Sequoia: stand-alone version

J Huisman

02 January 2024

Introduction

The standalone version of sequoia is largely identical to the R version, except for the input and output: these exist of plain text files and command line arguments, instead of arguments to R functions. For information not related to input and output, including a full description of the different variables, please see the documentation of the R package at CRAN.

One reason to use this standalone version is that for very large datasets (more than 10 000 individuals) it may not be possible or convenient to read the genetic data into R. Alternatively, a stand-alone Fortran program may happen to fit better into your data quality control and analysis pipeline. There is no appreciable difference in computational time, as also for the R package the bulk of the computations are done using Fortran code.

Note that the stand-alone version does not include all functionality that the R package offers, such as various data input checks, estimation of the ageprior, or comparisons between pedigrees. The latter two do not require the genetic data, and work for large pedigrees (up to 50K - 100K or so).

System requirements

You will need a Fortran compiler, e.g. gfortran. This is what turns the human-readable source code (.f90 file) into a computer-readable program (.exe or .dll). The compiler version should recognise the 2003 version of Fortran.

Windows

On windows, you could install the linux emulator cygwin, which includes the compiler gcc-fortran

Linux

`apt-get install gfortran` should do the trick, or a different fortran compiler of your choosing.

Mac

Similar to Linux?

Compiling sequoia

There are many different compiler options possible, but typically for sequoia you would use:

```
gfortran -O3 Sequoia_SA.f90 -o sequoia
```

where

- `gfortran` is the name of the compiler.

- `-O3` is the optimisation level, i.e. how hard the compiler tries to make the program as fast as possible. Omitting this or using `-O1` means quick compilation of a slow program, while using `-O3` or `-O4` (where available) means slow compilation of a faster program.
- `Sequoia_SA.f90` is the name of the source file (which is a plain text file)
- `-o sequoia` indicates that the output program should be called 'sequoia' (.exe on windows, .dll on linux).

Compilation will also create several `.mod` files; these are temporary files that can safely be deleted.

Test run

To check that it installed correctly, go to the folder where the compiled program is, and type `sequoia --help` or in windows + cygwin `./sequoia --help`. The `.` means 'in this folder'. If it displays the help text, everything should be OK. If it states 'no such file or directory' or similar, you are in a different directory as the program, or it did not compile.

You can also specify the full path, or that the program is 1 folder above your data (`./../sequoia --help`), in a different subfolder of the same main folder (`./../program/sequoia --help`), etc.

Data formats

To reconstruct a pedigree, sequoia requires text files with:

- genotype data
- life history data (sex + birth year) [optional]
- age priors: species-specific age-difference distribution among parent-offspring and sibling pairs
- parameter values

These files are described in detail below. You can generate templates for the required input files by running the R package on a subset of your data, or on the example data included with the package, and then use `writeSeq()`:

```
library(sequoia)
SeqOUT <- sequoia(SimGeno_example, LH_HSg5, Err = 0.005,
                  Module = "pre")     # only do the preparation steps
writeSeq(SeqList = SeqOUT,
         GenoM = SimGeno_example,
         folder = "SequoiaTemplatesFolder")
```

Genetic data

First create a subset of SNPs which are as informative (high MAF, high call rate), reliable (low error rate), and independent (low LD) as possible. For full pedigree reconstruction 400 – 700 good markers are sufficient, and fewer are necessary for only parentage assignment or monogamous systems.

One possible tool to perform such subsetting of SNPs is PLINK, with e.g. the following filtering steps:

```
plink --file mydata --geno 0.1 --maf 0.3 --indep 50 5 2
```

Then, the genetic data needs to be in the following format:

- 1 row per individual
- 1 column per SNP, coded as 0/1/2 copies of the reference allele
- missing values coded as -9
- no header row
- first column are IDs: maximum 40 characters long [1], no spaces.

[1] If you have longer IDs, change the global variable 'nchar_ID' at the top of the .f90 file & recompile.

This is very close to the output from PLINK's `--recodeA` option:

```
plink --file mydata --extract plink.prune.in --recodeA --out MyData
```

which will create a file with extension `.raw`. But in this file

- the first 6 column are family ID - Individual ID - father - mother - sex - phenotype
- there is a header row
- missing values are coded as NA.

Very large files can not be opened in most text editors to fix these things, but `sed` and `cut` can do the trick:

```
cat MyData.raw | tr -s ' ' | cut -d ' ' -f2,7- > MyData.txt
sed -i '1d' MyData.txt
sed -i 's/NA/-9/g' MyData.txt
```

which will

- replace multiple adjacent spaces to a single space; then take column 2 (individual ID) and column 7 onwards
- delete the first (header) row
- replace all `NA`'s by -9

Lifehistory data

This includes the sex and 'birth' 'year' (time unit of birth/hatching/...) of as many individuals as possible. They don't need to be in same order as in genotype data, non-genotyped IDs are ignored, and non-included genotyped individuals will have unknown sex and birth year

The file has a header row and 3, 5 or 6 columns:

1. ID: individual ID, max 40 characters, no spaces
2. Sex: 1 = female, 2 = male, 3 = unknown, 4 = hermaphrodite
3. BirthYear: Any positive integer (whole number) for known birth years, use negative integers for unknown.
4. BY.min: optional, set possible birth year range if birthyear is unknown
5. BY.max: optional
6. Year.last: Last year in which individual could have had offspring.

Column names are ignored, so column order is important. If you want to specify Year.last, you must also have columns BY.min & BY.max.

Columns 2–6 may **not** have any character values, which includes (from a Fortran perspective) `NA`'s ; these **must** all be replaced by 3 (Sex) or any negative value (birth year columns).

Time units

The time unit must be chosen such that offspring are never born in the same time unit as their parents. If for example your species may first breed at age 8 months, you should use half years or quarter years as your time unit, and start counting at or before the earliest possible date. You can e.g. use `lubridate::floor_date()` to transform birth dates to any time unit.

AgePriors

The agepriors matrix reflects the distribution of age differences between parent and offspring and between siblings. It will strongly affect pedigree reconstruction, and therefore care should be taken that an appropriate ageprior is used. Note that the same time units should be used as for the birth 'years' in the lifehistory file. Detailed information on the ageprior can be found at https://jiscah.github.io/articles/vignette_age/book/index.html ,

The R function `sequoia::MakeAgePrior()` can be used with and without a pedigree as input to create a (draft) ageprior, and does not not require genetic data:

```
## Ageprior: Flat 0/1, overlapping generations, MaxAgeParent = 3,3

##   M P FS MS PS
## 0 0 0  1  1  1
## 1 1 1  1  1  1
## 2 1 1  1  1  1
## 3 1 1  0  0  0
## 4 0 0  0  0  0
```

WARNING: The stand-alone version does *not* update `AgePriors.txt` between parentage assignment and full pedigree reconstruction, while the R function `sequoia()` does so automatically. To get the most out of pedigree reconstruction,

- generate a 'flat' ageprior, specifying only `MinAgeParent` and `MaxAgeParent`, and write it to a text file,
- run parentage assignment with this ageprior,
- in R, read in the assigned parents and run `MakeAgePrior()`
- write this ageprior a text file
- run full pedigree reconstruction with the dataset-specific ageprior

SequoiaSpecs.txt

Parameter values are read from `SequoiaSpecs.txt`, many of which can be overridden by command line arguments. The only mandatory arguments are `GenotypingErrorRate`, `MaxMismatch` (x3), `Tfilter`, and `Tassign`. `GenoFile` and `LHfile` are also mandatory, but can be specified via the command line.

Each line is in the form 'tag, value'. From the February 2021 version onwards, the tags are used when reading in the data. Any lines with tags not included in Table 1 are ignored. This includes misspelled tags, and matching is case sensitive!

```
Genofile    ,   Geno.txt
LHfile  ,   LifeHist.txt
GenotypingErrorRate ,   0.005
MaxMismatchDUP  ,   11
MaxMismatchOH   ,   4
MaxMismatchME   ,   7
Tfilter ,   -2.0
Tassign ,   0.5
MaxSibshipSize  ,   100
DummyPrefixFemale   ,   F
DummyPrefixMale ,   M
DummyPrefixHerm ,   H
Complexity  ,   2
Hermaphrodites, 0
UseAge  ,   1
FindMaybeRel    ,   0
CalcLLR ,   1
ErrFlavour , version2.0
```

Table 1: SequoiaSpecs.txt

	Default	Mandatory	Alias	Description
GenoFile	'NoFile'	(Yes)	Genofile	Max. 2000 characters, name of genotype file
LHfile	'NoFile'	(Yes)	LifeHist, LifeHistFile	Max. 2000 characters, name of lifehistory data file
GenotypingErrorRate		Yes	Err, ErrorRate	Real number(s) between 0 and 1
MaxMismatchDup		Yes	Max_dif_dup	Integer, max. mismatches to consider as duplicate
MaxMismatchOH		Yes	Max_OH_PO	Integer, max. mismatches to consider as parent-offspring pair
MaxMismatchME		Yes	Max_ME_PPO	Integer, max. mismatches to consider as parent-parent-offspring trio
Tfilter		Yes		Real, negative number, threshold to consider as potential relative
Tassign		Yes		Real, positive number, threshold to assign as relative
MaxSibshipSize	100		Max_n_offspring	Integer, max size of sibships (use a generous margin)
DummyPrefixFemale	'F'			1 or 2 characters
DummyPrefixMale	'M'			1 or 2 characters
DummyPrefixHerm	'H'			1 or 2 characters
Complexity	2		Complex, Complx	Integer, 0=monogamous, 1=simple (ignoring inbreeding), 2=full
Hermaphrodites	0		Herm	Integer, 0=no, 1=A, 2=B
UseAge	1		AgeEffect	Integer, 0=no, 1=yes, 2=extra
FindMaybeRel	0		GetMaybeRel, MaybeRel	Integer, 0=no, 1=yes
CalcLLR	1			Integer, 0=no, 1=yes
ErrFlavour	'2.0'			See ?sequoia::ErrToM

MaxMismatch

The maximum number of allowed mismatches (defined in Table 1) depends on the number of SNPs, their allele frequencies, and the genotyping error rate. The values can be estimated with R function `CalcMaxMismatch()`. In the R package the number of genotyped individuals is also taken into account, for a dataset-wide probability that this maximum is not exceeded. There, the quantile used is `qntl = 0.9999^(1/nIndiv)`.

WARNING: when 'GenotypingErrorRate' is changed, the 'MaxMismatch' values should be changed too, as this is not adjusted automatically. This can be done without reading the entire genotype dataset into R as follows:

```
system('plink --file mydata --freq --nonfounders --out MAF')
MAF <- read.table('MAF.frq', header=TRUE)[,'MAF']
nIndiv <- system('cat MyData.txt.txt | wc -l', intern=TRUE) %>% as.numeric
Err_hat <- 0.005  # for example
MaxMismatch <- sequoia::CalcMaxMismatch(Err = Err_hat,
                                        MAF = MAF,
                                        qntl = 0.9999^(1/nIndiv))
```

Back compatibility

In older versions, the order of the rows was crucial, and for back-compatibility deprecated arguments were therefore retained. Now, most can be omitted if you wish to use the default values.

Files generated by R package version 1.x will have 1 entry for `MaxMismatch` instead of the 3 entries shown here; back compatibility is implemented.

Table 2: Command line arguments

Category	Argument	Options	Details
General	–help		print help
File	–geno	<filename>	
	–lifehist		
	–ageprior		
	–pedigreeIN		
	–only		Only run analysis for these individuals
	–af		Population allele frequencies
	–mt		Mitochodrional haplotype sharing matrix
	–out		
What	–dup		duplicate check
	–par		parentage assignment
	–ped		full pedigree reconstruction
	–nopar		do not read parents.txt prior to reconstruction
	–resume	<x>	resume reconstruction at round <x>
	–noLLR		do not calculate parental LLR
	–maybePO	<max>	find likely parent-offspring pairs
	–maybeRel		find likely relatives pairs
	–pairs	<filename>	LLs for 7 relationships
How	–complex	mono, simp, full	Breeding system complexity
	–age	no, yes, extra	Weight of age info in assignments
	–herm	no, A, B	hermaphrodites
	–quiet		suppress (almost) all messages
	–verbose		print extra many messages
	–log		log with assignment steps for each individual

Running sequoia

Command line options

Many of the settings in `SequoiaSpecs.txt` can be overridden by command line options; if a argument is not specified on the command line, the value in `SequoiaSpecs.txt` is used. A concise overview of the various command line arguments is given in Table 2, and is also shown when you run `./sequoia --help`.

You can combine nearly all arguments together. Many non-sensible combinations will be caught and result in an error, but the checks are not exhaustive so please make sure that the combination of arguments in `SequoiaSpecs.txt` and on the command line is indeed what you intend to do. The settings used are printed to the screen at the start of the run for double checking, unless `--quiet` is specified.

The order of commands is irrelevant, execution will always be in the order:

- duplicate check (when –dup, –par and/or –ped)
- read pedigreeIN
- parentage assignment
- calculate parent LLR
- pedigree reconstruction
- calculate parent LLR
- MaybePO and/or MaybeRel
- Pairs LL

`--maybePO`, `--maybeRel`, `--pairs` will condition on `Pedigree_seq.txt` if run in combination with `--ped`, and otherwise on `Parents.txt` if run together with `--par`.

–pedigreeIN

What happens to the input pedigree `--pedigreeIN` depends on the other arguments:

- `--par`: use as pedigree-prior. A coarse check is performed to remove any definitely incorrect parents (OH count exceeds MaxMismatchOH, age difference is impossible), and the remainder are taken as starting point for further assignments. Useful in hermaphrodites with `Herm=='A'`, or potentially with many unknown birth years.
- `--ped`: starting point, instead of the default `Parents.txt`. The same check is performed as for –par.
- `--maybePO`, `--maybeRel`, `--pairs`: condition on this pedigree. No checking done if pedigree is consistent with genetic data!
- none of the above:
  - CalcLLR=1: calculate the parent LLRs for this pedigree
  - CalcLLR=0/`--noLLR`: only calculate parent-(parent-)offspring Mendelian errors for this pedigree (very fast, also for very large datasets)

Any parents in the pedigree that do not occur in the genotype file are turned into dummy individuals, except for `--par`.

–only

The input is a text file with a single column with IDs, no header. In combination with `--par` or `--ped`, only parents will be assigned to the listed individuals, the rest serves as candidate parents. In combination with `--maybePO` or `--maybeRel`, only pairs where one or both individuals are listed will be checked. (New from May 2023)

–af

Instead of estimating the allele frequencies from the genotype file, you can provide the population allele frequencies. This can e.g. be useful when running sequoia on only a small subset. The input is a text file

with each SNP on a separate row, either as a single column with no header, or as a file with header and a column labelled 'MAF', 'AF', or 'Frequency', such as the output from PLINK –freq.

Messages

Sequoia will by default be fairly verbose in the terminal when running. At the start it will display a brief summary of the input data and the parameter values it will be using. For `--par` and `--ped` it will keep you updated on the current total likelihood and the number of assigned parents. The time spent on each round will get shorter and shorter, and the increase in likelihood smaller and smaller, until the likelihood asymptotes, the parent LLRs are calculated, and the program finishes.

With `--verbose` it also shows which step it is currently working on (see Output: LogLik.txt below); the most time consuming in large datasets are often 'Find Pairs' in Round 1, and 'Sibship grandparents' in Round 2.

Cancelling a run

To stop a currently running instance of sequoia, press CTRL+C (So suppress the urge to copy messages from the Cygwin terminal with CTRL+C ...). Note that under Windows + Cygwin, the run is just halted, and you need to stop it properly via Task Manager.

Linux

Output

Duplicate check

`DuplicatesFound.txt`

- Type: GenoID or Genotype
- ID1, ID2
- Row1, Row2
- nDiffer: number of SNPs at which the pair differs, only counted where both are non-missing. Empty for Type = GenoID
- nBoth: number of SNPs at which the pair are both non-missing
- LLR: likelihood ratio same individual / most-likely relationship of 2 separate individuals

AgePrior

- `AgePriors_new.txt` : the input ageprior, with additional columns for grand-parental and avuncular pairs calculated from the input ageprior. This is the ageprior as used during pedigree reconstruction.

Parentage assignment

`Parents.txt`. Missing values are printed as `NA` in the dam and sire columns, `999.00` in the LLR columns, and `-9` in the OH and ME columns. The columns RowO, RowD and RowS indicate the row number in the genotype data of the Offspring, Dam and Sire, respectively.

Full pedigree reconstruction

In addition to `Pedigree_seq.txt` created at the very end, text files are created each iteration to be able to keep track of progress:

- `PairwiseLLR_<RR>.txt` : Pairs of likely siblings, to be considered during sibship clustering, identified at the start of round .
- `Pedigree_round<RR>.txt` : The pedigree as reconstructed at the end of round `<RR>`.

These files can be used to resume e.g. an interrupted run. `--resume <RR>` will read in `Pedigree_round<RR>.txt`, and if found also `PairwiseLLR_<RR+1>.txt`, and continue with Round <RR+1>. This differs subtly from `--pedigreeIN`, which will set the counter at RR=1, and e.g. not do sibship grandparent assignment in the initial round.[2]

At the very end, a set of additional files is also returned:

- `BirthYearProbabilities.txt` : a matrix with probabilities that individual i (rows) was born in year y (columns), for those individuals with unknown birth years (including dummy individuals). The first 3 columns are id, rowO (row number in genotype file), and Sex (inferred during pedigree reconstruction if missing in Lifehistory data)
- `DummyParents` : summarised info on each dummy individual. 'est.BY' is the mode of the probability distribution.
- `LogLik.txt` : Total log-likelihood after each step (columns) in each round (rows) of pedigree reconstruction, missing values given as 999. Should go fairly smoothly towards the final, maximum value. Repeated large downswings in one step followed by large upswings in an subsequent step indicates trouble finding the most-likely pedigree, due to insufficient or conflicting information. The column names are
  - start
  - cluster: after clustering of likely sibling pairs
  - GGpairs: grandparent–grandoffspring pairs. Not in Rounds 1 and 2
  - merge: merging of earlier formed sibships
  - sibPar: replace sibship dummy parents by real genotyped individuals
  - sibGP: for each sibship, subset candidate grandparents (dummy + real) and assign most likely one(s).
  - morePar: for each individual, subset candidate parents (dummy + real) and assign most likely one(s). May replace earlier assigned parents.

Note that since the 2021 April version the order of 'sibGP' and 'morePar' is swapped (sibGP used to be last); in conjunction with the other edits made in the update, this proved to increase correct assignments without affecting mis-assignments.

FindMaybeRel

`Unassigned_relatives_par.txt` or `Unassigned_relatives_full.txt`, for `--maybePO` and `--maybeRel` respectively. The output columns are described in the helpfile of `GetMaybeRel()`. These files can directly be used as input for `--Pairs`, i.e.

```
./sequoia --par --ped --maybeRel --pairs Unassigned_relatives_full.txt
```

Calculate parental likelihood or Mendelian errors only

When e.g. running `--pedigreeIN MyPedigree.txt`, the output file will be `MyPedigree_LLR.txt`.

For `--pedigreeIN MyPedigree.txt --noLLR`, the output file will be `MyPedigree_OH.txt`.

Pairwise likelihoods

When e.g. running `--pairs MyPairs.txt`, the output file will be `MyPairs_LL.txt`. The output columns are described in the helpfile of `CalcPairLL()`.

[2] intermediate results in subsequent rounds may differ from those when starting from scratch, but it (nearly) always asymptotes to the same pedigree.

Error messages

Often indicate something unusual with your data, or a bug. The latter are from the 2021 April version onwards clearly indicated with "Please report bug". In that case, or if you need help resolving the former, please file a bug report at https://github.com/JiscaH/sequoia/issues or send an email to jisca.huisman @ gmail.com .

Runtime and memory use

From the December 2023 version onwards, required computer memory is reduced by nearly 90%. This comes at the cost of increased runtime (about +10%).

R function vs. command line argument counterparts

Table 3: Counterparts

R function	Argument	Comments
CalcMaxMismatch		
CalcOHLLR(,LLR=FALSE)	–pedigreeIN <file> –noLLR	
CalcOHLLR(,LLR=TRUE)	–pedigreeIN <file>	
CalcPairLL	–pairs	
CheckGeno		
ErrToM	SequoiaSpecs.txt: ErrFlavour	Full flexibility not yet implemented
GenoConvert		cut/sed, see text
GetMaybeRel(,Module='par')	–maybePO	
GetMaybeRel(,Module='ped')	–maybeRel	
MakeAgePrior		Use R function, see text
sequoia(,Module='pre')		for file templates
sequoia(,Module='dup')	–dup	
sequoia(,Module='par')	–par	
sequoia(,Module='ped')	–ped	
SimGeno		
SnpStats		
SummarySeq		
writeSeq		writes file templates
[not available]	–only	only assign/check subset of individuals