Package 'ensemblVEP'

October 9, 2013

Version 1.0.5

Title R Interface to Ensembl Variant Effect Predictor

Author Valerie Obenchain <vobencha@fhcrc.org>,

Maintainer Valerie Obenchain <vobencha@fhcrc.org>

Depends methods, BiocGenerics, GenomicRanges, VariantAnnotation

Imports Biostrings, IRanges

Suggests RUnit, BiocGenerics

biocViews Annotation, Bioinformatics

Description Query the Ensembl Variant Effect Predictor via the perl API

SystemRequirements Ensembl VEP (API version 73) and the Perl package DBD::mysql must be installed. See the package README and Ensembl web site,http://www.ensembl.org/info/docs/variation/vep/index.html for installation instructions.

License Artistic-2.0

LazyLoad yes

R topics documented:

ensemblVEP *Query Ensembl Variant Effect Predictor*

Description

Retrieve variant annotation data from the Ensembl Variant Effect Predictor (VEP).

Usage

```
## S4 method for signature 'character'
ensemblVEP(file, param=VEPParam(), ...)
```

Arguments

file	A character specifying the full path to the file, including the file name. Valid input file types are described on the Ensembl VEP web page. http://www.ensembl.org/info/docs/variation/vep/vep_formats.html
param	An instance of VEPParam specifying runtime options.
...	Additional arguments passed to methods.

Details

The Ensembl VEP tool is described in detail on the home page. http://www.ensembl.org/info/docs/variation/vep/index.html

The ensemblVEP function wraps the perl API and requires a local install of the Ensembl VEP available in the user's path. The VEPParam class provides a way to specify runtime options. Results are returned from Ensembl VEP as GRanges (default) or VCF objects. Alternatively, results can be written directly to a file.

Value

Default behavior returns a GRanges object. Options can be set to return a VCF object or write a file to disk.

Author(s)

Valerie Obenchain <vobencha@fhcrc.org>

References

Ensembl VEP Home: http://www.ensembl.org/info/docs/variation/vep/index.html

Human Genome Variation Society (hgvs): http://www.hgvs.org/mutnomen/

See Also

VEPParam-class

Examples

```
## ---------------------------------------------------------------------
## Results returned as GRanges or VCF objects
## ---------------------------------------------------------------------
## The default behavior returns a GRanges with the consequence
## data as metadata columns.
file <- system.file("extdata", "ex2.vcf", package="VariantAnnotation")
gr <- ensemblVEP(file)
gr[1:3]

## When the 'vcf' option is TRUE, a VCF object is returned.
myparam <- VEPParam(dataformat=c(vcf=TRUE))
vcf <- ensemblVEP(file, param=myparam)
vcf

## The consequence data are returned as the 'CSQ' column in info.
info(vcf)$CSQ

## To parse this column use parseCSQToGRanges().
csq <- parseCSQToGRanges(vcf)
head(csq, 4)

## The columns returned are controlled by the 'fields' option.
## By default all fields are returned. See ?VEPParam for details.

## When comparing ensemblVEP() results to the data in the
## input vcf we see that variant 20:1230237 was not returned.
vcf_input <- readVcf(file, "hg19")
rowData(vcf_input)
rowData(vcf)

## This variant has no alternate allele and is called a
## monomorphic reference. The Ensembl VEP automatically
## drops these variants.
rowData(vcf)[,c("REF", "ALT")]

## ---------------------------------------------------------------------
## Results written to disk
## ---------------------------------------------------------------------
## Write a file to disk by providing a path and file name as 'output_file'.
## Different output file formats are specified using the 'dataformat'
## runtime options.

## Write a vcf file to myfile.vcf:
myparam <- VEPParam(dataformat=c(vcf=TRUE),
                    input=c(output_file="/path/myfile.vcf"))
## Write a gvf file to myfile.gvf:
myparam <- VEPParam(dataformat=c(gvf=TRUE),
                    input=c(output_file="/path/myfile.gvf"))

## ---------------------------------------------------------------------
## Runtime options
```

```
## ------------------------------------------------------------------------
## All runtime options are controlled by specifying a VEPParam.
## See ?VEPParam for complete details.
param <- VEPParam()

## Logical options are turned on/off with TRUE/FALSE. By
## default, 'quiet' is FALSE.
basic(param)$quiet

## Setting 'quiet' to TRUE will suppress all status and warnings.
basic(param)$quiet <- TRUE

## Characater options are turned on/off by specifying a character
## value or an empty character (i.e., character()). By default no
## 'sift' results are returned.
output(param)$sift

## Setting 'sift' to 'b' will return both predictions and scores.
output(param)$sift <- 'b'

## Return 'sift' to the original state of no results returned.
output(param)$sift <- character()
```

`parseCSQToGRanges` *Parse the CSQ column of a VCF object into a GRanges object*

Description

Parse the CSQ column in a VCF object returned from the Ensembl Variant Effect Predictor (VEP).

Usage

```
## S4 method for signature 'character'
parseCSQToGRanges(x, VCFRowID=character(), ...)
## S4 method for signature 'VCF'
parseCSQToGRanges(x, VCFRowID=character(), ...)
```

Arguments

`x`	The `character` name of a vcf file on disk or a `VCF` object
`VCFRowID`	A `character` vector of rownames from the original VCF. When provided, the result includes a metadata column named 'VCFRowID' which maps the result back to the row (variant) in the original VCF. When `VCFRowID` is not provided no 'VCFRowID' column is included.
`...`	Arguments passed to other methods. Currently not used.

Details

When ensemblVEP returns a VCF object, the consequence data are returned unparsed in the 'CSQ' INFO column. parseCSQToGRanges parses these data into a GRanges object that is expanded to match the dimension of the 'CSQ' data. Because each variant can have multiple matches, the ranges in the GRanges are repeated.

If rownames from the original VCF are provided as VCFRowID a metadata column is included in the result that maps back to the row (variant) in the original VCF.

Value

Returns a GRanges object with consequence data as the metadata columns.

Author(s)

Valerie Obenchain <vobencha@fhcrc.org>

References

Ensembl VEP Home: http://www.ensembl.org/info/docs/variation/vep/index.html

See Also

ensemblVEP VEPParam-class

Examples

```
file <- system.file("extdata", "ex2.vcf", package="VariantAnnotation")
vep <- ensemblVEP(file, param=VEPParam(dataformat=c(vcf=TRUE)))

## The returned 'CSQ' data are unparsed.
info(vep)$CSQ

## Parse into a GRanges and include the 'VCFRowID' column.
vcf <- readVcf(file, "hg19")
csq <- parseCSQToGRanges(vep, VCFRowID=rownames(vcf))
csq[1:4]
```

VEPParam-class *VEPParam objects*

Description

A VEPParam object is a container for storing runtime options for the the Ensembl Variant Effect Predictor.

Details

The VEPParam class stores runtime options for the ensemblVEP function which queries the Ensembl Variant Effect Predictor (VEP). Brief descriptions of the runtime options are provided below. For complete details, see the Ensembl VEP web page.

http://www.ensembl.org/info/docs/variation/vep/vep_script.html#running

Data in the VEPParam are organized into the following categories, 'basic', 'input', 'cache', 'output', 'identifier', 'colocatedVariants', 'dataformat', 'filterqc', 'database' and 'advanced'. Each category is a list of runtime options. logical options are turned on/off with TRUE and FALSE. character and numeric are 'on' when a character string is provided and 'off' when they contain an empty value (i.e., character() or numeric().

basic list of the following options:

- verbose: logical, default FALSE; output status messages
- quiet: logical, default FALSE; suppress status/warnings
- no_progress: logical, default FALSE; don't show progress bars
- config: character, default character(); name of config file
- everything: logical, default FALSE; shortcut to switch on 12 options (sift, polyphen, ccds, hgvs, hgnc, numbers, domains, regulatory, cell_type, canonical, protein and gmaf).
- fork: numeric, default numeric(); enable forking

input list of the the following options:

- species: character, default 'homo_sapiens'; species for the data
- format: character, default character(); one of the following input file formats, 'ensembl', 'vcf', 'pileup', 'hgvs', 'id' or 'vep'. By default the script auto-detects the input file format.
- output_file: character, default writes to temp file; path and file name of output file
- force_overwrite: logical, default FALSE; overwrite the output file if it currently exists
- stats_file: character, default character(); summary stats file name
- no_stats: logical, default FALSE; do not generate a stats file
- stats_text: logical, default FALSE; generate a plain text stats file instead of html
- html: logical, default FALSE; generate html version of the output file

cache list of the following options:

- cache: logical, default FALSE; enable use of cache
- dir: character, default '$HOME/.vep/'; cache/plugin to be used
- dir_cache: character, default '$HOME/.vep/'; cache to be used
- dir_plugins: character, default '$HOME/.vep/'; plugin to be used
- offline: logical, default FALSE; enable offline mode, no database connections will be made
- fasta: character, default character(); FASTA filename or directory to files to use for reference sequences

output list of the following options:

- sift: character, default character(); output prediction, score or both, valid strings are 'p', 's' or 'b'

- polyphen: `character`, default `character()`; output prediction, score or both, valid strings are 'p', 's' or 'b'
- regulatory: `logical`, default FALSE; identify overlaps with regulatory regions
- cell_type: `character`, default `character()`; only report regulatory regions found in the given cell type(s)
- custom: `character`, default `character()`; name of custom annotation file to add to output. Currently only a single annotation is supported.
- plugin: `character`, default `character()`; name of plugin module. Currently only a single module is supported.
- individual: `character`, default `character()`; consider only alternate alleles present in the genotypes of 'all' or a character vector of specified individuals
- phased: `logical`, default FALSE; force VCF genotypes to be interpreted as phased
- allele_number: `logical`, default FALSE; identify allele number from VCF input (1=first ALT, 2=second ALT, etc.)
- total_length: `character`, default `character()`; cDNA, CDS and protein positions as position/length
- numbers: `logical`, default FALSE; output affectd exon and intron numbering, format is Number/Total
- domains: `logical`, default FALSE; output names of overlapping protein domains
- no_escape: `logical`, default FALSE; don't URI escape HGVS string
- terms: `character`, default 'so'; type of consequence terms to output, valid strings are 'ensembl' or 'so'

`identifiers list` of the following options:

- hgvs: `logical`, default FALSE; add hgvs ID's
- protein: `logical`, default FALSE; add Ensembl protein ID's
- symbol: `logical`, default FALSE; add gene symbol (e.g. HGNC) (where available) to the output
- ccds: `logical`, default FALSE; add CCDS transcript ID's
- canonical: `logical`, default FALSE; indicate if transcript is cononical transcript for the gene
- biotype: `logical`, default FALSE; add biotype of transcript
- xref_seq: `logical`, default FALSE; output aligned refseq mRNA ID

`colocatedVariants list` of the following options:

- check_existing: `logical`, default FALSE; check for co-located variants
- check_alleles: `logical`, default FALSE; when checking for co-located variants only report them if none of the alleles supplied are novel
- check_svs: `logical`, default FALSE; check for structural variants that overlap the input variants
- gmaf: `logical`, default FALSE; add global minor allele frequence (MAF) from 1000 Genomes Phase 1 data
- maf_1kg: `logical`, default FALSE; add MAF from continental populations of 1000 Genomes Phase 1 data; must be use with –cache
- maf_esp: `logical`, default FALSE; add MAF from NHLBI-ESP populations; must be used with –cache

- pubmed: `logical`, default FALSE; report Pubmed IDs for publications that cite existing variant; must be used with –cache
- failed: `logical`, default FALSE; when checking for co-located variants include or exclude variants that have been flagged as failed

`dataformat list` of the following options:

- vcf: `logical`, default FALSE; write output in vcf format
- gvf: `logical`, default FALSE; write output in gcf format
- original: `logical`, default FALSE; writes output as filtered set of input. Must be used with –filter.
- fields: `character`, default fields are 'Uploaded_variation', 'Location', 'Allele', 'Gene', 'Feature', 'Feature_type', 'Consequence', 'cDNA_position', 'CDS_position', 'Protein_position', 'Amino_acids', 'Codons' and 'Extra'. See http://www.ensembl.org/info/docs/variation/vep/vep_formats.html#sv for details.
- convert: `character`, default `character()`; converts input file to one of 'ensembl', 'vcf', or 'pileup'

`filterqc list` of the following options:

- check_ref: `logical`, default FALSE; force check of supplied reference allele against the sequence stored in Ensembl Core database
- coding_only: `logical`, default FALSE; return consequences in coding regions only
- chr: `character`, default `character()`; select a subset of chromosomes to be analyzed
- no_intergenic: `logical`, default FALSE; do not include intergenic consequences
- most_severe: `logical`, default FALSE; output only most severe consequence per variation
- summary: `logical`, default FALSE; output a comma-separated list of all observed consequences per variation, transcript-specific columns will be left blank
- per_gene: `logical`, default FALSE; output only the most severe consequence per gene
- filter_common: `logical`, default FALSE; shortcut flag to turn on filters, See web page for details.
- check_frequency: `logical`, default FALSE; turn on frequency filtering, must also specify all of the –freq_* flags. See web page for details.
- freq_pop: `character`, default `character()`; population to use in frequency filter
- freq_freq: `numeric`, default `numeric()`; MAF to use in frequency filter
- freq_gt_lt: `character`, default `character()`; specify whether the frequency of the co-located variant must be greater than or less than the value specified. Values are 'gt' or 'lt'. in the `freq_freq` option.
- freq_filter: `character`, default `character()`; specify whether to exclude or include variants that pass the frequency filter. Values are 'exclude' or 'include'.
- filter: `character`, default `character()`; filter the output on consequence type. See web page for details.
- allow_non_variant: `logical`, default FALSE; when using VCF format as input and output, by default VEP will skip all non-variant lines of input (i.e., where the ALT is NULL). When this option is enabled, lines will be printed in the VCF output with no consequence data added.

`database list` of the following options:

- database: `logical`, default TRUE; enable the VEP to use local or remote databases
- host: `character`, default 'useast.ensembl.db.org'; database host
- user: `character` default `character()`; database user
- password: `character`, default `character()`; database password
- port: `numeric`, default `character()`; database port
- genomes: `logical`, default FALSE; override default connection settings with those for the Ensembl Genomces public MySQL server
- refseq: `logical`, default FALSE; use otherfeatures database to retrieve transcripts
- db_version: `numeric`, default `character()`; force connection to specific version
- registry: `character`, default `character()`; provide file to override default connection settings

`advanced list` of the following options:

- no_whole_genome: `logical`, default FALSE; run in non-whole genome mode, variants analyzed one at a time, no caching
- buffer_size: `numeric`, default 5000; internal buffer size corresponding to number of variations read into memory simultaneously
- write_cache: `logical`, default FALSE; enable writing to the cache
- build: `character`, default `character()`; build cache for the selected species from the database (See –chr flag)
- compress: `character`, default `character()`; specify utility to decompress cached files (zcat is default)
- skip_db_check: `logical`, default FALSE; force the script to use a cache built from a different host than specified with –host
- cache_region_size: `numeric`, default `numeric()`; size in base-pairs of the region covered by one file in the cache, see full description of this flag on the web site for details

Constructor

`VEPParam(basic=basicOpts(), input=inputOpts(), cache=cacheOpts(), output=outputOpts(),`
Creates a `VEPParam` object.

`basic list` of basic options

`input list` of input options

`cache list` of cache options

`output list` of output options

`identifier list` of identifier options

`colocatedVariants list` of colocatedVariants options

`dataformat list` of dataformat options

`filterqc list` of filterqc options

`database list` of database options

`advanced list` of advanced options

Accessors

In the following code, x is a VEPParam object and value is a named list or character vector.

basic(x), basic(x) <- value

input(x), input(x) <- value

cache(x), cache(x) <- value

output(x), output(x) <- value

identifier(x), identifier(x) <- value

colocatedVariants(x), colocatedVariants(x) <- value

dataformat(x), dataformat(x) <- value

filterqc(x), filterqc(x) <- value

database(x), database(x) <- value

advanced(x), advanced(x) <- value

Helper functions

These functions create a list of runtime options and are used in the VEPParam constructor.

basicOpts()

inputOpts()

cacheOpts()

outputOpts()

identifierOpts()

colocatedVariantsOpts()

dataformatOpts()

filterqcOpts()

databaseOpts()

advancedOpts()

Author(s)

Valerie Obenchain <vobencha@fhcrc.org>

See Also

ensemblVEP

Examples

```
## The VEPParam has five slots. By default, no options are
## set for 'basic' or 'filterqc'.
param <- VEPParam()
param

## View the values in 'basic' and 'input'.
basic(param)
input(param)


## Change the value of the 'everything' to TRUE.
basic(param)$everything
basic(param)$everything <- TRUE
basic(param)$everything

## Replace multiple values with a named list.
basic(param) <- list(verbose=TRUE, config="myconfig.txt")
basic(param)

## Write the output to myfile.vcf instead of returning a VCF object.
## Return the sift and polyphen predictions only (not scores).
VEPParam(input=c(output_file="path/myfile.vcf"),
         output=c(sift="p", polyphen="p"))

## 'sift' and 'polyphen' are runtime options that require
## a character value, (i.e., 's', 'p', or 'b').
output(param)$sift

## To turn off 'sift' or 'polyphen' set the value to an
## empty character (i.e., character()).
output(param)$sift <- character()
```

Index