Package 'subHMM'

November 13, 2018

Title The SubHMM Model for Identifying Tumor Subclones

Version 0.0.8

Date 2018-11-13

Author Bin Zhu, Paul Albert, Hyoyoung Choo-Wosoba

Description A Hidden Markov Modeling Approach for Identifying Tumor Subclones in Next-Generation Sequencing Studies.

Maintainer Bill Wheeler <wheelerb@imsweb.com>

Imports stats, utils, stringr, numDeriv

Suggests graphics, grDevices

License GPL-2

NeedsCompilation yes

R topics documented:

`subHMM-package` *The SubHMM Model for Identifying Tumor Subclones*

Description

A Hidden Markov Modeling Approach for Identifying Tumor Subclones in Next-Generation Sequencing Studies.

Details

Allele-specific copy number alteration (ASCNA) analysis is for identifying copy number abnormalities in tumor cells. Unlike normal cells, tumor cells are heterogeneous with a combination of dominant and minor subclones with distinct copy number profiles. Estimating the clonal proportion and identifying main and subclone genotypes across the genome is important for understanding tumor progression. Several ASCNA tools have recently been developed, but they have been limited to the identification of subclone regions, and not the genotype of the subclones. This package uses a hidden Markov model-based approach that estimates both sub-clone region as well as region-specific subclone genotype and clonal proportion. A hidden state variable is specified which

represents the conglomeration of clonal genotypes and subclone status. A two-step algorithm for parameter estimation is implemented, where in the first step, a standard hidden Markov model with this conglomerated state space is fit. Then, in the second step, region-specific estimates of the clonal proportions are obtained by maximizing region-specific pseudo-likelihoods.

Author(s)

Hyoyoung Choo-Wosoba, Paul S. Albert, and Bin Zhu <bin.zhu@nih.gov>

References

Choo-Wosoba H., Albert P.S., Zhu B. A Hidden Markov Modeling Approach for Identifying Tumor Subclones in Next-Generation Sequencing Studies.

`data` *Data for examples*

Description

Data for examples.

Details

The object contains `logR` and `logOR` values for `subhmm`.

See Also

`subhmm`

Examples

```
data(data, package="subHMM")

# Display some of the data
logR[1:10]
logOR[1:10]
```

`subhmm` *The SubHMM Model for Identifying Tumor Subclones*

Description

A Hidden Markov Modeling Approach for Identifying Tumor Subclones in Next-Generation Sequencing Studies.

Usage

```
subhmm(logR, logOR, purity=0.7, ploidy=1.5, clonal.prop=0.5,
                    logR.var=0.5, logOR.var=0.5, df=3,
                     genoStates=c("0", "A", "AA", "AB","AAA","AAB","AAAA","AABB",
                                  "AAAB","AAAAA","AAABB","AAAAB"), prob0=NULL,
                     mainclone.trans=NULL, subclone.trans=NULL,
                     subclone.prob=c(0.5, 0.5), subclone.probMat=NULL,
                     maxiter=1000, logOR2.min=1e-6, logOR.var.min=1e-5,
                     df.min=1e-5, df.max=100,
                     loglike.eps=0.01, parm.eps=1e-7, nLociRegion=50, print=1)
```

Arguments

`logR`	Vector of (non-missing) log(Ratio) values. No default
`logOR`	Vector of log(Odds Ratio) values. Note that this vector can have missing values and will be squared inside the function. No default.
`purity`	Initial value for the tumor purity. The default is 0.7.
`ploidy`	Initial value for the ploidy. The default is 1.5.
`clonal.prop`	Initial value for the constant clonal proportion parameter at the first stage (see details). The default is 0.5.
`logR.var`	Initial value for the variance component of logR. The default is 0.5.
`logOR.var`	Initial value for the variance component of logOR. The default is 0.5.
`df`	Initial value for the degrees of freedom. The default is 3.
`genoStates`	Character vector of hidden genotype states. The default is c("0", "A", "AA", "AB", "AAA", "AAB", "AAAA", "AABB", "AAAB", "AAAAA", "AAABB", "AAAAB") The state "0" denotes homozygous deletion, and will always be included..
`prob0`	NULL or a vector of initial probabilities for `genoStates`. If NULL, then it will be set to `rep(1/length(genoStates), length(genoStates))`. The default is NULL.

`mainclone.trans`

NULL or estimated transition matrix of mainclone genotypes. If `length(genoStates) = J`, then this matrix has the form:

	P(genoStates[1])	...	P(genoStates[J])
P(genoStates[1])	p11	...	p1J
...	...	...	...
P(genoStates[J])	pJ1	...	pJJ

If NULL, then it will be set to `matrix(c(rep(c(1-(J-1)/5000,rep(1/5000,J)),(J-1)),1-(J-1)/5000),J,J)`, where `J = length(genoStates)`. The default is NULL.

`subclone.trans`

NULL or estimated transition matrix of subclone existence. This matrix has the form:

	P(no subclone)	P(subclone)

P(no subclone)	p11	p12
P(subclone)	p21	p22

If NULL, then it will be set to `matrix(c(0.999, 0.001, 0.001, 0.999), nrow=2)`. The default is NULL.

`subclone.prob`
: Vector of initial probability of subclone existence, ie `c(P(no subclone), P(subclone))`. The default is `c(0.5, 0.5)`.

`subclone.probMat`
: NULL or multinomial probability matrix of dimension `c(J, J-1)` of subclone genotypes given subclone existence and a mainclone genotype, where `J = length(genoStates)`. For example if the genotype states are defined as in `genoStates` above, then the entry in the third row and second column of this matrix would be the probability of subclone genotype A occurring when the mainclone genotype is AA given subclone existence. If NULL, then the default value is `matrix(1/(J-1), J, J-1)`.

`maxiter`
: Maximum number of iterations for the algorithm. The default is 1000.

`logOR2.min`
: Minimum value for `logOR^2` to prevent numerical difficulties in the algorithm. The default is 1e-6.

`logOR.var.min`
: Minimum value for `logOR.var` to prevent numerical difficulties in the algorithm. The default is 1e-5.

`df.min`
: Lower bound for the `df` parameter in the optimization. The default is 1e-5.

`df.max`
: Upper bound for the `df` parameter in the optimization. The default is 100.

`loglike.eps`
: A stopping tolerance for the algorithm (see details). The algorithm will stop when two succesive log-likelihood values differ by less than `loglike.eps`. The default is 0.01.

`parm.eps`
: A stopping tolerance for the algorithm (see details). The algorithm will stop when the maximum difference between two succesive sets of estimated parameters differ by less than `parm.eps`. The default is 1e-7.

`nLociRegion`
: Minimum number of loci to define a region. The default is 50.

`print`
: Integer to print information to the console. More information is printed with a larger value for this option (see details). The default is 1.

Details

See the reference for details of the algorithm.
Missing values are allowed for `logOR` values as long as `logR` is observed. To prevent numerical errors in the `gamma` function, `df.min` should be a positive value, and `df.max` should not be too large. Similarly, a positive lower bound (`logOR.var.min`) needs to be set for `logOR.var` to prevent overflow errors and to prevent `dchisq` from taking a very long time to compute when the non-centrality parameter and argument are large. Internally, the algorithm transforms the parameters, so the value of the loglikelihood is based on the transformed parameters. However, parameter estimates that are printed to the console (`print` > 1) will be their untransformed values. The algorithm terminates when either a likelihood condition defined by `loglike.eps` or a parameter condition defined by `parm.eps` is satisfied. Either condition can be turned off by setting that stopping tolerance to a value less than or equal to zero.

Value

A list with the following names and descriptions:

- converged Convergence status
- parms Vector of estimated parameters ploidy, purity, logR.var, logOR.var, df.
- cov.parms Estimated covariance matrix of parms.
- se.parms Estimated standard errors of parms.
- asym.se.varlogR Estimated asymptotic standard error of the variance of logR.
- logR.est The expected value of logR
- logOR.est The expected value of logOR
- prob.stage1 Matrix of posterior probabilities for each genotype state at stage 1. The column names have the form P(M=G|no S) = the probability of mainclone genotype G without subclone existence, and P(S=G1|M=G2) = the probability of subclone genotype G1 given mainclone genotype G2 and subclone existence.
- mainclone.genotype Mainclone genotype index.
- subclone.prob Probabilities of subclone genotypes corresponding to each subclone region.
- subclone.ind Subclone region-locus-based indicator.
- subclone.genotype Subclone genotype corresponding to subclone regions by choosing the largest probability of subclone genotypes for each region according to subclone.prob.
- subclone.regions List of subclone region locations.
- clonal.prop.region Region-specific clonal proportion estimates.
- clonal.prop.est Clonal proportion estimate.
- loglike.vec Vector of log-likelihood values at each iteration.
- mainclone.trans.est Estimate of the mainclone.trans matrix.
- subclone.trans.est Estimate of the subclone.trans matrix.
- subclone.probMat.est Estimate of the subclone.probMat matrix.

NOTE: The value for logR.var in the object se.parms is the estimate of the standard error for the variance component of logR (logR.var). This is different from the estimated standard error of the variance of logR, which is asym.se.varlogR.

Author(s)

Hyoyoung Choo-Wosoba, Paul S. Albert, and Bin Zhu <bin.zhu@nih.gov>

References

Choo-Wosoba H., Albert P.S., Zhu B. A Hidden Markov Modeling Approach for Identifying Tumor Subclones in Next-Generation Sequencing Studies.

Examples

```
data(data, package="subHMM")

# Toy example so that it runs quickly
ret <- subhmm(logR, logOR, genoStates=c("", "A"))
ret[1:5]
```