AUG 14, 2023

OPEN ACCESS

NucBarcoder - a bioinformatic pipeline to characterise the genetic basis of plant species differences

Wu Huang[1], Alex Twyford[1], Peter Hollingsworth[1]

[1]Royal Botanic Garden Edinburgh, University of Edinburgh

Wu Huang: Corresponding email: hazelhuang1993@gmail.com

Wu Huang

University of Edinburgh, Royal Botanic Garden Edinburgh

Protocol Citation: Wu Huang, Alex Twyford, Peter Hollingsworth 2023. NucBarcoder - a bioinformatic pipeline to characterise the genetic basis of plant species differences. **protocols.io** https://protocols.io/view/nucbarcoder-a-bioinformatic-pipeline-to-characteri-cxqwxmxe

Protocol status: Working
We use this protocol and it's working

Created: Jul 26, 2023

Last Modified: Aug 14, 2023

DISCLAIMER

PROTOCOL integer ID:
85494

Keywords: Plant genomics, genetic differences among species, bioinformatic pipeline, multi-locus DNA barcoding

ABSTRACT

There is now a significant number of studies that have sequenced multiple loci from the nuclear genomes of plants and these are available in public databases or in private data repositories (if the study hasn't yet been published). Collectively these datasets have great potential for understanding the nature of genomic differences between plant species. To harness this existing information I have developed a set of workflows, with the logic behind my approach being to

● Develop key criteria for selecting suitable datasets focusing only on datasets which have sampled multiple individuals from multiple congeneric species

● Search the literature and public data repositories for datasets that match these criteria

● Obtain and organise the selected datasets

● Use these datasets to estimate the proportion of species that resolve as monophyletic units as one measure of species discrimination success

● Establish the frequency distribution of taxonomically informative nucleotide substitutions among taxa to better understand the genomic nature of differences between plant species

● Subsample the data to better understand the effectiveness of smaller numbers of loci in recovering maximal species discrimination, as well as evaluating the attributes of the gene regions that are most effective at telling species apart.

This protocol will focus on the last three bullet points. For the protocol for the first three bullet points please refer to another protocol "Acquiring and integrating data from multiple resources for meta-analysis".

GUIDELINES

All data used in this study are stored and run at the UK Crop Diversity platform (www.cropdiversity.ac.uk).
Python (version 3+) and bash shell scripts were used to process the data.
The R Statistical Programing Language (version 4.1.0) and the Rstudio integrated development environment (Racine, 2012) were used for most of the plotting along with Excel (version 16.58) charts.
All code scripts, workflow, and an example to run the pipeline are available at https://github.com/Hazelhuangup/Species_specific_alleles_analysis.
All data used in this study are stored and run at the UK Crop Diversity platform (www.cropdiversity.ac.uk).
Python (version 3+) and bash shell scripts were used to process the data.
The R Statistical Programing Language (version 4.1.0) and the Rstudio integrated development environment (Racine, 2012) were used for most of the plotting along with Excel (version 16.58) charts.
All code scripts, workflow, and an example to run the pipeline are available at https://github.com/Hazelhuangup/NucBarcoder/tree/main/src.

Overview of the data management and the structure of Nuc..

1

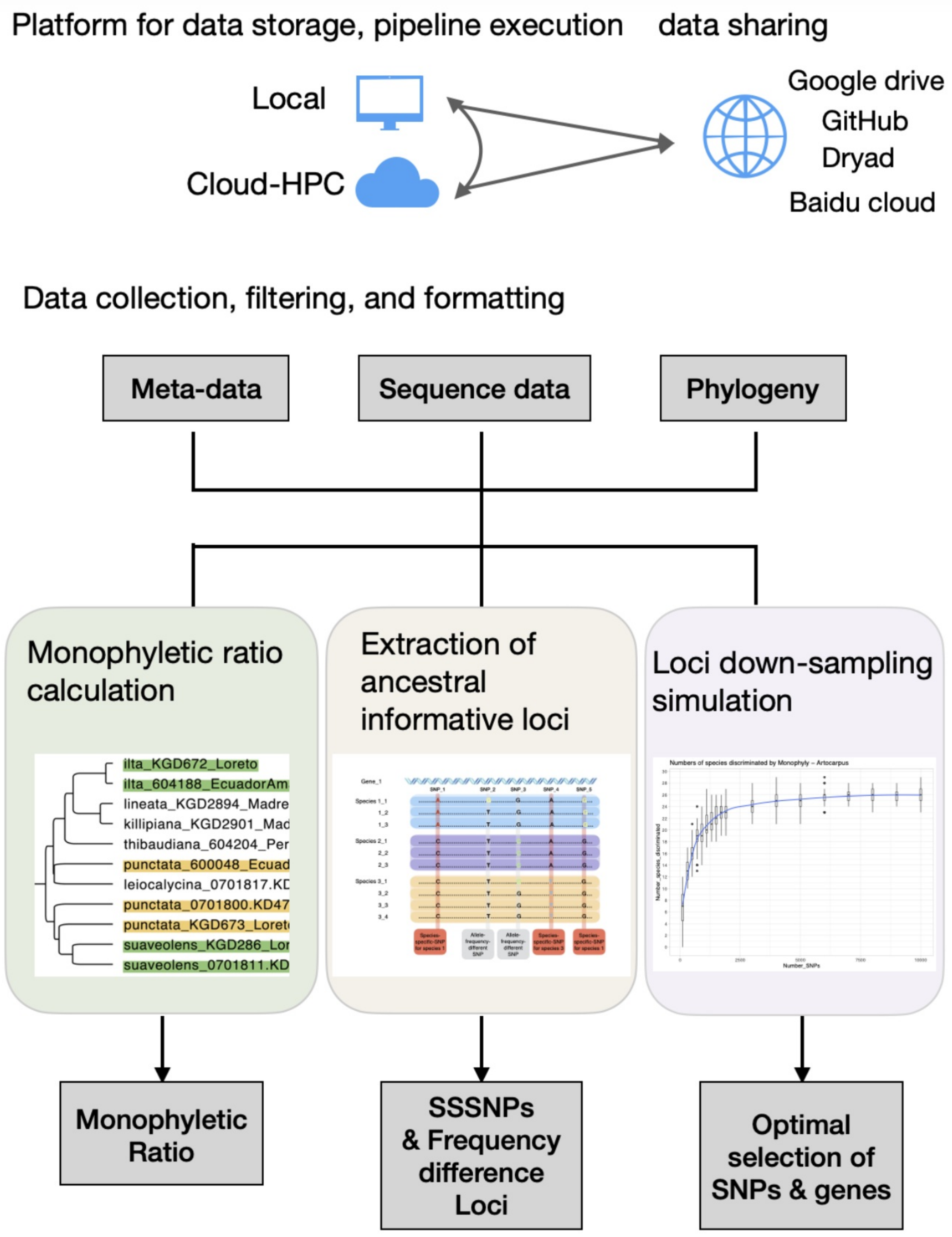


Figure 1. Overview of the data management and the structure of NucBarcoder. The data storage and pipeline execution are performed on both a local computer and a High-Performance Computing system via cloud service. The original data from public and private providers is deposited and downloaded from Google Drive, Dryad, and Baidu cloud. The data underwent format tidying up and filtering locally and was then uploaded to HPC for intensive processing and analytical tasks. The Nucbarcoder pipeline comprises three main parts: Monophyletic ratio calculation, Extraction of ancestry informative loci, and loci down-sampling simulation (sub-sampling of the data).

Data cleaning and filtering

Data cleaning and filtering were performed at both the individual and locus levels.

2 Different clean-up and filtering tools were used for aligned sequence formats and SNP matrices. VCFtools (0.1.17) (Danecek et al., 2011) was used to deal with the vcf format and self-written scripts were used for parsing the aligned sequences files, including format conversion, removing individuals, and removing sites. The main steps are listed in Table 2.

Data format	Tools or scripts	Useful parameters and their functions	Usage and description
vcf	vcftools	--remove --min-meanDP --max-meanDP --minDP --max-missing --remove-indels --min-alleles	Specify individuals to remove from the dataset; Remove sites by read depth and missing data rate; Remove all insertions and deletions; Specify to include only bi-allelic sites.
fasta, phylip, nex	rm_site_hign_N_rate.py	default	Remove sites based on missing data rate >40% (adaptable);
	fa2phy.py nex2fa.py phy2fa_oneline.py formatting_ipyrad_alleles2fasta.py	default	Convert the file format for specific usage.
	give_me_mul_seq.py	--seqfile --seqlist	Run the scripts with the command 'give_me_mul_seq.py -f input.fasta -l IDs_to_keep -o output.fasta'

Table 2. Tools or scripts used for data cleanup and filtering

2.1 Upon acquiring each dataset, we examined how taxon re-identifications were performed and any ambiguous samples were removed based on the information given in the metadata of collaborators and in publications. Any individuals with unresolved species names (including cf., aff., and other uncategorised naming methods) and hybrids were removed from the matrix. Individuals that were outside of the focal genus of each dataset were removed except for the purpose of acting as outgroups in phylogenetic analyses. Multiple varieties or subspecies in one species were treated as a single taxon at the species level. Individuals with a high proportion of missing data were also removed, the threshold ranges from 75 ~ 80% data presence according to the data quality after manual checking. Further details on the removal of individuals and why they were removed from each genus are provided in Table 2.

2.2 After the filtering of individuals, we undertook locus filtering. For target capture or transcriptome data where genomic segment information is available, loci that were missing from 80% of all individuals were deleted. This also applies to RAD-seq/GBS data where stack or assembly information on each locus was provided. For datasets without genomic segment information, such as vcf and concatenated consensus fasta files, a nucleotide site was removed when over 40% of the individuals are missing. When the depth and quality information was accessible, usually in vcf format, nucleotide sites were retained only when the depth is within a reasonable range (lower depth limit of 2 to an upper limit of 2 to 3 times the average sequencing depth) according to the sequencing depth reported by the collaborators.

2.3 The programming scripts used in Step 1 could be found at https://github.com/Hazelhuangup/NucBarcoder/tree/main/src.
An example of using VCFtools:

Command

Filtering data (VCFtools 0.1.17)

```
vcftools
--gzvcf ./01_raw_data/allsamples121217_raw_m.vcf.gz
--remove ./01_raw_data/ind_to_be_removed_from_vcf_file.list
--remove-indels
--min-alleles 2
--min-meanDP 3
--max-meanDP 60
--minDP 2
--recode
--max-missing 0.6
--out ./01_raw_data/Antirrhinum_SNP_minmeanDP3-60_meanDP2_max-missing0.6
```

Command

Get multiple sequences by sequence ID (give_me_mul_seq.py)

```
give_me_mul_seq.py
-f input.fasta
-l IDs_to_keep
-o output.fasta
```

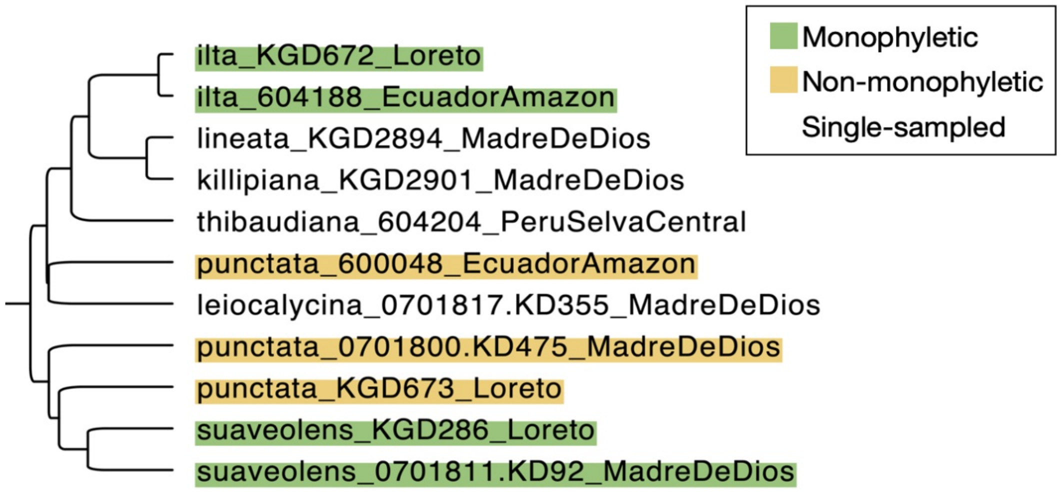
Command

Convert the file format for specific usage

```
formatting_ipyrad_alleles2fasta.py input_file output_file
```

Monophyletic Ratio

3 The Monophyletic Ratio (MR) was calculated as a simple measure of species discrimination success from different data sets. This is defined as the number of species resolved as monophyletic on a given phylogeny divided by the total number of multiple-sampled species (individuals ≥2 samples per species) from a genus. Figure 3.6. demonstrates an example where 3 species have multiple sampled individuals, 2 of them are monophyletic on the phylogeny. In this case, the monophyletic ratio is 2/3 = 0.67.



ilta_KGD672_Loreto
ilta_604188_EcuadorAmazon
lineata_KGD2894_MadreDeDios
killipiana_KGD2901_MadreDeDios
thibaudiana_604204_PeruSelvaCentral
punctata_600048_EcuadorAmazon
leiocalycina_0701817.KD355_MadreDeDios
punctata_0701800.KD475_MadreDeDios
punctata_KGD673_Loreto
suaveolens_KGD286_Loreto
suaveolens_0701811.KD92_MadreDeDios
Monophyletic
Non-monophyletic
Single-sampled

Figure 3. Phylogeny of the 11 individuals from the genus Inga featuring monophyletic and non-monophyletic taxa. The green highlights individuals from species Inga ilta and I. suaveoolens which
form monophyletic clades, and yellow highlights individuals from I. *puctata* that form a non-monophyletic clade.

3.1 If a phylogeny is in visualised format only, the identification could only be done by manual calculation.

3.2 With phylogenies provided in newick or other text-based formats, this process could be automated using MonoPhy (Schwery et al., 2016), which is a quick and user-friendly method for assessing the monophyly of taxa in a given phylogeny. MonoPhy builds on the existing packages ape 5.0, phytools (Revell, 2012), phangorn, RColorBrewer (https://CRAN.R-project.org/package=RColorBrewer) and taxize (Chamberlain et al., 2013), and the installation of these packages is also required.

Data format	Tools or scripts	Parameters	Usage and package requirements
rooted newick tree ID corresponding file	MonoPhy	default	Rscript MonoPhy.r Input.treefile outgroup ID output ape, phytools, phangorn, RColorBrewer, and taxize packages required

Table 3.2.1. Tools or scripts used for identifying monophyletic clades

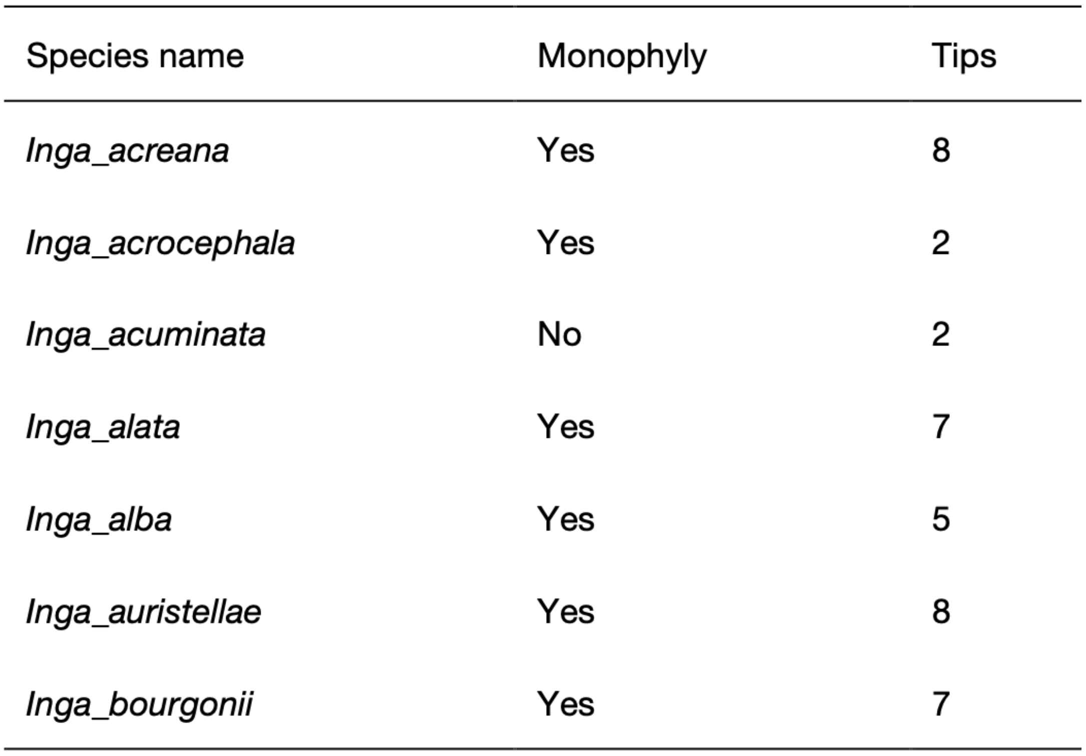
Command

Identify monophyletic clades based on Newick phylogeny and calculate the number automatically using MonoPhy

```
Rscript MonoPhy.r Input.treefile outgroup ID MonoPhy_output.txt


awk '$4>1{print $1"\t"$2"\t"$4}'  MonoPhy_output.txt |sort > MonoPhy_YN_result.txt
```

The output of MonoPhy indicates taxa are either monophyletic, non-monophyletic or monotypic. By counting the number of species that are scored “Yes” for ‘monophyly’, divided by the number of species with ‘tips’ > 1 (= species with multiple-samples), gives the Monophyletic Ratio. For the example shown in Table 3.5, the monophyletic ratio is 6/7 = 0.86. A tutorial on running MonoPhy and how to interpret the results is given at https://cran.r-project.org/web/packages/MonoPhy/vignettes/MonoPhyVignette.html.



Species name	Monophyly	Tips
Inga_acreana	Yes	8
Inga_acrocephala	Yes	2
Inga_acuminata	No	2
Inga_alata	Yes	7
Inga_alba	Yes	5
Inga_auristellae	Yes	8
Inga_bourgonii	Yes	7

Table 3.2.2. An example of the output of MonoPhy

Running the main analysis command ‘AssessMonophyly’ on 393 individuals using standard settings is very rapid, and used only 0.19 seconds on a MacBook Pro with 2.2 GHz Intel 6-Core i7 and 16 GB RAM. MonoPhy cannot be run in parallel by default.

4 Monophyletic clades were first counted from published phylogenies based on the data and format available. This involved either manual scoring of species resolving as monophyletic when only graphic representations of trees were available, and automated extraction of the MR for machine readable data (see section 3.4.3). Where phylogenies were not available for a given study, then we built a basic phylogeny using the aligned sequence data. Where sequence alignments were not available, we discarded these datasets due to the time constraints of producing high quality alignments from multiple different sources.

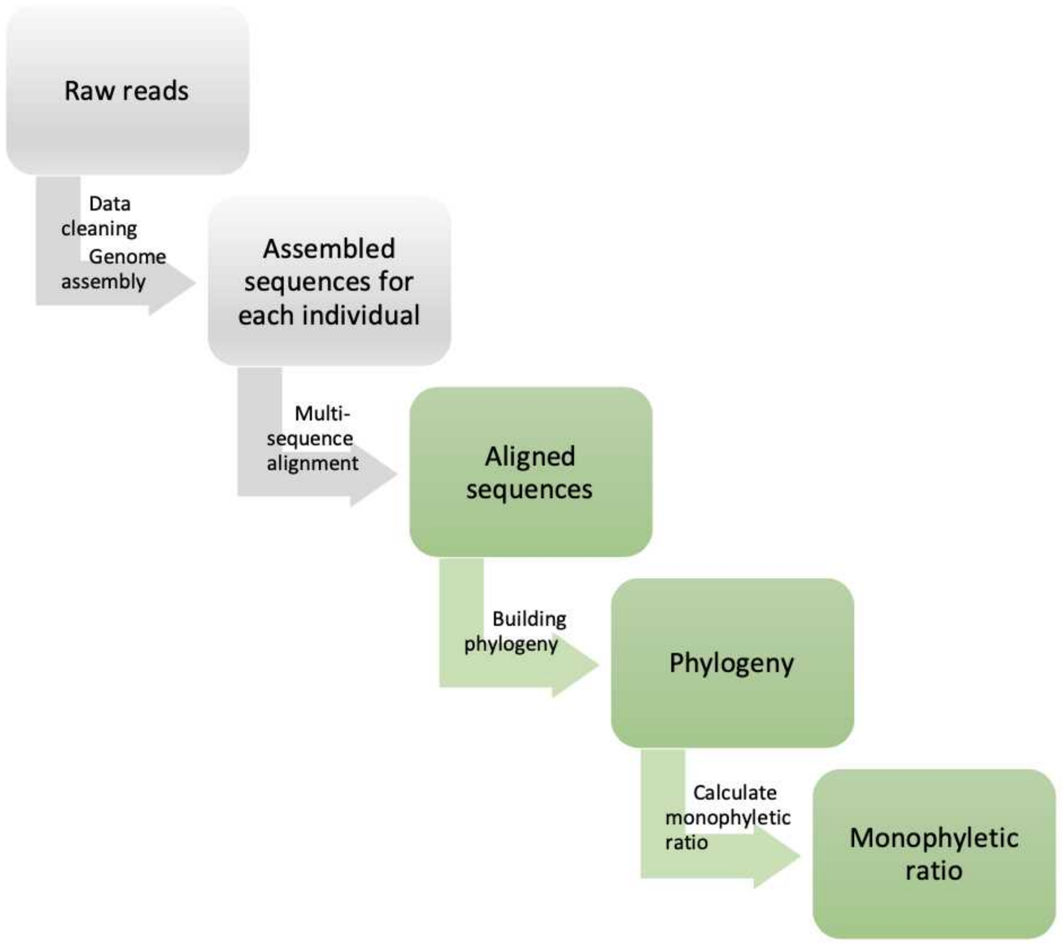
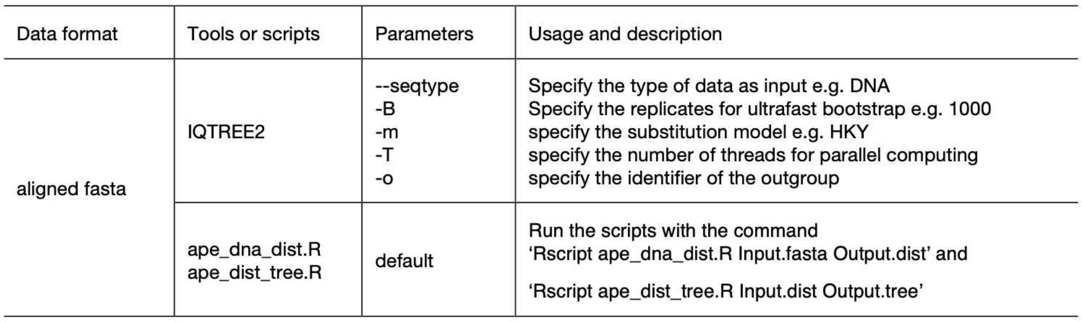


Figure 4. Steps to a successful monophyletic ratio calculation. Green colour blocks indicate where We accepted data, and grey blocks shows the stages where to discard datasets if these were the only data available.

4.1 We built phylogenies with the aligned sequences as input using IQTREE2 (Minh et al., 2020). IQTREE (Nguyen et al., 2015) is a widely used software package for phylogenetic inference using maximum likelihood. The second version incorporates Polymorphism-aware phylogenetic models (PoMo) to parse the IUPAC Ambiguity Codes (Schrempf et al., 2016), which is useful when the dataset maintains heterozygosity information. We also built quick UPGMA trees using R package ape v.5.0 (Paradis et al., 2019) and phangorn (Schliep, 2011). The test data for evaluating the tree-building pipeline consisted of 393 individuals with 1,313,489 base pairs per individual. To build a tree using this dataset, IQTREE2 requires a minimum of 7 Gb RAM and 131 hours CPU time. With 64 CPU for parallel computing, it took a total of 2 hours and 34 minutes to finish the task. Building quick trees with ape requires less computing power because it doesn't have a model testing and bootstrap stage. Running the whole dataset to build a UPGMA tree took 34 minutes CPU time.



Data format	Tools or scripts	Parameters	Usage and description
aligned fasta	IQTREE2	--seqtype -B -m -T -o	Specify the type of data as input e.g. DNA Specify the replicates for ultrafast bootstrap e.g. 1000 specify the substitution model e.g. HKY specify the number of threads for parallel computing specify the identifier of the outgroup
	ape_dna_dist.R ape_dist_tree.R	default	Run the scripts with the command 'Rscript ape_dna_dist.R Input.fasta Output.dist' and 'Rscript ape_dist_tree.R Input.dist Output.tree'

Table 4.1. Tools or scripts used for building phylogeny

Command

Build phylogeny with iqtree2 (IQ-TREE multicore version 2.1.2)

```
iqtree2  ##500M data 2h34min * 64 CPU ^11G
-s 04_phylogeny/Inga_810genes_293inds_concat.fasta
--seqtype DNA
-B 1000
-m HKY
--prefix 04_phylogeny/Inga_810genes_293inds_concat
-T 64
-o Zygia_917
```

The extraction of taxonomically informative Loci (Species-s…

5 To provide a quantification of the distribution of taxonomically informative polymorphism, we extracted information on the frequency distribution of ancestry informative loci that were either fixed (Species-specific) or with marked nucleotide frequency differences.
Specifically, we divided SNPs into two categories: a) species-specific-SNPs which are fixed in all individuals from one species and distinct from all other ingroup species, and b) Frequency different SNPs where SNPs show a significant allele frequency difference in one species in comparison to other groups.

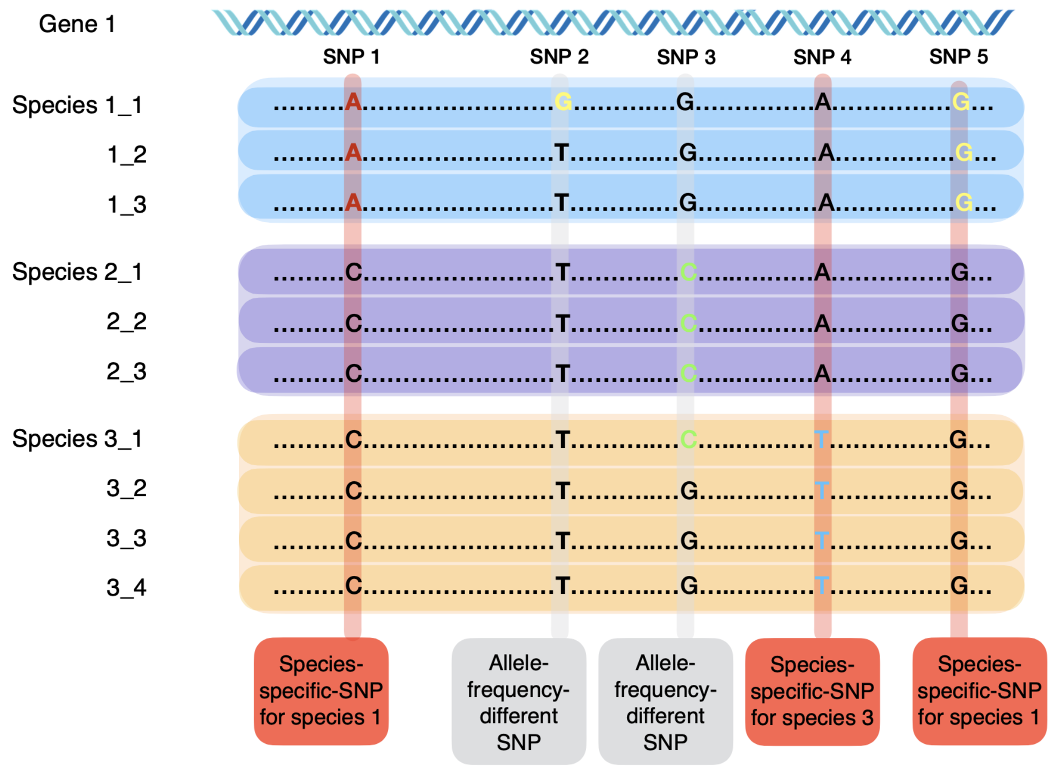

Gene 1
SNP 1
SNP 2
SNP 3
SNP 4
SNP 5
Species 1_1
1_2
1_3
Species 2_1
2_2
2_3
Species 3_1
3_2
3_3
3_4
.........A.......................G.............G...............A...............G...
.........A.......................T.............G...............A...............G...
.........A.......................T.............G...............A...............G...
.........C.......................T.............C...............A...............G...
.........C.......................T.............C...............A...............G...
.........C.......................T.............C...............A...............G...
.........C.......................T.............C...............T...............G...
.........C.......................T.............G...............T...............G...
.........C.......................T.............G...............T...............G...
.........C.......................T.............G...............T...............G...
Species-specific-SNP for species 1
Allele-frequency-different SNP
Allele-frequency-different SNP
Species-specific-SNP for species 3
Species-specific-SNP for species 1

Figure 5. Diagram of two categories of polymorphic sites. SNPs (1,4,5) that are highlighted in red are species-specific, SNPs (2,3) that are in grey are SNPs with allele frequency differences.

5.1 To extract both types of SNPs from the aligned sequenced files, a Python script extract_ancestral_informative_SNPs*.py was developed with the following steps:

1) For each locus, there should be at least 6 taxa whose locus coverages are greater than one (less than ~80% missing data)

2) Calculate allele frequency for the target taxon. If the allele frequency (AF) of the major allele (M_A) is higher than 87.5%, progress to stage 3).

3) Calculate allele frequency for the M_A across all other taxa (aggregated). If less than the 10% threshold then proceed to 4).

4) Calculate allele frequency across other species with multi-sampled populations. If the number of individuals in this group is greater than 4 (included), and A_M is lower than 12.5% (at most only 1/8 allele belongs to the minor one), then retain the site. Otherwise, only if the minor alleles are homogeneous in a taxon would this site be retained.

The pipeline also includes the following extra calculations steps:

5) Count the number of SSSNPs for each species that has multiple sampled individuals (e.g. the number of SNPs that are fixed and different compared to all other sampled species).

6) Calculate the density of SSSNPs by dividing the number of SSSNPs by the average total nucleotides that do not contain missing data in that species.

Scripts for extracting ancestry informative SNPs are available in two versions, one is for the situation where the input is in vcf format (extract_ancestral_informative_SNPs_vcf.py), and one is for fasta, phylip, and nexus format (extract_ancestral_informative_SNPs_hete.py). Both of these two versions are heterozygous aware, and can accommodate heterozygosity in the designation of SSSNP calling.

Data format	Tools or scripts	Parameters	Description
vcf	extract_ancestral_informative_SNPs_vcf.py	-f --name --selectedspps --selectedsample	Specify the input file in vcf/fasta format. The vcf file need GT and DP columns Specify the sample ID to species names corresponding file Give a list of spps names that are multiple sampled Give a list of sample names that belongs to target genus (ID)
fasta, phylip, and nexus	extract_ancestral_informative_SNPs_hete.py		

Table 5.1. Tools or scripts used for extracting ancestral informative SNPs

5.2 An example:

Command

extract_ancestral_informative_SNPs

```
python ../00_src/extract_ancestral_informative_SNPs_hete.py
-f 01_raw_data/Ant_concat.fa
-n 01_raw_data/ID_to_scientific_name.txt
-sp 01_raw_data/selected_species.txt
-sm 01_raw_data/selected_samples.txt
-o 03_output/Ant
> Ant.midfile.txt
```

The essential output will be in the file *.AIL.list Example.AIL.list.txt

Calculate the number of SSSNPs and frequency different loci.

Command

basic statistical analysis and visualization of SSA

```
for i in `cat 01_raw_data/selected_species.txt`;do
    grep $i 03_output/Ant.AIL.list |awk '$4=="ssa"'|wc -l >>  03_output/All.AIL.SSA.txt
    grep $i 03_output/Ant.AIL.list |awk '$4=="."'|wc -l >>  03_output/All.AIL.freq_diff.txt
    done
```

And record these basic statistics into a table.

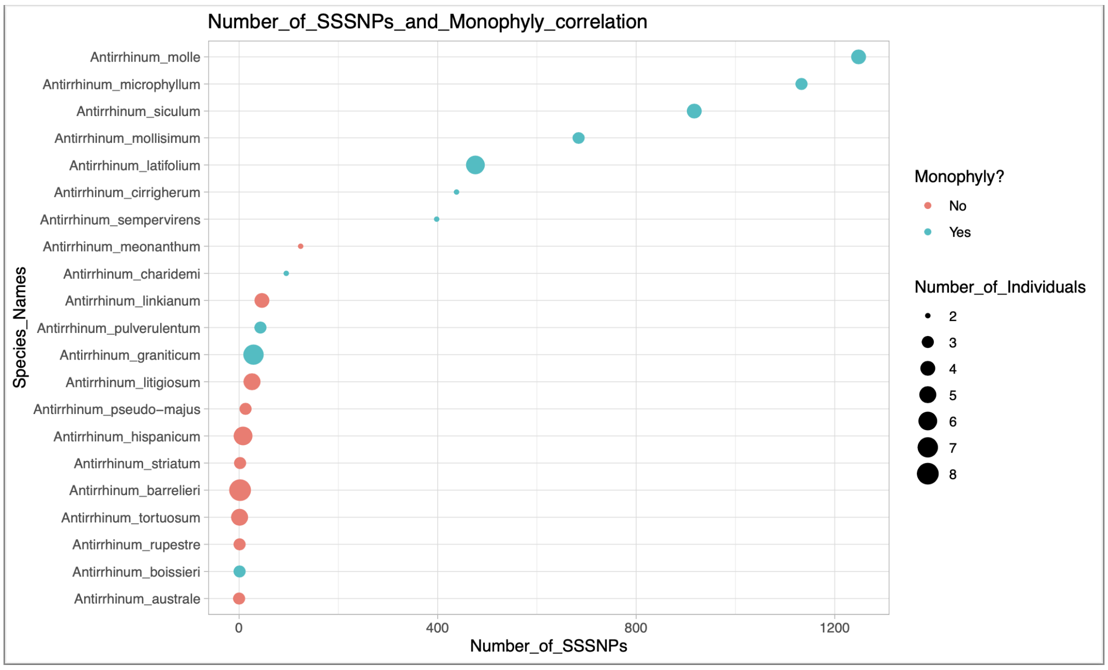
Basic_statistics_of_Ant.xlsx

To visualise and compare the distribution of SSSNPs across all species in available datasets, we produced a script plotting the density distribution of SSSNPs from all multiple sampled species annotated with whether a given species resolved as monophyletic or not, using ggplot2 embedded in the script Number_of_SSSNPs_vs_Monophyly.R.

Command

Visualise the distribution of the number of SSSNPs

```
Rscript Number_of_SSSNPs_vs_Monophyly.R Basic_statistics_of_Ant.xlsx
Number_of_SSSNPs_vs_Monophyly_Ant.pdf 10 6
```



Figur 5.2 The distribution of the number of SSSNPs (Antirrhinum)

5.3 In very large data sets consisting of many hundreds of thousands of nucleotides, one might expect to encounter shared SNPs between any group of samples purely due to random mutations. To distinguish the biological signal of SSNPs from random sampling artefacts, we developed an approach to test whether the number of SSSNPs is greater than expected due to chance alone based on a random distribution of the data. This was implemented by randomising the label of the sequences using the shuf function in bash command lines and repeating the same analysis counting the distribution of SSSNPs. We then compared the number of SSSNPs extracted from the original datasets versus the randomised label datasets using Wilcox signed-rank test (Knapp, 2017) with wilcox.test in R (parameters: paired =

FALSE, alternative = "greater").

Command

wilcox t-test

```
wilcox.test(file$Number_of_SSSNPs,file$Number_of_SSSNPs_Random_label, paired =
FALSE, alternative = "greater")
```

5.4 Computing requirements for extracting species-specific SNPs and allele-frequency-different SNPs depended largely on the dataset size, i.e. the number of sites to parse, and to some extent on the number of multiple-sampled species. Table 3.7. illustrates the run times for data sets involved in this study.

Dataset	Data size (Mb)	Input format	Number of multiple-samples species	Running time
Brownea	1.3	vcf	11	42s
Euphrasia	1.8	vcf	4	22s
Commiphora	1.9	fasta	22	2min12s
Syagrus	3.5	vcf	17	3min
Linanthus	4.4	fasta	20	2min37s
Bee_orchid	6	vcf	4	1min40s
Cornus	9.5	fasta	18	4min32s
Polemonium	9.9	fasta	12	3min1s
Aesculus	17	fasta	15	7min12s
Tsuga	38	fasta	8	8min35s
Quercus	52	fasta	7	10min55s
Vitis	76	vcf	8	38min10s
Antirrhinum	119	vcf	21	3h59min20s
Capurodendron	148	fasta	20	51min12s
Artocarpus	157	fasta	42	2h58mins
Salix	190	vcf	23	2h3min40s
Linaria	235	fasta	13	54min12s
Inga	576	fasta	69	16h8min30s
Geonoma	969	fasta	44	16h12min18s

Table 5.3. The running time for extracting ancestral informative SNPs from different datasets

Genomic region down-sampling and the impact on species r..

6 Performance of each locus in telling species apart

To assess the range of variation in species discrimination power among loci within datasets, we compared the discrimination success of individual loci/genes. This step is important in identifying the attributes of loci that are most informative in species discrimination which could help inform the choice of markers for future use. To do this, we built phylogenetic trees based on each single locus and looked at how many species resolved as monophyletic (compared to the equivalent figure using the total dataset). Given the scale of this task, tree-building was restricted to building quick UPGMA trees using ape_dna_dist.R and ape_dist_tree.R, and then quantifying the species resolved as monophyletic using MonoPhy.

Command

Build quick phylogeny (ape)

```
Rscript ape_dna_dist.R Input.fasta Output.dist
Rscript ape_dist_tree.R Input.dist Output.tree
```

To assess the relationship between nucleotide diversity of individual loci and their performance in telling species apart, we calculated the average nucleotide diversity of each single locus among all individuals in a genus using nuc.div function in R package pegas (Paradis, 2010) (script calculate_pi.R).

Command

Calculate the average nucleotide diversity of each single locus among all individuals in a genus

```
## calculate nucleotide diversity
Rscript ../00_src/calculate_pi.R 01_raw_data/Geonoma_795gene_326inds.fasta
06_nuc_div/Geonoma_795gene_326inds.nuc.div
python ../00_src/seq_each_spps.py -f 01_raw_data/Geonoma_795gene_326inds.fasta -n
01_raw_data/ID_to_scientific_name.txt -sp 01_raw_data/selected_species.txt -o
06_nuc_div/Geonoma_all_loci
grep -v "x" 06_nuc_div/Geonoma_all_loci*.div >
06_nuc_div/Geonoma_all_loci_inds_spps_nuc_div.txt
#rm 06_nuc_div/*fasta 06_nuc_div/*div
mkdir single_spps
for i in `cat 01_raw_data/selected_species.txt`;do
        grep $i 06_nuc_div/single_gene/* >06_nuc_div/single_spps/$i.nuc_div.txt
        done
```

We then plotted the number of monophyletic species resolved by this locus, against the nucleotide diversity, and the density of SSSNPs at this locus, using ggplot2 scripted in NucDiv_DensSSSNPs_No_spps_mono_by_each_gene.R. Regression curves were then fitted for both the series of nucleotide diversity and the density of SSSNPs against the number of species that are resolved as monophyletic using the CORREL function in Excel.

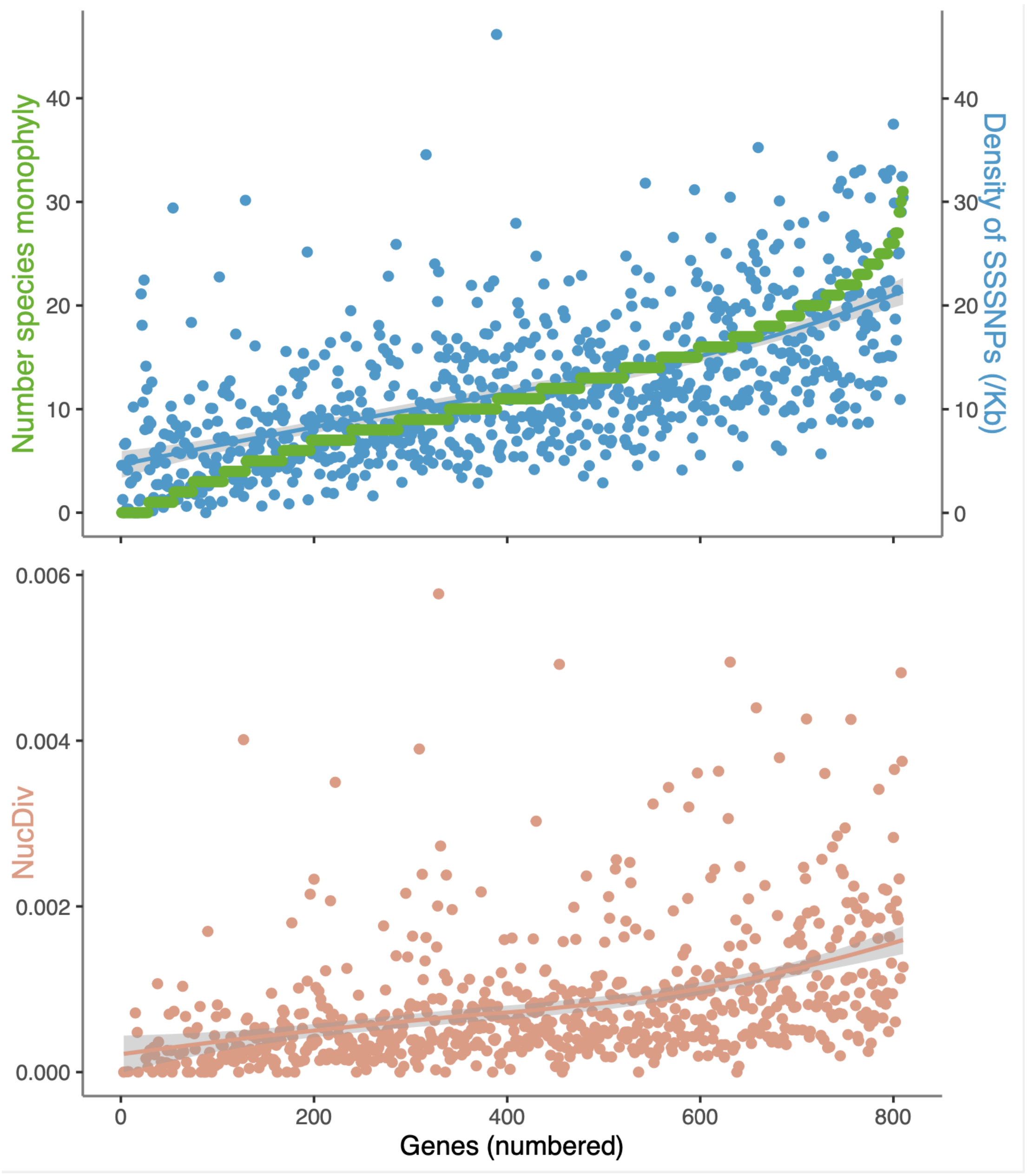


Figure 6. Correlation of Nucleotide Diversity, Density of SSSNPs, and the Number of Monophyletic Clades by each gene

7 **Species discrimination success with down-sampled sequence data**

To further explore how much data is needed to tell species apart, we subsampled each dataset to reduce the amount of sequence information and assess the consequent impact on the species discrimination success.

Specifically, we developed a pipeline (DS_snp.sh, available on *GitHub page, example in the code box) to downsample the data at laddered intervals. To model a reduced amount of sequence information, we tested down-sampling in terms of a) the number of SNPs, and b) the number of DNA segments (genes, exon, or assembled RAD-tags) where this information is available. The steps in both regards are the same:

1) Select a specified number of SNPs/DNA segments randomly using python script.
2) Build a fast UPGMA tree using R package ape and phangorn.
3) Identify monophyletic clades using MonoPhy and count the number of monophyletic clades

with python.

4) Repeat step 1-3 for 50 times (bootstrap = 50).
5) Change the number of SNPs/DNA segments specified and repeat steps 1-4.
6) Visualize the result using ggplot2 (Wickham, 2011).

Command

Testing the impact on species resolving power with down sampled SNPs

```
## The execution will be in the subfolder in parallel with the raw data folder
mkdir random_sampled_sequence

for (( n = 100; n <=2000; n += 200 ));do
        echo $n > Number_of_spps_mono_$n\SNPs.list
        for i in {0..50};do
                python ../../00_src/down_sampling_snp_seq_only.py
../01_raw_data/Inga_810genes_phy-snp-sites.fasta $n
./random_sampled_sequence/sub_fasta_len_$n\_$i.fasta
                Rscript ../../00_src/ape_dna_dist.R
./random_sampled_sequence/sub_fasta_len_$n\_$i.fasta
./random_sampled_sequence/sub_fasta_len_$n\_$i.dist
                rm ./random_sampled_sequence/sub_fasta_len_$n\_$i.fasta ## it can take a huge
disk space
                Rscript ../../00_src/ape_dist_tree.R
random_sampled_sequence/sub_fasta_len_$n\_$i.dist
random_sampled_sequence/sub_fasta_len_$n\_$i.mono.tre
                rm ./random_sampled_sequence/sub_fasta_len_$n\_$i.dist
                Rscript ../../00_src/monophy.R
random_sampled_sequence/sub_fasta_len_$n\_$i.mono.tre KD_284
../01_raw_data/ID_to_scientific_name_phy.txt
random_sampled_sequence/sub_fasta_len_$n\_$i.mono.txt
                python ../../00_src/count_Yes_multiple_sampled_clades.py
random_sampled_sequence/sub_fasta_len_$n\_$i.mono.txt all_species.txt |wc -l >>
Number_of_spps_mono_$n\SNPs.list
                done
        done
paste `ls -v *list` > Number_of_spps_mono_for_boxplot.txt
```

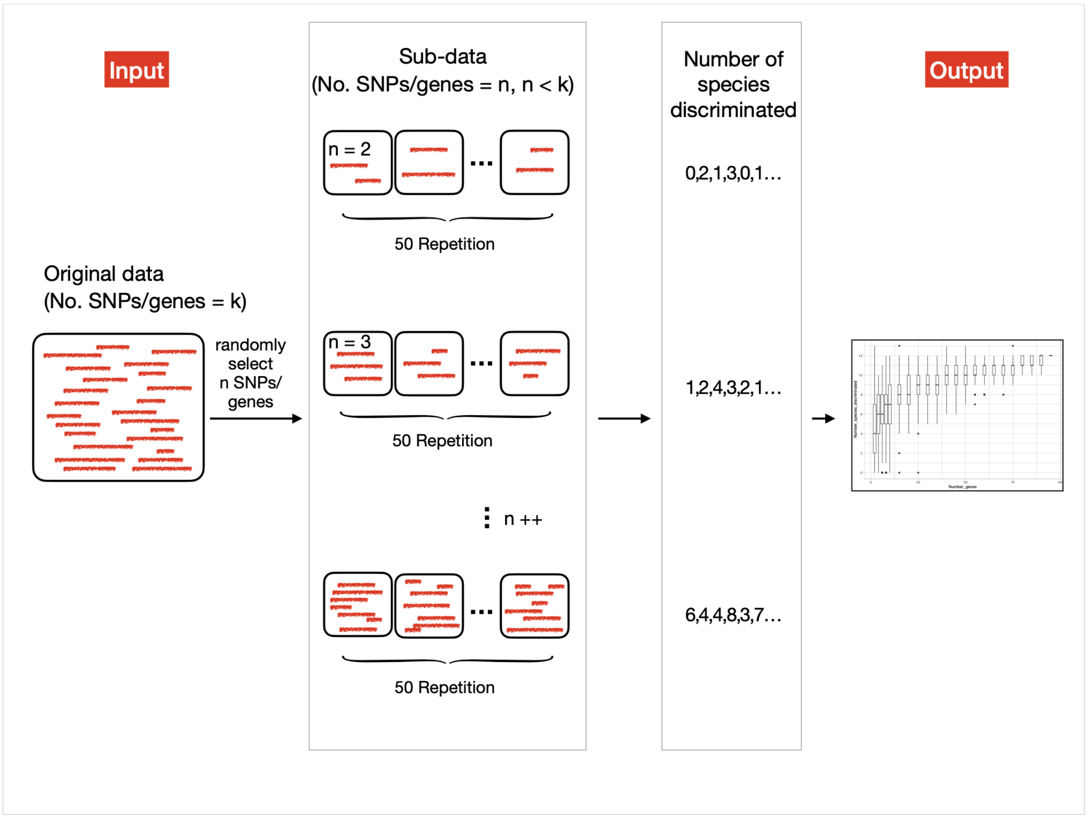


Figure 7. Diagram for the main steps of down-sampling simulation. For both SNP and segment-based down-sampling, the input is the original comprehensive collection of SNPs/DNA segments. A specified number of SNPs/DNA segments starting from 100 SNPs/10 segments were then randomly selected, and at each laddered step I increased the dataset size by randomly selecting a further 100 SNPs/10 segments, and at each step we recorded the number of species being told apart. The random sampling was repeated 50 times at each step. The resulting data were used to plot the distribution of the number of species being told apart with a given number of randomly selected SNPs/segments. The x-axis of the output figure is the number of SNPs/DNA segments from 0 to the maximum the dataset allows, and the y-axis is the distribution of the number of species being told apart based on 50 random samples of SNPs/DNA segments.

7.1 An example of down-sampling.

Output file: Number_of_spps_mono_for_boxplot.txt

Visualisation: with the Number_of_spps_mono_for_boxplot.txt as input, the data is visualised using an interactive R script No_spps_discriminated_vs_No_SNPs_boxplot_v_mac.R with R studio.

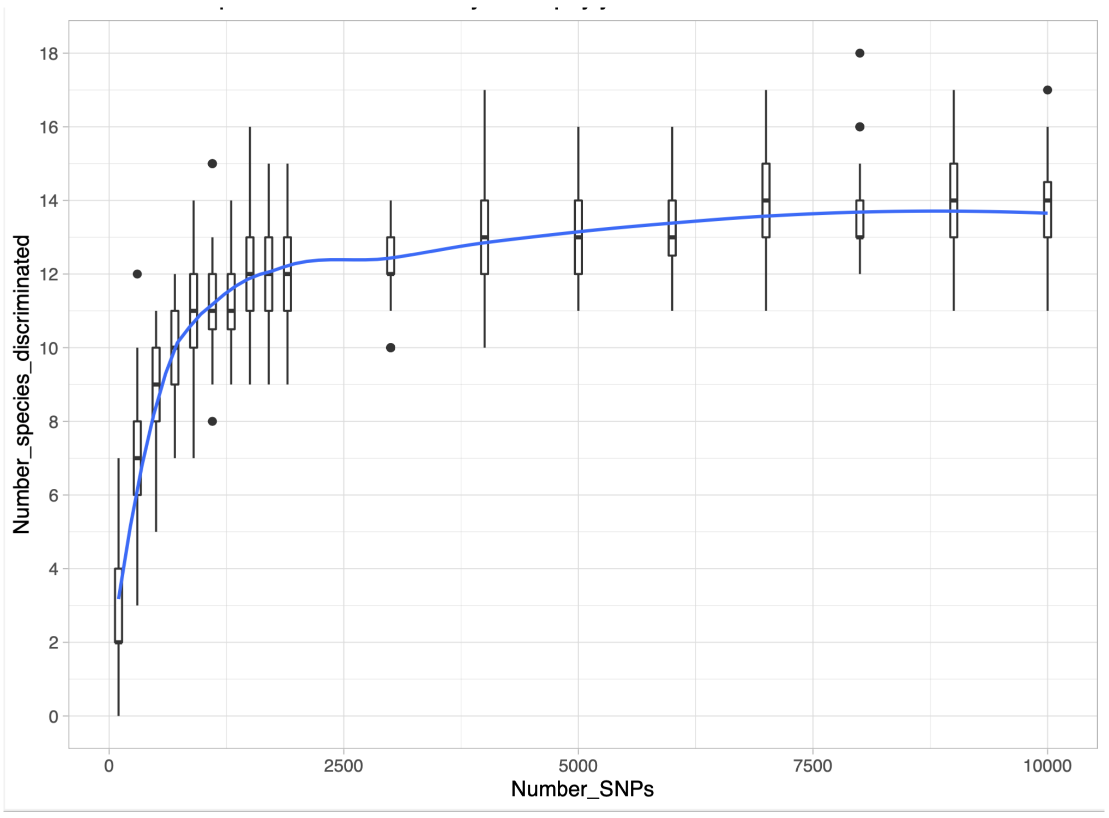


Figure 7.1 Numbers of species discriminated by Monophyly with down-sampled SNPs.

7.2 For datasets in which the performance of individual genes was analysed (see section 6), we conducted a further analysis focusing on the best-performing genes. Here I started with the genes which showed the maximum amount of species discrimination, and sequentially added one gene at a time, selecting at each stage the next best-performing locus. At each stage, I recorded the number of genes used as well as the number of species that resolved as monophyletic. This analysis complements the preceding random selection of loci, by instead assessing the minimal number of best-performing loci required to achieve the maximum amount of species discrimination from a given data set.