Package 'VPA'

November 4, 2011

Type Package

Title Variant Pattern Analyzer for next-generation sequencing study

Version 0.3.4

Date 2011-11-04

Author Qiang Hu

Maintainer Qiang Hu <huqmail@gmail.com>

Depends R (>= 2.12.1), snowfall

Suggests Rsamtools

Enhances Rmpi, snow

Description VPA (Variant Pattern Analyzer) is a package for prioritizing variants with specified frequency pattern from multiple study subjects in next-generation sequencing study. To install VPA package in Unix-like operation system, the version of R should be >= 2.12.1. To install VPA package in Windows operation system, the version of R should be >=2.14.

License GPL-2

LazyLoad yes

R topics documented:

VPA-package 2
filterpos 2
filtervcf 4
gefreq 5
getref 6
LoadFiltering 7
Patterning 9
Pos2Gene 10
pos2seq 11
read.vcf 12
subvcf 13
vcfreq 14
write.vcf 15

Index **16**

`VPA-package` *Extract variants from VCF data with specified variant frequency pattern*

Description

VPA (Variant Pattern Analyzer) is a package for prioritizing variants with user-specified frequency pattern from multiple study subjects in next-generation sequencing study. The package starts from individual files of sequence and variant calls and the extract variants with user-specified frequency pattern across the study subjects of interest. The frequency pattern can be analyzed at both variant level and gene level, and functions are provided to assess the statistical significance of observed frequency difference.

Details

Package:	VPA
Type:	Package
Version:	0.3.4
Date:	2011-11-04
License:	GPL-2
LazyLoad:	yes

Author(s)

Qiang Hu

Maintainer: Qiang Hu <huqmail@gmail.com>

Examples

```
#setwd(system.file("extdata", package="VPA"))
#varflt <- LoadFiltering(file="index1.txt", filtering=TRUE)
#pattern <- cbind(A=c(1/4,1), B=c(0,0))
#varRes1 <- Patterning(varflt, pattern)
```

`filterpos` *Filter variants against known SNP dataset*

Description

The function is used to filter variants against known SNP dataset in VCF, bed, gff or user-specified position files. For example, variants in VCF format can be filtered against dbSNP, 1000 genome project dataset, customized VCF data and so on.

Usage

```
filterpos(vcf, position=NULL, file="", type="vcf", tbi=FALSE, chr=TRUE,
tabix="tabix", ...)
```

Arguments

vcf	A VCF object for filtering.
position	A data.frame or matrix with chromosome names in the first column, start positions in the second column and end positions in the third column (1-based). This can be used to filter against customized VCF data.
file	The file containing the known SNPs.
type	The date format of input file. It can be 'vcf', 'bed' or 'gff'.
tbi	Logical value. If TRUE, the input file should be indexed by tabix for efficient information retrieval.
chr	Logical value. If TRUE, the chromosome names of the input file should have the prefix of 'chr', e.g. 'chr1'. If FALSE, the chromosome names don't have the 'chr' prefix.
tabix	The path of tabix function. if NULL, scanTabix function from Rsamtools will be used instead.
...	More arguments for read.table when reading the input file.

Details

Variants can be filtered against dbSNP and 1000 genome project dataset to eliminate common variants.

For example, the dbSNP 132 can be download from UCSC (http://hgdownload.cse.ucsc.edu/goldenPath/hg19/database/snp132.txt.gz). The 2nd-5th columns of the dataset can be extracted easily using 'cut' or 'awk' into a bed format file. The bed file can be indexed by 'tabix' for efficient information retrieval. The filterpos function can be used to eliminate the variants observed in the indexed dataset file, with arguments type="bed" and tbi=TRUE.

Value

The input vcf data will be filtered against known SNP database or user-specified position files. A list including filtered vcf data and dropped vcf data will be returned.

Author(s)

Qiang Hu

See Also

pos2seq

Examples

```
vcffile1 <- system.file("extdata", "1151HZ0001.flt.vcf", package="VPA")
vcfdata1 <- read.vcf(vcffile1)
vcffile2 <- system.file("extdata", "1151HZ0006.flt.vcf", package="VPA")
vcfdata2 <- read.vcf(vcffile2)
vcf <- filterpos(vcfdata1, position=cbind(vcfdata2$CHROM, vcfdata2$POS,
```

```
vcfdata2$POS), chr=FALSE)
```

`filtervcf` *Filter variants with user-specified quality criteria*

Description

VCF format file contains various score to assess the positional-level quality of variant and sequence call. The function filtervcf can be used to filter variants with user-specified quality criteria.

Usage

```
filtervcf(vcf, alter = NULL, alter.PL=20, alter.AD=3, alter.ADP=NULL,
QUAL = 20, DP = c(10,500), GQ = NULL, FILTER = NULL, INDEL = NULL)
```

Arguments

`vcf`	A VCF object for filtering.
`alter`	Logical value. If TRUE, the variant positions are kept. If FALSE, the variant positions are discarded. If NULL, the option will be ignored.
`alter.PL`	Phred-scaled genotype likelihoods of variant call to define a variant. The PL information can be extracted from PL column (both GATK and Samtools) in the VCF data.
`alter.AD`	The minimum depth of variant allele when alter is TRUE. The information of variant allele depth can be extracted from AD (GATK) or DP4 (Samtools) column in the VCF data.
`alter.ADP`	The minimum percentage of read depth containing variant allele.
`QUAL`	phred-scaled variant likelihoods of variant call. The QUAL information can be extracted from QUAL column (both GATK and Samtools) in the VCF data.
`DP`	The minimum and maximum of position-level read depth. The DP information can be extracted from DP column (both GATK and Samtools) in the VCF data.
`GQ`	Phred-scaled score for most likely genotype at position of interest. The GQ information can be extracted from GQ column (both GATK and Samtools) in the VCF data. If NULL, the option will be ignored.
`FILTER`	'NULL' or 'PASS'. The VCF format of variant call produced by GATK will label quality status of each position. This information can be extracted from FILTER column (GATK) in the VCF data. If the VCF data is produced by Samtools, FILTER column will contain empty information. If 'NULL' is set, all variants will be parsed. If 'PASS' is set, only variant with 'PASS' label will be parsed.
`INDEL`	Logical value. If TRUE, only INDELs are evaluated. If FALSE, only point variants are evaluated. If NULL, the option will be ignored.

Value

The input vcf data will be filtered by user-specified quality criteria. A list including filtered vcf data and dropped vcf data will return.

Author(s)

Qiang Hu

See Also

subvcf

Examples

```
#Filter alignment artifacts to get promising SNP
vcffile <- system.file("extdata", "1151HZ0001.flt.vcf", package="VPA")
vcfdata <- read.vcf(vcffile)
vcflt <- filtervcf(vcfdata, alter=TRUE, alter.AD=3, QUAL=20,
DP=c(10,500), GQ=20, INDEL=FALSE)$filtered
write.vcf(vcflt)
```

---

gefreq *Frequency analysis at gene level*

Description

To summarize the frequency of variants in gene level and estimate the statistical significance of frequency difference.

Usage

```
gefreq(vcf, method="fisher.test", p=1, level="gene", ref="hg19", ...)
```

Arguments

vcf	A varlist object.
method	The test method for mutated genes. This must be one of "fisher.test" and "chisq.test".
p	The maximum of the p values.
level	The annotation level for variants. It can be either "gene" (i.e., including intron region) or "exon" only (i.e., without including intron region).
ref	The version of reference genome, e.g. "hg19".
...	More arguments for the test method.

Value

A list contains a dataframe of frequencies between groups and an annotation list of each samples.

frequency	A data frame to list the gene name, variation distribution, variation frequency and p.value of all genes with variants across study groups.
otherfreq	A data frame to list the frequency results of the genes that are not in the specified level.
annotation	A list including gene annotations for the variants of each sample.

Author(s)

Qiang Hu

See Also

vcfreq

Examples

```
#gefreq(varRes1)
```

---

getref *Download reference gene annotation*

Description

To download reference gene table from UCSC genome browser golden path.

Usage

```
getref(ref="hg19")
```

Arguments

ref	The version of reference genome, such as hg18, hg19, etc.

Details

List of reference gene table can be found at: http://hgdownload.cse.ucsc.edu/goldenPath/.

Value

A data.frame from flat table of reference gene annotation.

Author(s)

Qiang Hu

See Also

Pos2Gene

Examples

```
#gereq(varRes1)
```

`LoadFiltering` *To load and filter variants in batch mode*

Description

To load data from study subjects and perform position-level quality filtering. The index.txt file contains group status and VCF file location of each subject. The function take index.txt file as input to load variant and sequence call files automatically.

Usage

```
LoadFiltering(file, datadir=NULL, filtering=TRUE, alter.PL=20,
alter.AD=3, alter.ADP=NULL, QUAL=20, DP=c(10,500), GQ=20, FILTER=NULL,
tabix="tabix", parallel=FALSE, pn=4, type=NULL, ...)
```

Arguments

`file`	Formatted input file including the annotation information of study subjects.
`datadir`	The work directory of the index file and variants data. If it is NULL, the absolute path of variants files should be provided in the annotation file.
`filtering`	Logical value. Whether to filter VCF data by specified quality criteria.
`alter.PL`	Phred-scaled genotype likelihoods of variant call to define a variant. The PL information can be extracted from PL column (both GATK and Samtools) in the VCF data.
`alter.AD`	The minimum depth of variant allele when alter is TRUE. The information of variant allele depth can be extracted from AD (GATK) or DP4 (Samtools) column in the VCF data.
`alter.ADP`	The minimum percentage of read depth containing variant allele.
`QUAL`	Phred-scaled variant likelihoods of variant call. The QUAL information can be extracted from QUAL column (both GATK and Samtools) in the VCF data.
`DP`	The minimum and maximum of position-level read depth. The DP information can be extracted from DP column (both GATK and Samtools) in the VCF data.
`GQ`	Phred-scaled score for most likely genotype at position of interest. The GQ information can be extracted from GQ column (both GATK and Samtools) in the VCF data. If NULL, the option will be ignored.
`FILTER`	'NULL' or 'PASS'. The VCF format of variant call produced by GATK will label quality status of each position. This information can be extracted from FILTER column (GATK) in the VCF data. If the VCF data is produced by Samtools, FILTER column will contain empty information. If 'NULL' is set, all variants will be parsed. If 'PASS' is set, only variant with 'PASS' label will be parsed.
`tabix`	The file path of executable tabix.
`parallel`	If TRUE, the function will run in parallel model.
`pn`	The CPU numbers to be used if parallel is TRUE.
`type`	MPI type. See detail in `help(sfInit)`
`...`	Arguments to pass to the method `sfInit` of the snowfall package.

Details

`file` The input file contains the annotation information of each sample. Each row is for one sample. The four columns are separated by tab, including sample name (required), group status (required), variant call file name (required) and sequence call file name (optional). Sample name column lists the sample name. Group status column lists the status (e.g., aggressive, benign or normal) of group each sample belongs to. Variant call file name column lists the path of VCF formatted variant call file. Sequence call file name column lists the path of compressed VCF sequence call file. The high-volume data in tab-delimited VCF formats can be efficiently compressed by bgzip program and retrieved through tabix program from open-source Samtools package. If the VCF format file is compressed by bgzip, tabix should be installed. The path of tabix should be specified in the function if it is not in the PATH system environment.

`Quality criteria` The detail of quality scores in VCF data can be found at http://www.1000genomes.org/node/101.

`parallel` This function will extract calls in sequential mode. If parallel is true, the function will extract calls in parallel mode. The package `Rmpi` and `snowfall` are required for parallel mode.

Value

The value returned is a varlist, including `vcflist`, `VarIndex` and `Samples`.

`varlist`	A list of vcf objects, one for each sample. If the filtering is true, the variant data are filtered by specified quality criteria.
`VarIndex`	The indexes for all variant positions. TRUE denotes the presence of variant. FALSE denotes the absence of variant. NA denotes low coverage.
`Sample`	Samples annotation from the input index file.

Author(s)

Qiang Hu

See Also

`filtervcf`

Examples

```
setwd(system.file("extdata", package="VPA"))
varflt <- LoadFiltering(file="index1.txt", filtering=TRUE, alter.PL=20,
alter.AD=3)
pattern <- cbind(A=c(1/4,1), B=c(0,0))
varRes1 <- Patterning(varflt, pattern, var.PL=c(FALSE, TRUE))
```

`Patterning` *Extract variants with user-specified variant pattern*

Description

To prioritize variants in user-specified frequency pattern.

Usage

```
Patterning(x, pattern, paired=FALSE, not.covered=NULL, var.PL=NULL)
```

Arguments

`x`	A `varlist` class data from the function `LoadFiltering`.
`pattern`	The variant frequency matrix. Each column of the matrix is defined as the minimum and maximum value of variant frequency for each group of interest.
`paired`	Logical value. Whether cases and controls are paired. If paired is TRUE, control group label in index file should be marked as control. Sample names should be matched between case and its matched control.
`not.covered`	Logical value for the position with sequence coverage less than specified depth. If TRUE, such low-coverage positions will be considered as variant. If FALSE, such low-coverage positions will be considered as reference. If it's NULL (default), such low-coverage positions will be filtered.
`var.PL`	A TRUE or FALSE vector for each group in the order of `pattern`. PL is used to label possible variant when `alter.PL` is not NULL in the function LoadFiltering. When filtering variants with specified frequency pattern, possible variants are considered as variants if TRUE. If FALSE, possible variants are considered as non-variants. If NULL, possible variants are considered as non-variants in all groups.

Details

This function is used to extract variant with user-specified frequency pattern across study subjects. The pattern matrix is specified by users in advance. The column names should be matched with sample group names.

Value

The value returned is a `varlist`, including `VarVCF`, `VarFrequency`, `Pattern` and `Samples`.

`VarVCF`	A list variants with user-specified frequency pattern in each sample.
`VarFrequency`	Variant frequencies for input positions.
`Sample`	Samples annotation from input file.

Author(s)

Qiang Hu

See Also

`LoadFiltering`

Examples

```
#setwd(system.file("extdata", package="VPA"))
#varflt <- LoadFiltering(file="index1.txt", filtering=TRUE, alter.PL=20,
#alter.AD=3)
#pattern <- cbind(A=c(1/4,1), B=c(0,0))
#varRes1 <- Patterning(varflt, pattern, var.PL=c(FALSE, TRUE))
```

`Pos2Gene` *Map sequencing variants to gene*

Description

The function is used to annotate a variant to its gene. The annotation information is based on the refseq table downloaded from the UCSC genome browser.

Usage

```
Pos2Gene(chr, pos, level="gene", show.dist=FALSE, ref="hg19")
```

Arguments

`chr`	Chromosome name of the variant, such as 'chr1' or '1'.
`pos`	The location of variant in the Chromosome.
`level`	The annotation level for variants. It can be either "gene" (i.e., including intron region) or "exon" (i.e., without including intron region).
`show.dist`	Logical value. When a position is mapped as inter-gene, whether to show the distances from the two genes.
`ref`	The version of reference genome.

Value

A vector including gene annotation results.

Author(s)

Qiang Hu

See Also

`getref`

Examples

```
#Pos2Gene("1", "1000000", level="exon", show.dist=TRUE, ref="hg19")
```

pos2seq *Positions to sequencing calls*

Description

Function to retrieve variants or sequence calls of interested positions from tabix indexed files.

Usage

```
pos2seq(Pos, Seqfile, file="", tabix="tabix", region = 5000)
```

Arguments

Pos	A two columns data.frame or matrix includes chromosome and position for each variant (or sequence) call.
Seqfile	A tabix indexed VCF file include all variant (or sequence) calls to retrieve.
file	File path to write out the retrieved results as a plain file in VCF format.
tabix	The path of tabix function. if NULL, scanTabix function from Rsamtools will be used instead.
region	The number of positions for tabix to retrieve at the same time. Too big number will not work for tabix. The default is 5000.

Details

pos2seq requires tabix function from SAMtools. The path of tabix could be specified in the optional argument of function if it is not in the PATH system environment. More details: http://samtools.sourceforge.net/tabix.shtml.

Value

A list includes header and VCF data. The results can also be outputted as a plain text file in VCF format.

Author(s)

Qiang Hu

See Also

LoadFiltering

Examples

```
vcffile <- system.file("extdata", "1151HZ0001.flt.vcf", package="VPA")
vcfdata <- read.vcf(vcffile)

##extract calls from tabix indexed data
Pos <- cbind(vcfdata$CHROM, vcfdata$POS)
gzfile <- system.file("extdata", "1151HZ0006.vcf.gz", package="VPA")
calls <- pos2seq(Pos, gzfile)
```

`read.vcf` *Read VCF file*

Description

Load VCF format file of variant and/or sequence calls into a flexible VCF object in R environment.

Usage

```
read.vcf(file, VCF=NULL, INFOID = NULL, FORMATID = NULL, ...)
```

Arguments

`file`	VCF format file of sequence calls from tools such as samtools and GATK.
`VCF`	An object list from the function `pos2seq` output.
`INFOID`	Characters. Only specified elements in INFO column of VCF file will be read.
`FORMATID`	Characters. Only specified elements in FORMAT column of VCF file will be read.
`...`	not used.

Value

A vcf object. Also it is a list. Each element of the list is from a column of the VCF file. See HEAD for details.

Author(s)

Qiang Hu

See Also

`write.vcf`

Examples

```
##read example vcf file in data directory.
vcffile <- system.file("extdata", "1151HZ0001.flt.vcf", package="VPA")
vcfdata <- read.vcf(vcffile)
summary(vcfdata)
write.vcf(vcfdata)
```

subvcf *Subset of VCF data*

Description

To obtain a subset of a vcf object data.

Usage

```
subvcf(vcf, CHR = NULL, POS = NULL, CHRPOS=NULL, samples = NULL, TF =
NULL)
```

Arguments

vcf	A vcf object data.
CHR	Chromosome. To get the subset of input vcf data based on specified chromosome(s).
POS	Position. To get the subset of input vcf data based on specified positions.
CHRPOS	Position within a chromosome separated by colon.
samples	To specify sample(s) of interest.
TF	A vector of logical values. To define which of the corresponding positions will be extracted.

Value

A vcf object returned.

Author(s)

Qiang Hu

See Also

filtervcf

Examples

```
vcffile <- system.file("extdata", "1151HZ0001.flt.vcf", package="VPA")
vcfdata <- read.vcf(vcffile)

#extract calls in position 1:985999
subvcf(vcfdata, CHRPOS="1:985999")

#extract calls by TF
tf <- c(rep(TRUE, 10), rep(FALSE, length(vcfdata$POS)-10))
subvcf(vcfdata, TF=tf)
```

`vcfreq` *Variant frequency of a varlist*

Description

To summarize the frequency of variants and estimate the statistical significance of frequency difference.

Usage

```
vcfreq(vcf, method="fisher.test", p=1, ...)
```

Arguments

`vcf`	A varlist object.
`method`	The allele frequency test method to be used. This must be one of "fisher.test" and "chisq.test".
`p`	The maximum of the p values.
`...`	More arguments for the test method.

Value

A data frame to list the position, REF, genotypes, variant allele frequencies and p.value of all variants across study groups.

Author(s)

Qiang Hu

See Also

`gefreq`

Examples

```
##read example vcf file in data directory.
#vcfreq(varRes1)
```

`write.vcf` *Write VCF object*

Description

To write a vcf object to VCF format file

Usage

```
write.vcf(x, file = "", HEAD=TRUE, ...)
```

Arguments

`x`	A vcf object.
`file`	Character. The file (including path) to which a vcf object will be written.
`HEAD`	Logical value. If TRUE, head information will be output.
`...`	not used.

Author(s)

Qiang Hu

See Also

`read.vcf`

Examples

```
vcffile <- system.file("extdata", "1151HZ0001.flt.vcf", package="VPA")
vcfdata <- read.vcf(vcffile)
summary(vcfdata)
write.vcf(vcfdata)
```

Index

∗Topic **annotation**
  Pos2Gene, 10
∗Topic **filter**
  filterpos, 2
  filtervcf, 4
∗Topic **frequency**
  gefreq, 5
  vcfreq, 14
∗Topic **package**
  VPA-package, 2
∗Topic **pattern**
  Patterning, 9
∗Topic **reference genome**
  getref, 6
∗Topic **subset**
  subvcf, 13
∗Topic **tabix**
  pos2seq, 11
∗Topic **variant**
  LoadFiltering, 7
  Patterning, 9
∗Topic **vcf**
  read.vcf, 12
  write.vcf, 15

filterpos, 2
filtervcf, 4, *8*, *13*

gefreq, 5, *14*
getref, 6, *10*

LoadFiltering, 7, *9*, *11*

Patterning, 9
Pos2Gene, *6*, 10
pos2seq, *3*, 11
print.vcf *(write.vcf)*, 15

read.vcf, 12, *15*

subvcf, *5*, 13
summary.vcf *(write.vcf)*, 15

vcfreq, *6*, 14
VPA *(VPA-package)*, 2
VPA-package, 2

write.vcf, *12*, 15