Package 'UPDhmm'

October 17, 2024

Title Detecting Uniparental Disomy through NGS trio data

Version 1.0.0

BugReports <https://github.com/martasevilla/UPDhmm/issues>

Description Uniparental disomy (UPD) is a genetic condition
where an individual inherits both copies
of a chromosome or part of it from one parent,
rather than one copy from each parent. This package
contains a HMM for detecting UPDs through HTS
(High Throughput Sequencing) data from trio assays.
By analyzing the genotypes in the trio, the model
infers a hidden state (normal, father isodisomy,
mother isodisomy, father heterodisomy and mother heterodisomy).

biocViews Software, HiddenMarkovModel, Genetics

License MIT + file LICENSE

Encoding UTF-8

LazyData false

RoxygenNote 7.3.1

Depends R (>= 4.3.0)

Imports HMM, utils, VariantAnnotation, GenomicRanges, S4Vectors,
IRanges, stats

Suggests knitr, testthat (>= 2.1.0), BiocStyle, rmarkdown, markdown,
karyoploteR, regioneR, dplyr

VignetteBuilder knitr

Roxygen list(markdown = TRUE)

URL <https://github.com/martasevilla/UPDhmm>

Language en-US

git_url https://git.bioconductor.org/packages/UPDhmm

git_branch RELEASE_3_19

git_last_commit 5bcdf84

git_last_commit_date 2024-04-30

Author Marta Sevilla [aut, cre] (<<https://orcid.org/0009-0005-0179-920X>>),
        Carlos Ruiz-Arenas [aut] (<<https://orcid.org/0000-0002-6014-3498>>)

Maintainer Marta Sevilla <marta.sevilla@upf.edu>

Contents

UPDhmm-package   *UPDhmm: Detecting Uniparental Disomy through NGS trio data*

Description

Uniparental disomy (UPD) is a genetic condition where an individual inherits both copies of a chromosome or part of it from one parent, rather than one copy from each parent. This package contains a HMM for detecting UPDs through HTS (High Throughput Sequencing) data from trio assays. By analyzing the genotypes in the trio, the model infers a hidden state (normal, father isodisomy, mother isodisomy, father heterodisomy and mother heterodisomy).

Author(s)

Maintainer: Marta Sevilla <marta.sevilla@upf.edu> (ORCID)

Authors:

- Carlos Ruiz-Arenas <cruizarenas@unav.es> (ORCID)

See Also

Useful links:

- <https://github.com/martasevilla/UPDhmm>

- Report bugs at <https://github.com/martasevilla/UPDhmm/issues>

addOr	*Function to transform a large collapsed VCF into a dataframe, incorporating predicted states along with the log-likelihood ratio and p-value.*

Description

Function to transform a large collapsed VCF into a dataframe, incorporating predicted states along with the log-likelihood ratio and p-value.

Usage

```
addOr(filtered_def_blocks_states, largeCollapsedVcf, hmm, genotypes)
```

Arguments

filtered_def_blocks_states
:   data.frame object containing the blocks

largeCollapsedVcf
:   Input VCF file

hmm
:   Hidden Markov Model used to infer the events. The format should adhere to the general HMM format from HMM package with a series of elements:

    1. The hidden states names in the "States" vector.
    2. All possible observations in the "Symbols" vector.
    3. Start probabilities of every hidden state in the "startProbs" vector.
    4. Transition probabilities matrix of the hidden states in "transProbs".
    5. Probabilities associated between every hidden state and all possible observations in the "emissionProbs" matrix.

genotypes
:   Possible GT formats and its correspondence with the hmm

Value

data.frame containing the transformed information.

applyViterbi	*Apply the hidden Markov model using the Viterbi algorithm.*

Description

Apply the hidden Markov model using the Viterbi algorithm.

Usage

```
applyViterbi(largeCollapsedVcf, hmm, genotypes)
```

Arguments

largeCollapsedVcf

> input vcf file

hmm   Hidden Markov Model used to infer the events

genotypes   Possible GT formats and its correspondence with the hmm

Value

largeCollapsedVcf

---

asDfVcf   *Function to transform a large collapsed VCF into a dataframe with*
                                   predicted states, including chromosome, start position, end position
                                   and metadata.

Description

Function to transform a large collapsed VCF into a dataframe with predicted states, including chromosome, start position, end position and metadata.

Usage

```
asDfVcf(largeCollapsedVcf, genotypes)
```

Arguments

largeCollapsedVcf

> Name of the large collapsed VCF file.

genotypes   Possible GT formats and its correspondence with the hmm

Value

dataframe

---

blocksVcf	*Function to simplify contiguous variants with the same state into blocks.*

Description

Function to simplify contiguous variants with the same state into blocks.

Usage

```
blocksVcf(df)
```

Arguments

df   data.frame resulting from the `as_df_vcf` function.

Value

data.frame containing information on the chromosome, start #' position of the block, end position of the block, and predicted state.

---

calculateEvents	*Calculate UPD events in trio VCFs.*

Description

This function predicts the hidden states by applying the Viterbi algorithm using the Hidden Markov Model (HMM) from the UPDhmm package. It takes the genotypes of the trio as input and includes a final step to simplify the results into blocks.

Usage

```
calculateEvents(largeCollapsedVcf, hmm = NULL)
```

Arguments

largeCollapsedVcf

           The VCF file in the general format (largeCollapsedVcf) with VariantAnnotation package. Previously edited with `vcfCheck()` function from UPDhmm package

hmm   Default = NULL. If no arguments are added, the package will use the default HMM already implemented, based on Mendelian inheritance. If an optional HMM is desired, it should adhere to the general HMM format from `HMM` package with the following elements inside a list:

      1. The hidden state names in the "States" vector.
      2. All possible observations in the "Symbols" vector.

3. Start probabilities of every hidden state in the "startProbs" vector.
4. Transition probabilities matrix between states in "transProbs".
5. Probabilities associated between every hidden state and all possible observations in the "emissionProbs" matrix.

Value

A data.frame object containing all detected events in the provided trio. If no events are found, the function will return an empty data.frame.

Examples

```
file <- system.file(package = "UPDhmm", "extdata", "test_het_mat.vcf.gz")
vcf <- VariantAnnotation::readVcf(file)
processedVcf <- vcfCheck(vcf,
    proband = "NA19675", mother = "NA19678",
    father = "NA19679"
)
```

---

hmm   *HMM data for predicting UPD events in trio genomic data*

Description

This dataset provides Hidden Markov Model (HMM) parameters for predicting uniparental disomy (UPD) events in trio genomic data.

states Five different possible states.

symbols Code symbols used for genotype combinations.

startProbs The initial probabilities of each state.

transProbs Probabilities of transitioning from one state to another.

emissionProbs Given a certain genotype combination, the odds of each possible state.

Usage

```
data(hmm)
```

Format

A list with 5 different elements

Source

Created in-house based on basic Mendelian rules for calculating UPD events.

Examples

```
data(hmm)
```

---

vcfCheck	*Check quality parameters (optional) and change IDs.*

Description

This function takes a VCF file and converts it into a largeCollapsedVcf object using the VariantAn-
notation package. It also rename the sample for subsequent steps needed in UPDhmm package.
Additionally, it features an optional parameter, quality_check, which triggers warnings when vari-
ants lack sufficient quality based on RD and GQ parameters in the input VCF.

Usage

```
vcfCheck(largeCollapsedVcf, father, mother, proband, check_quality = FALSE)
```

Arguments

largeCollapsedVcf

                The file in largeCollapsedVcf format.

father   Name of the father's sample.

mother   Name of the mother's sample.

proband   Name of the proband's sample.

check_quality   Optional argument. TRUE/FALSE. If quality parameters want to be measured.
Default = FALSE

Value

largeCollapsedVcf (VariantAnnotation VCF format).

Examples

```
fl <- system.file("extdata", "test_het_mat.vcf.gz", package = "UPDhmm")
vcf <- VariantAnnotation::readVcf(fl)
processedVcf <-
    vcfCheck(vcf, proband = "Sample1", mother = "Sample3", father = "Sample2")
```

Index