

# Package ‘scoreInvHap’

July 24, 2025

**Title** Get inversion status in predefined regions

**Version** 1.30.0

**Maintainer** Dolors Pelegri-Siso <dolors.pelegri@isglobal.org>

**Description** scoreInvHap can get the samples' inversion status of known inversions. scoreInvHap uses SNP data as input and requires the following information about the inversion: genotype frequencies in the different haplotypes, R2 between the region SNPs and inversion status and heterozygote genotypes in the reference. The package include this data for 21 inversions.

**Depends** R (>= 3.6.0)

**License** file LICENSE

**Encoding** UTF-8

**LazyData** true

**RoxygenNote** 7.1.1

**Imports** Biostrings, methods, snpStats, VariantAnnotation,  
GenomicRanges, BiocParallel, graphics, SummarizedExperiment

**Suggests** testthat, knitr, BiocStyle, rmarkdown

**VignetteBuilder** knitr

**biocViews** SNP, Genetics, GenomicVariation

**git\_url** <https://git.bioconductor.org/packages/scoreInvHap>

**git\_branch** RELEASE\_3\_21

**git\_last\_commit** 6877ed1

**git\_last\_commit\_date** 2025-04-15

**Repository** Bioconductor 3.21

**Date/Publication** 2025-07-23

**Author** Carlos Ruiz [aut],  
Dolors Pelegrí [aut],  
Juan R. Gonzalez [aut, cre]

Contents

|                              |           |
|------------------------------|-----------|
| adaptRefs . . . . .          | 2         |
| checkSNPs . . . . .          | 3         |
| classifSNPs . . . . .        | 4         |
| computeScore . . . . .       | 5         |
| correctAlleleTable . . . . . | 5         |
| getAlleleTable . . . . .     | 6         |
| getGenotypesTable . . . . .  | 6         |
| getInvStatus . . . . .       | 7         |
| hetRefs . . . . .            | 7         |
| info . . . . .               | 8         |
| inversionGR . . . . .        | 8         |
| prepareMap . . . . .         | 9         |
| Refs . . . . .               | 9         |
| scoreInvHap . . . . .        | 10        |
| scoreInvHapRes . . . . .     | 11        |
| SNPsR2 . . . . .             | 13        |
| <b>Index</b>                 | <b>14</b> |

---

|           |                                         |
|-----------|-----------------------------------------|
| adaptRefs | <i>Adapt references to imputed data</i> |
|-----------|-----------------------------------------|

---

Description

Internal

Usage

adaptRefs(Refs, alleletable, haploid = FALSE)

Arguments

- |             |                                                           |
|-------------|-----------------------------------------------------------|
| Refs        | List with the allele frequencies                          |
| alleletable | Data.frame with the alleles per SNP (from getAlleleTable) |
| haploid     | Logical. If TRUE, modify references for haploid samples   |

Value

List with the same values than Refs but adapted to imputation data

---

|           |                              |
|-----------|------------------------------|
| checkSNPs | <i>Check genotype object</i> |
|-----------|------------------------------|

---

## Description

This function checks the genotype object before passing the SNPs to 'scoreInvHap'. The function removes SNPs with different alleles or different allele frequencies. Nonetheless, it is possible that these SNPs could be recovered after an examination of the results. Be aware that testing of allele frequencies might fail for small datasets.

## Usage

```
checkSNPs(SNPobj, checkAlleleFreqs = TRUE)
```

## Arguments

|                  |                                                    |
|------------------|----------------------------------------------------|
| SNPobj           | List with SNPs data from plink or VCF-class.       |
| checkAlleleFreqs | Should allele frequencies be check (Default: TRUE) |

## Value

List containing the SNPs prepared for scoreInvHap

- **genos**: Object with genotype data ready for scoreInvHap
- **wrongAlleles**: Character vector with the SNPs discarded due to having alleles different to reference
- **wrongFreqs**: Character vector with the SNPs discarded due to having allele frequencies different to reference

## Examples

```
## Run method
if(require(VariantAnnotation)){
  vcf <- readVcf(system.file("extdata", "example.vcf", package = "scoreInvHap"), "hg19")
  resList <- checkSNPs(vcf)
  resList
}
```

---

|             |                                              |
|-------------|----------------------------------------------|
| classifSNPs | <i>Get similarity scores and probability</i> |
|-------------|----------------------------------------------|

---

### Description

This function computes the similarity scores between the sample SNPs and the haplotype's reference.

### Usage

```
classifSNPs(
  genos,
  R2,
  refs,
  alleletable,
  BPPARAM = BiocParallel::SerialParam()
)

classifSNPsImpute(genos, R2, refs, BPPARAM = BiocParallel::SerialParam())
```

### Arguments

|             |                                                                                                              |
|-------------|--------------------------------------------------------------------------------------------------------------|
| genos       | Matrix with the samples genotypes. It is the result of <code>getGenotypesTable</code>                        |
| R2          | Vector with the R2 between the SNPs and the inversion status.                                                |
| refs        | List of matrices. Each matrix has, for an SNP, the frequencies of each genotype in the different haplotypes. |
| alleletable | Data frame with the reference alleles computed with <code>getAlleleTable</code> .                            |
| BPPARAM     | A <code>BiocParallelParam</code> instance. Used to parallelize computation                                   |

### Details

`classifSNPs` computes, for each individual, similarity scores for all the present haplotypes. For each SNP, we compute as many similarity scores as haplotypes present in the reference. We have defined the similarity score as the frequency of this genotype in the different haplotype population. To compute the global similarity score, we have computed a mean of the scores by SNP weighted by the R2 between the SNP and the haplotype classification.

`classifSNPsImpute` is a version of `classifSNPs` that works with posterior probabilities of imputed genotypes.

### Value

List with the results:

- `scores`: Matrix with the similarity scores of the individuals
- `numSNPs`: Vector with the number of SNPs used in each computation

---

|              |                                                   |
|--------------|---------------------------------------------------|
| computeScore | <i>Compute all similarity scores for a sample</i> |
|--------------|---------------------------------------------------|

---

**Description**

Internal

**Usage**

```
computeScore(geno, refs, R2)
```

**Arguments**

|      |                                                                                                              |
|------|--------------------------------------------------------------------------------------------------------------|
| geno | Vector with the sample genotypes. It is the result of getGenotypesTable                                      |
| refs | List of matrices. Each matrix has, for an SNP, the frequencies of each genotype in the different haplotypes. |
| R2   | Vector with the R2 between the SNPs and the inversion status                                                 |

**Value**

List with the results:

- scores: Vector with the similarity scores of the sample
- numSNPs: Numeric with the number of SNPs used in the computation

---

|                    |                                      |
|--------------------|--------------------------------------|
| correctAlleleTable | <i>Solve genotypes discrepancies</i> |
|--------------------|--------------------------------------|

---

**Description**

This function tries to solve discrepancies between the reference and sample genotypes. The cause of these discrepancies is that samples and references have used different strands to codify the SNP. This function get the complement genotypes for the discordant SNPs and checks if discordancies are solved.

**Usage**

```
correctAlleleTable(alleleTable, hetRefs, map)
```

**Arguments**

|             |                                                                    |
|-------------|--------------------------------------------------------------------|
| alleleTable | Data.frame with the alleles per SNP (from getAlleleTable)          |
| hetRefs     | Character vector with the heterozygous genotypes in the reference. |
| map         | Data.frame with the annotation of the SNPs (from plink format)     |

**Value**

alleletable without discrepancies between these genotypes and the references.

---

|                |                                 |
|----------------|---------------------------------|
| getAlleleTable | <i>Compute the allele table</i> |
|----------------|---------------------------------|

---

**Description**

Get a data.frame that maps the numeric genotype of a SNPmatrix (0, 1, 2) into the real genotype. Heterozygous genotypes are ordered alphabetically.

**Usage**

```
getAlleleTable(map)
```

**Arguments**

|     |                                                                |
|-----|----------------------------------------------------------------|
| map | Data.frame with the annotation of the SNPs (from plink format) |
|-----|----------------------------------------------------------------|

**Value**

Data.frame with genotypes map

---

|                   |                            |
|-------------------|----------------------------|
| getGenotypesTable | <i>Get genotypes table</i> |
|-------------------|----------------------------|

---

**Description**

Get a matrix with the sample genotypes from all SNP.

**Usage**

```
getGenotypesTable(geno, allele)
```

**Arguments**

|        |                                                           |
|--------|-----------------------------------------------------------|
| geno   | SnpMatrix (from plink format)                             |
| allele | Data.frame with the alleles per SNP (from getAlleleTable) |

**Value**

Character matrix with the samples genotypes

---

`getInvStatus`*Get the inversion status of a sample*

---

**Description**

This function estimates the inversion status of the samples using the probabilities computed in `classifSNPs`

**Usage**

```
getInvStatus(scores)
```

**Arguments**

`scores`                      Matrix of probabilities (from `classifSNPs`)

**Value**

List with the results:

- `class`: Vector with the most probable classification
- `certainty`: Vector with the certainty of the most probable classification

---

`hetRefs`*Heterozygote genotypes in the references*

---

**Description**

Dataset with the heterozygote genotypes of all the SNPs used in any of the references. This dataset include all the SNPs that are present inside the inversion's region in 1000 Genomes Phase 3.

**Usage**

```
hetRefs
```

**Format**

List of character vectors with the heterozygous genotypes of the SNPs present included the region of 21 inversions. Each element is named with the SNPs names.

---

|      |                                  |
|------|----------------------------------|
| info | <i>SNP reference description</i> |
|------|----------------------------------|

---

**Description**

Description of the SNPs included in scoreInvHap references. The description includes the coordinates in hg19, the dbSNP identifier, the reference and alternative allele and the allele frequency in the European Samples of 1000 Genomes.

**Usage**

info

**Format**

data.frame

---

|             |                                |
|-------------|--------------------------------|
| inversionGR | <i>Inversions' description</i> |
|-------------|--------------------------------|

---

**Description**

Description of the 21 human inversions whose references are included in scoreInvHap. The description includes the cytogenic location, the coordinates in hg19, the number of alleles and the number of SNPs with a MAF > 5 Samples of 1000 Genomes.

**Usage**

inversionGR

**Format**

GenomicRanges with the inversions' description in the metadata

---

|            |                                     |
|------------|-------------------------------------|
| prepareMap | <i>Modify feature data from VCF</i> |
|------------|-------------------------------------|

---

**Description**

Internal. Modify feature data from VCF to comply with scoreInvHap requirements.

**Usage**

prepareMap(vcf)

**Arguments**

vcf                      VCF object

**Value**

Data.frame with the feature data

---

|      |                                         |
|------|-----------------------------------------|
| Refs | <i>Genotype frequency in references</i> |
|------|-----------------------------------------|

---

**Description**

Dataset with the genotype frequencies in the different haplotype populations. These frequencies have been computed using the European samples of 1000 Genomes Phase 3 data. Real inversion status have been obtained from invFEST and 1000Genomes.

**Usage**

Refs

**Format**

List of matrices for 20 inversions. Each matrix has the frequency of each genotype in each haplotype.

scoreInvHap

*scoreInvHap: package to get inversion status of predefined regions.***Description**

scoreInvHap can get the samples' inversion status of known inversions. scoreInvHap uses SNP data as input and requires the following information about the inversion: genotype frequencies in the different inversion groups, R2 between the region SNPs and inversion status, heterozygote genotypes in the reference, allele frequencies in the reference population and inversion frequencies. The package include this data for 21 inversions.

This is the main function of 'scoreInvHap' package. This function accepts SNPs data in a plink or a VCF format and compute the inversion prediction. The list of available inversions is included in a GenomicRanges called 'inversionGR'.

**Usage**

```
scoreInvHap(
  SNPlist,
  inv = NULL,
  SNPsR2,
  hetRefs,
  Refs,
  R2 = 0,
  probs = FALSE,
  BPPARAM = BiocParallel::SerialParam(),
  verbose = FALSE
)
```

**Arguments**

|         |                                                                                                                                                          |
|---------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
| SNPlist | List with SNPs data from plink or VCF-class.                                                                                                             |
| inv     | Character with the name of the inversion to genotype. The available inversions are included in a table in the main vignette.                             |
| SNPsR2  | Vector with the R2 of the SNPs of the region                                                                                                             |
| hetRefs | Vector with the heterozygote form of the SNP in the inversion                                                                                            |
| Refs    | List with the allele frequencies in the references                                                                                                       |
| R2      | Vector with the R2 between the SNPs and the inversion status                                                                                             |
| probs   | Logical. If TRUE, scores are computed using posterior probabilities. If FALSE, scores are computed using best guess. Only applied when SNPlist is a VCF. |
| BPPARAM | A BiocParallelParam instance. Used to parallelize computation                                                                                            |
| verbose | Should message be shown?                                                                                                                                 |

**Value**

A scoreInvHap object

**Examples**

```
# See list of inversions
data(inversionGR)
inversionGR

## Run method
if(require(VariantAnnotation)){
  vcf <- readVcf(system.file("extdata", "example.vcf", package = "scoreInvHap"), "hg19")
  res <- scoreInvHap(vcf, inv = "inv7_005")
}
```

---

|                |                                 |
|----------------|---------------------------------|
| scoreInvHapRes | <i>scoreInvHapRes instances</i> |
|----------------|---------------------------------|

---

**Description**

Container with the results of the classification pipeline

**Usage**

```
## S4 method for signature 'scoreInvHapRes'
classification(object, minDiff = 0, callRate = 0, inversion = TRUE)

## S4 method for signature 'scoreInvHapRes'
certainty(object)

## S4 method for signature 'scoreInvHapRes'
diffscores(object)

## S4 method for signature 'scoreInvHapRes'
maxscores(object)

## S4 method for signature 'scoreInvHapRes'
numSNPs(object)

## S4 method for signature 'scoreInvHapRes'
plotCallRate(object, callRate = 0.9, ...)

## S4 method for signature 'scoreInvHapRes'
plotScores(object, minDiff = 0.1, ...)

## S4 method for signature 'scoreInvHapRes'
propSNPs(object)

## S4 method for signature 'scoreInvHapRes'
scores(object)
```

**Arguments**

|                        |                                                                                                                    |
|------------------------|--------------------------------------------------------------------------------------------------------------------|
| <code>object</code>    | <code>scoreInvHapRes</code>                                                                                        |
| <code>minDiff</code>   | Numeric with the threshold of the minimum difference between the top and the second score. Used to filter samples. |
| <code>callRate</code>  | Numeric with the threshold of the minimum call rate of the samples. Used to filter samples.                        |
| <code>inversion</code> | Logical. If true, haplotypes classification is adapted to return inversion status. (Default: TRUE)                 |
| <code>...</code>       | Further parameters passed to plot function.                                                                        |

**Value**

A `scoreInvHapRes` instance

**Methods (by generic)**

- `classification`: Get classification
- `certainty`: Get classification certainty
- `diffscores`: Get maximum similarity scores
- `maxscores`: Get maximum similarity scores
- `numSNPs`: Get number of SNPs used in computation
- `plotCallRate`: Plot call rate based QC
- `plotScores`: Plot scores based QC
- `propSNPs`: Get proportions of SNPs used in computation
- `scores`: Get similarity scores

**Slots**

`classification` Factor with the individuals classification

`scores` Similarity scores for the different haplotypes.

`numSNPs` Numeric with SNPs used to compute the scores.

`certainty` Numeric with the certainty of the classification for each individual.

**Examples**

```
if(require(VariantAnnotation)){
  vcf <- readVcf(system.file("extdata", "example.vcf", package = "scoreInvHap"), "hg19")

  ## Create scoreInvHapRes class from pipeline
  res <- scoreInvHap(vcf, inv = "inv7_005")

  ## Print object
  res

  ## Get haplotype classification
```

```
classification(res)

## Get similiraty scores
scores(res)
}
```

---

SNPsR2

*R2 between the SNPs and the inversion status*

---

### Description

Dataset with R2 between the SNPs and the inversion status. This values are used to weigth similarity scores. These values have been computed using the European samples of 1000 Genomes Phase 3 data. Real inversion status have been estimated using invClust.

### Usage

SNPsR2

### Format

List of numeric vectors for 21 inversions

# Index

## \* datasets

- hetRefs, [7](#)
- info, [8](#)
- inversionGR, [8](#)
- Refs, [9](#)
- SNPsR2, [13](#)

adaptRefs, [2](#)

certainty (scoreInvHapRes), [11](#)  
certainty, scoreInvHapRes-method  
(scoreInvHapRes), [11](#)

checkSNPs, [3](#)

classification (scoreInvHapRes), [11](#)  
classification, scoreInvHapRes-method  
(scoreInvHapRes), [11](#)

classifSNPs, [4](#)  
classifSNPsImpute (classifSNPs), [4](#)  
computeScore, [5](#)  
correctAlleleTable, [5](#)

diffscores (scoreInvHapRes), [11](#)  
diffscores, scoreInvHapRes-method  
(scoreInvHapRes), [11](#)

getAlleleTable, [6](#)  
getGenotypesTable, [6](#)  
getInvStatus, [7](#)

hetRefs, [7](#)

info, [8](#)  
inversionGR, [8](#)

maxscores (scoreInvHapRes), [11](#)  
maxscores, scoreInvHapRes-method  
(scoreInvHapRes), [11](#)

numSNPs (scoreInvHapRes), [11](#)  
numSNPs, scoreInvHapRes-method  
(scoreInvHapRes), [11](#)

plotCallRate (scoreInvHapRes), [11](#)  
plotCallRate, scoreInvHapRes-method  
(scoreInvHapRes), [11](#)

plotScores (scoreInvHapRes), [11](#)  
plotScores, scoreInvHapRes-method  
(scoreInvHapRes), [11](#)

prepareMap, [9](#)

propSNPs (scoreInvHapRes), [11](#)  
propSNPs, scoreInvHapRes-method  
(scoreInvHapRes), [11](#)

Refs, [9](#)

scoreInvHap, [10](#)  
scoreInvHapRes, [11](#)  
scoreInvHapRes-class (scoreInvHapRes),  
[11](#)  
scoreInvHapRes-methods  
(scoreInvHapRes), [11](#)  
scores (scoreInvHapRes), [11](#)  
scores, scoreInvHapRes-method  
(scoreInvHapRes), [11](#)  
SNPsR2, [13](#)