchid populations*, edited by P. Kindlmann, J. H. Willems and D. F. Whigham, 3-22. Leiden: Backhuys Publishers.

Oliveira, C., L. F. Almeida-Toledo, L. Mori, and S. A. Toledo-Filho. 1992. "Extensive chromosomal rearrangements and nuclear DNA content changes in the evolution of the armoured catfishes genus *Corydoras* (Pisces, Siluriformes, Callichthyidae)." *Journal of Fish Biology* 40:419-431.

Oliveira, Claudio, Lurdes F. Almeida-Toledo, Lyria Mori, and Silvio Almeida Toledo-Filho. 1993. "Cytogenetic and DNA content in six genera of the family Callichthyidae (Pisces, Siluriformes)." *Caryologia* 46 (2-3):171-188. doi: 10.1080/00087114.1993.10797258.

Otto, Sarah P., and Jeannette Whitton. 2000. "Polyploid incidence and evolution." *Annual Review of Genetics* 34 (1):401-437.

Pante, Eric, and Benoit Simon-Bouhet. 2013. "marmap: a package for importing, plotting and analyzing bathymetric and topographic data in R." *PLoS One* 8 (9):e73051.

Past Interglacials Working Group of PAGES. 2016. "Interglacials of the last 800,000 years." *Reviews of Geophysics* 54 (1):162-219. doi: 10.1002/2015RG000482.

Paun, Ovidiu, Richard M. Bateman, Michael F. Fay, Mikael Hedrén, Laure Civeyrel, and Mark W. Chase. 2010. "Stable Epigenetic Effects Impact Adaptation in Allopolyploid Orchids (*Dactylorhiza*: Orchidaceae)." *Molecular Biology and Evolution* 27 (11):2465-2473. doi: 10.1093/molbev/msq150.

Paun, Ovidiu, Michael F. Fay, Douglas E. Soltis, and Mark W. Chase. 2007. "Genetic and epigenetic alterations after hybridization and genome doubling." *Taxon* 56 (3):649-656.

Pillon, Yohan, Michael F. Fay, Mikael Hedrén, Richard M. Bateman, Dion S. Devey, Alexey B. Shipunov, Michelle van der Bank, and Mark W. Chase. 2007. "Evolution and temporal diversification of western European polyploid species complexes in *Dactylorhiza* (Orchidaceae)." *Taxon* 56 (4):1185-1208.

Pritchard, Jonathan K., Matthew Stephens, and Peter Donnelly. 2000. "Inference of Population Structure Using Multilocus Genotype Data." *Genetics* 155 (2):945-959.

R: A language and environment for statistical computing. v. 3.4.4, available from https://www.R-project.org/. R Foundation for Statistical Computing, Vienna, Austria.

R: A language and environment for statistical computing. v. 4.0.2, available from https://www.R-project.org/. R Foundation for Statistical Computing, Vienna, Austria.

Ramsey, Justin, and Douglas W. Schemske. 2002. "Neopolyploidy in flowering plants." *Annual Review of Ecology and Systematics* 33 (1):589-639.

Rasmussen, Hanne N. 1995. *Terrestial Orchid: From Seed to Mycotrophic Plant*. Cambridge: Cambridge University Press.

Schinkel, Christoph C. F., Bernhard Kirchheimer, Stefan Dullinger, Danny Geelen, Nico De Storme, and Elvira Hörandl. 2017. "Pathways to polyploidy: indications of a female triploid bridge in the alpine species *Ranunculus kuepferi* (Ranunculaceae)." *Plant Systematics and Evolution* 303 (8):1093-1108. doi: 10.1007/s00606-017-1435-6.

Shackleton, Nicholas John, and Richard Gilbert West. 1969. "The last interglacial in the marine and terrestrial records." *Proceedings of the Royal Society of London. Series B. Biological Sciences* 174 (1034):135-154. doi: 10.1098/rspb.1969.0085.

Soltis, Douglas E., and Pamela S. Soltis. 1993. "Molecular data and the dynamic nature of polyploidy." *Critical Reviews in Plant Sciences* 12 (3):243-273.

Soltis, Douglas E., and Pamela S. Soltis. 1999. "Polyploidy: recurrent formation and genome evolution." *Trends in Ecology & Evolution* 14 (9):348-352.

Soltis, Douglas E., Pamela S. Soltis, J. Chris Pires, Ales Kovarik, Jennifer A. Tate, and Evgeny Mavrodiev. 2004. "Recent and recurrent polyploidy in *Tragopogon* (Asteraceae): cytogenetic, genomic and genetic comparisons." *Biological Journal of the Linnean Society* 82 (4):485-501.

Soltis, Douglas E., Clayton J. Visger, and Pamela S. Soltis. 2014. "The polyploidy revolution then…and now: Stebbins revisited." *American Journal of Botany* 101 (7):1057-1078. doi: 10.3732/ajb.1400178.

South, Andy. 2011. "rworldmap: A New R package for Mapping Global Data." *The R Journal* 3 (1):35-43.

South, Andy. 2012. "rworldxtra: Country boundaries at high resolution. R package version 1.01."

Ståhlberg, David. 2007. "Systematics, phylogeography and polyploid evolution in the *Dactylorhiza maculata* complex (Orchidaceae)." PhD, Department of Ecology, Lund University.

Stebbins Jr, G. Ledyard. 1947. "Types of polyploids: their classification and significance." *Advances in Genetics* 1:403-429.

Summerhayes, Victor S. 1951. *Wild orchids of Britain*. London.

Tzedakis, P. C. 2010. "The MIS 11–MIS 1 analogy, southern European vegetation, atmospheric methane and the" early anthropogenic hypothesis"." *Climate of the Past* 6 (2):131-144.

Warnes, Gregory R., Ben Bolker, Lodewijk Bonebakker, Robert Gentleman, Wolfgang Huber, Andy Liaw, Thomas Lumley, Martin Maechler, Arni Magnusson, Steffen Moeller, Marc Schwartz, and Bill Venables. 2019. "gplots: Various R Programming Tools for Plotting Data." *R package version 3.0.1.1*.

Weiss-Schneeweiss, H., K. Emadzade, T. S. Jang, and G. M. Schneeweiss. 2013. "Evolutionary Consequences, Constraints and Potential of Polyploidy in Plants." *Cytogenetic and Genome Research* 140 (2-4):137-150. doi: 10.1159/000351727.

Wolfe, Thomas M., Francisco Balao, Emiliano Trucchi, Gert Bachmann, Wenjia Gu, Juliane Baar, Mikael Hedrén, Wolfram Weckwerth, Andrew R. Leitch, and Ovidiu Paun. 2021. "Recurrent allopolyploidization events diversify eco-physiological traits in marsh orchids." *bioRxiv*:2021.08.28.458039. doi: 10.1101/2021.08.28.458039.

Wu, Chung-I. 2001. "The genic view of the process of speciation." *Journal of Evolutionary Biology* 14 (6):851-865. doi: 10.1046/j.1420-9101.2001.00335.x.

Xiong, Zhiyong, Robert T. Gaeta, Patrick P. Edger, Yao Cao, Kanglu Zhao, Siqi Zhang, and J. Chris Pires. 2021. "Chromosome inheritance and meiotic stability in allopolyploid *Brassica napus*." *G3 Genes|Genomes|Genetics* 11 (2). doi: 10.1093/g3journal/jkaa011.

Yeung, Edward C. 2017. "A perspective on orchid seed and protocorm development." *Botanical Studies* 58 (1):33. doi: 10.1186/s40529-017-0188-4.

Appendix

Supplementary figures and tables

Table S 1: Collection data for the used accessions. Collectors are: EW, Erik Westberg; FB, Francisco Balao; LW, Łukasz Wilk; MH, Mikael Hedrén; OP, Ovidiu Paun; RD, R. Dundr; RMB, Richard M. Bateman; SN, Sofie Nordström; SS, S. Sczepanski; UH, U. Heidtke.

Species	Pop. acronym	Locality	Country	Latitude	Longitude	Collector	No. accessions	Accession numbers
fuchsii	**en1f**	North York Moors, Sand Dale	England	54.2528	-0.6851	OP	1	1247
fuchsii	**en2f**	North York Moors, Seive Dale Fen	England	54.2816	-0.6897	OP	1	1259
fuchsii	**en12f**	Box Hill, Hampshire	England	51.2333	-0.3167	MH	5	717-721
fuchsii	**ee1f**	Saaremaa, Viidumäe	Estonia	58.2833	22.1333	MH	3	5237-5238, 5253
fuchsii	**fr4f**	Isola2000	France	44.1852	7.1617	MH	3	7799-7800, 7810
fuchsii	**fr7f**	Belesta	France	42.8945	1.9294	OP	5	1707-1708, 1710, 1713-1714
fuchsii	**it1f**	Ahrntal-Növes	Italy	46.9971	11.9567	MH	4	12694-12695, 12697-12698
fuchsii	**ne2f**	Voorne	Netherlands	51.8253	5.4388	MH	3	5618, 5620-5621
fuchsii	**no1f**	Talvik	Norway	70.0500	22.9500	MH	2	5962, 5964
fuchsii	**no2f**	Gjellebekk, Griserud	Norway	59.8167	10.3000	MH	5	12415-12419
fuchsii	**pl2f**	Sopot, Pomorskie	Poland	54.4667	18.5500	MH	1	5521
fuchsii	**ru1f**	Paksuniemi, Karelia	Russia	61.6667	31.3667	MH	4	1395-1396, 1398-1399
fuchsii	**ru6f**	Vourijärvi I, Kola Peninsula	Russia	66.7833	30.1500	MH	2	5913, 5916

Table S 1 (continued).

fuchsii	**sc3f**	N Uist	Scotland	57.6854	-7.2057	OP	2	1855, 1858
fuchsii	**se30f**	Måryd	Sweden	55.6833	13.3833	MH	5	1376, 1378-1381
fuchsii	**se31f**	Grinduga	Sweden	60.6386	17.3101	MH	2	7225, 7227
incarnata	**at9i**	Inzing	Austria	47.2817	11.1846	OP	5	1585-1586, 1588-1589, 1594
incarnata	**en4i**	Ainsdale hills, Merseyside	England	53.6530	-3.0650	MH/SN/RMB	2	6129, 6132
incarnata	**en6i**	Beaulieu Road, Hampshire	England	50.8166	-1.4833	MH/SN/RMB	2	6103, 6105
incarnata	**fi1i**	Moskuvaara	Finland	67.5666	26.8666	SN	3	2861-2862, 2864
incarnata	**fr1i**	along D29	France	42.8616	1.9808	OP	4	1012, 1717, 1735, 1738
incarnata	**hu2i**	near Sopron	Hungary	47.6540	16.7265	OP, FB	4	1769-1772
incarnata	**hu3i**	C Hungary, W Dabas, dairy meadow	Hungary	47.2532	19.1928	RMB	1	14560
incarnata	**ne3i**	Voorne	Netherlands	51.8969	4.0858	MH	1	6035
incarnata	**no2i**	Gjellebekk, Griserud	Norway	59.8167	10.3000	MH	4	2891-2893, 12484
incarnata	**no4i**	Obrestad	Norway	58.6500	5.5667	MH	5	12467-12470, 12472
incarnata	**pl3i**	Prysniewo, Pomorskie	Poland	54.6519	18.1461	MH	1	8818
incarnata	**ru7i**	Kosalma, Uksijärvi, Karelia	Russia	62.1167	34.0000	MH	2	2828, 2830
incarnata	**sc3i**	N Uist	Scotland	57.6854	-7.2057	OP	5	1870, 1873, 1875-1877
incarnata	**se2i**	Kristianstadt, E shore of Lyngsjön	Sweden	55.9310	14.0683	MH, OP	4	1281-1283, 8799
incarnata	**se3i**	Lojsthajd, Gotland	Sweden	57.3402	18.3212	MH, OP	4	1412, 1904-1905, 2032

Table S 1 (continued).

incarnata	**se4i**	Kauparve, Rute, Gotland	Sweden	57.8171	18.8954	MH, OP	5	1906-1908, 10271-10272
incarnata	**se7i**	Ansätten, Kattygelmyren	Sweden	63.8500	14.0333	MH	3	8032-8034
incarnata	**se9i**	Algutsrum, Nötbrunnskärret, Öl	Sweden	56.6772	16.5507	MH	1	10576
majalis	**at1m**	Altenberg	Austria	47.68553	15.65581	OP	6	1031-1032, 1034, 1567, 1569-1570
majalis	**at2m**	Mooshuben	Austria	47.74486	15.35083	OP	5	1476-1477, 1572-1574
majalis	**at3m**	St. Ulrich	Austria	47.52928	12.57853	OP	5	1638, 1660-1661, 1666, 1669
majalis	**at4m**	Fuschlsee	Austria	47.81667	13.2500	OP	9	1693, 1695-1697, 1699, 1703, 1775-1777
majalis	**at5m**	Josersee	Austria	47.58567	15.0995	OP	6	1131-1132, 1134-1135, 1137, 1139
majalis	**at6m**	Rinnerbergerbaches	Austria	47.90542	14.1655	OP	5	1142, 1145, 1148-1150
majalis	**at8m**	Kitzbühel	Austria	47.4610	12.36564	OP	5	1651-1653, 1655, 1657
majalis	**at10m**	Gusswerk, Greith	Austria	47.71667	15.21667	OP	3	1943, 1950-1951
majalis	**be3m**	Saint Hubert	Belgium	50.03194	5.3737	MH	5	14114-14117, 14119
majalis	**de1m**	Hamburg, Over	Germany	53.43333	10.1000	EW	5	3879-3880, 3883, 3887-3888
majalis	**fr2m**	Bourg D'oueil	France	42.85872	0.4953	OP	5	1743, 1745-1748
majalis	**fr1m**	Belcaire-Espezel, D29	France	42.86164	1.9808	OP	6	1007, 1010, 1015, 1715-1716, 1722
majalis	**pl1m**	Lubiana, Pomorskie	Poland	54.11661	17.88331	MH	5	3107, 3109-3112
majalis	**pl5m**	Szlembark, Tatra Mts, Małopolskie	Poland	49.47389	20.21228	LW	5	3130-3132, 3134-3135
majalis	**se1m**	Lanskrona, Saxtorp	Sweden	55.81781	12.94561	OP	5	1763-1767

Table S 1 (continued).

majalis	**se2m**	Kristianstadt, Lyngsjön	Sweden	55.9310	14.06831	OP	6	1288-1289, 1291, 1758, 1760-1761
purpurella	**en3p**	Alston Moor	England	54.78407	-2.4210	OP	5	1785-1788, 1790
purpurella	**sc3p**	N Uist	Scotland	57.68539	-7.2057	OP	5	1863-1864, 1867-1869
traunsteineri	**at3t**	St. Ulrich	Austria	47.52928	12.57853	OP	5	1623, 1668, 1670-1672
traunsteineri	**at7t**	Prilau	Austria	47.34131	12.80525	OP	11	1383, 1388, 1390, 1508, 1529-1533, 1552-1553
traunsteineri	**at8t**	Kitzbühel	Austria	47.4610	12.36564	OP	6	1429, 1432, 1639, 1641-1643
traunsteineri	**at10t**	Gusswerk, Greith	Austria	47.71667	15.21667	OP	9	1940-1942, 1944-1946, 1949, 1956-1957
traunsteineri	**cz2t**	Baronský pond, Jestrebí/Ceská Lípa	Czech Republic	50.05681	15.71175	RD	5	6354-6358
traunsteineri	**de3t**	Spiesswald, Miesau-Buchholz, Kaiserslautern	Germany	49.53333	7.9500	UH	4	10131-10134
traunsteineri	**de4t**	Peenewiesen bei Gütskow	Germany	53.92056	13.43331	SS	3	10118, 10120-10121
traunsteineri	**de6t**	Eppenbrunn	Germany	49.09369	7.5653	Lo206	5	10998-10999, 12000-12002
traunsteineri	**ee1t**	Viidumäe, Saaremaa	Estonia	58.28331	22.13331	MH	5	4187, 4192-4193, 4196-4197
traunsteineri	**en1t**	North York Moors, Sand Dale	England	54.25278	-0.6851	OP	5	1233, 1238, 1240, 1798, 1803
traunsteineri	**en2t**	North York Moors, Seive Dale Fen	England	54.28158	-0.6898	OP	5	1805-1809
traunsteineri	**fi1t**	Moskuvaara	Finland	67.56661	26.86661	SN	5	4146-4150
traunsteineri	**ie1t**	Crockravar, NW Carnlough, Antrim, Ulster	Ireland	54.99081	-5.9934	RMB/ID	5	14506-14510

Table S 1 (continued).

traunsteineri	**no2t**	Gjellebekk, Griserud	Norway	59.81667	10.3000	MH	5	12612-12616
traunsteineri	**no6t**	Nordmarka i Surnadal kommune	Norway	63.03333	8.8667	MH	4	8507-8508, 8510-8511
traunsteineri	**no10t**	Sola	Norway	58.88406	5.6040	MH	5	12535-12539
traunsteineri	**ro5t**	Feleacu, Valea Morii	Romania	46.71494	23.62167	MH	5	4432-4436
traunsteineri	**ru4t**	Petrozavodsk, Matrosy, Karelia	Russia	61.78333	33.8000	MH	4	3010-3013
traunsteineri	**sc1t**	Applecross	Scotland	57.4220	-5.8193	OP	5	1180-1181, 1812, 1814-1815
traunsteineri	**sc2t**	Loch Kernsary	Scotland	57.7670	-5.5690	OP	5	1182-1184, 1810-1811
traunsteineri	**sc3t**	N Uist	Scotland	57.68539	-7.2057	OP	6	1173, 1826, 1828, 1830, 1832-1833
traunsteineri	**se3t**	Gotland, Lojsthajd	Sweden	57.34019	18.32119	OP	6	1404, 1894-1896, 1901-1902
traunsteineri	**se4t**	Gotland, Kauparve	Sweden	57.81708	18.89536	OP	5	1414, 1417, 1909-1910, 1914
traunsteineri	**se5t**	ML	Sweden	60.61472	17.55833	OP	5	1918-1921, 2143
traunsteineri	**se6t**	Armasjärvi	Sweden	66.32275	23.5085	MH	5	4084-4086, 4088, 4093
traunsteineri	**se7t**	Ansätten, Kattygelmyren	Sweden	63.8500	14.03333	MH	5	8020-8024
traunsteineri	**se13t**	Ramsele, Långmyran	Sweden	63.51667	16.41667	MH	3	3680-3682
traunsteineri	**se15t**	Sjogerstad	Sweden	58.31872	13.8030	MH	4	10323, 10325-10327

Table S 2: Overview of the diploid accessions used to generate in-silico allotetraploids for testing the accuracy of ebg alloSNP.

ID	***fuchsii*-"mother"**	**No. *fuchsii* reads**	***incarnata*-"father"**	**No. *incarnata* reads**
Alexej	f_1396_ru1	873,794	i_2861_fi1	1,082,278
Brida	f_718_en12	807,365	i_6132_en4	1,000,000
Carla	f_5621_ne2	855,151	i_6035_ne3	1,059,187
Dave	f_1259_en2	708,060	i_1870_sc3	877,000
Edda	f_1376_se30	764,837	i_1281_se2	947,324
Freya	f_12416_no2	791,114	i_12484_no2	979,871
Gary	f_1855_sc3	875,712	i_1873_sc3	1,084,654
Hertha	f_5964_no1	872,808	i_2891_no2	1,081,057
Isa	f_7800_fr4	807,365	i_1738_fr1	1,000,000
Jean	f_1708_fr7	877,256	i_1717_fr1	1,086,566

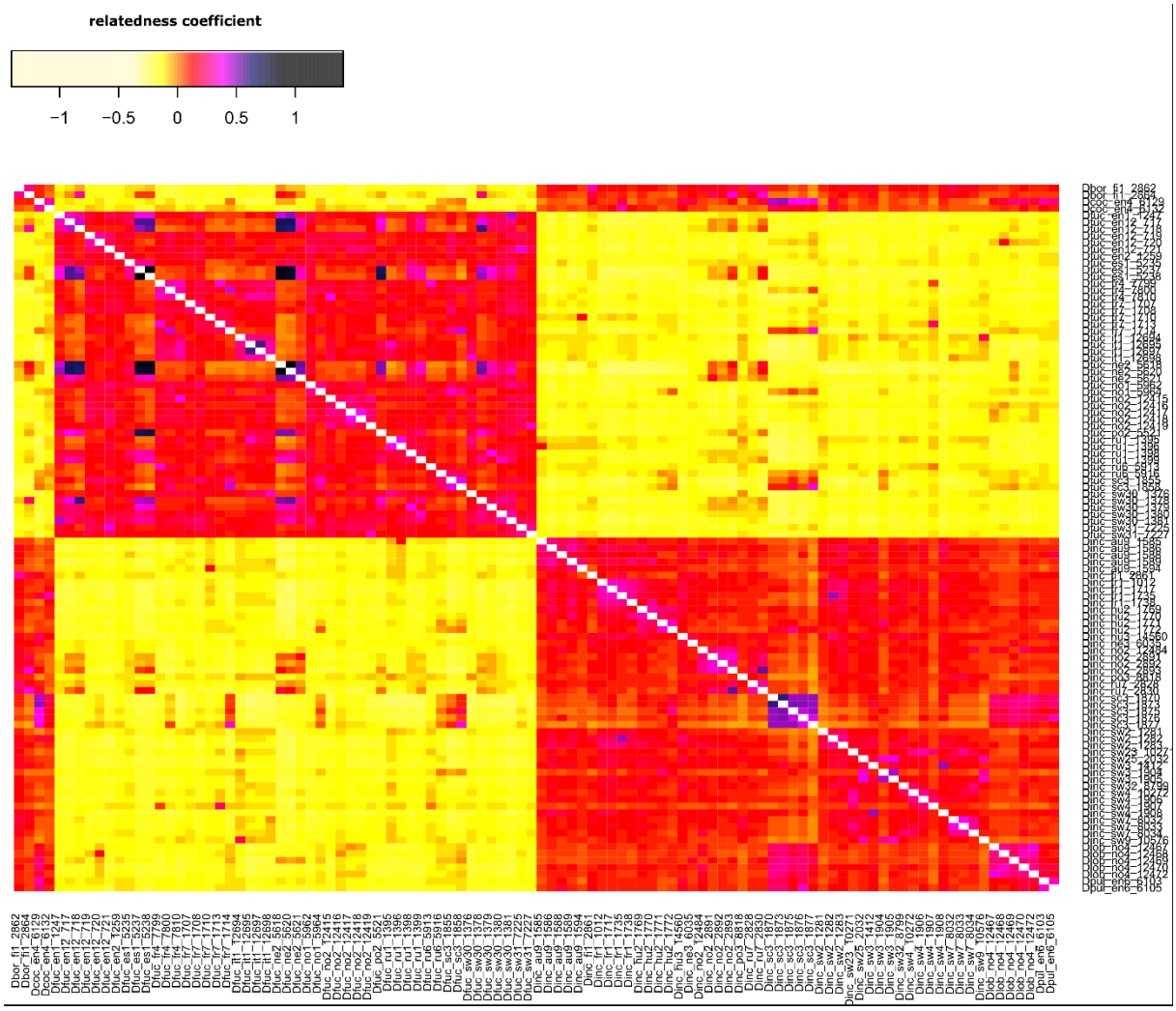

Figure S 1: Heatmap of pairwise relatedness of the 104 diploid accessions from *Dactylorhiza fuchsii* and *D. incarnata*. Colours indicate degrees of relatedness according to the legend. The estimates on the diagonal have been excluded to optimise for colour resolution.

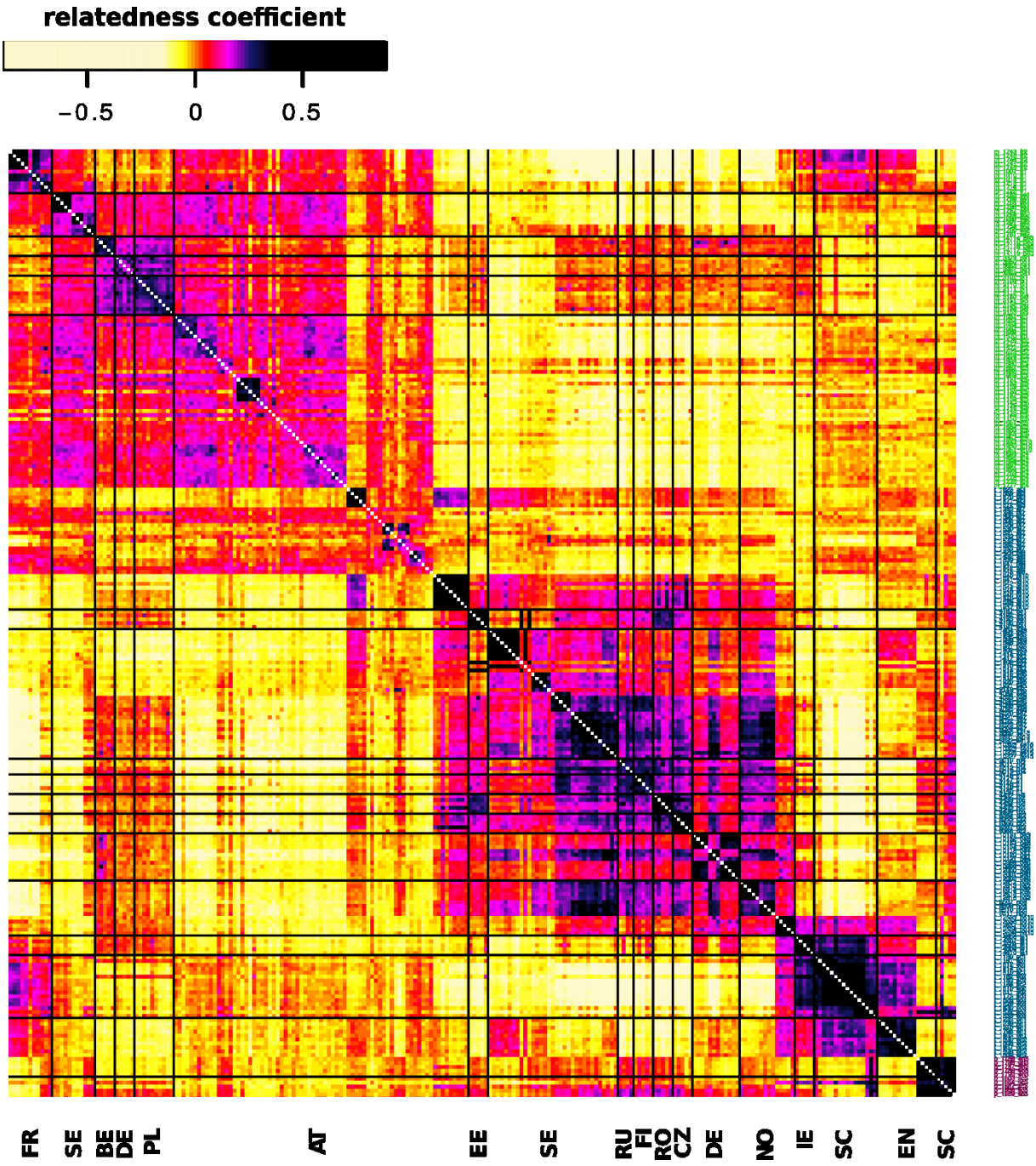

Figure S 2: Heatmap of pairwise relatedness (Huang et al. 2014) between 241 allotetraploid *Dactylorhiza* accessions showing individual accession numbers. Colours indicate degrees of relatedness according to the legend. The estimates on the diagonal have been excluded to optimise for colour resolution. On the lower x-axis country codes are given, on the right side accession numbers are given.

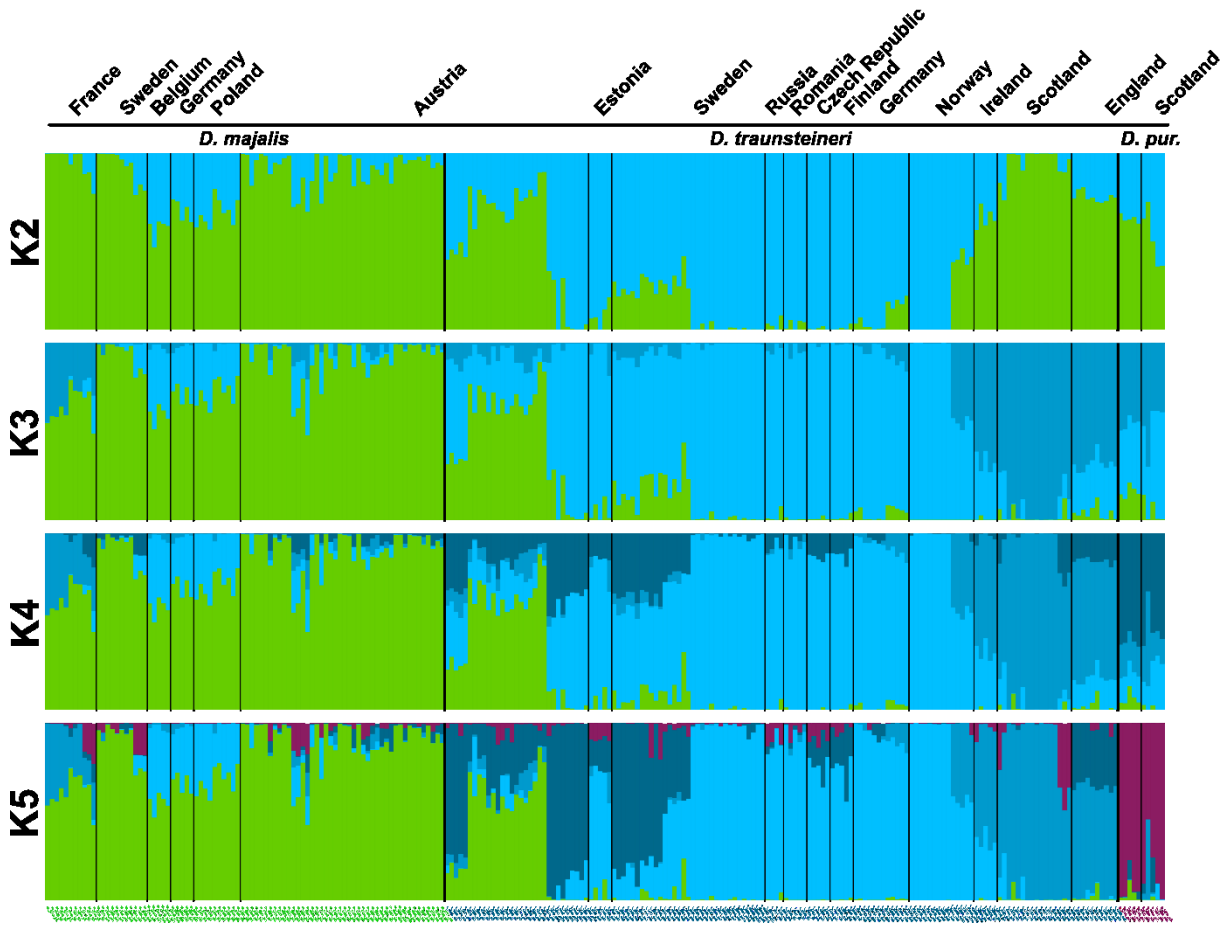

Figure S 3: STRUCTURE results for the tetraploid-encoded data set (i.e., both homeologs together) of 241 individuals for *K* = 2 to *K* = 5 with individual accession numbers. Colours represent different gene pools. *D. pur.* abbreviates *Dactylorhiza purpurella*.

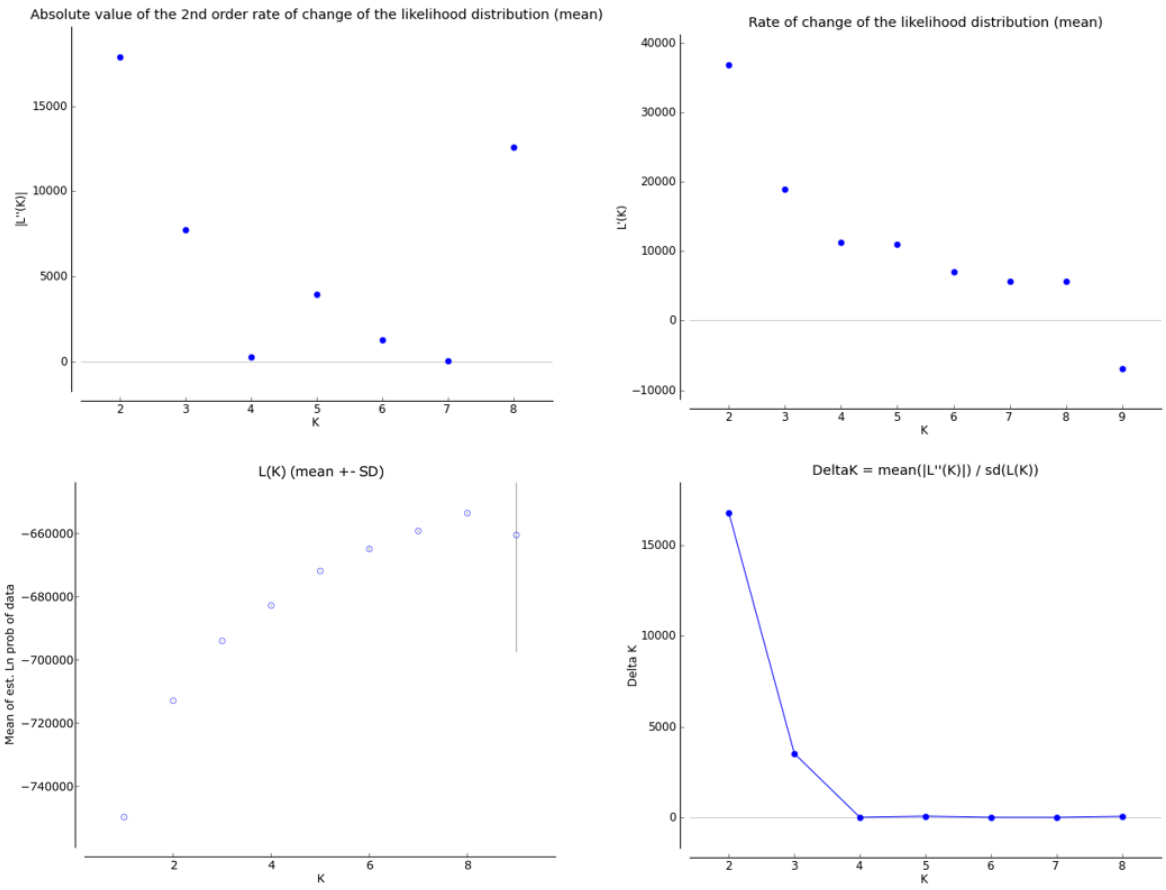

Figure S 4: Likelihood values following Evanno's method for *K* = 2 to *K* = 8 of the allotetraploid STRUCTURE analyses.

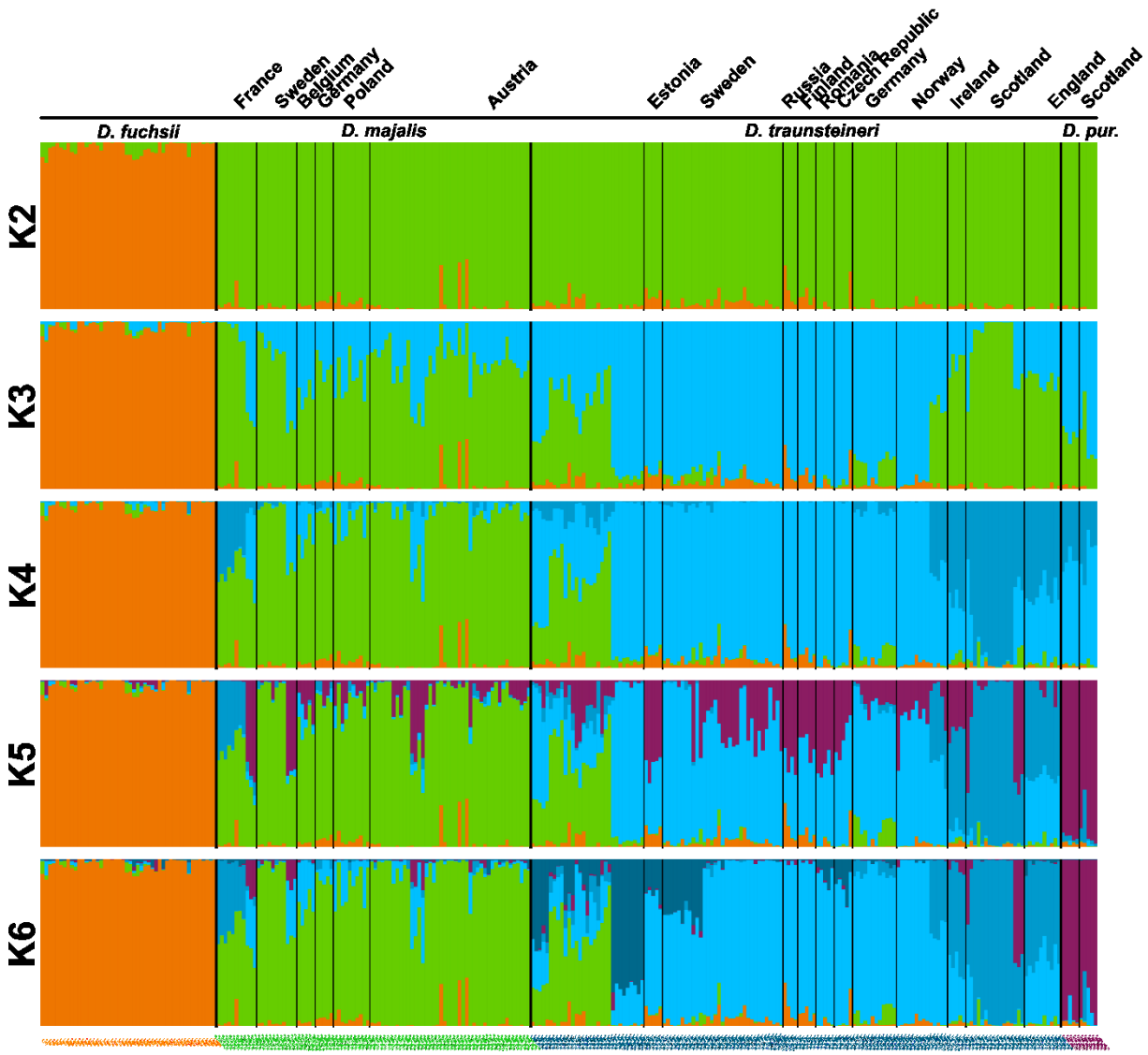

Figure S 5: STRUCTURE results for 48 diploid *Dactylorhiza fuchsii* accessions (maternal lineage), along with the diploid maternal part of allotetraploid accessions of 241 individuals for *K* = 2 to *K* = 6 showing individual accession numbers. Colours represent different gene pools. *D. pur.* abbreviates *Dactylorhiza purpurella*.

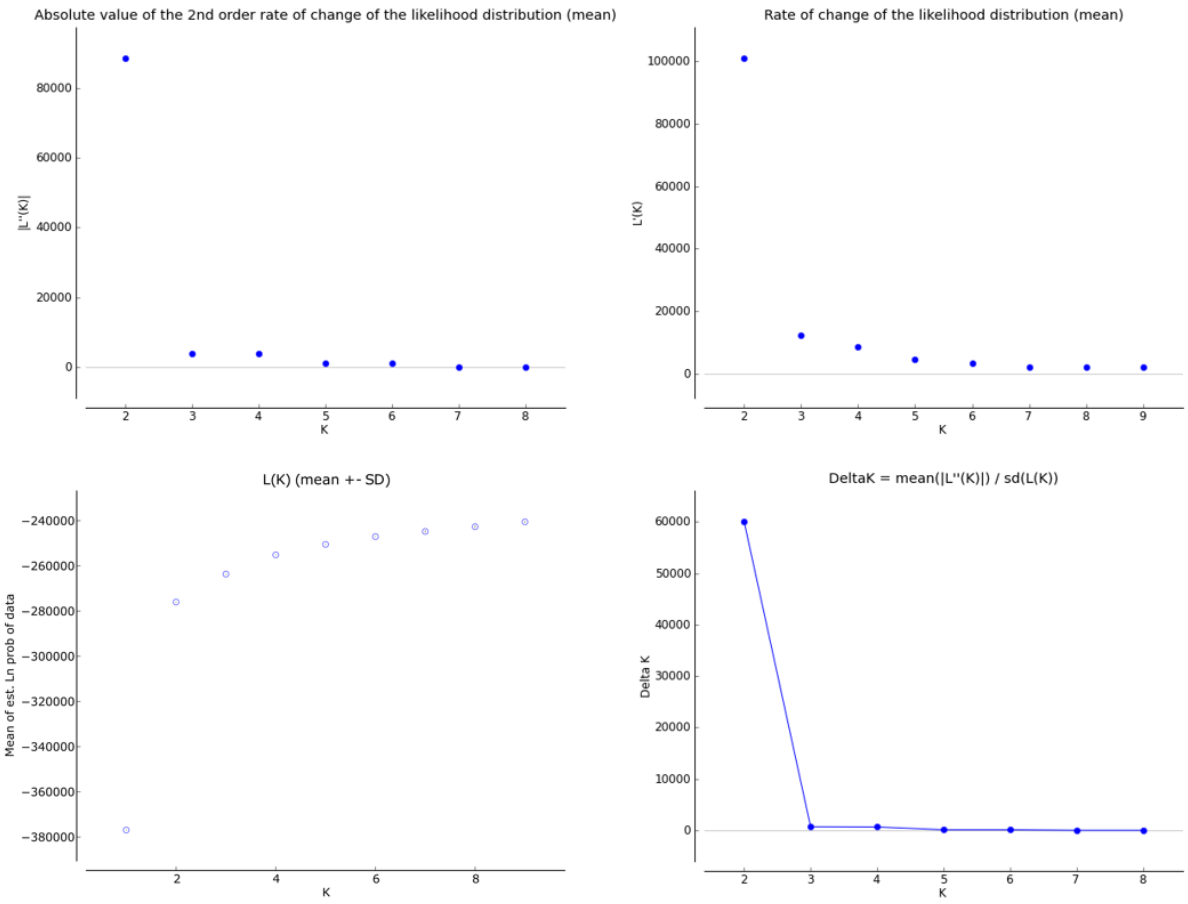

Figure S 6: Likelihood values following Evanno's method for *K* = 2 to *K* = 8 of the STRUCTURE analyses of the maternal side.

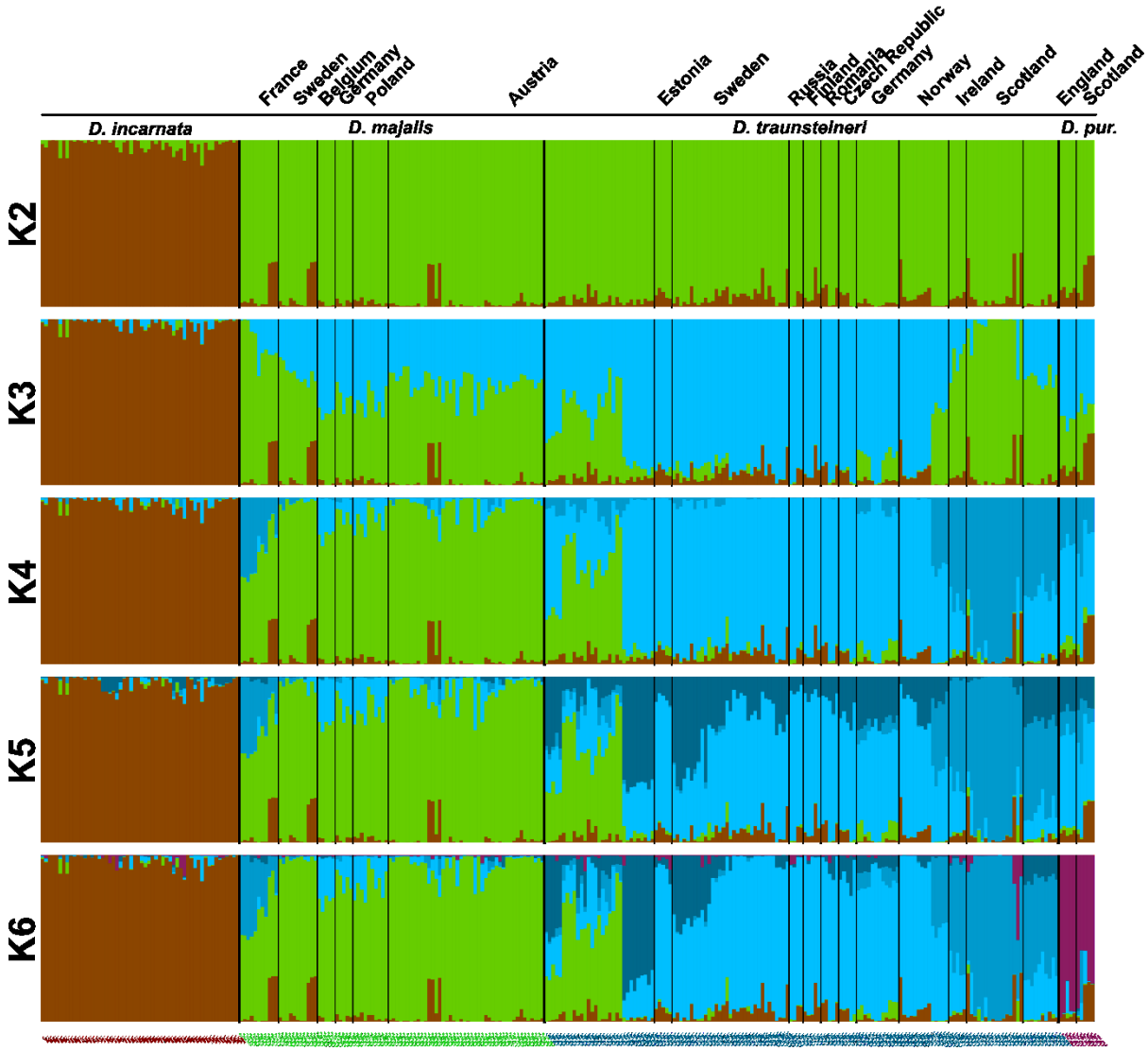

Figure S 7: STRUCTURE results for 56 diploid *Dactylorhiza incarnata* accessions (paternal lineage), along with the diploid paternal part of allotetraploid accessions of 241 individuals for *K* = 2 to *K* = 6 showing individual accession numbers. Colours represent different gene pools. *D. pur.* abbreviates *Dactylorhiza purpurella*.

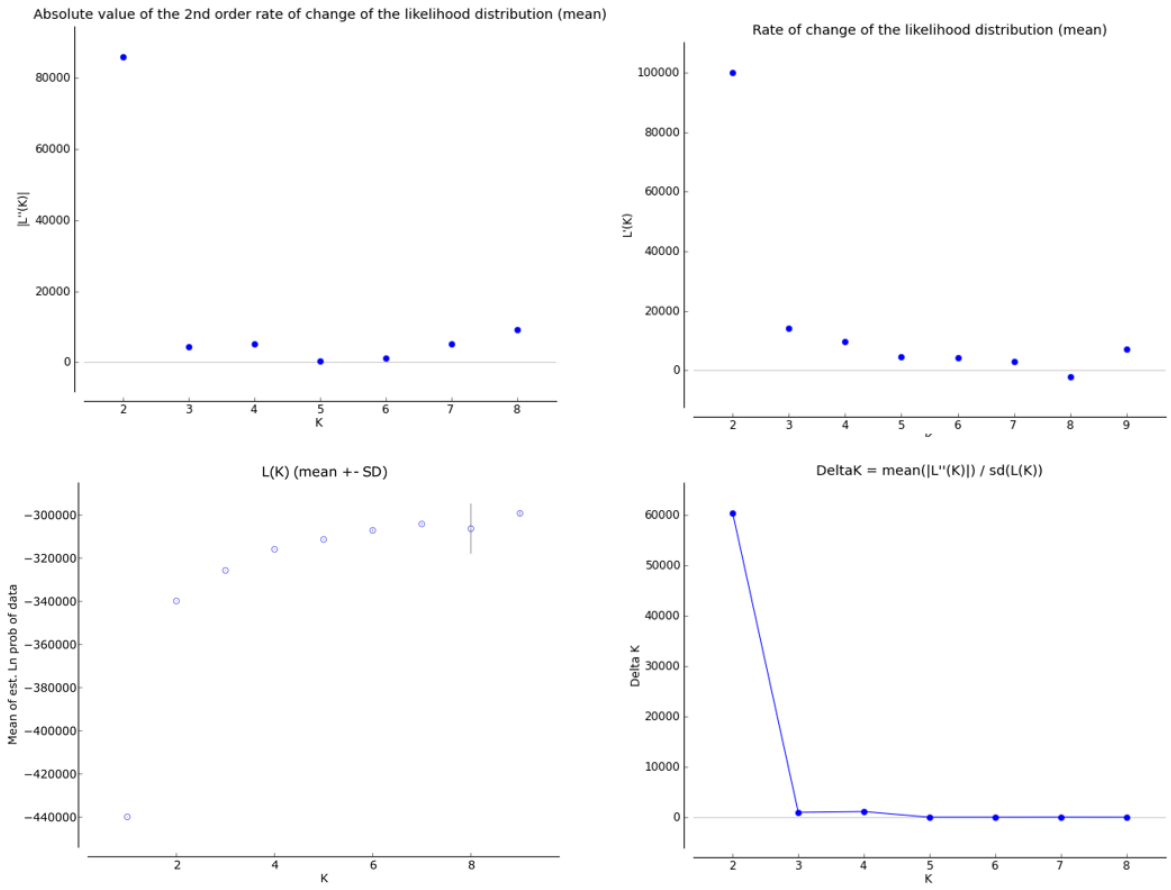

Figure S 8: Likelihood values following Evanno's method for *K* = 2 to *K* = 8 of the STRUCTURE analyses of the paternal side.

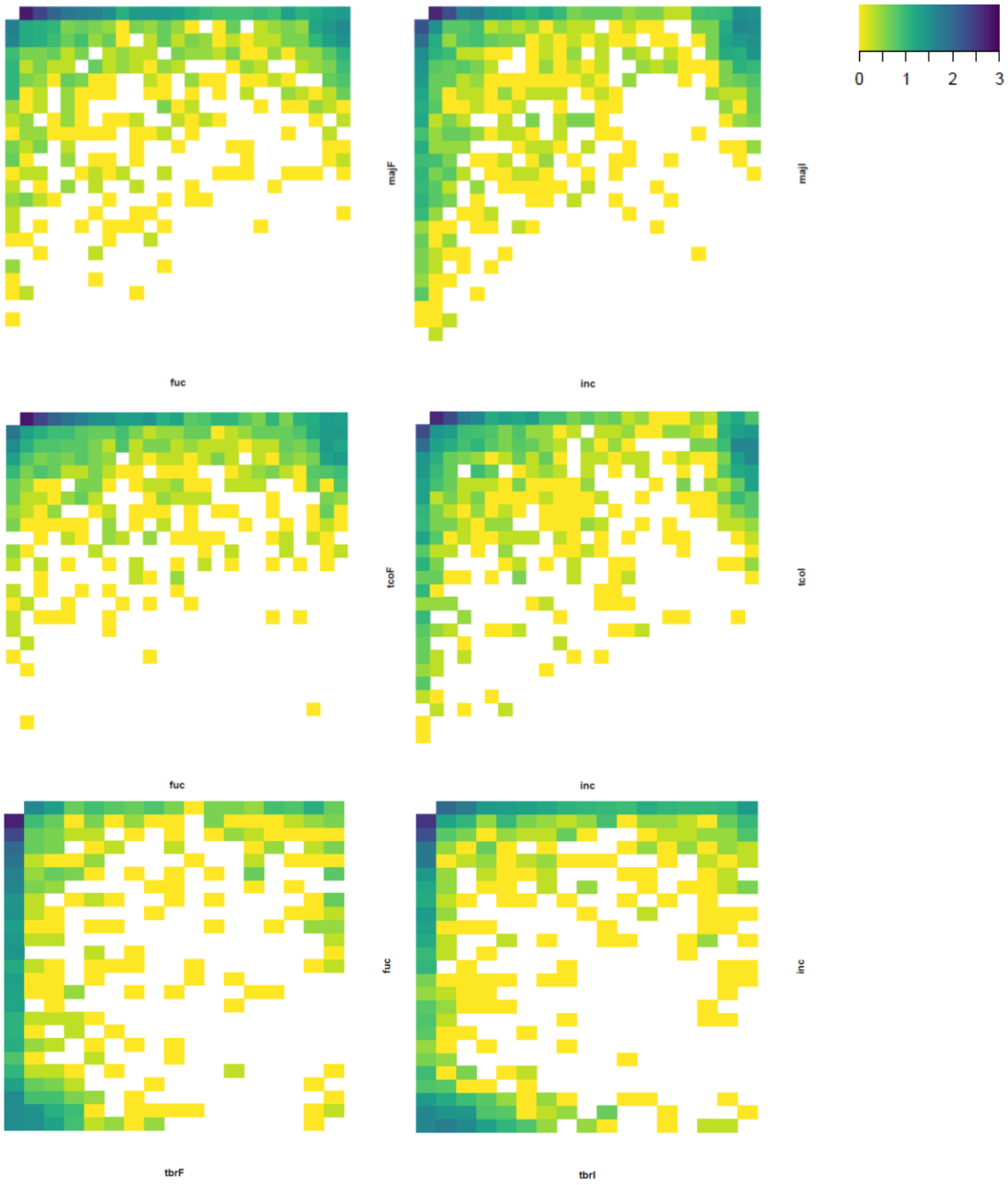

Figure S 9: Folded joint site frequency spectra for group pairs (log10-transformed, maternal (F) and paternal (I) genotypes each). Abbrev.: fuc, *Dactylorhiza fuchsii*; inc, *D. incarnata*; maj, *D. majalis*; tco, *D. traunsteineri* continental European group; tbr, *D. traunsteineri* British group; pur, *D. purpurella*.

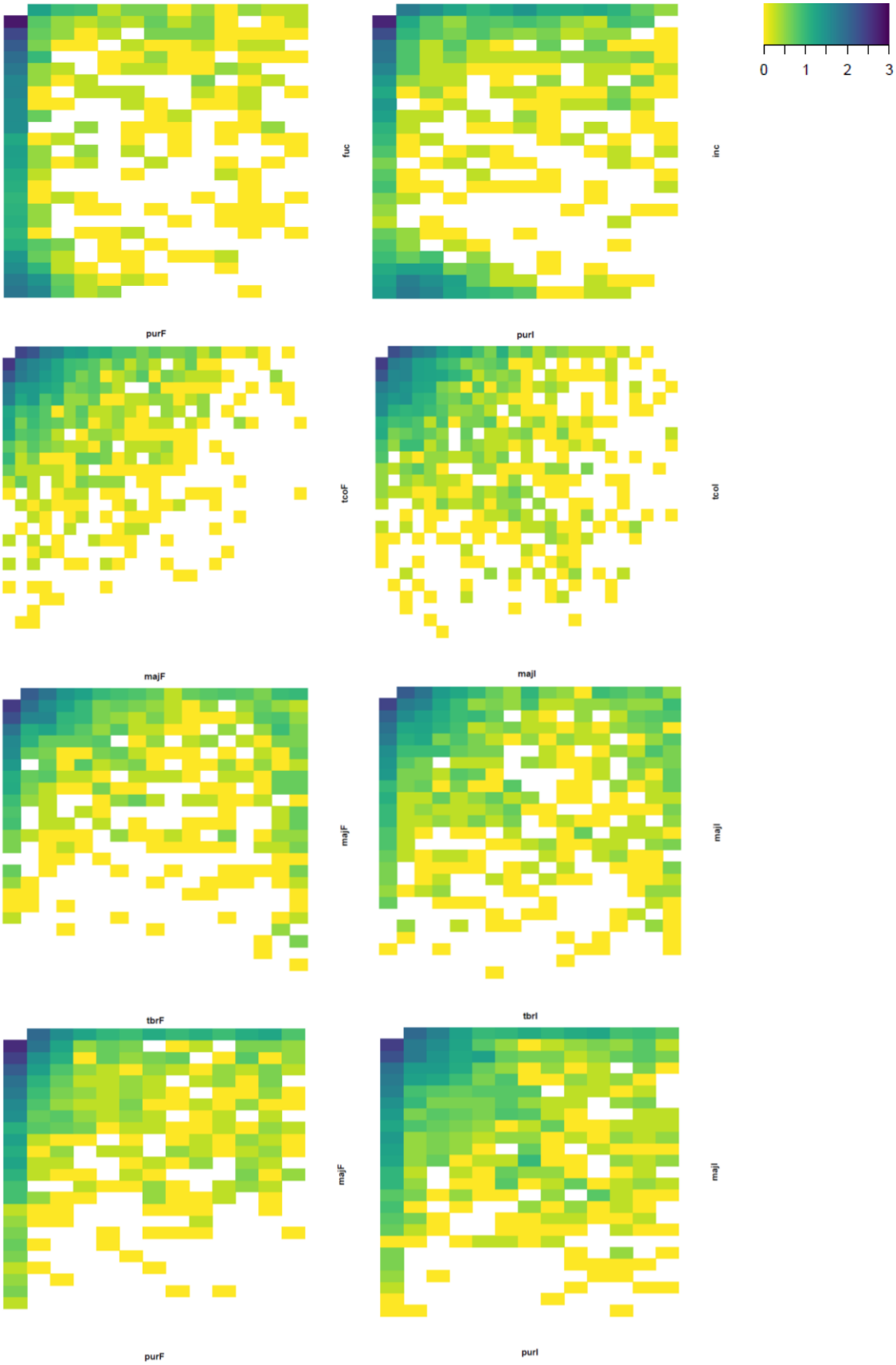

Figure S 8 (continued).

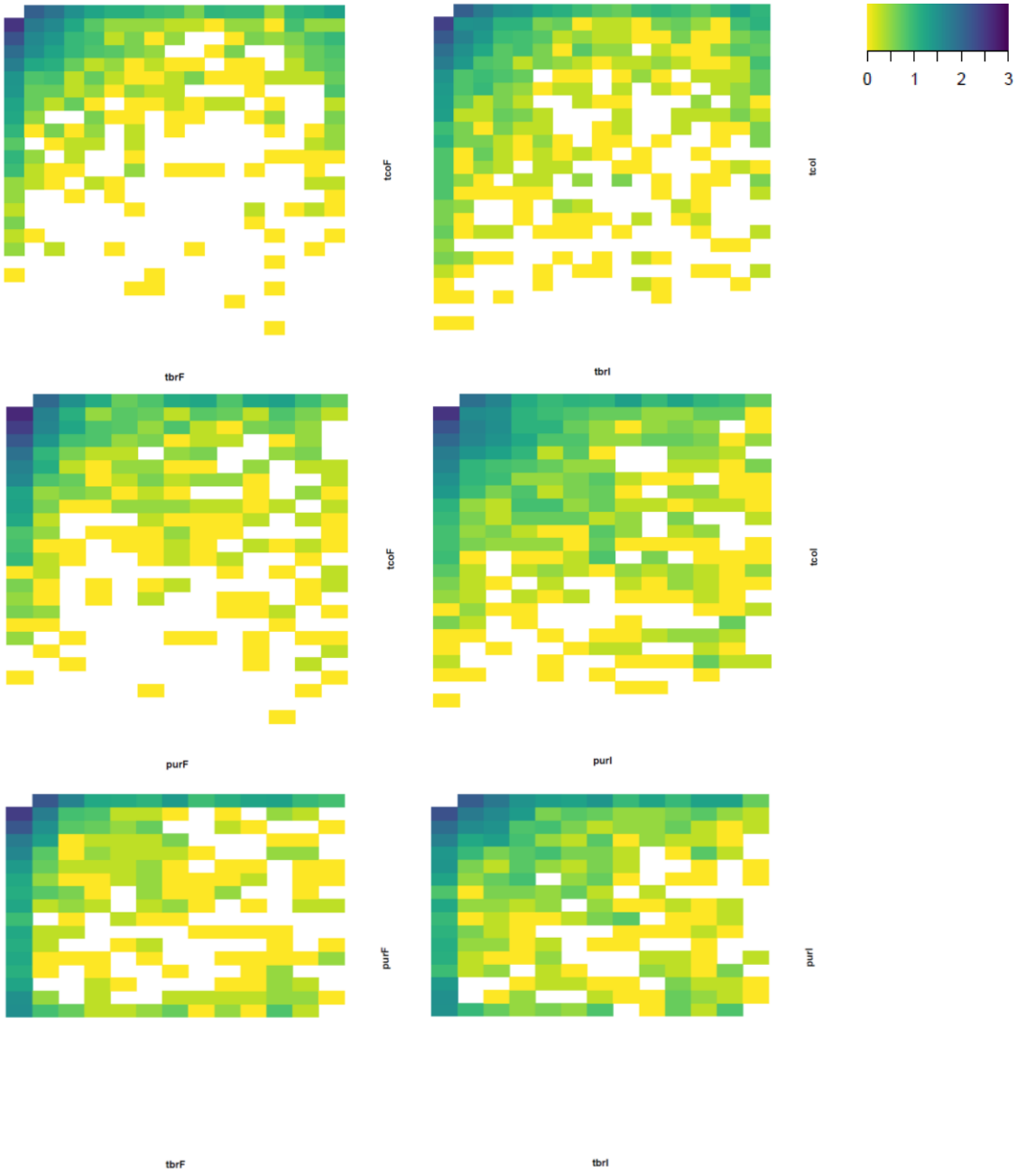

Figure S 8 (continued).

Bioinformatic pipelines and R scripts

Data processing, genotype calling, separating subgenomes & analyses

```
########## pipeline_angsd.txt ##########
########## excluding introgressed diploids with angsd v. 0.928 ##########

#create list of mapped bam files (normally when in parent directory where the directory
with bam files is placed called for example "bams"):
ls ./readGroups/*.bam > listExclude-5
#use a text editor to clean the list to the part of the name needed by using serach and
replace. Save as indlist

#infer genotype likelihoods (change -minInd to half individuals, and -minMaf for frequency
of 2 individuals):
#with GATK model (angsd v. 0.928 (htslib: 1.9), "http://www.popgen.dk/angsd")
angsd -bam listExclude-5 -GL 2 -doMajorMinor 1 -doMaf 1 -SNP_pval 1e-6 -minMapQ 20 -minQ 20
-minInd 52 -minMaf 0.019 -doGlf 2 -out gatk50excluded-5

#before excluding <-5
#total number of sites analyzed: 914463
#number of sites retained after filtering: 8069
#after excluding <-5
#total number of sites analyzed: 886663
#number of sites retained after filtering: 6334

#calculate covariance matrix. Use the same -minMaf as above
python /usr/local/pcangsd/pcangsd.py -beagle gatk50excluded-5.beagle.gz -o gatk50excluded-5
-minMaf 0.019


#plot covariance matrix as heatmap. Go to R or Rstudio (R v. 3.4.4, 2018)
setwd("/mnt/data2/Anna/")
gatk50excluded <- as.matrix(read.table("gatk50excluded-5NA.cov"))
indexcluded <- read.table("indlistExclude-5")
ind.labexcluded<-as.vector(ind[,1])

install.packages("gplots")
library(gplots)

heatmap <- heatmap.2(gatk50excluded, trace="none", Rowv=NA, Colv=NA, cexRow=0.6 ,cexCol =
0.6, labRow = ind.labexcluded, labCol = ind.labexcluded, col=
colorRampPalette(c("lemonchiffon", "lemonchiffon", "lemonchiffon", "lemonchiffon",
"lemonchiffon", "yellow", "red", "magenta", "midnightblue", "black", "black",
"black"))(100))


gatk50 <- as.matrix(read.table("gatk50NA.cov"))
ind <- read.table("indlist")
ind.lab<-as.vector(ind[,1])

heatmap <- heatmap.2(gatk50, trace="none", Rowv=NA, Colv=NA, cexRow=0.6, cexCol = 0.6,
labRow = ind.lab, labCol = ind.lab, col= colorRampPalette(c("lemonchiffon", "lemonchiffon",
"lemonchiffon", "lemonchiffon", "lemonchiffon", "yellow", "red", "magenta", "midnightblue",
"black", "black", "black"))(100))
#####################


########## pipeline_ReadGroups.txt ##########
########## adding ReadGroups with AddOrReplaceReadGroups from picard v. 2.1.0 ##########

cd sorted
for file in *.bam; do echo $file; nice -n 19 java -Xmx60G -jar /usr/local/picard-tools-
2.1.0/picard.jar AddOrReplaceReadGroups I=$file O=../readGroups/${file/bam/gr.bam}
RGID=${file/_uniqsort.bam/} RGLB=${file/_uniqsort.bam/} RGPL=illumina
RGPU=${file/_uniqsort.bam/} RGSM=${file/_uniqsort.bam/}; done
```

```
cd mnt/data2/Anna/
mkdir readGroupsPolyplTest
cd mnt/data2/Anna/sortedPolyplTest
for file in *.bam; do echo $file; nice -n 19 java -Xmx60G -jar /usr/local/picard-tools-
2.1.0/picard.jar AddOrReplaceReadGroups I=$file
O=../readGroupsPolyplTest/${file/bam/gr.bam} RGID=${file/_uniqsort.bam/}
RGLB=${file/_uniqsort.bam/} RGPL=illumina RGPU=${file/_uniqsort.bam/}
RGSM=${file/_uniqsort.bam/}; done

cd mnt/data2/Anna
mkdir 2UsePolypl
cd ./sorted2UsePolypl
for file in *.bam; do echo $file; nice -n 19 java -Xmx60G -jar /usr/local/picard-tools-
2.1.0/picard.jar AddOrReplaceReadGroups I=$file O=../2UsePolypl/${file/bam/gr.bam}
RGID=${file/_uniqsort.bam/} RGLB=${file/_uniqsort.bam/} RGPL=illumina
RGPU=${file/_uniqsort.bam/} RGSM=${file/_uniqsort.bam/}; done

#view if sorted with samtools v. 1.3
cd /mnt/data2/Anna/readGroupsPolyplTest
samtools view -H /mnt/data2/Anna/readGroupsPolyplTest/Dmaj_au1_1031_uniqsort.gr.bam | head

cd /mnt/data2/Anna/2Use
samtools view -H
/mnt/data1/botanik/Dactylorhiza/RADseq_PART_II/0_STACKS_demultiplexed/samples_used_for_popu
lation_genetic_studies/PolyploidPopulations_forward/pyRAD_ref/Dang_fr13_8523_uniqsort.bam |
head

#view read groups
cd /mnt/data2/Anna/readGroupsPolyplTest
samtools view -H /mnt/data2/Anna/readGroupsPolyplTest/Dmaj_au1_1031_uniqsort.gr.bam | tail
####################


########## pipeline_mapping.txt ##########
########## mapping to reference genome (Dactylorhiza incarnata) ##########

#index reference bwa mem v. 0.7.12; samtools v. 1.3

#While being in the reference folder do the following commands. Replace reference.fasta
with your reference. May need to run locate picard.jar to find the path to picard for the
3rd command.

bwa index -a bwtsw ./reference.fasta

samtools faidx ./reference.fasta

java -Xmx60G -jar /usr/local/picard.jar CreateSequenceDictionary R=./reference.fasta
o=./reference.dict


#mapping itself. Replace .fq with .fastq if your files carry such an extension. Adjust -t
according to how many cores you have available.

#bwa mem v. 0.7.12

cd /mnt/data2/Anna/

mkdir mapped

mkdir ./mapped/fuc ./mapped/inc ./mapped/polypl

mkdir ./mapped/fuc/single ./mapped/inc/single ./mapped/polypl/single


#fuc

cd ./samples/fuc/
```

```
#for mapping paired end reads:

for file in *.1.fq; do echo $file; nice -n 19 bwa mem -M -t 16
../../referenceCurated/curated.fasta $file ${file/.1.fq/.2.fq} >
../../mapped/fuc/${file/.1.fq/.sam}; done


#for mapping single end reads:

cd ./singleFiles/

for file in *.fq; do echo $file; nice -n 19 bwa mem -M -t 16
../../../referenceCurated/curated.fasta $file >
../../../mapped/fuc/single/${file/.fq/.sam}; done


#inc

#for mapping paired end reads:

cd ../../inc/

for file in *.1.fq; do echo $file; nice -n 19 bwa mem -M -t 16
../../referenceCurated/curated.fasta $file ${file/.1.fq/.2.fq} >
../../mapped/inc/${file/.1.fq/.sam}; done


#for mapping single end reads:

cd ./singleFiles/

for file in *.fq; do echo $file; nice -n 19 bwa mem -M -t 16
../../../referenceCurated/curated.fasta $file >
../../../mapped/inc/single/${file/.fq/.sam}; done


#polypl

#for mapping paired end reads:

cd ../../polypl/

for file in *.1.fq; do echo $file; nice -n 19 bwa mem -M -t 16
../../referenceCurated/curated.fasta $file ${file/.1.fq/.2.fq} >
../../mapped/polypl/${file/.1.fq/.sam}; done


#for mapping single end reads:

cd ./singleFiles/

for file in *.fq; do echo $file; nice -n 19 bwa mem -M -t 16
../../../referenceCurated/curated.fasta $file >
../../../mapped/polypl/single/${file/.fq/.sam}; done


#post-mapping processing

#sorting the aligned SAM by coordinate while converting to BAM

cd mnt/data2/Anna/

mkdir mappedSort

mkdir ./mappedSort/fuc ./mappedSort/inc ./mappedSort/polypl

mkdir ./mappedSort/fuc/single ./mappedSort/inc/single ./mappedSort/polypl/single
```

```
cd ./mapped/fuc/

for file in *.sam; do echo $file; nice -n 19 java -Xmx40G -jar
/home/botanik/Downloads/picard-tools-2.4.1/picard.jar SortSam I=$file
O=../../mappedSort/fuc/${file/.sam/.bam} SO=coordinate; done

cd ./single/

for file in *.sam; do echo $file; java -Xmx40G -jar /home/botanik/Downloads/picard-tools-
2.4.1/picard.jar SortSam I=$file O=../../../mappedSort/fuc/single/${file/.sam/.bam}
SO=coordinate; done

cd ../../inc/

for file in *.sam; do echo $file; java -Xmx60G -jar /home/botanik/Downloads/picard-tools-
2.4.1/picard.jar SortSam I=$file O=../../mappedSort/inc/${file/.sam/.bam} SO=coordinate;
done

cd ./single/

for file in *.sam; do echo $file; java -Xmx60G -jar /home/botanik/Downloads/picard-tools-
2.4.1/picard.jar SortSam I=$file O=../../../mappedSort/inc/single/${file/.sam/.bam}
SO=coordinate; done

cd ../../polypl/

for file in *.sam; do echo $file; java -Xmx60G -jar /home/botanik/Downloads/picard-tools-
2.4.1/picard.jar SortSam I=$file O=../../mappedSort/polypl/${file/.sam/.bam} SO=coordinate;
done

cd ./single/

for file in *.sam; do echo $file; java -Xmx60G -jar /home/botanik/Downloads/picard-tools-
2.4.1/picard.jar SortSam I=$file O=../../../mappedSort/polypl/single/${file/.sam/.bam}
SO=coordinate; done


#checking mapping percentage

samtools flagstat file


#add read groups with picard v. 2.20.6

cd mnt/data2/Anna/

mkdir mappedSortGroup

mkdir ./mappedSortGroup/fuc ./mappedSortGroup/inc ./mappedSortGroup/polypl

mkdir ./mappedSortGroup/fuc/single ./mappedSortGroup/inc/single
./mappedSortGroup/polypl/single

cd ./mappedSort/fuc/

for file in *.bam; do echo $file; nice -n 19 java -Xmx60G -jar /usr/local/picard.jar
AddOrReplaceReadGroups I=$file O=../../mappedSortGroup/fuc/${file/.bam/.gr.bam}
RGID=${file/.bam/} RGLB=${file/.bam/} RGPL=illumina RGPU=${file/.bam/} RGSM=${file/.bam/};
done

cd ./mappedSort/fuc/single/

for file in *.bam; do echo $file; nice -n 19 java -Xmx60G -jar /usr/local/picard.jar
AddOrReplaceReadGroups I=$file O=../../../mappedSortGroup/fuc/single/${file/.bam/.gr.bam}
RGID=${file/.bam/} RGLB=${file/.bam/} RGPL=illumina RGPU=${file/.bam/} RGSM=${file/.bam/};
done
```

```
cd ./mappedSort/inc/

for file in *.bam; do echo $file; nice -n 19 java -Xmx60G -jar /usr/local/picard.jar
AddOrReplaceReadGroups I=$file O=../../mappedSortGroup/inc/${file/.bam/.gr.bam}
RGID=${file/.bam/} RGLB=${file/.bam/} RGPL=illumina RGPU=${file/.bam/} RGSM=${file/.bam/};
done

cd ./mappedSort/inc/single/

for file in *.bam; do echo $file; nice -n 19 java -Xmx60G -jar /usr/local/picard.jar
AddOrReplaceReadGroups I=$file O=../../../mappedSortGroup/inc/single/${file/.bam/.gr.bam}
RGID=${file/.bam/} RGLB=${file/.bam/} RGPL=illumina RGPU=${file/.bam/} RGSM=${file/.bam/};
done

cd ./mappedSort/polypl/

for file in *.bam; do echo $file; nice -n 19 java -Xmx60G -jar /usr/local/picard.jar
AddOrReplaceReadGroups I=$file O=../../mappedSortGroup/polypl/${file/.bam/.gr.bam}
RGID=${file/.bam/} RGLB=${file/.bam/} RGPL=illumina RGPU=${file/.bam/} RGSM=${file/.bam/};
done

cd ./mappedSort/polypl/single/

for file in *.bam; do echo $file; nice -n 19 java -Xmx60G -jar /usr/local/picard.jar
AddOrReplaceReadGroups I=$file
O=../../../mappedSortGroup/polypl/single/${file/.bam/.gr.bam} RGID=${file/.bam/}
RGLB=${file/.bam/} RGPL=illumina RGPU=${file/.bam/} RGSM=${file/.bam/}; done


#indel realignment. This may take a long time. Make sure you are using GATK version 3.8,
but not versions later than that.

#samtools v. 1.3

#GATK v. 3.8

cd mnt/data1/botanik/Anna/

mkdir realigned

mkdir ./realigned/fuc ./realigned/inc ./realigned/polypl

mkdir ./realigned/fuc/single ./realigned/inc/single ./realigned/polypl/single


cd ./mappedSortGroup/fuc/

for file in *.bam; do echo $file; samtools index $file; done

for file in *.bam; do echo $file; nice -n 19 java -Xmx60G -jar
/usr/local/GenomeAnalysisTK.jar -T RealignerTargetCreator -R
../../referenceCurated/curated.fasta -I $file -o ${file/.bam/.intervals}; done

for file in *.bam; do echo $file; nice -n 19 java -Xmx60G  -jar
/usr/local/GenomeAnalysisTK.jar -T IndelRealigner -R ../../referenceCurated/curated.fasta -
I $file -targetIntervals ${file/.bam/.intervals} -maxReads 100000 -o
../../realigned/fuc/$file; done

cd ./single

for file in *.bam; do echo $file; samtools index $file; done

for file in *.bam; do echo $file; nice -n 19 java -Xmx60G -jar
/usr/local/GenomeAnalysisTK.jar -T RealignerTargetCreator -R
../../../referenceCurated/curated.fasta -I $file -o ${file/.bam/.intervals}; done
```

```
for file in *.bam; do echo $file; nice -n 19 java -Xmx60G  -jar
/usr/local/GenomeAnalysisTK.jar -T IndelRealigner -R
../../../referenceCurated/curated.fasta -I $file -targetIntervals ${file/.bam/.intervals} -
maxReads 100000 -o ../../../realigned/fuc/single/$file; done


cd ./mappedSortGroup/inc/

for file in *.bam; do echo $file; samtools index $file; done

for file in *.bam; do echo $file; nice -n 19 java -Xmx60G -jar
/usr/local/GenomeAnalysisTK.jar -T RealignerTargetCreator -R
../../referenceCurated/curated.fasta -I $file -o ${file/.bam/.intervals}; done

for file in *.bam; do echo $file; nice -n 19 java -Xmx60G  -jar
/usr/local/GenomeAnalysisTK.jar -T IndelRealigner -R ../../referenceCurated/curated.fasta -
I $file -targetIntervals ${file/.bam/.intervals} -maxReads 100000 -o
../../realigned/inc/$file; done

cd ./single

for file in *.bam; do echo $file; samtools index $file; done

for file in *.bam; do echo $file; nice -n 19 java -Xmx60G -jar
/usr/local/GenomeAnalysisTK.jar -T RealignerTargetCreator -R
../../../referenceCurated/curated.fasta -I $file -o ${file/.bam/.intervals}; done

for file in *.bam; do echo $file; nice -n 19 java -Xmx60G  -jar
/usr/local/GenomeAnalysisTK.jar -T IndelRealigner -R
../../../referenceCurated/curated.fasta -I $file -targetIntervals ${file/.bam/.intervals} -
maxReads 100000 -o ../../../realigned/inc/single/$file; done


cd ./mappedSortGroup/polypl/

for file in *.bam; do echo $file; samtools index $file; done

for file in *.bam; do echo $file; nice -n 19 java -Xmx60G -jar
/usr/local/GenomeAnalysisTK.jar -T RealignerTargetCreator -R
../../referenceCurated/curated.fasta -I $file -o ${file/.bam/.intervals}; done

for file in *.bam; do echo $file; nice -n 19 java -Xmx60G  -jar
/usr/local/GenomeAnalysisTK.jar -T IndelRealigner -R ../../referenceCurated/curated.fasta -
I $file -targetIntervals ${file/.bam/.intervals} -maxReads 100000 -o
../../realigned/polypl/$file; done

cd ./single

for file in *.bam; do echo $file; samtools index $file; done

for file in *.bam; do echo $file; nice -n 19 java -Xmx60G -jar
/usr/local/GenomeAnalysisTK.jar -T RealignerTargetCreator -R
../../../referenceCurated/curated.fasta -I $file -o ${file/.bam/.intervals}; done

for file in *.bam; do echo $file; nice -n 19 java -Xmx60G  -jar
/usr/local/GenomeAnalysisTK.jar -T IndelRealigner -R
../../../referenceCurated/curated.fasta -I $file -targetIntervals ${file/.bam/.intervals} -
maxReads 100000 -o ../../../realigned/polypl/single/$file; done
####################
```

```
########## pipeline_ebg.txt ##########
########## prepare files for ebg, then run ebg, polyrelatedness and STRUCTURE ##########

#use the GATK UnifiedGenotyper to make a VCF file for all BAM files
cd mnt/data2/Anna/
mkdir ebgTest

#creating .fai (fasta index file) for reference fasta file with samtools v. 1.3
cd mnt/data2/Anna/
samtools faidx pyRAD.fasta

#creating .dict (fasta sequence dictionary file) for reference fasta file with
CreateSequenceDictionary from picard v. 2.1.0
java -jar /usr/local/picard-tools-2.1.0/picard.jar CreateSequenceDictionary R= pyRAD.fasta
O= pyRAD.dict

#creating .bai files (indexed bam files) with samtools v. 1.3
##diploids
cd /mnt/data2/Anna/readGroups2useTest/
for file in *.bam; do echo $file; samtools index $file ${file/.bam/.bam.bai}; done
##polyploids
cd /mnt/data2/Anna/readGroupsPolyplTest/
for file in *.bam; do echo $file; samtools index $file ${file/.bam/.bam.bai}; done

#creating .vcf files for 2x and 4x
##diploids & polyploids together as diploids with UnifiedGenotyper from GATK 3.8-1-0-
gf15c1c3ef
cd /mnt/data1/botanik/Anna/dataEBG/
java -jar /usr/local/GenomeAnalysisTK.jar -T UnifiedGenotyper -R
/mnt/data1/botanik/Anna/referenceCurated/curated.fasta $(for f in
/mnt/data1/botanik/Anna/dataEBG/*bam; do printf "%sI $f " '-'; done) -o
/mnt/data1/botanik/Anna/ebg/2x.vcf -ploidy 2 -nt 5 -nct 2

##diploids & polyploids together as polyploids with UnifiedGenotyper from GATK 3.8-1-0-
gf15c1c3ef
cd /mnt/data1/botanik/Anna/dataEBG/
java -jar /usr/local/GenomeAnalysisTK.jar -T UnifiedGenotyper -R
/mnt/data1/botanik/Anna/referenceCurated/curated.fasta $(for f in
/mnt/data1/botanik/Anna/dataEBG/*bam; do printf "%sI $f " '-'; done) -o
/mnt/data1/botanik/Anna/ebg/4x.vcf -ploidy 4 -nt 5 -nct 2

mkdir /mnt/data1/botanik/Anna/ebg/200429_finalData

##excluding introgressed individuals from 2x.vcf and 4x.vcf according to
DactRAD_sampleinfo_MKB_newSamplesAnna_OP.xls

##extract data from .vcf files
botanik@A772-Marie:/mnt/data1/botanik/Anna/ebg/200429_finalData$ wc -l 2x.vcf
#16969393 2x.vcf
botanik@A772-Marie:/mnt/data1/botanik/Anna/ebg/200429_finalData$ wc -l 4x.vcf
#17423275 4x.vcf
head -913 4x.vcf > 4xhead
head -913 4xhead | tail -1 > 4xindividuals
head -911 2x.vcf > 2xhead

head -17423275 4x.vcf | tail -17422362 > 4xdata.vcf
wc -l 4xdata.vcf
#17422362 4xdata.vcf
cat 4xindividuals 4xdata.vcf > 4xdataInd.vcf

head -16969393 2x.vcf | tail -16968482 > 2xdata.vcf
wc -l 2xdata.vcf
#16968482 2xdata.vcf
cat 2xindividuals 2xdata.vcf > 2xdataInd.vcf
```

```
botanik@A772-Marie:/mnt/data1/botanik/Anna/ebg/200429_finalData$ head -n 3 4xdataInd.vcf |
awk '{print NF}'
383
383
383
botanik@A772-Marie:/mnt/data1/botanik/Anna/ebg/200429_finalData$ head -n 3 2xdataInd.vcf |
awk '{print NF}'
383
383
383

sed 's/\t/\n/g' 4xindividuals > 4xindividualsList
sed 's/\t/\n/g' 2xindividuals > 2xindividualsList

#extracting col 1 and replacing the | in scaffold labels
cut -f 1 4xdataInd.vcf | sed 's/|/_/g' > 4xCHROM
cut -f 1 2xdataInd.vcf | sed 's/|/_/g' > 2xCHROM

#extract 345 ind and order 4x by locality
cd /mnt/data1/botanik/Anna/190903_pipelines
sh ./2xextract345ind.sh
sh ./4xextract345ind.sh

#checking number of lines
wc -l 4x345ind.vcf
#17422363 4x345ind.vcf
wc -l 2x345ind.vcf
#16968483 2x345ind.vcf

#checking sequence of individuals
head -1 4x345ind.vcf > 4x345indHead
head -1 2x345ind.vcf > 2x345indHead

#extract data without labels
head -17422363 4x345ind.vcf | tail -17422362 > 4x345indData.vcf
head -16968483 2x345ind.vcf | tail -16968482 > 2x345indData.vcf

#checking number of lines
wc -l 4x345indData.vcf
#17422362 4x345ind.vcf
wc -l 2x345indData.vcf
#16968482 2x345ind.vcf

#prepare file with new labels (newLabels)

#splitting the .vcf file
mkdir ./splitting
split -l 1800000 4x345indData.vcf ./splitting/4x345indData.
split -l 3600000 2x345indData.vcf ./splitting/2x345indData.

#remove last line from header (individual labels)
head -912 4xhead > 4xheadNoLabels
head -910 2xhead > 2xheadNoLabels

#adding the header + new labels file to all splitted files, e.g.
cat 4xheadNoLabels newLabels ./splitting/4x345indData.aa >
./splitting/4x345indData.aa.head.vcf
cat 2xheadNoLabels newLabels ./splitting/2x345indData.aa >
./splitting/2x345indData.aa.head.vcf

#open rstudio
sudo rstudio
#open usr/local/polyploid-genotyping/helper-scripts/filter-vcf.R to filter the .vcf files
#####minQ def.="100"
#####minDP def.="5"
#####missing def.="173" <-half the number of individuals
```

```
##concatenating splitted & filtered .vcf files again
cat 2x345indData.aa.filtered.vcf 2x345indData.ab.filtered.vcf 2x345indData.ac.filtered.vcf
2x345indData.ad.filtered.vcf 2x345indData.ae.filtered.vcf > 2x345indData.filtered.vcf
cat 4x345indData.aa.filtered.vcf 4x345indData.ab.filtered.vcf 4x345indData.ac.filtered.vcf
4x345indData.ad.filtered.vcf 4x345indData.ae.filtered.vcf 4x345indData.af.filtered.vcf
4x345indData.ag.filtered.vcf 4x345indData.ah.filtered.vcf 4x345indData.ai.filtered.vcf
4x345indData.aj.filtered.vcf > 4x345indData.filtered.vcf
##adding the header to filtered .vcf files (not necessary at this point)
#cat 4xheadNoLabels newLabels 4x345indData.filtered.vcf > 4x345indData.filtered.headed.vcf
#cat 2xheadNoLabels newLabels 2x345indData.filtered.vcf > 2x345indData.filtered.headed.vcf

#open usr/local/polyploid-genotyping/helper-scripts/intersect-vcf.R
#run with 2x345indData.filtered.headed.vcf as vcf1 & 4x345indData.filtered.headed.vcf as
vcf2
#make output files accessible
sudo chown botanik:botanik shared*

#remove individuals of alternative ploidy from each file
cut -f 1-9 shared345ind-2x.vcf > shared345ind-2xInfocols
cut -f 1-9 shared345ind-4x.vcf > shared345ind-4xInfocols
cut -f 10-113 shared345ind-2x.vcf > shared345ind-2xOnly
cut -f 114-354 shared345ind-4x.vcf > shared345ind-4xOnly

#adding infocols to data again
paste -d '\t' shared345ind-2xInfocols shared345ind-2xOnly > shared345ind-2xOnly+Infocols
paste -d '\t' shared345ind-4xInfocols shared345ind-4xOnly > shared345ind-4xOnly+Infocols

#splitting newLabels by ploidy
cut -f 1-9 newLabels > newLabelsInfocols
cut -f 10-113 newLabels > newLabels2xOnly
cut -f 114-354 newLabels > newLabels4xOnly

#adding infocols to indlabels again
paste -d '\t' newLabelsInfocols newLabels2xOnly > newLabels2x
paste -d '\t' newLabelsInfocols newLabels4xOnly > newLabels4x

#adding the header
cat 2xheadNoLabels newLabels2x shared345ind-2xOnly+Infocols > shared345ind-2x.final.vcf
cat 4xheadNoLabels newLabels4x shared345ind-4xOnly+Infocols > shared345ind-4x.final.vcf

#read counts from vcf
/usr/local/polyploid-genotyping/helper-scripts/read-counts-from-vcf.py --vcf shared345ind-
2x.final.vcf -n 104 -s 45318 --prefix shared345ind-2x_readCounts
/usr/local/polyploid-genotyping/helper-scripts/read-counts-from-vcf.py --vcf shared345ind-
4x.final.vcf -n 241 -s 45318 --prefix shared345ind-4x_readCounts

#gt from vcf
/usr/local/polyploid-genotyping/helper-scripts/gt-from-vcf.py --vcf shared345ind-
2x.final.vcf --prefix shared345ind-2x_readCounts
/usr/local/polyploid-genotyping/helper-scripts/gt-from-vcf.py --vcf shared345ind-
4x.final.vcf --prefix shared345ind-4x_readCounts

#replace _ by | in shared-variants list again as mpileup doesn't work with _!!!
sed 's/_/|/g' ../../ebg/200429_finalData/shared345ind-variants.txt >
../../ebg/200429_finalData/shared345ind-variantsPipe.txt

#mpileup
cd /mnt/data1/botanik/Anna/dataEBG/200508_finalBams #(folder where .bam files are)
samtools mpileup -x -I -f /mnt/data1/botanik/Anna/referenceCurated/curated.fasta -l
../../ebg/200429_finalData/shared345ind-variantsPipe.txt $(ls *.bam) -o
../../ebg/200429_finalData/shared345ind.pileup

#error file
/usr/local/polyploid-genotyping/helper-scripts/per-locus-err.py -i
/mnt/data1/botanik/Anna/ebg/200429_finalData/shared345ind.pileup -n 345 >
/mnt/data1/botanik/Anna/ebg/200429_finalData/errshared345ind
```

```
#run ebg (ebg v. 0.3.2-alpha (September 2018))
cd /mnt/data1/botanik/Anna/ebg/200429_finalData
ebg gatk -t shared345ind-4x_readCounts-tot.txt -l 45318 -n 241 -a shared345ind-
4x_readCounts-alt.txt -e errshared345ind -p 4 --iters 10000 --prefix shared345ind-4x
ebg gatk -t shared345ind-2x_readCounts-tot.txt -l 45318 -n 104 -a shared345ind-
2x_readCounts-alt.txt -e errshared345ind -p 2 --iters 10000 --prefix shared345ind-2x

##filter for maf 0.02
#transpose genos file
awk '
{
    for (i=1; i<=NF; i++)  {
        a[NR,i] = $i
    }
}
NF>p { p = NF }
END {
    for(j=1; j<=p; j++) {
        str=a[1,j]
        for(i=2; i<=NR; i++){
            str=str" "a[i,j];
        }
        print str
    }
}' shared345ind-4x-genos.txt > shared345ind-4x-genosT.txt
#replace spaces by tabs
sed 's/\s/\t/g' shared345ind-4x-genosT.txt > shared345ind-4x-genosTtab.txt
#replace -9 with NA to calculate column sums
sed 's/-9/NA/g' shared345ind-4x-genosTtab.txt > shared345ind-4x-genosTtabNA.txt
#go to R (script filter.maf.R)
#extract column sums and count -9 (NA's) per column
#get 2 files
#concatenate these 2 files horizontally
paste -d '\t' rowSums345ind.txt NAperLocus345ind.txt > calcMaf345ind.ods
#calcMaf345ind1.ods
#copy shared345ind-variants.txt in libreoffice (sequence of scaffolds)
#extract lines according to calcMaf1.ods with extract18879loci.sh
cd /mnt/data1/botanik/Anna/190903_pipelines/
sh ./extract18879loci.sh
#resulting file: shared345ind-4x-genosTtabMaf0.02.txt

#adapting genos files for polyrelatedness
mkdir ./polyrel
sed 's/-9/9999/g' shared345ind-4x-genosTtabMaf0.02.txt | sed 's/0/5555/g' | sed
's/1/5556/g' | sed 's/2/5566/g' | sed 's/3/5666/g' | sed 's/4/6666/g' >
./polyrel/shared345ind-4x-genosTtabMaf0.02.4dig.txt
#replacing 9999 by 0 (missingallele)
sed 's/9999/0/g' ./polyrel/shared345ind-4x-genosTtabMaf0.02.4dig.txt >
./polyrel/shared345ind-4x-genosTtabMaf0.02.4digMiss.txt
#transpose again
awk '
{
    for (i=1; i<=NF; i++)  {
        a[NR,i] = $i
    }
}
NF>p { p = NF }
END {
    for(j=1; j<=p; j++) {
        str=a[1,j]
        for(i=2; i<=NR; i++){
            str=str" "a[i,j];
        }
        print str
    }
}' shared345ind-4x-genosTtabMaf0.02.4digMiss.txt > shared345ind-4x-
genosTtabMaf0.02.4digMissT.txt
```

```
#replace spaces by tabs again
sed 's/\s/\t/g' shared345ind-4x-genosTtabMaf0.02.4digMissT.txt > shared345ind-4x-
genosTtabMaf0.02.4digMissTtab.txt
#adding id and pop (m/p/t/mxt)
sed 's/\t/\n/g' ../newLabels4xOnly > indlabTemp
sed 's/_/\t/g' indlabTemp > indlabTemp1
cut -f 1 indlabTemp1 > poplabTemp
paste -d '\t' indlabTemp poplabTemp > indlabels
paste -d '\t' indlabels shared345ind-4x-genosTtabMaf0.02.4digMissTtab.txt > temp1
#make locilist with gedit/libreoffice
#generating final polyrelatedness input file

cat polyrelHeader locilistTtab temp1 endfile > 200609_shared241ind18879loci-
genosPolyrel.txt

#replacing spaces by tabs
sed 's/\s/\t/g' temp1 > temp1tab
cat polyrelHeader18879loci temp1tab endfile > 200527_shared241ind18879loci-genosPolyrel.txt

cd /usr/local/polyrelatednes
sudo cp /mnt/data1/botanik/Anna/ebg/200429_finalData/polyrel/200527_shared241ind18879loci-
genosPolyrel.txt ./200527_shared241ind18879loci-genosPolyrel.txt

#paste -d '\t' indlabelsShort shared345ind-4x-genosTtabMaf0.02.4digMissTtab.txt >
temp1short
#cat polyrelHeader18879loci temp1short endfile > 200512_shared241ind18879loci-
genosPolyrel.txt

#copy input file in polyrelatedness folder
cd /usr/local/polyrelatednes
sudo cp /mnt/data1/botanik/Anna/ebg/200429_finalData/polyrel/200512_shared241ind18879loci-
genosPolyrel.txt ./200512_shared241ind18879loci-genosPolyrel.txt

#run polyrelatedness v. 1.8
sudo ./PolyRelatedness.out
#11
#input: 200609_shared241ind18879loci-genosPolyrel.txt
#output: 200609_241ind18879loci-MOMout.txt
#1 (MOM estimator)

#copy to working directory
sudo cp ./200609_241ind18879lociMOMout.txt
/mnt/data1/botanik/Anna/ebg/200429_finalData/polyrel/



#STRUCTURE input file preparation
#filter by maf 0.02 and 1 ev. 10 kB according to calcMaf345ind1.ods
cd /mnt/data1/botanik/Anna/190903_pipelines/
sh ./extract2466loci.sh
#extract2466loci.sh is a bash script just with sed -n -e '21p;...(all lines that contain
loci which should be retained, here 2466 lines/loci)' infile > outfile

#extract genotypes fields from vcf file without header
grep -o -e './././.' -e './.' shared241ind2466loci > shared241ind2466lociGenotypes
#check line number = number of loci * number of individuals
#replace './.' by '-9/-9/-9/-9'
sed 's/\.\/\./-9\/-9\/-9\/-9/g' shared241ind2466lociGenotypes >
shared241ind2466lociGenotypesMiss
grep '\.\/\.' shared241ind2466lociGenotypes | wc -l
#70,707
grep -e '-9\/-9\/-9\/-9' shared241ind2466lociGenotypesMiss | wc -l
#70,707
#split one individual over four lines
sed 's/\//\n/g' shared241ind2466lociGenotypesMiss >
shared241ind2466lociGenotypesMissAllelePerLine
```

```
wc -l shared241ind2466lociGenotypesMiss
#594,306 shared241ind2466lociGenotypesMiss
wc -l shared241ind2466lociGenotypesMissAllelePerLine
#2,377,224 shared241ind2466lociGenotypesMissAllelePerLine
#split into one file per locus 241*4=964
mkdir ./temp
split -l 964 shared241ind2466lociGenotypesMissAllelePerLine ./temp/allGenotypesSingleLocus_
#run script to horizontally concatenate allGenotypesSingleLocus files
cd /mnt/data1/botanik/Anna/190903_pipelines/
sh 200519_allGenotypesInCol.sh
mv /mnt/data1/botanik/Anna/ebg/200429_finalData/STRUCT/temp1/allGenotypesAllLoci
/mnt/data1/botanik/Anna/ebg/200429_finalData/STRUCT/allGenotypesAllLoci
#replace 1 by 2 and 0 by 1
sed 's/1/2/g' allGenotypesAllLoci | sed 's/0/1/g' > allGenotypesAllLoci12
#prepare individual & loc cols
#transpose
awk '
{
    for (i=1; i<=NF; i++)  {
        a[NR,i] = $i
    }
}
NF>p { p = NF }
END {
    for(j=1; j<=p; j++) {
        str=a[1,j]
        for(i=2; i<=NR; i++){
            str=str" "a[i,j];
        }
        print str
    }
}' ../newLabels4xOnly > newLabels4xOnlyT
#get loc col
sed 's/_/\t/g' newLabels4xOnlyT | cut -f 3 > newLabels4xOnlyTLoc
#multiply lines for each individual to 4
cd /mnt/data1/botanik/Anna/190903_pipelines/
sh ./200520_4xindividualsMultiplyLines.sh >
/mnt/data1/botanik/Anna/ebg/200429_finalData/STRUCT/newLabels4xOnlyT4lines
#change input file in 4xindividualsMultiplyLines.sh
sh ./200520_4xindividualsMultiplyLines.sh >
/mnt/data1/botanik/Anna/ebg/200429_finalData/STRUCT/newLabels4xOnlyTLoc4lines
#200520_4xindividualsMultiplyLines.sh is a bash script:
#file=$(cat /mnt/data1/botanik/Anna/ebg/200429_finalData/STRUCT/newLabels4xOnlyTLoc)
#
#for line in $file
#      do
#              echo "$line"
#              printf "%s\n" "$line" "$line" "$line"
#      done

#generate input file with ind and data only
sed 's/_//g' newLabels4xOnlyT4lines > indlabels
paste -d '\t' indlabels allGenotypesAllLoci12 > 200520_allGenotypesAllLociInd_final.txt

#generate a mainparams file according to manual -> mainparams
#extraparams file is empty, but file have to be provided to the program

##test run
sudo /usr/local/Structure/bin/structure -m
/mnt/data1/botanik/Anna/ebg/200429_finalData/STRUCT/200520_testInput/paramsfiles/mainparams
-e
/mnt/data1/botanik/Anna/ebg/200429_finalData/STRUCT/200520_testInput/paramsfiles/extraparam
s -K 2 -L 2466 -N 241 -i
/mnt/data1/botanik/Anna/ebg/200429_finalData/STRUCT/200520_testInput/200520_allGenotypesAll
LociInd_final.txt -o
/mnt/data1/botanik/Anna/ebg/200429_finalData/STRUCT/200520_testOutput/4xSTRUCTURE_2466loci_
2_1
```

```
#works!

#prepare STRUCTURE slrms to run on LiSC cube

#copy files to cube (directory on computer)
scp /mnt/data1/botanik/Anna/ebg/200429_finalData/STRUCT/200520_cubeInput/slrms/*.slrm
paun@vlogin3.csb.univie.ac.at:/scratch/ovidiu/structure/200520/slrms/
scp /mnt/data1/botanik/Anna/ebg/200429_finalData/STRUCT/200520_cubeInput/*
paun@vlogin3.csb.univie.ac.at:/scratch/ovidiu/structure/200520/

#enter cube
ssh paun@vlogin3.csb.univie.ac.at
cd /scratch/ovidiu/structure
mkdir 200520
mkdir ./200520/slrms
#copy files from computer to cube
botanik@A772-Marie:/mnt/data1/botanik/Anna/ebg/200429_finalData/STRUCT/200520_cubeInput$
scp ./200520_allGenotypesAllLociInd_final.txt
paun@vlogin3.csb.univie.ac.at:/scratch/ovidiu/structure/200520
botanik@A772-Marie:/mnt/data1/botanik/Anna/ebg/200429_finalData/STRUCT/200520_cubeInput$
scp ./mainparams  paun@vlogin3.csb.univie.ac.at:/scratch/ovidiu/structure/200520
botanik@A772-Marie:/mnt/data1/botanik/Anna/ebg/200429_finalData/STRUCT/200520_cubeInput$
scp ./extraparams paun@vlogin3.csb.univie.ac.at:/scratch/ovidiu/structure/200520
botanik@A772-Marie:/mnt/data1/botanik/Anna/ebg/200429_finalData/STRUCT/200520_cubeInput$
scp ./slrms/*.slrm paun@vlogin3.csb.univie.ac.at:/scratch/ovidiu/structure/200520/slrms
#submit a job
cd ./slrms
sbatch STRUCTURE_K1.slrm
#check queue on cube
squeue -p basic -u paun

#copy files from cube to computer (be in directory on the computer, not in the cube!)
scp paun@vlogin3.csb.univie.ac.at:/scratch/ovidiu/structure/200608_cubeInput_ffK/*_f
/mnt/data1/botanik/Anna/ebg/200429_finalData/STRUCT/200702_cubeOutput_ffK/

#run structure harvester with .zip
#http://taylor0.biology.ucla.edu/structureHarvester/#, 01.07.2020.

#run CLUMPP for K2, K3, K4 & K5 with Greedy
#prepare paramfile
/usr/local/CLUMPP_Linux64.1.1.2/CLUMPP paramfileAdaptedK2
#...
#change K value in paramfile accordingly

#plot CLUMPP results in R
#prepare input file from .output file in libreoffice & gedit
#prepare ind labels list
sed 's/\t/\n/g' ../../newLabels4xOnly > ./newLabels4xOnlyT
#add colours and additional pop col

#run ebg alloSNP to separate the tetraploid genotypes into diploid subgenomes
#separate inc and fuc in .vcf file
mkdir alloSNP
cd ./alloSNP
#generate a list of colpositions in vcf of inc and fuc
sed 's/\t/\n/g' ../newLabels2xOnly | awk '{print NR "\t" $s}' > ./2xorderOfInds
#separate accordingly to this list
awk '{print
$1,$2,$3,$4,$5,$6,$7,$8,$9,$10,$11,$12,$13,$14,$15,$16,$17,$18,$19,$20,$21,$22,$23,$24,$25,
$26,$27,$28,$29,$30,$31,$32,$33,$34,$35,$36,$37,$38,$39,$40,$41,$42,$43,$44,$45,$46,$47,$48
}' ../shared345ind-2xOnly > ./shared345indFuc
awk '{print
$49,$50,$51,$52,$53,$54,$55,$56,$57,$58,$59,$60,$61,$62,$63,$64,$65,$66,$67,$68,$69,$70,$71
,$72,$73,$74,$75,$76,$77,$78,$79,$80,$81,$82,$83,$84,$85,$86,$87,$88,$89,$90,$91,$92,$93,$9
4,$95,$96,$97,$98,$99,$100,$101,$102,$103,$104}' ../shared345ind-2xOnly > ./shared345indInc
#separate the header
cut -f 2 2xorderOfInds | head -48 > 2xindsFuc
```

```
cut -f 2 2xorderOfInds | head -104 | tail -56 > 2xindsInc
#replace \n by tabs
#concat header to files
cat 2xindsFuc shared345indFuc > shared345indFucHeaded
cat 2xindsInc shared345indInc > shared345indIncHeaded
#concat header of infocols and infocols
cat ../newLabelsInfocols ../shared345ind-2xInfocols > ./shared345ind-2xInfocolsHeaded
#replace _ by | in infocols
sed 's/_/|/g' shared345ind-2xInfocolsHeaded > shared345ind-2xInfocolsHeadedPipe
#put infocols and data together
paste -d '\t' shared345ind-2xInfocolsHeadedPipe shared345indFucHeaded >
shared345indFucHeaded+Infocols
paste -d '\t' shared345ind-2xInfocolsHeadedPipe shared345indIncHeaded >
shared345indIncHeaded+Infocols
#replace spaces by tabs
sed 's/\s/\t/g' shared345indFucHeaded+Infocols > shared345indFucHeaded+InfocolsTab
sed 's/\s/\t/g' shared345indIncHeaded+Infocols > shared345indIncHeaded+InfocolsTab
#add header to both files
cat ../2xheadNoLabels shared345indFucHeaded+InfocolsTab > shared48indFuc_finalTab.vcf
cat ../2xheadNoLabels shared345indIncHeaded+InfocolsTab > shared56indInc_finalTab.vcf
#outputting allele frequencies with vcftools v. 0.1.15 for inc and fuc separately
vcftools --vcf shared48indFuc_finalTab.vcf --freq2 --out shared48indFuc_final.freq
vcftools --vcf shared56indInc_finalTab.vcf --freq2 --out shared56indInc_final.freq
#extract reference frequencies from freq files
cut -f 5 shared48indFuc_final.freq.frq > shared2xfuc_ref-freq.txt
#delete header row in nano
cut -f 5 shared56indInc_final.freq.frq > shared2xinc_ref-freq.txt
#delete header row in nano

#127 nan in fuc ref freq and 5 in inc ref freq -> remove those loci in all 5 input files!
#get row numbers without nans in libre office (removeNAN.ods)
#transpose tot and alt files to get loci in rows
awk '
{
    for (i=1; i<=NF; i++)  {
        a[NR,i] = $i
    }
}
NF>p { p = NF }
END {
    for(j=1; j<=p; j++) {
        str=a[1,j]
        for(i=2; i<=NR; i++){
            str=str" "a[i,j];
        }
        print str
    }
}' ../shared345ind-4x_readCounts-tot.txt > ../shared345ind-4x_readCounts-totT.txt

awk '
{
    for (i=1; i<=NF; i++)  {
        a[NR,i] = $i
    }
}
NF>p { p = NF }
END {
    for(j=1; j<=p; j++) {
        str=a[1,j]
        for(i=2; i<=NR; i++){
            str=str" "a[i,j];
        }
        print str
    }
}' ../shared345ind-4x_readCounts-alt.txt > ../shared345ind-4x_readCounts-altT.txt
```

```
#replace spaces by tabs in transposed files
sed 's/\s/\t/g' ../shared345ind-4x_readCounts-totT.txt > ../shared345ind-4x_readCounts-
totTtab.txt
sed 's/\s/\t/g' ../shared345ind-4x_readCounts-altT.txt > ../shared345ind-4x_readCounts-
altTtab.txt
#extract 45186 loci in all 5 files
sh ./removeNANinRows.sh
#transpose total and alternative read counts files back and replace spaces by tabs
awk '
{
    for (i=1; i<=NF; i++)  {
        a[NR,i] = $i
    }
}
NF>p { p = NF }
END {
    for(j=1; j<=p; j++) {
        str=a[1,j]
        for(i=2; i<=NR; i++){
            str=str" "a[i,j];
        }
        print str
    }
}' ../shared345ind-4x_readCounts-totTtab45186loci.txt > ../shared345ind-4x_readCounts-
tottab45186loci.txt

awk '
{
    for (i=1; i<=NF; i++)  {
        a[NR,i] = $i
    }
}
NF>p { p = NF }
END {
    for(j=1; j<=p; j++) {
        str=a[1,j]
        for(i=2; i<=NR; i++){
            str=str" "a[i,j];
        }
        print str
    }
}' ../shared345ind-4x_readCounts-altTtab45186loci.txt > ../shared345ind-4x_readCounts-
alttab45186loci.txt

sed 's/\s/\t/g' ../shared345ind-4x_readCounts-tottab45186loci.txt > ../shared345ind-
4x_readCounts-tot45186loci.txt
sed 's/\s/\t/g' ../shared345ind-4x_readCounts-alttab45186loci.txt > ../shared345ind-
4x_readCounts-alt45186loci.txt

#run ebg in alloSNP mode
mkdir 200525_input
cd /mnt/data1/botanik/Anna/ebg/200429_finalData/alloSNP/200525_input/
ebg alloSNP -f shared2xinc_ref-freq45186loci.txt -n 241 -l 45186 -t shared345ind-
4x_readCounts-tot45186loci.txt -a shared345ind-4x_readCounts-alt45186loci.txt -e
errshared345ind45186loci -p1 2 -p2 2 --iters 1000 --prefix 200525_alloSNPinc

ebg alloSNP -f shared2xfuc_ref-freq45186loci.txt -n 241 -l 45186 -t shared345ind-
4x_readCounts-tot45186loci.txt -a shared345ind-4x_readCounts-alt45186loci.txt -e
errshared345ind45186loci -p1 2 -p2 2 --iters 1000 --prefix 200525_alloSNPfuc

#remove nan loci from diploid files to generate structure plots with diploids and
subgenomes
```

```
#transpose diploid genos file to get loci in rows
awk '
{
    for (i=1; i<=NF; i++)  {
        a[NR,i] = $i
    }
}
NF>p { p = NF }
END {
    for(j=1; j<=p; j++) {
        str=a[1,j]
        for(i=2; i<=NR; i++){
            str=str" "a[i,j];
        }
        print str
    }
}' ../../shared345ind-2x-genos.txt > ../../shared345ind-2x-genosT.txt
#replace spaces by tabs in transposed file
sed 's/\s/\t/g' ../../shared345ind-2x-genosT.txt > ../../shared345ind-2x-genosTtab.txt
#run script to exclude nan loci
sh ./removeNANinRowsDiploids.sh

#split fuc from inc
cut -f 1-48 shared345ind-2x-genosTtab45186loci.txt > shared345ind-2x-
genosTtab45186lociFuc.txt
cut -f 49-104 shared345ind-2x-genosTtab45186loci.txt > shared345ind-2x-
genosTtab45186lociInc.txt

#transpose files back
awk '
{
    for (i=1; i<=NF; i++)  {
        a[NR,i] = $i
    }
}
NF>p { p = NF }
END {
    for(j=1; j<=p; j++) {
        str=a[1,j]
        for(i=2; i<=NR; i++){
            str=str" "a[i,j];
        }
        print str
    }
}' ./shared345ind-2x-genosTtab45186lociFuc.txt > ./shared345ind-2x-genostab45186lociFuc.txt
awk '
{
    for (i=1; i<=NF; i++)  {
        a[NR,i] = $i
    }
}
NF>p { p = NF }
END {
    for(j=1; j<=p; j++) {
        str=a[1,j]
        for(i=2; i<=NR; i++){
            str=str" "a[i,j];
        }
        print str
    }
}' ./shared345ind-2x-genosTtab45186lociInc.txt > ./shared345ind-2x-genostab45186lociInc.txt
#replace spaces by tabs in transposed file
sed 's/\s/\t/g' ./shared345ind-2x-genostab45186lociFuc.txt > ./shared345ind-2x-
genos45186lociFuc.txt
sed 's/\s/\t/g' ./shared345ind-2x-genostab45186lociInc.txt > ./shared345ind-2x-
genos45186lociInc.txt
```

```
##filter for maf 0.02 in diploid and subgenomes files
#transpose genos files
awk '
{
    for (i=1; i<=NF; i++)  {
        a[NR,i] = $i
    }
}
NF>p { p = NF }
END {
    for(j=1; j<=p; j++) {
        str=a[1,j]
        for(i=2; i<=NR; i++){
            str=str" "a[i,j];
        }
        print str
    }
}' shared345ind-2x-genos45186lociFuc.txt > shared345ind-2x-genos45186lociFucT.txt

awk '
{
    for (i=1; i<=NF; i++)  {
        a[NR,i] = $i
    }
}
NF>p { p = NF }
END {
    for(j=1; j<=p; j++) {
        str=a[1,j]
        for(i=2; i<=NR; i++){
            str=str" "a[i,j];
        }
        print str
    }
}' shared345ind-2x-genos45186lociInc.txt > shared345ind-2x-genos45186lociIncT.txt

awk '
{
    for (i=1; i<=NF; i++)  {
        a[NR,i] = $i
    }
}
NF>p { p = NF }
END {
    for(j=1; j<=p; j++) {
        str=a[1,j]
        for(i=2; i<=NR; i++){
            str=str" "a[i,j];
        }
        print str
    }
}' 200525_alloSNPfuc-g1.txt > 200525_alloSNPfuc-g1T.txt
```

```
awk '
{
    for (i=1; i<=NF; i++)  {
        a[NR,i] = $i
    }
}
NF>p { p = NF }
END {
    for(j=1; j<=p; j++) {
        str=a[1,j]
        for(i=2; i<=NR; i++){
            str=str" "a[i,j];
        }
        print str
    }
}' 200525_alloSNPfuc-g2.txt > 200525_alloSNPfuc-g2T.txt

awk '
{
    for (i=1; i<=NF; i++)  {
        a[NR,i] = $i
    }
}
NF>p { p = NF }
END {
    for(j=1; j<=p; j++) {
        str=a[1,j]
        for(i=2; i<=NR; i++){
            str=str" "a[i,j];
        }
        print str
    }
}' 200525_alloSNPinc-g1.txt > 200525_alloSNPinc-g1T.txt

awk '
{
    for (i=1; i<=NF; i++)  {
        a[NR,i] = $i
    }
}
NF>p { p = NF }
END {
    for(j=1; j<=p; j++) {
        str=a[1,j]
        for(i=2; i<=NR; i++){
            str=str" "a[i,j];
        }
        print str
    }
}' 200525_alloSNPinc-g2.txt > 200525_alloSNPinc-g2T.txt

#replace spaces by tabs
sed 's/\s/\t/g' shared345ind-2x-genos45186lociIncT.txt > shared345ind-2x-
genos45186lociIncTtab.txt
sed 's/\s/\t/g' shared345ind-2x-genos45186lociFucT.txt > shared345ind-2x-
genos45186lociFucTtab.txt
sed 's/\s/\t/g' 200525_alloSNPfuc-g1T.txt > 200525_alloSNPfuc-g1Ttab.txt
sed 's/\s/\t/g' 200525_alloSNPfuc-g2T.txt > 200525_alloSNPfuc-g2Ttab.txt
sed 's/\s/\t/g' 200525_alloSNPinc-g1T.txt > 200525_alloSNPinc-g1Ttab.txt
sed 's/\s/\t/g' 200525_alloSNPinc-g2T.txt > 200525_alloSNPinc-g2Ttab.txt

#replace -9 with NA to calculate column sums
sed 's/-9/NA/g' shared345ind-2x-genos45186lociIncTtab.txt > shared345ind-2x-
genos45186lociIncTtabNA.txt
sed 's/-9/NA/g' shared345ind-2x-genos45186lociFucTtab.txt > shared345ind-2x-
genos45186lociFucTtabNA.txt
sed 's/-9/NA/g' 200525_alloSNPfuc-g1Ttab.txt > 200525_alloSNPfuc-g1TtabNA.txt
```

```
sed 's/-9/NA/g' 200525_alloSNPfuc-g2Ttab.txt > 200525_alloSNPfuc-g2TtabNA.txt
sed 's/-9/NA/g' 200525_alloSNPinc-g1Ttab.txt > 200525_alloSNPinc-g1TtabNA.txt
sed 's/-9/NA/g' 200525_alloSNPinc-g2Ttab.txt > 200525_alloSNPinc-g2TtabNA.txt

#put files together as they will be plotted later on
#explanation of labelling files, e.g.
#'known' when fuchsii subgenome estimated from fuchsii reference allele freq.,
#'inferred' when fuchsii subgenome estimated from incarnata reference allele freq.

#fuc-fucSGknown - 48fuc+241fucSGkn=289ind
paste -d '\t' shared345ind-2x-genos45186lociFucTtabNA.txt 200525_alloSNPfuc-g1TtabNA.txt >
fuc-fucSGkn.txt

#fuc-fucSGinferred - 48fuc+241fucSGinf=289ind
paste -d '\t' shared345ind-2x-genos45186lociFucTtabNA.txt 200525_alloSNPinc-g2TtabNA.txt >
fuc-fucSGinf.txt

#inc-incSGknown - 56inc+241incSGkn=297ind
paste -d '\t' shared345ind-2x-genos45186lociIncTtabNA.txt 200525_alloSNPinc-g1TtabNA.txt >
inc-incSGkn.txt

#inc-incSGinferred - 56inc+241incSGinf=297ind
paste -d '\t' shared345ind-2x-genos45186lociIncTtabNA.txt 200525_alloSNPfuc-g2TtabNA.txt >
inc-incSGinf.txt

#inc-incSGknown-fucSGknown-fuc - 56inc+241incSGkn+241fucSGkn+48fuc=586ind
paste -d '\t' shared345ind-2x-genos45186lociIncTtabNA.txt 200525_alloSNPinc-g1TtabNA.txt
200525_alloSNPfuc-g1TtabNA.txt shared345ind-2x-genos45186lociFucTtabNA.txt > inc-incSGkn-
fucSGkn-fuc.txt

#inc-incSGinferred-fucSGinferred-fuc - 56inc+241incSGinf+241fucSGinf+48fuc=586ind
paste -d '\t' shared345ind-2x-genos45186lociIncTtabNA.txt 200525_alloSNPfuc-g2TtabNA.txt
200525_alloSNPinc-g2TtabNA.txt shared345ind-2x-genos45186lociFucTtabNA.txt > inc-incSGinf-
fucSGinf-fuc.txt

#go to R (script filter.maf.R)
#extract row sums and count -9 (NA's) per column
#get 2 files
#concatenate these 2 files horizontally
paste -d '\t' fuc-fucSGkn.rowSums.txt fuc-fucSGkn.NAperLocus.txt > fuc-fucSGkn.calcMaf.txt
paste -d '\t' fuc-fucSGinf.rowSums.txt fuc-fucSGinf.NAperLocus.txt > fuc-
fucSGinf.calcMaf.txt
paste -d '\t' inc-incSGkn.rowSums.txt inc-incSGkn.NAperLocus.txt > inc-incSGkn.calcMaf.txt
paste -d '\t' inc-incSGinf.rowSums.txt inc-incSGinf.NAperLocus.txt > inc-
incSGinf.calcMaf.txt
paste -d '\t' inc-incSGkn-fucSGkn-fuc.rowSums.txt inc-incSGkn-fucSGkn-fuc.NAperLocus.txt >
inc-incSGkn-fucSGkn-fuc.calcMaf.txt
paste -d '\t' inc-incSGinf-fucSGinf-fuc.rowSums.txt inc-incSGinf-fucSGinf-
fuc.NAperLocus.txt > inc-incSGinf-fucSGinf-fuc.calcMaf.txt
#get loci labels from -variants file
sh ./removeNANinRowsDiploids.sh #same structure as extract2466loci.sh (explained earlier)

#calcMaf345ind1.ods
#copy sequence of scaffolds in libreoffice
#extract lines according to calcMaf345ind1.ods with extractLociSGstruct.sh
cd /mnt/data1/botanik/Anna/190903_pipelines/
sh ./extractLociSGstruct.sh

#change NA back to -9 (missing data)
sed 's/NA/-9/g' fuc-fucSGinf2390loci.txt > fuc-fucSGinf2390loci-9.txt
sed 's/NA/-9/g' fuc-fucSGkn2608loci.txt > fuc-fucSGkn2608loci-9.txt
sed 's/NA/-9/g' inc-incSGkn2378loci.txt > inc-incSGkn2378loci-9.txt
sed 's/NA/-9/g' inc-incSGinf2355loci.txt > inc-incSGinf2355loci-9.txt
sed 's/NA/-9/g' inc-incSGkn-fucSGkn-fuc2608loci.txt > inc-incSGkn-fucSGkn-fuc2608loci-9.txt
sed 's/NA/-9/g' inc-incSGinf-fucSGinf-fuc2375loci.txt > inc-incSGinf-fucSGinf-fuc2375loci-
9.txt
```

```
#transpose all -9 files
awk '
{
    for (i=1; i<=NF; i++)  {
        a[NR,i] = $i
    }
}
NF>p { p = NF }
END {
    for(j=1; j<=p; j++) {
        str=a[1,j]
        for(i=2; i<=NR; i++){
            str=str" "a[i,j];
        }
        print str
    }
}' fuc-fucSGinf2390loci-9.txt > fuc-fucSGinf2390loci-9T.txt

awk '
{
    for (i=1; i<=NF; i++)  {
        a[NR,i] = $i
    }
}
NF>p { p = NF }
END {
    for(j=1; j<=p; j++) {
        str=a[1,j]
        for(i=2; i<=NR; i++){
            str=str" "a[i,j];
        }
        print str
    }
}' fuc-fucSGkn2608loci-9.txt > fuc-fucSGkn2608loci-9T.txt

awk '
{
    for (i=1; i<=NF; i++)  {
        a[NR,i] = $i
    }
}
NF>p { p = NF }
END {
    for(j=1; j<=p; j++) {
        str=a[1,j]
        for(i=2; i<=NR; i++){
            str=str" "a[i,j];
        }
        print str
    }
}' inc-incSGkn2378loci-9.txt > inc-incSGkn2378loci-9T.txt
```

```
awk '
{
    for (i=1; i<=NF; i++)  {
        a[NR,i] = $i
    }
}
NF>p { p = NF }
END {
    for(j=1; j<=p; j++) {
        str=a[1,j]
        for(i=2; i<=NR; i++){
            str=str" "a[i,j];
        }
        print str
    }
}' inc-incSGinf2355loci-9.txt > inc-incSGinf2355loci-9T.txt

awk '
{
    for (i=1; i<=NF; i++)  {
        a[NR,i] = $i
    }
}
NF>p { p = NF }
END {
    for(j=1; j<=p; j++) {
        str=a[1,j]
        for(i=2; i<=NR; i++){
            str=str" "a[i,j];
        }
        print str
    }
}' inc-incSGkn-fucSGkn-fuc2608loci-9.txt > inc-incSGkn-fucSGkn-fuc2608loci-9T.txt

awk '
{
    for (i=1; i<=NF; i++)  {
        a[NR,i] = $i
    }
}
NF>p { p = NF }
END {
    for(j=1; j<=p; j++) {
        str=a[1,j]
        for(i=2; i<=NR; i++){
            str=str" "a[i,j];
        }
        print str
    }
}' inc-incSGinf-fucSGinf-fuc2375loci-9.txt > inc-incSGinf-fucSGinf-fuc2375loci-9T.txt

#replace spaces by tabs
sed 's/\s/\t/g' fuc-fucSGinf2390loci-9T.txt > fuc-fucSGinf2390loci-9Ttab.txt
sed 's/\s/\t/g' fuc-fucSGkn2608loci-9T.txt > fuc-fucSGkn2608loci-9Ttab.txt
sed 's/\s/\t/g' inc-incSGkn2378loci-9T.txt > inc-incSGkn2378loci-9Ttab.txt
sed 's/\s/\t/g' inc-incSGinf2355loci-9T.txt > inc-incSGinf2355loci-9Ttab.txt
sed 's/\s/\t/g' inc-incSGkn-fucSGkn-fuc2608loci-9T.txt > inc-incSGkn-fucSGkn-fuc2608loci-
9Ttab.txt
sed 's/\s/\t/g' inc-incSGinf-fucSGinf-fuc2375loci-9T.txt > inc-incSGinf-fucSGinf-
fuc2375loci-9Ttab.txt

#add indlabels to each file
sed 's/\t/\n/g' /mnt/data1/botanik/Anna/ebg/200429_finalData/newLabels4xOnly | sed 's/_//g'
> ./newLabels4xOnlyT
sed 's/\t/\n/g' /mnt/data1/botanik/Anna/ebg/200429_finalData/newLabels2xOnly | sed 's/_//g'
> ./newLabels2xOnlyT
#prepare indlabels in libreoffice
```

```
#duplicate each row
sed 'p' labels.fuc-fucSGkn.txt > labels.fuc-fucSGkn2lines.txt
sed 'p' labels.fuc-fucSGinf.txt > labels.fuc-fucSGinf2lines.txt
sed 'p' labels.inc-incSGkn.txt > labels.inc-incSGkn2lines.txt
sed 'p' labels.inc-incSGinf.txt > labels.inc-incSGinf2lines.txt
sed 'p' labels.inc-incSGkn-fucSGkn-fuc.txt > labels.inc-incSGkn-fucSGkn-fuc2lines.txt
sed 'p' labels.inc-incSGinf-fucSGinf-fuc.txt > labels.inc-incSGinf-fucSGinf-fuc2lines.txt

#print each line in data file two times
sed 's/0/5_5/g' fuc-fucSGkn2608loci-9Ttab.txt | sed 's/1/5_6/g' | sed 's/2/6_6/g' | sed
's/-9/-9_-9/g' | sed 'p' |
sed '0~2s/5_//g' | sed '0~2s/6_//g' | sed '0~2s/-9_//g' | sed 's/_5//g' | sed 's/_6//g' |
sed 's/_-9//g' > ../fuc-fucSGkn2608loci-9Ttab2lines.txt

sed 's/0/5_5/g' fuc-fucSGinf2390loci-9Ttab.txt | sed 's/1/5_6/g' | sed 's/2/6_6/g' | sed
's/-9/-9_-9/g' | sed 'p' |
sed '0~2s/5_//g' | sed '0~2s/6_//g' | sed '0~2s/-9_//g' | sed 's/_5//g' | sed 's/_6//g' |
sed 's/_-9//g' > ../fuc-fucSGinf2390loci-9Ttab2lines.txt

sed 's/0/5_5/g' inc-incSGkn2378loci-9Ttab.txt | sed 's/1/5_6/g' | sed 's/2/6_6/g' | sed
's/-9/-9_-9/g' | sed 'p' |
sed '0~2s/5_//g' | sed '0~2s/6_//g' | sed '0~2s/-9_//g' | sed 's/_5//g' | sed 's/_6//g' |
sed 's/_-9//g' > ../inc-incSGkn2378loci-9Ttab2lines.txt

sed 's/0/5_5/g' inc-incSGinf2355loci-9Ttab.txt | sed 's/1/5_6/g' | sed 's/2/6_6/g' | sed
's/-9/-9_-9/g' | sed 'p' |
sed '0~2s/5_//g' | sed '0~2s/6_//g' | sed '0~2s/-9_//g' | sed 's/_5//g' | sed 's/_6//g' |
sed 's/_-9//g' > ../inc-incSGinf2355loci-9Ttab2lines.txt

sed 's/0/5_5/g' inc-incSGkn-fucSGkn-fuc2608loci-9Ttab.txt | sed 's/1/5_6/g' | sed
's/2/6_6/g' | sed 's/-9/-9_-9/g' | sed 'p' |
sed '0~2s/5_//g' | sed '0~2s/6_//g' | sed '0~2s/-9_//g' | sed 's/_5//g' | sed 's/_6//g' |
sed 's/_-9//g' > ../inc-incSGkn-fucSGkn-fuc2608loci-9Ttab2lines.txt

sed 's/0/5_5/g' inc-incSGinf-fucSGinf-fuc2375loci-9Ttab.txt | sed 's/1/5_6/g' | sed
's/2/6_6/g' | sed 's/-9/-9_-9/g' | sed 'p' |
sed '0~2s/5_//g' | sed '0~2s/6_//g' | sed '0~2s/-9_//g' | sed 's/_5//g' | sed 's/_6//g' |
sed 's/_-9//g' > ../inc-incSGinf-fucSGinf-fuc2375loci-9Ttab2lines.txt

#paste indlabels to files
paste -d '\t' labels.fuc-fucSGkn2lines.txt fuc-fucSGkn2608loci-9Ttab2lines.txt >
200608_fuc-fucSGkn2608loci_final.txt
paste -d '\t' labels.fuc-fucSGinf2lines.txt fuc-fucSGinf2390loci-9Ttab2lines.txt >
200608_fuc-fucSGinf2390loci_final.txt
paste -d '\t' labels.inc-incSGkn2lines.txt inc-incSGkn2378loci-9Ttab2lines.txt >
200608_inc-incSGkn2378loci_final.txt
paste -d '\t' labels.inc-incSGinf2lines.txt inc-incSGinf2355loci-9Ttab2lines.txt >
200608_inc-incSGinf2355loci_final.txt
paste -d '\t' labels.inc-incSGkn-fucSGkn-fuc2lines.txt inc-incSGkn-fucSGkn-fuc2608loci-
9Ttab2lines.txt > 200608_inc-incSGkn-fucSGkn-fuc2608loci_final.txt
paste -d '\t' labels.inc-incSGinf-fucSGinf-fuc2lines.txt inc-incSGinf-fucSGinf-fuc2375loci-
9Ttab2lines.txt > 200608_inc-incSGinf-fucSGinf-fuc2375loci_final.txt

#run ffK on cube
#enter cube
ssh paun@vlogin3.csb.univie.ac.at
cd /scratch/ovidiu/structure
mkdir 200608_cubeInput_ffK
mkdir ./200608_cubeInput_ffK/slrms
```

```
#copy files from computer to cube
botanik@A772-
Marie:/mnt/data1/botanik/Anna/ebg/200429_finalData/STRUCT/200606_subgenomesParents$ scp
./200608_cubeInput_ffK/*
paun@vlogin3.csb.univie.ac.at:/scratch/ovidiu/structure/200608_cubeInput_ffK
botanik@A772-
Marie:/mnt/data1/botanik/Anna/ebg/200429_finalData/STRUCT/200606_subgenomesParents$ scp
./200608_cubeInput_ffK/slrms/*
paun@vlogin3.csb.univie.ac.at:/scratch/ovidiu/structure/200608_cubeInput_ffK/slrms
#submit a job
cd ./slrms
sbatch STRUCTURE_K1.slrm
#check queue on cube
squeue -p basic -u paun

#copy files from cube to computer (from computer directory!)
scp paun@vlogin3.csb.univie.ac.at:/scratch/ovidiu/structure/200520/*_f
/mnt/data1/botanik/Anna/ebg/200429_finalData/STRUCT/200520_cubeOutput/
####################
```

Plotting the sampling localities onto a geographical map

```
########## samplingMaps.R ##########
########## R script for plotting sampling localities from different taxa onto a map
##########

setwd("C:/R_files/DactPhylogenomics/210514_samplingMaps/")


#install and load required packages

install.packages("rworldmap")

install.packages("rworldxtra")

install.packages("marmap")

library(rworldmap)

library(rworldxtra)

library(marmap)


#load data

sampling.data <- read.table(file="210812_finalCoordData.txt",head=T)


#plot map with pie charts

pdf(file="samplingMap.pdf",paper="a4r",width=10,height=10)


worldmap <- getMap(resolution="high")

plot(worldmap,xlim=c(min(sampling.data[,3]),max(sampling.data[,3])),

     ylim=c(min(sampling.data[,2]),max(sampling.data[,2])),

     border="darkgrey")
```

```
space.pies(x=sampling.data$lon,y=sampling.data$lat,
           pie.slices=sampling.data[,4:8],
           pie.colors=sampling.data[,9:13],
           pie.radius=0.6,
           pie.space=0.0000000000001,link=F)

#remove labels from the Alpine region -> visible only in the separate map
sampling.dataNoAT <- sampling.data[1:10,1:3] == NA
sampling.dataNoAT <- data.frame(rbind(sampling.dataNoAT,sampling.data[-(1:10),1:3]))

text(sampling.dataNoAT$lon,sampling.dataNoAT$lat,
     labels=sampling.dataNoAT$loc,cex=1,pos=3,offset=.5)

legend("topleft",
       legend=c("D. fuchsii",
                "D. incarnata",
                "D. majalis",
                "D. traunsteineri",
                "D. purpurella"),
       col=c("darkorange4",
             "darkorange2",
             "chartreuse3",
             "deepskyblue3",
             "maroon4"),
       pch=16,cex=2,bty='o',bg="white",box.col="white")

axis(side=1,at=seq(from=(-20),to=40,by=5))
axis(side=2,at=seq(from=40,to=70,by=5))


#plot map for Alps separately
plot(worldmap,xlim=c(min(sampling.data[c(1:10,29,30,32),3]-
1),max(sampling.data[c(1:10,29,30,32),3]+1)),
     ylim=c(min(sampling.data[c(1:10,29,30,32),2]-
1),max(sampling.data[c(1:10,29,30,32),2]+1)),
     border="darkgrey")
```

```
space.pies(x=sampling.data$lon[c(1:10,29,30,32)],y=sampling.data$lat[c(1:10,29,30,32)],

           pie.slices=sampling.data[c(1:10,29,30,32),4:8],

           pie.colors=sampling.data[c(1:10,29,30,32),9:13],

           pie.radius=0.1,

           pie.space=0.0000000000001,link=F)


text(sampling.data$lon[c(1:10,29,30,32)],sampling.data$lat[c(1:10,29,30,32)],

     labels=sampling.data$loc[c(1:10,29,30,32)],cex=1,pos=3,offset=.5)


#add longitudinal and latitudinal axes onto the plot

axis(side=1)

axis(side=2)


dev.off()
####################
```

Plotting the heatmap of pairwise relatedness

```
########## heatmapPolyrelatedness.R ##########
########## plotting a heatmap of pairwise relatedness from a covariance matrix generated
with polyRelatedness v. 1.8 ##########

#install and load required package
install.packages("gplots")
library(gplots)

setwd("/mnt/data1/botanik/Anna/ebg/200429_finalData/polyrel/")
polyplMOMoutNA241ind <- as.matrix(read.table("200609_241ind18879lociMOMoutNA.txt"))
#diagonal def. as NA (in libreoffice)

polyplIndLoc241ind <- read.table("newLabels4xOnlyT")
polyplIndLoc241ind.lab <- as.vector(polyplIndLoc241ind[,1]) #accession numbers
polyplIndLoc241ind.col <- as.vector(polyplIndLoc241ind[,3]) #colours
polyplIndLoc241ind.pop <- as.vector(polyplIndLoc241ind[,4]) #locality labels
polyplIndLoc241ind.cou <- as.vector(polyplIndLoc241ind[,5]) #countries

#heatmap without dendrogram, sorted by loc, full accession numbers
pdf(file="200628_241ind18879loci-MOMheatmapInd.pdf")
heatmapMOM241ind <- heatmap.2(polyplMOMoutNA241ind,

colsep=c(11,22,27,32,42,117,122,155,159,164,169,174,186,200,205,221,236),
rowsep=c(11,22,27,32,42,117,122,155,159,164,169,174,186,200,205,221,236), sepcolor="black",
sepwidth=c(0.02,0.02),
                              trace="none",
                              Rowv=NA, Colv=NA, cexRow=0.2 ,cexCol=0.7, labRow =
polyplIndLoc241ind.lab, labCol = polyplIndLoc241ind.cou, colRow = polyplIndLoc241ind.col,
colCol = "black",
                              col= colorRampPalette(c("lemonchiffon", "lemonchiffon",
"lemonchiffon", "lemonchiffon", "lemonchiffon", "lemonchiffon", "lemonchiffon",
"lemonchiffon", "yellow", "red", "magenta", "midnightblue", "black", "black", "black",
"black", "black", "black"))(100))
title("MOM
      241 polyploids,
      18,879 loci (filtered by maf = 0.02)")
```

```
dev.off()

#heatmap without dendrogram, sorted by loc, locality labels
pdf(file="200628_241ind18879loci-MOMheatmapLoc.pdf")
heatmapMOM241ind <- heatmap.2(polyplMOMoutNA241ind,

colsep=c(11,22,27,32,42,117,122,155,159,164,169,174,186,200,205,221,236),
rowsep=c(11,22,27,32,42,117,122,155,159,164,169,174,186,200,205,221,236), sepcolor="black",
sepwidth=c(0.02,0.02),
                              trace="none",
                              Rowv=NA, Colv=NA, cexRow=0.5 ,cexCol=0.7, labRow =
polyplIndLoc241ind.pop, labCol = polyplIndLoc241ind.cou, colRow = col, colCol = "black",
adjRow = c(1,0), adjCol = c(1,0.5),
                              col= colorRampPalette(c("lemonchiffon", "lemonchiffon",
"lemonchiffon", "lemonchiffon", "lemonchiffon", "lemonchiffon", "lemonchiffon",
"lemonchiffon", "yellow", "red", "magenta", "midnightblue", "black", "black", "black",
"black", "black", "black"))(100))
title("MOM
      241 polyploids,
      18,879 loci (filtered by maf = 0.02)")
dev.off()
####################
```

Plotting STRUCTURE results

```
########## plottingSTRUCTURE-copy.R ##########
########## plotting STRUCTURE results ##########

#set working directory
setwd("/mnt/data1/botanik/Anna/ebg/200429_finalData/STRUCT/200701_cubeOutputPolypl/RinputPo
lypl/")

#import indlist
ind <- read.table("newLabels4xOnlyT")
col <- as.vector(ind[,3])

#import K2, K3, K4, K5 (results from CLUMPP in a table)
K2 <- read.table(file="K2.Rinput.txt",head=F)
K3 <- read.table(file="K3.Rinput.txt",head=F)
K4 <- read.table(file="K4.Rinput.txt",head=F)
K5 <- read.table(file="K5.Rinput.txt",head=F)

#with sampling site labels
pdf(file="200701_241ind2466loci-Polypl-K2345_Loc.pdf",paper="a4")
par(mfrow=c(4,1))

barplot(t(K2),col=c("red3","chartreuse3"),space=0,border=NA,axes=F)
abline(v=c(11,22,27,32,42,117,122,155,159,164,169,174,186,200,205,221,236),col="black")
text(x=1:nrow(ind),y=1.3,labels=ind[,5],srt=0,pos=1,xpd=T,cex=0.8,col="black")
text(x=1:nrow(ind),y=-0.02,labels=ind[,4],srt=0,pos=1,xpd=T,cex=0.5,col=col)
title("K=2",cex.main=1)

barplot(t(K3),col=c("red3","chartreuse3","deepskyblue3"),space=0,border=NA,axes=F)
abline(v=c(11,22,27,32,42,117,122,155,159,164,169,174,186,200,205,221,236),col="black")
text(x=1:nrow(ind),y=1.3,labels=ind[,5],srt=0,pos=1,xpd=T,cex=0.8,col="black")
text(x=1:nrow(ind),y=-0.02,labels=ind[,4],srt=0,pos=1,xpd=T,cex=0.5,col=col)
title("K=3",cex.main=1)

barplot(t(K4),col=c("red3","chartreuse3","deepskyblue3","orange"),space=0,border=NA,axes=F)
abline(v=c(11,22,27,32,42,117,122,155,159,164,169,174,186,200,205,221,236),col="black")
text(x=1:nrow(ind),y=1.3,labels=ind[,5],srt=0,pos=1,xpd=T,cex=0.8,col="black")
text(x=1:nrow(ind),y=-0.02,labels=ind[,4],srt=0,pos=1,xpd=T,cex=0.5,col=col)
title("K=4",cex.main=1)
```

```
barplot(t(K5),col=c("red3","chartreuse3","deepskyblue3","orange","darkorchid3"),space=0,bor
der=NA,axes=F)
abline(v=c(11,22,27,32,42,117,122,155,159,164,169,174,186,200,205,221,236),col="black")
text(x=1:nrow(ind),y=1.3,labels=ind[,5],srt=0,pos=1,xpd=T,cex=0.8,col="black")
text(x=1:nrow(ind),y=-0.02,labels=ind[,4],srt=0,pos=1,xpd=T,cex=0.5,col=col)
title("K=5",cex.main=1)

dev.off()

#with accession numbers
pdf(file="200701_241ind2466loci-Polypl-K2345_Ind.pdf",paper="a4")
par(mfrow=c(4,1))

barplot(t(K2),col=c("red3","chartreuse3"),space=0,border=NA,axes=F)
abline(v=c(11,22,27,32,42,117,122,155,159,164,169,174,186,200,205,221,236),col="black")
text(x=1:nrow(ind),y=1.3,labels=ind[,5],srt=0,pos=1,xpd=T,cex=0.8,col="black")
text(x=1:nrow(ind),y=-0.02,labels=ind[,1],srt=-45,pos=1,xpd=T,cex=0.1,col=col)
title("K=2",cex.main=1)

barplot(t(K3),col=c("red3","chartreuse3","deepskyblue3"),space=0,border=NA,axes=F)
abline(v=c(11,22,27,32,42,117,122,155,159,164,169,174,186,200,205,221,236),col="black")
text(x=1:nrow(ind),y=1.3,labels=ind[,5],srt=0,pos=1,xpd=T,cex=0.8,col="black")
text(x=1:nrow(ind),y=-0.02,labels=ind[,1],srt=-45,pos=1,xpd=T,cex=0.1,col=col)
title("K=3",cex.main=1)

barplot(t(K4),col=c("red3","chartreuse3","deepskyblue3","orange"),space=0,border=NA,axes=F)
abline(v=c(11,22,27,32,42,117,122,155,159,164,169,174,186,200,205,221,236),col="black")
text(x=1:nrow(ind),y=1.3,labels=ind[,5],srt=0,pos=1,xpd=T,cex=0.8,col="black")
text(x=1:nrow(ind),y=-0.02,labels=ind[,1],srt=-45,pos=1,xpd=T,cex=0.1,col=col)
title("K=4",cex.main=1)

barplot(t(K5),col=c("red3","chartreuse3","deepskyblue3","orange","darkorchid3"),space=0,bor
der=NA,axes=F)
abline(v=c(11,22,27,32,42,117,122,155,159,164,169,174,186,200,205,221,236),col="black")
text(x=1:nrow(ind),y=1.3,labels=ind[,5],srt=0,pos=1,xpd=T,cex=0.8,col="black")
text(x=1:nrow(ind),y=-0.02,labels=ind[,1],srt=-45,pos=1,xpd=T,cex=0.1,col=col)
title("K=5",cex.main=1)

dev.off()

#same procedure for subgenome plots, not shown here
###################
```

Filtering for, generating and plotting joint Site Frequency Spectra (jSFS)

```
########## jSFSfiltering_final_v2.R ##########
########## filtering data for constructing joint site frequency spectra ##########

#script starts with inputting the ebg output file of all ind. with q>.8/.75 with -9
replaced by NA &

#ends with the filtered data set for constructing the jSFS


#set working directory

setwd("C:/R_files/DactPhylogenomics/201203_SFSscript/")


#install and load required packages

install.packages("tidyverse")
library(tidyverse)
```

```
#load data (without any header)
#fuchsii side
data.fuc <- read.table(file="48ind_fuc.txt",head=F,na.strings="-9")
data.majF <- read.table(file="53ind_majF.txt",head=F,na.strings="-9")
data.tauF <- read.table(file="8ind_tauF.txt",head=F,na.strings="-9")
data.tcoF <- read.table(file="50ind_tcoF.txt",head=F,na.strings="-9")
data.tbrF <- read.table(file="11ind_tbrF.txt",head=F,na.strings="-9")
data.purF <- read.table(file="8ind_purF.txt",head=F,na.strings="-9")

all.FF <- cbind(data.fuc,data.majF,data.tauF,data.tcoF,data.tbrF,data.purF)
compl1.FF <- as.matrix(
  ifelse((rowSums(all.FF,na.rm=T)) > (ncol(all.FF) - rowSums(is.na(all.FF))),
         1,0)) #1 is major allele freq, 0 is minor allele

#incarnata side
data.inc <- read.table(file="56ind_inc.txt",head=F,na.strings="-9")
data.majI <- read.table(file="53ind_majI.txt",head=F,na.strings="-9")
data.tauI <- read.table(file="8ind_tauI.txt",head=F,na.strings="-9")
data.tcoI <- read.table(file="50ind_tcoI.txt",head=F,na.strings="-9")
data.tbrI <- read.table(file="11ind_tbrI.txt",head=F,na.strings="-9")
data.purI <- read.table(file="8ind_purI.txt",head=F,na.strings="-9")

all.II <- cbind(data.inc,data.majI,data.tauI,data.tcoI,data.tbrI,data.purI)
compl1.II <- as.matrix(
  ifelse((rowSums(all.II,na.rm=T)) > (ncol(all.II) - rowSums(is.na(all.II))),
         1,0)) #1 is major allele freq, 0 is minor allele
#load loci list with 2 columns (scaffold and position)
lociList45186 <- read.table(file="lociList45186.txt",head=F)
##defining filter functions
##get minor allele if ebg alloSNP outputs the major
convert2minor <- function(data,compl1){
  newState <- ifelse(as.matrix(compl1) == 1,
                     2 - data,
                     data)
  return(as.matrix(newState))
}
```

```
##pathway to filter for missing data

missingData <- function(data,maxMiss){

  ifelse(sum(!is.na(data)) >= maxMiss,0,1) #1 = more than 8 ind. with missing data per
locus, 0 = less than 8 ind. with missing data per locus

}

##prepare data for subsetting to 6/8

preSubsetCombined <- function(dataDemeA,dataDemeB,lociList,maxMissA,maxMissB){ #include
loci list to retain only those columns in it which are needed for 1 ev. 1000 bp

  missingDataApplyA <- apply(X=dataDemeA,MAR=1,FUN=missingData,maxMiss=maxMissA) #marks
loci with more than "maxMissA" loci missing in deme A with a "1"

  missingDataApplyB <- apply(X=dataDemeB,MAR=1,FUN=missingData,maxMiss=maxMissB) #marks
loci with more than "maxMissB" loci missing in deme B with a "1"

  codedMissing <-
as.data.frame(cbind(lociList,dataDemeA,dataDemeB,missingDataApplyA,missingDataApplyB))
#combine the locilist with the data and the missingness category (0/1)

  colnames(codedMissing) <- seq(from=1,to=ncol(codedMissing)) #rename the column names to
be able to apply filter()

  missingExcl <-
filter(codedMissing,codedMissing[,(ncol(lociList)+ncol(dataDemeA)+ncol(dataDemeB)+1)] ==
"0" & codedMissing[,(ncol(lociList)+ncol(dataDemeA)+ncol(dataDemeB)+2)] =="0") #only retain
loci which have data for mind. 8 individuals in both demes

  missingExclData <- missingExcl[,-
c((ncol(lociList)+ncol(dataDemeA)+ncol(dataDemeB)+1),(ncol(lociList)+ncol(dataDemeA)+ncol(d
ataDemeB)+2))] #exclude column with the missingness category (0/1)

  print(paste(nrow(missingExclData),"of",nrow(dataDemeA),"loci retained
(missingness)",sep=" ")) #documents how many loci are retained after this step is done

  return(missingExclData) #outputs the data filtered by missingness

}

##subsample large dataset randomly

randomSubset <- function(data,ind){ #ind is the number of ind. you want to keep for further
analysis (here 6 or 8)

  subs <- sample(data[!is.na(data)],size=ind) #loops through every row (locus per locus)
and retains #"ind" (here 6 or 8) random values per row

  subs <- t(subs) #transpose

  return(subs) #returns a random subset of #"ind"

}
```

```
##exclude monomorphic loci

monoPoly <- function(data){

  count0 <- ifelse(sum(data == 0,na.rm=T) < sum(!is.na(data)),0,1) #marks loci with 0 in
all columns (individuals) with a "1"

  count2 <- ifelse(sum(data == 2,na.rm=T) < sum(!is.na(data)),0,1) #marks loci with 2 in
all columns (individuals) with a "1"

  sum02 <- sum(c(count0,count2)) #1 = monomorphic, 0 = polymorphic, every locus with either
0 only or 2 only will be marked with a "1"

  return(sum02) #outputs a single column with categorised loci

}

#apply monoPoly function to both demes contemporarily

getPolymorphicData <- function(dataDemeA,dataDemeB,lociList){

  monoPolyApplyAB <- apply(X=cbind(dataDemeA,dataDemeB),MAR=1,FUN=monoPoly) #categorises
loci according to monomorphy/polymorphy across both demes

  codedMonoPoly <- as.data.frame(cbind(lociList,dataDemeA,dataDemeB,monoPolyApplyAB))
#combines locilist, data and category as before (missingness function)

  colnames(codedMonoPoly) <- seq(from=1,to=ncol(codedMonoPoly)) #renaming column names to
be able to apply filter()

  monoExcl <-
filter(codedMonoPoly,codedMonoPoly[,(ncol(lociList)+ncol(dataDemeA)+ncol(dataDemeB)+1)] ==
"0") #retains only polymorphic loci beforehand marked with a 0 in the last column of the
data frame constructed before

  monoExclData <- monoExcl[,-c(((ncol(lociList)+ncol(dataDemeA)+ncol(dataDemeB))+1))]
#excludes the last column with filter category again

  print(paste(nrow(monoExclData),"of",nrow(dataDemeA),"loci retained (polymorphic
loci)",sep=" ")) #documents number of retained loci

  return(monoExclData) #outputs data with polymorphic loci only

}

##pathway to take 1 SNP ev. x bp

filter1SNPevXbp <- function(data,lociList,oneEvery){

  scaffold <- as.matrix(lociList[,1]) #define column where scaffold names are to be found

  position <- as.matrix(lociList[,2]) #define column where the locus positions are to be
found

  keepList <- data.frame(keep0=0) #start a data frame with a random value (which is deleted
later on and has no influence on the filtering itself)

  for(locus in 1:(nrow(data)-1)){ ##loop through every row (=locus) directly within this
function -> no requirement for apply() later on!!!

    filterFunction <- ifelse(scaffold[locus] == scaffold[locus+1], #compare the current
locus with the subsequent one

                             ifelse(position[locus+1] - position[locus] < oneEvery,1,0),
#if the current locus is on the same scaffold as the next scaffold, then consider the
position
```

```
                             0) #if the loci are on different scaffolds, keep the one you
are looking at currently

    keepList <- rbind(keepList,filterFunction) #adds the result (0/1) for every locus to
the keepList

  }

  #last locus is omitted by filterFunction, because there is no subsequent locus anymore ->
include by taking forthcoming locus into consideration

  filterLastLocus <- ifelse(scaffold[nrow(data)] == scaffold[(nrow(data)-1)],

                            ifelse(position[nrow(data)] - position[(nrow(data)-1)] <
oneEvery,1,0),

                            0)

  keepList <- rbind(keepList,filterLastLocus) #add the filter category of the last locus to
the rest

  return(keepList[-1,]) #outputs the keepList without the first random value

}

##apply the above function to filter the actual data set

keep1SNPevXbp <- function(data,lociList,oneEvery){

  filter1SNPevXbpRun <- filter1SNPevXbp(data=data,lociList=lociList,oneEvery=oneEvery)
#outputs a single column with 0/1 filtering category (keep 0)

  coded1evXbp <- cbind(lociList,data,filter1SNPevXbpRun)

  colnames(coded1evXbp) <- seq(from=1,to=ncol(coded1evXbp))

  oneSNPevXbpExcl <- filter(coded1evXbp,coded1evXbp[,(ncol(data)+3)] == "0") #retain loci
marked with a "0"

  oneSNPevXbpExclData <- oneSNPevXbpExcl[,-(ncol(data)+3)] #exclude column with filter
category (0/1)

  print(paste(nrow(oneSNPevXbpExclData),"of",nrow(data),"loci retained (1
every",oneEvery,"bp)",sep=" ")) #document number of retained loci
  return(oneSNPevXbpExclData) #outputs data set filtered by 1 every x bp

}

##put all filter functions together

filteredData <-
function(dataDemeA,dataDemeB,compl1,maxMissA,maxMissB,indA,indB,lociList,oneEvery){



  minorA <- apply(dataDemeA,MAR=2,FUN=convert2minor,compl1)

  minorB <- apply(dataDemeB,MAR=2,FUN=convert2minor,compl1)

  print(paste("deme A:",colSums(as.matrix(compl1)),"of",nrow(minorA),"loci were converted
from major to minor",sep=" ")) #documents how many loci are converted

  print(paste("deme B:",colSums(as.matrix(compl1)),"of",nrow(minorB),"loci were converted
from major to minor",sep=" ")) #documents how many loci are converted
```

```
  preSubs <-
preSubsetCombined(dataDemeA=minorA,dataDemeB=minorB,lociList=lociList,maxMissA=maxMissA,max
MissB=maxMissB)



  preSubsAloci <- preSubs[,c(1,2,3:(ncol(dataDemeA)+2))]

  preSubsA <- preSubs[,3:(ncol(dataDemeA)+2)]

  preSubsBloci <- preSubs[,c(1,2,((ncol(dataDemeA)+3)):ncol(preSubs))]

  preSubsB <- preSubs[,(ncol(dataDemeA)+3):ncol(preSubs)]



  randomSubsetA <- t(apply(X=preSubsA,MAR=1,FUN=randomSubset,ind=indA))

  print(paste("deme A:",nrow(randomSubsetA),"loci in",ncol(randomSubsetA),"individuals
retained",sep=" "))

  randomSubsetB <- t(apply(X=preSubsB,MAR=1,FUN=randomSubset,ind=indB))

  print(paste("deme B:",nrow(randomSubsetB),"loci in",ncol(randomSubsetB),"individuals
retained",sep=" "))



  poly <-
getPolymorphicData(dataDemeA=randomSubsetA,dataDemeB=randomSubsetB,lociList=preSubsAloci[,1
:2])



  keep1SNPevXbpAB <- keep1SNPevXbp(data=poly[,-
(1:2)],lociList=poly[,1:2],oneEvery=oneEvery)



  return(keep1SNPevXbpAB)

}

#generate data for each pair, only one pair shown

#fuchsii side

dataForSFS.fuc.majF <- filteredData(dataDemeA=data.fuc,dataDemeB=data.majF,

                                    compl1=compl1.FF,

                                    maxMissA=12,maxMissB=12,

                                    indA=12,indB=12,

                                    lociList=lociList45186,oneEvery=1000)


#same procedure for the incarnata side, not shown here

#save files

#fuchsii side, only one pair shown

write.table(dataForSFS.fuc.majF,

            file="./210203_resultsF/dataForSFS_12fuc_12majF.txt",
```

```
            row.names=F,col.names=F,sep="\t")


#same procedure for the incarnata side, not shown here
####################



########## jSFS_matrix12-24.R ##########
########## R script generating a joint site frequency spectrum with 6 individuals in the
one deme & 12 in the other ##########

#results in a matrix with 13 columns and 25 rows
#other scripts (jSFS_matrix16-24.R and jSFS_matrix24-24.R) not shown, but work in the same
way

setwd("C:/R_files/DactPhylogenomics/201203_SFSscript/")

#allelesPerDeme

allelesPerDeme <- function(data){

  alleleSum <- as.matrix(rowSums(data,na.rm=T))

  return(alleleSum)

}

#jSFS

jSFS <- function(dataDemeA,dataDemeB){

  allelesDemeA <- allelesPerDeme(dataDemeA)

  allelesDemeB <- allelesPerDeme(dataDemeB)

  allelesCombined <- as.matrix(paste(allelesDemeA,allelesDemeB,sep=","))



  cat00 <- sum(allelesCombined == '0,0')
  cat01 <- sum(allelesCombined == '0,1')

  cat02 <- sum(allelesCombined == '0,2')

  cat03 <- sum(allelesCombined == '0,3')

  cat04 <- sum(allelesCombined == '0,4')

  cat05 <- sum(allelesCombined == '0,5')

  cat06 <- sum(allelesCombined == '0,6')

  cat07 <- sum(allelesCombined == '0,7')

  cat08 <- sum(allelesCombined == '0,8')

  cat09 <- sum(allelesCombined == '0,9')

  cat010 <- sum(allelesCombined == '0,10')

  cat011 <- sum(allelesCombined == '0,11')

  cat012 <- sum(allelesCombined == '0,12')

  cat013 <- sum(allelesCombined == '0,13')
```

```
  cat014 <- sum(allelesCombined == '0,14')

  cat015 <- sum(allelesCombined == '0,15')

  cat016 <- sum(allelesCombined == '0,16')

  cat017 <- sum(allelesCombined == '0,17')

  cat018 <- sum(allelesCombined == '0,18')

  cat019 <- sum(allelesCombined == '0,19')

  cat020 <- sum(allelesCombined == '0,20')

  cat021 <- sum(allelesCombined == '0,21')

  cat022 <- sum(allelesCombined == '0,22')

  cat023 <- sum(allelesCombined == '0,23')

  cat024 <- sum(allelesCombined == '0,24')

  countsCol0 <-
cbind(c(cat00,cat01,cat02,cat03,cat04,cat05,cat06,cat07,cat08,cat09,cat010,cat011,cat012,ca
t013,cat014,cat015,cat016,cat017,cat018,cat019,cat020,cat021,cat022,cat023,cat024))



  cat10 <- sum(allelesCombined == '1,0')

  cat11 <- sum(allelesCombined == '1,1')

  cat12 <- sum(allelesCombined == '1,2')

  cat13 <- sum(allelesCombined == '1,3')

  cat14 <- sum(allelesCombined == '1,4')

  cat15 <- sum(allelesCombined == '1,5')

  cat16 <- sum(allelesCombined == '1,6')
  cat17 <- sum(allelesCombined == '1,7')

  cat18 <- sum(allelesCombined == '1,8')

  cat19 <- sum(allelesCombined == '1,9')

  cat110 <- sum(allelesCombined == '1,10')

  cat111 <- sum(allelesCombined == '1,11')

  cat112 <- sum(allelesCombined == '1,12')

  cat113 <- sum(allelesCombined == '1,13')

  cat114 <- sum(allelesCombined == '1,14')

  cat115 <- sum(allelesCombined == '1,15')

  cat116 <- sum(allelesCombined == '1,16')

  cat117 <- sum(allelesCombined == '1,17')

  cat118 <- sum(allelesCombined == '1,18')

  cat119 <- sum(allelesCombined == '1,19')
```

```
  cat120 <- sum(allelesCombined == '1,20')

  cat121 <- sum(allelesCombined == '1,21')

  cat122 <- sum(allelesCombined == '1,22')

  cat123 <- sum(allelesCombined == '1,23')

  cat124 <- sum(allelesCombined == '1,24')

  countsCol1 <-
cbind(c(cat10,cat11,cat12,cat13,cat14,cat15,cat16,cat17,cat18,cat19,cat110,cat111,cat112,ca
t113,cat114,cat115,cat116,cat117,cat118,cat119,cat120,cat121,cat122,cat123,cat124))



  cat20 <- sum(allelesCombined == '2,0')

  cat21 <- sum(allelesCombined == '2,1')

  cat22 <- sum(allelesCombined == '2,2')

  cat23 <- sum(allelesCombined == '2,3')

  cat24 <- sum(allelesCombined == '2,4')

  cat25 <- sum(allelesCombined == '2,5')

  cat26 <- sum(allelesCombined == '2,6')

  cat27 <- sum(allelesCombined == '2,7')

  cat28 <- sum(allelesCombined == '2,8')

  cat29 <- sum(allelesCombined == '2,9')

  cat210 <- sum(allelesCombined == '2,10')

  cat211 <- sum(allelesCombined == '2,11')

  cat212 <- sum(allelesCombined == '2,12')
  cat213 <- sum(allelesCombined == '2,13')

  cat214 <- sum(allelesCombined == '2,14')

  cat215 <- sum(allelesCombined == '2,15')

  cat216 <- sum(allelesCombined == '2,16')

  cat217 <- sum(allelesCombined == '2,17')

  cat218 <- sum(allelesCombined == '2,18')

  cat219 <- sum(allelesCombined == '2,19')

  cat220 <- sum(allelesCombined == '2,20')

  cat221 <- sum(allelesCombined == '2,21')

  cat222 <- sum(allelesCombined == '2,22')

  cat223 <- sum(allelesCombined == '2,23')

  cat224 <- sum(allelesCombined == '2,24')
```

```
  countsCol2 <-
cbind(c(cat20,cat21,cat22,cat23,cat24,cat25,cat26,cat27,cat28,cat29,cat210,cat211,cat212,ca
t213,cat214,cat215,cat216,cat217,cat218,cat219,cat220,cat221,cat222,cat223,cat224))



  cat30 <- sum(allelesCombined == '3,0')

  cat31 <- sum(allelesCombined == '3,1')

  cat32 <- sum(allelesCombined == '3,2')

  cat33 <- sum(allelesCombined == '3,3')

  cat34 <- sum(allelesCombined == '3,4')

  cat35 <- sum(allelesCombined == '3,5')

  cat36 <- sum(allelesCombined == '3,6')

  cat37 <- sum(allelesCombined == '3,7')

  cat38 <- sum(allelesCombined == '3,8')

  cat39 <- sum(allelesCombined == '3,9')

  cat310 <- sum(allelesCombined == '3,10')

  cat311 <- sum(allelesCombined == '3,11')

  cat312 <- sum(allelesCombined == '3,12')

  cat313 <- sum(allelesCombined == '3,13')

  cat314 <- sum(allelesCombined == '3,14')

  cat315 <- sum(allelesCombined == '3,15')

  cat316 <- sum(allelesCombined == '3,16')

  cat317 <- sum(allelesCombined == '3,17')

  cat318 <- sum(allelesCombined == '3,18')
  cat319 <- sum(allelesCombined == '3,19')

  cat320 <- sum(allelesCombined == '3,20')

  cat321 <- sum(allelesCombined == '3,21')

  cat322 <- sum(allelesCombined == '3,22')

  cat323 <- sum(allelesCombined == '3,23')

  cat324 <- sum(allelesCombined == '3,24')

  countsCol3 <-
cbind(c(cat30,cat31,cat32,cat33,cat34,cat35,cat36,cat37,cat38,cat39,cat310,cat311,cat312,ca
t313,cat314,cat315,cat316,cat317,cat318,cat319,cat320,cat321,cat322,cat323,cat324))



  cat40 <- sum(allelesCombined == '4,0')

  cat41 <- sum(allelesCombined == '4,1')

  cat42 <- sum(allelesCombined == '4,2')
```

```
  cat43 <- sum(allelesCombined == '4,3')

  cat44 <- sum(allelesCombined == '4,4')

  cat45 <- sum(allelesCombined == '4,5')

  cat46 <- sum(allelesCombined == '4,6')

  cat47 <- sum(allelesCombined == '4,7')

  cat48 <- sum(allelesCombined == '4,8')

  cat49 <- sum(allelesCombined == '4,9')

  cat410 <- sum(allelesCombined == '4,10')

  cat411 <- sum(allelesCombined == '4,11')

  cat412 <- sum(allelesCombined == '4,12')

  cat413 <- sum(allelesCombined == '4,13')

  cat414 <- sum(allelesCombined == '4,14')

  cat415 <- sum(allelesCombined == '4,15')

  cat416 <- sum(allelesCombined == '4,16')

  cat417 <- sum(allelesCombined == '4,17')

  cat418 <- sum(allelesCombined == '4,18')

  cat419 <- sum(allelesCombined == '4,19')

  cat420 <- sum(allelesCombined == '4,20')

  cat421 <- sum(allelesCombined == '4,21')

  cat422 <- sum(allelesCombined == '4,22')

  cat423 <- sum(allelesCombined == '4,23')

  cat424 <- sum(allelesCombined == '4,24')
  countsCol4 <-
cbind(c(cat40,cat41,cat42,cat43,cat44,cat45,cat46,cat47,cat48,cat49,cat410,cat411,cat412,ca
t413,cat414,cat415,cat416,cat417,cat418,cat419,cat420,cat421,cat422,cat423,cat424))



  cat50 <- sum(allelesCombined == '5,0')

  cat51 <- sum(allelesCombined == '5,1')

  cat52 <- sum(allelesCombined == '5,2')

  cat53 <- sum(allelesCombined == '5,3')

  cat54 <- sum(allelesCombined == '5,4')

  cat55 <- sum(allelesCombined == '5,5')

  cat56 <- sum(allelesCombined == '5,6')

  cat57 <- sum(allelesCombined == '5,7')

  cat58 <- sum(allelesCombined == '5,8')
```

```
  cat59 <- sum(allelesCombined == '5,9')

  cat510 <- sum(allelesCombined == '5,10')

  cat511 <- sum(allelesCombined == '5,11')

  cat512 <- sum(allelesCombined == '5,12')

  cat513 <- sum(allelesCombined == '5,13')

  cat514 <- sum(allelesCombined == '5,14')

  cat515 <- sum(allelesCombined == '5,15')

  cat516 <- sum(allelesCombined == '5,16')

  cat517 <- sum(allelesCombined == '5,17')

  cat518 <- sum(allelesCombined == '5,18')

  cat519 <- sum(allelesCombined == '5,19')

  cat520 <- sum(allelesCombined == '5,20')

  cat521 <- sum(allelesCombined == '5,21')

  cat522 <- sum(allelesCombined == '5,22')

  cat523 <- sum(allelesCombined == '5,23')

  cat524 <- sum(allelesCombined == '5,24')

  countsCol5 <-
cbind(c(cat50,cat51,cat52,cat53,cat54,cat55,cat56,cat57,cat58,cat59,cat510,cat511,cat512,ca
t513,cat514,cat515,cat516,cat517,cat518,cat519,cat520,cat521,cat522,cat523,cat524))



  cat60 <- sum(allelesCombined == '6,0')

  cat61 <- sum(allelesCombined == '6,1')

  cat62 <- sum(allelesCombined == '6,2')
  cat63 <- sum(allelesCombined == '6,3')

  cat64 <- sum(allelesCombined == '6,4')

  cat65 <- sum(allelesCombined == '6,5')

  cat66 <- sum(allelesCombined == '6,6')

  cat67 <- sum(allelesCombined == '6,7')

  cat68 <- sum(allelesCombined == '6,8')

  cat69 <- sum(allelesCombined == '6,9')

  cat610 <- sum(allelesCombined == '6,10')

  cat611 <- sum(allelesCombined == '6,11')

  cat612 <- sum(allelesCombined == '6,12')

  cat613 <- sum(allelesCombined == '6,13')

  cat614 <- sum(allelesCombined == '6,14')
```

```
  cat615 <- sum(allelesCombined == '6,15')

  cat616 <- sum(allelesCombined == '6,16')

  cat617 <- sum(allelesCombined == '6,17')

  cat618 <- sum(allelesCombined == '6,18')

  cat619 <- sum(allelesCombined == '6,19')

  cat620 <- sum(allelesCombined == '6,20')

  cat621 <- sum(allelesCombined == '6,21')

  cat622 <- sum(allelesCombined == '6,22')

  cat623 <- sum(allelesCombined == '6,23')

  cat624 <- sum(allelesCombined == '6,24')

  countsCol6 <-
cbind(c(cat60,cat61,cat62,cat63,cat64,cat65,cat66,cat67,cat68,cat69,cat610,cat611,cat612,ca
t613,cat614,cat615,cat616,cat617,cat618,cat619,cat620,cat621,cat622,cat623,cat624))



  cat70 <- sum(allelesCombined == '7,0')

  cat71 <- sum(allelesCombined == '7,1')

  cat72 <- sum(allelesCombined == '7,2')

  cat73 <- sum(allelesCombined == '7,3')

  cat74 <- sum(allelesCombined == '7,4')

  cat75 <- sum(allelesCombined == '7,5')

  cat76 <- sum(allelesCombined == '7,6')

  cat77 <- sum(allelesCombined == '7,7')

  cat78 <- sum(allelesCombined == '7,8')
  cat79 <- sum(allelesCombined == '7,9')

  cat710 <- sum(allelesCombined == '7,10')

  cat711 <- sum(allelesCombined == '7,11')

  cat712 <- sum(allelesCombined == '7,12')

  cat713 <- sum(allelesCombined == '7,13')

  cat714 <- sum(allelesCombined == '7,14')

  cat715 <- sum(allelesCombined == '7,15')

  cat716 <- sum(allelesCombined == '7,16')

  cat717 <- sum(allelesCombined == '7,17')

  cat718 <- sum(allelesCombined == '7,18')

  cat719 <- sum(allelesCombined == '7,19')

  cat720 <- sum(allelesCombined == '7,20')
```

```
  cat721 <- sum(allelesCombined == '7,21')

  cat722 <- sum(allelesCombined == '7,22')

  cat723 <- sum(allelesCombined == '7,23')

  cat724 <- sum(allelesCombined == '7,24')

  countsCol7 <-
cbind(c(cat70,cat71,cat72,cat73,cat74,cat75,cat76,cat77,cat78,cat79,cat710,cat711,cat712,ca
t713,cat714,cat715,cat716,cat717,cat718,cat719,cat720,cat721,cat722,cat723,cat724))



  cat80 <- sum(allelesCombined == '8,0')

  cat81 <- sum(allelesCombined == '8,1')

  cat82 <- sum(allelesCombined == '8,2')

  cat83 <- sum(allelesCombined == '8,3')

  cat84 <- sum(allelesCombined == '8,4')

  cat85 <- sum(allelesCombined == '8,5')

  cat86 <- sum(allelesCombined == '8,6')

  cat87 <- sum(allelesCombined == '8,7')

  cat88 <- sum(allelesCombined == '8,8')

  cat89 <- sum(allelesCombined == '8,9')

  cat810 <- sum(allelesCombined == '8,10')

  cat811 <- sum(allelesCombined == '8,11')

  cat812 <- sum(allelesCombined == '8,12')

  cat813 <- sum(allelesCombined == '8,13')

  cat814 <- sum(allelesCombined == '8,14')
  cat815 <- sum(allelesCombined == '8,15')

  cat816 <- sum(allelesCombined == '8,16')

  cat817 <- sum(allelesCombined == '8,17')

  cat818 <- sum(allelesCombined == '8,18')

  cat819 <- sum(allelesCombined == '8,19')

  cat820 <- sum(allelesCombined == '8,20')

  cat821 <- sum(allelesCombined == '8,21')

  cat822 <- sum(allelesCombined == '8,22')

  cat823 <- sum(allelesCombined == '8,23')

  cat824 <- sum(allelesCombined == '8,24')

  countsCol8 <-
cbind(c(cat80,cat81,cat82,cat83,cat84,cat85,cat86,cat87,cat88,cat89,cat810,cat811,cat812,ca
t813,cat814,cat815,cat816,cat817,cat818,cat819,cat820,cat821,cat822,cat823,cat824))
```

```
  cat90 <- sum(allelesCombined == '9,0')

  cat91 <- sum(allelesCombined == '9,1')

  cat92 <- sum(allelesCombined == '9,2')

  cat93 <- sum(allelesCombined == '9,3')

  cat94 <- sum(allelesCombined == '9,4')

  cat95 <- sum(allelesCombined == '9,5')

  cat96 <- sum(allelesCombined == '9,6')

  cat97 <- sum(allelesCombined == '9,7')

  cat98 <- sum(allelesCombined == '9,8')

  cat99 <- sum(allelesCombined == '9,9')

  cat910 <- sum(allelesCombined == '9,10')

  cat911 <- sum(allelesCombined == '9,11')

  cat912 <- sum(allelesCombined == '9,12')

  cat913 <- sum(allelesCombined == '9,13')

  cat914 <- sum(allelesCombined == '9,14')

  cat915 <- sum(allelesCombined == '9,15')

  cat916 <- sum(allelesCombined == '9,16')

  cat917 <- sum(allelesCombined == '9,17')

  cat918 <- sum(allelesCombined == '9,18')

  cat919 <- sum(allelesCombined == '9,19')

  cat920 <- sum(allelesCombined == '9,20')
  cat921 <- sum(allelesCombined == '9,21')

  cat922 <- sum(allelesCombined == '9,22')

  cat923 <- sum(allelesCombined == '9,23')

  cat924 <- sum(allelesCombined == '9,24')

  countsCol9 <-
cbind(c(cat90,cat91,cat92,cat93,cat94,cat95,cat96,cat97,cat98,cat99,cat910,cat911,cat912,ca
t913,cat914,cat915,cat916,cat917,cat918,cat919,cat920,cat921,cat922,cat923,cat924))



  cat100 <- sum(allelesCombined == '10,0')

  cat101 <- sum(allelesCombined == '10,1')

  cat102 <- sum(allelesCombined == '10,2')

  cat103 <- sum(allelesCombined == '10,3')

  cat104 <- sum(allelesCombined == '10,4')

  cat105 <- sum(allelesCombined == '10,5')
```

```
  cat106 <- sum(allelesCombined == '10,6')

  cat107 <- sum(allelesCombined == '10,7')

  cat108 <- sum(allelesCombined == '10,8')

  cat109 <- sum(allelesCombined == '10,9')

  cat1010 <- sum(allelesCombined == '10,10')

  cat1011 <- sum(allelesCombined == '10,11')

  cat1012 <- sum(allelesCombined == '10,12')

  cat1013 <- sum(allelesCombined == '10,13')

  cat1014 <- sum(allelesCombined == '10,14')

  cat1015 <- sum(allelesCombined == '10,15')

  cat1016 <- sum(allelesCombined == '10,16')

  cat1017 <- sum(allelesCombined == '10,17')

  cat1018 <- sum(allelesCombined == '10,18')

  cat1019 <- sum(allelesCombined == '10,19')

  cat1020 <- sum(allelesCombined == '10,20')

  cat1021 <- sum(allelesCombined == '10,21')

  cat1022 <- sum(allelesCombined == '10,22')

  cat1023 <- sum(allelesCombined == '10,23')

  cat1024 <- sum(allelesCombined == '10,24')

  countsCol10 <-
cbind(c(cat100,cat101,cat102,cat103,cat104,cat105,cat106,cat107,cat108,cat109,cat1010,cat10
11,cat1012,cat1013,cat1014,cat1015,cat1016,cat1017,cat1018,cat1019,cat1020,cat1021,cat1022,
cat1023,cat1024))



  cat110 <- sum(allelesCombined == '11,0')

  cat111 <- sum(allelesCombined == '11,1')

  cat112 <- sum(allelesCombined == '11,2')

  cat113 <- sum(allelesCombined == '11,3')

  cat114 <- sum(allelesCombined == '11,4')

  cat115 <- sum(allelesCombined == '11,5')

  cat116 <- sum(allelesCombined == '11,6')

  cat117 <- sum(allelesCombined == '11,7')

  cat118 <- sum(allelesCombined == '11,8')

  cat119 <- sum(allelesCombined == '11,9')

  cat1110 <- sum(allelesCombined == '11,10')
```

```
  cat1111 <- sum(allelesCombined == '11,11')

  cat1112 <- sum(allelesCombined == '11,12')

  cat1113 <- sum(allelesCombined == '11,13')

  cat1114 <- sum(allelesCombined == '11,14')

  cat1115 <- sum(allelesCombined == '11,15')

  cat1116 <- sum(allelesCombined == '11,16')

  cat1117 <- sum(allelesCombined == '11,17')

  cat1118 <- sum(allelesCombined == '11,18')

  cat1119 <- sum(allelesCombined == '11,19')

  cat1120 <- sum(allelesCombined == '11,20')

  cat1121 <- sum(allelesCombined == '11,21')

  cat1122 <- sum(allelesCombined == '11,22')

  cat1123 <- sum(allelesCombined == '11,23')

  cat1124 <- sum(allelesCombined == '11,24')

  countsCol11 <-
cbind(c(cat110,cat111,cat112,cat113,cat114,cat115,cat116,cat117,cat118,cat119,cat1110,cat11
11,cat1112,cat1113,cat1114,cat1115,cat1116,cat1117,cat1118,cat1119,cat1120,cat1121,cat1122,
cat1123,cat1124))



  cat120 <- sum(allelesCombined == '12,0')

  cat121 <- sum(allelesCombined == '12,1')

  cat122 <- sum(allelesCombined == '12,2')
  cat123 <- sum(allelesCombined == '12,3')

  cat124 <- sum(allelesCombined == '12,4')

  cat125 <- sum(allelesCombined == '12,5')

  cat126 <- sum(allelesCombined == '12,6')

  cat127 <- sum(allelesCombined == '12,7')

  cat128 <- sum(allelesCombined == '12,8')

  cat129 <- sum(allelesCombined == '12,9')

  cat1210 <- sum(allelesCombined == '12,10')

  cat1211 <- sum(allelesCombined == '12,11')

  cat1212 <- sum(allelesCombined == '12,12')

  cat1213 <- sum(allelesCombined == '12,13')

  cat1214 <- sum(allelesCombined == '12,14')

  cat1215 <- sum(allelesCombined == '12,15')

  cat1216 <- sum(allelesCombined == '12,16')
```

```
  cat1217 <- sum(allelesCombined == '12,17')

  cat1218 <- sum(allelesCombined == '12,18')

  cat1219 <- sum(allelesCombined == '12,19')

  cat1220 <- sum(allelesCombined == '12,20')

  cat1221 <- sum(allelesCombined == '12,21')

  cat1222 <- sum(allelesCombined == '12,22')

  cat1223 <- sum(allelesCombined == '12,23')

  cat1224 <- sum(allelesCombined == '12,24')

  countsCol12 <-
cbind(c(cat120,cat121,cat122,cat123,cat124,cat125,cat126,cat127,cat128,cat129,cat1210,cat12
11,cat1212,cat1213,cat1214,cat1215,cat1216,cat1217,cat1218,cat1219,cat1220,cat1221,cat1222,
cat1223,cat1224))



  counts <-
cbind(countsCol0,countsCol1,countsCol2,countsCol3,countsCol4,countsCol5,countsCol6,countsCo
l7,countsCol8,countsCol9,countsCol10,countsCol11,countsCol12)

  return(counts)

}

#plot deme with 6 ind on x, with 12 on y

#fuchsii side, only one shown

tauF.tcoF <- read.table(file="210203_resultsF/dataForSFS_6tauF_12tcoF.txt",head=F)

head(tauF.tcoF)

tauF <- tauF.tcoF[,3:8]

tcoF <- tauF.tcoF[,9:20]

jSFS.tauF.tcoF <- jSFS(dataDemeA=tauF,dataDemeB=tcoF)

write.table(jSFS.tauF.tcoF,file="210203_resultsF/jSFS_6tauF_12tcoF.txt",

            sep='\t',row.names=F,col.names=F)

#same procedure for the incarnata side, not shown here
####################



########## jSFS_matrixPlots.R ##########
########## R script plotting jSFS ##########

setwd("C:/R_files/DactPhylogenomics/201203_SFSscript/")

#install and load required packages

install.packages("gplots")

library(gplots)

install.packages("viridis")
```

```
library(viridis)

#fuchsii side, only one shown

#demeA gets plotted on the y-axis, demeB on the x-axis

jSFS.fuc.majF <- read.table(file="210203_resultsF/jSFS_12fuc_12majF.txt",head=F)


#plot all jSFS log10 scale

pdf(file="210203_resultsF/210802_jSFSplots_polyplFFLog10.pdf",paper="a4r",width=8,height=8)

jSFS.fuc.majF.matrix <- as.matrix(jSFS.fuc.majF)

jSFS.fuc.majF.matrix[jSFS.fuc.majF.matrix==0] <- NA

colourrange = c(seq(0,1,length=100),

                seq(1.01,2,length=100),

                seq(2.01,3,length=100))

heatmap.2(log10(jSFS.fuc.majF.matrix),

          xlab=expression(bold("fuc")),ylab=expression(bold("majF")),

          Rowv=NA,Colv=NA,

          col=viridis(3*100-1,begin=1,end=0),

          #col=rainbow(1000,start=0.1,end=1),

          breaks=colourrange,

          key=T,key.title="number of loci",density.info="none",

          key.xlab=NA,key.ylab=NA,keysize=1,

          labRow=NA,labCol=NA,

          trace="none",dendrogram="none")

dev.off()

#same procedure for the incarnata side, not shown here
####################
```