

Package ‘IsoformSwitchAnalyzeR’

September 30, 2022

Type Package

Title Identify, Annotate and Visualize Alternative Splicing and Isoform Switches with Functional Consequences from both short- and long-read RNA-seq data.

Version 1.19.0

Description Analysis of alternative splicing and isoform switches with predicted functional consequences (e.g. gain/loss of protein domains etc.) from quantification of all types of RNASeq by tools such as Kallisto, Salmon, StringTie, Cufflinks/Cuffdiff etc.

URL <http://bioconductor.org/packages/IsoformSwitchAnalyzeR/>

BugReports <https://github.com/kvittingseerup/IsoformSwitchAnalyzeR/issues>

License GPL (>= 2)

Depends R (>= 3.6), limma, DEXSeq, ggplot2

Imports methods, BSgenome, plyr, reshape2, gridExtra, Biostrings (>= 2.50.0), IRanges, GenomicRanges, DRIMSeq, RColorBrewer, rtracklayer, VennDiagram, DBI, grDevices, graphics, stats, utils, GenomeInfoDb, grid, tximport (>= 1.7.1), tximeta (>= 1.7.12), edgeR, futile.logger, stringr, dplyr, magrittr, readr, tibble, XVector, BiocGenerics, RCurl, Biobase

Suggests knitr, BSgenome.Hsapiens.UCSC.hg19, rmarkdown

VignetteBuilder knitr

Collate classes.R methods.R import_data.R test_isoform_switches.R analyze_ORF.R analyze_external_sequence_analysis.R analyze_switch_consequences.R isoform_plots.R plot_all_iso_switch.R high_level_functions.R tools.R analyze_alternative_splicing.R

biocViews GeneExpression, Transcription, AlternativeSplicing, DifferentialExpression, DifferentialSplicing, Visualization, StatisticalMethod, TranscriptomeVariant, BiomedicalInformatics, FunctionalGenomics, SystemsBiology, Transcriptomics, RNASeq, Annotation, FunctionalPrediction, GenePrediction, DataImport, MultipleComparison, BatchEffect, ImmunoOncology

RoxygenNote 6.0.1

git_url <https://git.bioconductor.org/packages/IsoformSwitchAnalyzeR>
git_branch master
git_last_commit 9791e03
git_last_commit_date 2022-04-26
Date/Publication 2022-09-30
Author Kristoffer Vitting-Seerup [cre, aut]
 (<<https://orcid.org/0000-0002-6450-0608>>)
Maintainer Kristoffer Vitting-Seerup <k.vitting.seerup@gmail.com>

R topics documented:

addORFfromGTF	3
analyzeAlternativeSplicing	5
analyzeCPAT	8
analyzeCPC2	10
analyzeIUPred2A	12
analyzeNetSurfP2	15
analyzeNovelIsoformORF	18
analyzeORF	21
analyzePFAM	24
analyzeSignalP	27
analyzeSwitchConsequences	29
CDSSet	38
createSwitchAnalyzeRlist	39
exampleData	43
extractConsequenceEnrichment	44
extractConsequenceEnrichmentComparison	47
extractConsequenceGenomeWide	49
extractConsequenceSummary	53
extractGeneExpression	55
extractSequence	57
extractSplicingEnrichment	61
extractSplicingEnrichmentComparison	64
extractSplicingGenomeWide	67
extractSplicingSummary	70
extractSwitchOverlap	72
extractSwitchSummary	74
extractTopSwitches	76
getCDS	78
importCufflinksFiles	79
importGTF	83
importIsoformExpression	86
importRdata	90
importSalmonData	97
isoformSwitchAnalysisCombined	99
isoformSwitchAnalysisPart1	102

isoformSwitchAnalysisPart2	104
isoformSwitchTestDEXSeq	108
isoformSwitchTestDRIMSeq	112
isoformToGeneExp	117
isoformToIsoformFraction	120
preFilter	122
prepareSalmonFileDataFrame	125
subsetSwitchAnalyzeRlist	127
switchPlot	128
switchPlotFeatureExp	131
switchPlotTopSwitches	134
switchPlotTranscript	137

Index**141**

addORFfromGTF	<i>Add CDS from a GTF file to a switchAnalyzeRlist.</i>
---------------	---

Description

Function for importing annotated CDS from a (gzipped) GTF file and add it to a switchAnalyzeRlist. This function is made to help annotate isoforms if you have performed (guided) de-novo isoform reconstruction (isoform deconvolution). This function will annotate all known transcripts but non of the novel transcripts identified by the isoform deconvolution. To analyse the novel transcripts the [analyzeNovelIsoformORF](#) function can be used after you have run this function.

Usage

```
addORFfromGTF(
  ### Core arguments
  switchAnalyzeRlist,
  pathToGTF,

  ### Advanced argument
  overwriteExistingORF = FALSE,
  onlyConsiderFullORF = FALSE,
  removeNonConventionalChr = FALSE,
  ignoreAfterBar = TRUE,
  ignoreAfterSpace = TRUE,
  ignoreAfterPeriod = FALSE,
  PTCDistance = 50,
  quiet = FALSE
)
```

Arguments

switchAnalyzeRlist	A switchAnalyzeRlist object.
pathToGTF	A string indicating the full path to the (gzipped or unpacked) GTF file which contains the the known annotation (aka from a official source) which was used to guided the transcript assembly (isoform deconvolution).
overwriteExistingORF	A logic indicating whether to overwirte existing ORF annoation. The main reason for the argument is to prevent accidental overwriting.
onlyConsiderFullORF	A logic indicating whether the ORFs added should only be added if they are fully annotated. Here fully annotated is defined as those that both have a annotated 'start_codon' and 'stop_codon' in the 'type' column (column 3). This argument is only considered if onlyConsiderFullORF=TRUE. Default is FALSE.
removeNonConvensionalChr	A logic indicating whether non-conventional chromosomes, here defined as chromosome names containing either a '_' or a period ('.'). These regions are typically used to annotate regions that cannot be associated to a specific region (such as the human 'chr1_gl000191_random') or regions quite different due to different haplotypes (e.g. the 'chr6_cox_hap2'). Default is FALSE.
ignoreAfterBar	A logic indicating whether to subset the isoform ids by ignoring everything after the first bar (" "). Useful for analysis of GENCODE files. Default is TRUE.
ignoreAfterSpace	A logic indicating whether to subset the isoform ids by ignoring everything after the first space (" "). Useful for analysis of gffutils generated GTF files. Default is TRUE.
ignoreAfterPeriod	A logic indicating whether to subset the gene/isoform is by ignoring everything after the first period ("."). Should be used with care. Default is FALSE.
PTCDistance	Only considered if addAnnotatedORFs=TRUE. A numeric giving the premature termination codon-distance: The minimum distance from the annotated STOP to the final exon-exon junction, for a transcript to be marked as NMD-sensitive. Default is 50
quiet	A logic indicating whether to avoid printing progress messages. Default is FALSE.

Details

The GTF file must have the following 3 annotation in column 9: 'transcript_id', 'gene_id', and 'gene_name'. Furthermore if addAnnotatedORFs is to be used the 'type' column (column 3) must contain the features marked as 'CDS'. If the onlyConsiderFullORF argument should work the GTF must also have 'start_codon' and 'stop_codon' annotated in the 'type' column (column 3).

Value

The switchAnalyzeRlist given as input is annotated with ORF information, as described in the details section of the [analyzeORF](#) documentation, and returned.

Author(s)

Kristoffer Vitting-Seerup

References

- This function : Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. Mol. Cancer Res. (2017).
- Information about NMD : Weischenfeldt J, et al: Mammalian tissues defective in nonsense-mediated mRNA decay display highly aberrant splicing patterns. Genome Biol. 2012, 13:R35.

See Also

[analyzeNovelIsoformORF](#)

Examples

```
### Please note the way of importing files in the following example with
# "system.file('pathToFile', package="IsoformSwitchAnalyzeR") is
# specialized way of accessing the example data in the IsoformSwitchAnalyzeR package
# and not something you need to do - just supply the string e.g.
# pathToGTF = "myAnnotation/knwon_annotation.gtf" to the functions

### Load example data
data("exampleSwitchListIntermediary")

### Remove ORF annotation
exampleSwitchListIntermediary$orfAnalysis <- NULL
exampleSwitchListIntermediary$isoformFeatures$PTC <- NULL

### Add ORF back in from GTF
exampleSwitchListIntermediary <- addORFfromGTF(
  exampleSwitchListIntermediary,
  pathToGTF = system.file("extdata/example.gtf.gz", package="IsoformSwitchAnalyzeR")
)
```

analyzeAlternativeSplicing

Analyse alternative splicing (including intron retention(s))

Description

These function utilize the analysis of alternative splicing previously implemented in the now deprecated splicer package which compares each isoform in a gene to the hypothetical pre-RNA generated by combining all the exons within a gene and classify the changes in alternative splicing. Not this version also only considered expressed (aka analyzed in the switchAnalyzeRlist).

Usage

```

analyzeAlternativeSplicing(
  switchAnalyzeRlist,
  onlySwitchingGenes=TRUE,
  alpha=0.05,
  dIFcutoff = 0.1,
  showProgress=TRUE,
  quiet=FALSE
)

analyzeIntronRetention(
  switchAnalyzeRlist,
  onlySwitchingGenes = TRUE,
  alpha = 0.05,
  dIFcutoff = 0.1,
  showProgress = TRUE,
  quiet = FALSE
)

```

Arguments

switchAnalyzeRlist	A switchAnalyzeRlist object.
onlySwitchingGenes	A logic indicating whether to only analyze genes with isoform switches (as indicated by the alpha and dIFcutoff parameters). Default is FALSE.
alpha	The Cutoff used on the FDR correct p-values (q-values) for calling significance. Default is 0.05.
dIFcutoff	Cutoff used for minimum changes in (absolute) isoform usage before an isoform is considered eligible for switch testing. This cutoff can remove cases where isoforms with extremely low IF values are deemed significant and thereby included in the downstream analysis. This cutoff is analogous to having a (log2) fold change in a normal differential expression analysis of genes to ensure the DE genes have a certain effect size. Default is 0.1 (10%).
showProgress	A logic indicating whether to make a progress bar (if TRUE) or not (if FALSE). Default is TRUE.
quiet	A logic indicating whether to avoid printing progress messages. Default is FALSE

Details

The `analyzeIntronRetention()` is just a convenient wrapper for the `analyzeIntronRetention()` function to ensure backward compatibility.

Alternative splicing (including alternative transcription start sites (ATSS) and alternative transcription termination sites (ATTS)) are classified for each isoform comparing that isoform to the hypothetical pre-RNA generated by combining all the exons (after exclusion of retained introns) within

a gene. Retained introns is defined as when one "exon" of one isoform overlaps two separate exons in other isoform.

Since the comparison is to the hypothetical pre-RNA the interpretation of an event is as follows:

- ES: Exon Skipping. Compared to the hypothetical pre-RNA a single exon was skipped in the isoform analyzed (for every ES event annotated).
- MEE: Mutually exclusive exon. Special case were two isoforms from the same gene contains two mutually exclusive exons and which are not found in any of the other isoforms from that gene.
- MES: Multiple Exon Skipping. Compared to the hypothetical pre-RNA multiple consecutive exon was skipped in the isoform analyzed (for every MES event annotated).
- IR: Intron Retention. Compared to the hypothetical pre-RNA an intron was retained in the isoform analyzed.
- A5: Alternative 5' end donor site. Compared to the hypothetical pre-RNA an alternative 5' end donor site was used. Since it is compared to the pre-RNA, the donor site used is per definition more upstream than the the pre-RNA (the upstream exon is shorter).
- A3: Alternative 3' end acceptor site. Compared to the hypothetical pre-RNA an alternative 3' end acceptor site was used. Since it is compared to the pre-RNA, the donor site used is per definition more downstream than the the pre-RNA (the downstream exon is shorter).
- ATSS: Alternative Transcription Start Sites. Compared to the hypothetical pre-RNA an alternative transcription start sites was used. Since it is compared to the pre-RNA, the ATSS site used is per definition more downstream than the the pre-RNA .
- ATTS: Alternative Transcription Termination Sites. Compared to the hypothetical pre-RNA an alternative transcription Termination sites was used. Since it is compared to the pre-RNA, the ATTS site used is per definition more upstream than the the pre-RNA.

Value

A `switchAnalyzeRlist` where the column IR indicating the number of Intron Retentions found in each transcript have been added to the `isoform_features` entry. NA is used if the transcript was not analyzed. Furthermore a data.frame (called 'AlternativeSplicingAnalysis'), where for each `isoform_id` containing the number of alternative splicing events found as well as the genomic coordinates of the affected region(s), is added to the `switchAnalyzeRlist`. In this data.frame genomic coordinates for each splice event are separated by ";" except for cases where there are multiple MES, then each set of coordinates belonging to a MES is separated by ',' (and then the coordinates belong to a specific MES is separated by ';').

Author(s)

Kristoffer Vitting-Seerup

References

- Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. *Mol. Cancer Res.* (2017).
- Vitting-Seerup et al. IsoformSwitchAnalyzeR: Analysis of changes in genome-wide patterns of alternative splicing and its functional consequences. *bioRxiv* (2018).

See Also

```

extractSplicingSummary
extractSplicingEnrichment
extractSplicingEnrichmentComparison
extractSplicingGenomeWide

```

Examples

```

### Load data
data("exampleSwitchListIntermediary")

### Perform analysis
exampleSwitchListAnalyzed <- analyzeAlternativeSplicing(exampleSwitchListIntermediary, quiet=TRUE)

### Inspect result
head(exampleSwitchListAnalyzed$AlternativeSplicingAnalysis) # the first 6 does not have any intron retentions (IR)
table(exampleSwitchListAnalyzed$AlternativeSplicingAnalysis$IR) # there appear to be 7 transcripts that have an in

```

analyzeCPAT

Import Result of External Sequence Analysis

Description

Allows for easy integration of the result of CPAT (external sequence analysis of coding potential) in the IsoformSwitchAnalyzeR workflow. Please note that due to the 'removeNoncodinORFs' option we recommend using analyzeCPAT before analyzePFAM and analyzeSignalP if you have predicted the ORFs with analyzeORF. This is an alternative to analyzing CPC2 results with analyzeCPC2.

Usage

```

analyzeCPAT(
  switchAnalyzeRlist,
  pathToCPATresultFile,
  codingCutoff,
  removeNoncodinORFs,
  quiet=FALSE
)

```

Arguments

```

switchAnalyzeRlist
  :A switchAnalyzeRlist object

pathToCPATresultFile
  : A string indicating the full path to the CPAT result file. See details for
  suggestion of how to run and obtain the result of the CPAT tool.

```

- `codingCutoff` : Numeric indicating the cutoff used by CPAT for distinguishing between coding and non-coding transcripts. The cutoff is dependent on species analyzed. Our analysis suggest that the optimal cutoff for overlapping coding and non-coding isoforms are 0.725 for human and 0.721 for mouse - HOWEVER the suggested cutoffs from the CPAT article (see references) derived by comparing known genes to random non-coding regions of the genome is 0.364 for human and 0.44 for mouse.
- `removeNoncodingORFs` : A logic indicating whether to remove ORF information from the isoforms which the CPAT analysis classifies as non-coding. This can be particular useful if the isoform (and ORF) was predicted de-novo but is not recommended if ORFs was imported from a GTF file. This will affect all downstream analysis and plots as both analysis of domains and signal peptides requires that ORFs are annotated (e.g. `analyzeSwitchConsequences` will not consider the domains (potentially) found by Pfam if the ORF have been removed).
- `quiet` : A logic indicating whether to avoid printing progress messages (incl. progress bar). Default is FALSE

Details

Notes for how to run the external tools: Use default parameters. If the webserver (<http://lilab.research.bcm.edu/cpat/>) was used download the tab-delimited result file (from the bottom of the result page). If a stand-alone version was just supply the path to the result file.

Please note that the `analyzeCPAT()` function will automatically only import the CPAT results from the isoforms stored in the `switchAnalyzeRlist` - even if many more are stored in the result file.

Value

Two columns are added to `isoformFeatures`: `'codingPotentialValue'` and `'codingPotential'` containing the predicted coding potential values and a logic indicating whether the isoform is coding or not respectively (based on the supplied cutoff).

Author(s)

Kristoffer Vitting-Seerup

References

- This function : Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. Mol. Cancer Res. (2017).
- CPAT : Wang et al. CPAT: Coding-Potential Assessment Tool using an alignment-free logistic regression model. Nucleic Acids Res. 2013, 41:e74.

See Also

[createSwitchAnalyzeRlist](#)
[extractSequence](#)
[analyzePFAM](#)
[analyzeNetSurfP2](#)

```
analyzeCPC2
analyzeSignalP
analyzeSwitchConsequences
```

Examples

```
### Load example data (matching the result files also store in IsoformSwitchAnalyzer)
data("exampleSwitchListIntermediary")
exampleSwitchListIntermediary

### Add CPAT analysis
exampleSwitchListAnalyzed <- analyzeCPAT(
  switchAnalyzeRlist = exampleSwitchListIntermediary,
  pathToCPATresultFile = system.file("extdata/cpat_results.txt", package = "IsoformSwitchAnalyzer"),
  codingCutoff = 0.364, # the coding potential cutoff suggested for human
  removeNoncodinORFs = TRUE # Because ORF was predicted de novo
)

exampleSwitchListAnalyzed
```

```
analyzeCPC2
```

```
Import Result of External Sequence Analysis
```

Description

Allows for easy integration of the result of CPC2 (external sequence analysis of coding potential) in the IsoformSwitchAnalyzer workflow. Please note that due to the 'removeNoncodinORFs' option we recommend using analyzeCPC2 before analyzePFAM and analyzeSignalP if you have predicted the ORFs with analyzeORF. This is an alternative to analyzing CPAT results with analyzeCPAT.

Usage

```
analyzeCPC2(
  switchAnalyzeRlist,
  pathToCPC2resultFile,
  codingCutoff = 0.5,
  removeNoncodinORFs,
  quiet=FALSE
)
```

Arguments

```
switchAnalyzeRlist
      :A switchAnalyzeRlist object

pathToCPC2resultFile
      : A string indicating the full path to the CPC2 result file. See details for
      suggestion of how to run and obtain the result of the CPAT tool.
```

- `codingCutoff` : Numeric indicating the cutoff used by CPC2 for distinguishing between coding and non-coding transcripts. The cutoff appears to be species independent. Default is 0.5.
- `removeNoncodingORFs` : A logic indicating whether to remove ORF information from the isoforms which the CPC2 analysis classifies as non-coding. This can be particularly useful if the isoform (and ORF) was predicted de-novo but is not recommended if ORFs were imported from a GTF file. This will affect all downstream analysis and plots as both analysis of domains and signal peptides requires that ORFs are annotated (e.g. `analyzeSwitchConsequences` will not consider the domains (potentially) found by Pfam if the ORF have been removed).
- `quiet` : A logic indicating whether to avoid printing progress messages (incl. progress bar). Default is FALSE

Details

Notes for how to run the external tools: Use default parameters and if required select the most similar species. If the [webserver](<http://cpc2.cbi.pku.edu.cn/batch.php>) (batch submission) was used, download the tab-delimited result file (via the "Download the result" button). If a stand-alone version was just supply the path to the result file.

Please note that the `analyzeCPC2()` function will automatically only import the CPC2 results from the isoforms stored in the `switchAnalyzeRlist` - even if many more are stored in the result file.

Value

Two columns are added to `isoformFeatures`: `'codingPotentialValue'` and `'codingPotential'` containing the predicted coding potential values and a logic indicating whether the isoform is coding or not respectively (based on the supplied cutoff).

Author(s)

Kristoffer Vitting-Seerup

References

- This function : Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. *Mol. Cancer Res.* (2017).
- CPC2 : Kang et al CPC2: a fast and accurate coding potential calculator based on sequence intrinsic features. *Nucleic Acids Res.* 2017

See Also

[createSwitchAnalyzeRlist](#)
[extractSequence](#)
[analyzePFAM](#)
[analyzeNetSurfP2](#)
[analyzeSignalP](#)
[analyzeCPAT](#)
[analyzeSwitchConsequences](#)

Examples

```

### Load example data (matching the result files also store in IsoformSwitchAnalyzer)
data("exampleSwitchListIntermediary")
exampleSwitchListIntermediary

### Add CPC2 analysis
exampleSwitchListAnalyzed <- analyzeCPC2(
  switchAnalyzeRlist = exampleSwitchListIntermediary,
  pathToCPC2resultFile = system.file("extdata/cpc2_result.txt", package = "IsoformSwitchAnalyzer"),
  removeNoncodingORFs = TRUE # because ORF was predicted de novo
)

exampleSwitchListAnalyzed

```

analyzeIUPred2A

Import Result of IUPred2A analysis

Description

Allows for easy integration of the result of IUPred2A (performing external sequence analysis of Intrinsically Disordered Regions (IDR) and Intrinsically Disordered Binding Regions (IDBR)) in the IsoformSwitchAnalyzer workflow. This function also supports using a sliding window to extract IDRs. Please note that due to the 'removeNoncodingORFs' option in analyzeCPAT and analyzeCPC2 we recommend using analyzeCPC2/analyzeCPAT before using analyzeIUPred2A, analyzePFAM and analyzeSignalP if you have predicted the ORFs with analyzeORF.

Usage

```

analyzeIUPred2A(
  switchAnalyzeRlist,
  pathToIUPred2AresultFile,
  smoothingWindowSize = 5,
  probabilityCutoff = 0.5,
  minIdrSize = 30,
  annotateBindingSites = TRUE,
  minIdrBindingSize = 15,
  minIdrBindingOverlapFrac = 0.8,
  showProgress = TRUE,
  quiet = FALSE
)

```

Arguments

switchAnalyzeRlist

A switchAnalyzeRlist object

pathToIUPred2AresultFile

A string indicating the full path to the IUPred2A result file. If multiple result files were created (multiple web-server runs) just supply all the paths as a vector of strings. Can both be gzipped or unpacked.

smoothingWindowSize	An integer indicating how large a sliding window should be used to calculate a smoothed (via mean) disordered probability score of a particular position in a peptide. This has as a smoothing effect which prevents IDRs from not being detected (or from being split into sub-IDRs) by a single residue with low probability. The trade off is worse accuracy of detecting the exact edges of the IDRs. To turn of smoothing simply set to 1. Default is 5 amino acids.
probabilityCutoff	A double indicating the cutoff applied to the (smoothed) disordered probability score (see "smoothingWindowSize" argument above) for calling a residue as "disordered". The default, 30 amino acids, is an accepted standard for long IDRs.
minIdrSize	An integer indicating how long a stretch of disordered amino acid constitute the "region" part of the Intrinsically Disordered Region (IDR) definition. The default, 30 amino acids, is an accepted standard for long IDRs.
annotateBindingSites	An logic indicating whether to also integrate the ANCHOR2 prediction of Intrinsically Disordered Binding Regions (IDBRs). See details for more info. Default is TRUE.
minIdrBindingSize	An integer indicating how long a stretch of binding site the "region" part of the Intrinsically Disordered Binding Regions (IDBR) is defined as. Default is 15 AA.
minIdrBindingOverlapFrac	An numeric indicating the min fraction of a predicted IDBR must also be within a IDR before the IDR is considered as a an IDR with a binding region. See details for more info. Default is 0.8 (aka 80%).
showProgress	A logic indicating whether to make a progress bar (if TRUE) or not (if FALSE). Default is TRUE.
quiet	A logic indicating whether to avoid printing progress messages (incl. progress bar). Default is FALSE

Details

Intrinsically Disordered Regions (IDR) are regions of a protein which does not have a fixed three-dimensional structure (opposite protein domains). Such regions are thought to play important roles in all aspects of biology (and when it goes wrong) through multiple different functional aspects. One such functional aspect is facilitating protein interactions via regions called Intrinsically Disordered Binding Regions (IDBR).

The IUPred2A webserver is somewhat strict with regards to the number of sequences in the files uploaded so we suggest multiple runs each with one of the the files contain subsets. See [extractSequence](#) for info on how to split the amino acid fasta files.

Notes for how to run the webserver:

- 1) Go to <https://iupred2a.elte.hu>
- 2) Upload the amino avoid file (_AA) created with extractSequence.
- 3) Add your email (you will receive a notification when the job is done).
- 4) Use default parameters ("IUPred2 long disorder" as prediction type and "ANCHOR2" for context dependent prediction):
- 4) In the email you receive when the results are done use the link given after "The text file can be

found here:" and save the result (right click on a blank space and use "save as" or use the keyboard shortcut "Ctrl/cmd + s" (or use wget etc to download in the first place)) 5) Supply a string indicating the path to the downloaded file directly to the "pathToIUPred2AresultFile" argument. If multiple files are created (multiple web-server runs) just supply the path to all of them as a string.

IDR are only added to isoforms annotated as having an ORF even if other isoforms exists in the result file. This means if you quantify the same isoform many times (many different Isoform-SwitchAnalyzeR workflows on the same organism) you can just run IUPred2A once on all isoforms and then supply the entire file to pathToIUPred2AresultFile.

Please note that the analyzeIUPred2A() function will automatically only import the IUPred2A results from the isoforms stored in the switchAnalyzeRlist - even if many more are stored in the result file.

Notes on Intrinsically Disordered Binding Regions (IDBR): As the prediction of IDR and IDBR are done by two different tools we require two things to annotate the Intrinsically Disordered Binding Regions (IDBR). Firstly the IDBR (predicted by ANCHOR2) must be a region of at least minIdrBindingSize amino acids (after the smoothing) - note this is different from the minIdrSize parameter. Secondly the fraction of the IDBR which overlaps the IDR predictions (done by IUPred2, again after smoothing) must be at least minIdrBindingOverlapFrac. When that is the case the IDR type will be annotated as "IDR_w_binding_region" instead of just "IDR". The current default parameters have not been rigorously tested and should be considered experimental.

Value

A column called 'idr_identified' is added to isoformFeatures containing a binary indication (yes/no) of whether a transcript contains any IDR regions or not. Furthermore the data.frame 'idr-Analysis' is added to the switchAnalyzeRlist containing positional data of each IDR identified.

The data.frame added have one row per isoform and contains the columns:

- isoform_id: The name of the isoform analyzed. Matches the 'isoform_id' entry in the 'isoformFeatures' entry of the switchAnalyzeRlist
- orf_aa_start: The start coordinate given as amino acid position (of the ORF).
- orf_aa_end: The end coordinate given as amino acid position (of the ORF).
- idr_type: A text string indicating the IDR type (one of 'IDR' or 'IDR_w_binding_region'.
- nr_idr_binding_sites_overlapping: (Only if annotateBindingSites=TRUE). An integer indicating the number of IDBRs predicted within the IDR.
- max_fraction_of_idr_binding_sites_overlapping: (Only if annotateBindingSites=TRUE). A fraction indicating the the largest overlap any IDBR had with the IDR.
- transcriptStart: The transcript coordinate of the start of the IDR.
- transcriptEnd: The transcript coordinate of the end of the IDR.
- idrStarExon: The exon index in which the start of the IDR is located.
- idrEndExon: The exon index in which the end of the IDR is located.
- idrStartGenomic: The genomic coordinate of the start of the IDR.
- idrEndGenomic: The genomic coordinate of the end of the IDR.

Author(s)

Kristoffer Vitting-Seerup

References

- This function: Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. Mol. Cancer Res. (2017).
- IUPred2A : Meszaros et al. IUPred2A: Context-dependent prediction of protein disorder as a function of redox state and protein binding. Nucleic Acids Res (2018).

See Also

[createSwitchAnalyzeRlist](#)
[extractSequence](#)
[analyzeCPAT](#)
[analyzeSignalP](#)
[analyzePFAM](#)
[analyzeNetSurfP2](#)
[analyzeSwitchConsequences](#)

Examples

```

### Load example data (matching the result files also store in IsoformSwitchAnalyzeR)
data("exampleSwitchListIntermediary")
exampleSwitchListIntermediary

### Add IUPred2A analysis
exampleSwitchListAnalyzed <- analyzeIUPred2A(
  switchAnalyzeRlist = exampleSwitchListIntermediary,
  pathToIUPred2AresultFile = system.file("extdata/iupred2a_result.txt.gz", package = "IsoformSwitchAnalyzeR")
)

exampleSwitchListAnalyzed

```

analyzeNetSurfP2	<i>Import Result of NetSurfP2 analysis</i>
------------------	--

Description

Allows for easy integration of the result of NetSurfP2 (performing external sequence analysis which include Intrinsically Disordered Regions (IDR)) in the IsoformSwitchAnalyzeR workflow. This function also supports using a sliding window to extract IDRs. Please note that due to the 'removeNoncodingORFs' option in analyzeCPAT and analyzeCPC2 we recommend using analyzeCPC2/analyzeCPAT before using analyzeNetSurfP2, analyzePFAM and analyzeSignalP if you have predicted the ORFs with analyzeORF.

Usage

```

analyzeNetSurfP2(
  switchAnalyzeRlist,
  pathToNetSurfP2resultFile,

```

```

smoothingWindowSize = 5,
probabilityCutoff = 0.5,
minIdrSize = 30,
showProgress = TRUE,
quiet = FALSE
)

```

Arguments

switchAnalyzeRlist	A switchAnalyzeRlist object
pathToNetSurfP2resultFile	A string indicating the full path to the NetSurfP-2 result file. Can be gzipped. If multiple result files were created (multiple web-server runs) just supply all the paths as a vector of strings.
smoothingWindowSize	An integer indicating how large a sliding window should be used to calculate a smoothed (via mean) disordered probability score of a particular position in a peptide. This has as a smoothing effect which prevents IDRs from not being detected (or from being split into sub-IDRs) by a single residue with low probability. The trade off is worse accuracy of detecting the exact edges of the IDRs. To turn of smoothing simply set to 1. Default is 5 amino acids.
probabilityCutoff	A double indicating the cutoff applied to the (smoothed) disordered probability score (see "smoothingWindowSize" argument above) for calling a residue as "disordered". The default, 30 amino acids, is an accepted standard for long IDRs.
minIdrSize	An integer indicating how long a stretch of disordered amino acid constitute the "region" part of the Intrinsically Disordered Region (IDR) definition. The default, 30 amino acids, is an accepted standard for long IDRs.
showProgress	A logic indicating whether to make a progress bar (if TRUE) or not (if FALSE). Default is TRUE.
quiet	A logic indicating whether to avoid printing progress messages (incl. progress bar). Default is FALSE

Details

Intrinsically Disordered Regions (IDR) are regions of a protein which does not have a fixed three-dimensional structure (opposite protein domains). Such regions are thought to play important roles in all aspects of biology (and when it goes wrong) through multiple different functional aspects - including facilitating protein interactions.

The NetSurfP web-server currently have a restriction of max 4000 sequences in the file uploaded. If you have more than that (one for each isoform in `summary(switchAnalyzeRlist)`) we recommend multiple runs each with one of the the files containing subsets - else you can just run the combined fasta file. See [extractSequence](#) for info on how to split the amino acid fasta files.

Notes for how to run the external tools:

Use default parameters. If you want to use the webserver it is easily done as follows: 1) Go to <http://>

[//www.cbs.dtu.dk/services/NetSurfP-2.0/](http://www.cbs.dtu.dk/services/NetSurfP-2.0/) 2) Upload the amino avoid file (_AA) created with extractSequence. 3) Submit your job. 4) Wait till job is finished (if you submit your email you will receive a notification). 5) In the top-right corner of the result site use the "Export All" button to download the results as a CNV file. 6) Supply a string indicating the path to the downloaded cnv file directly to the "pathToNetSurfP2resultFile" argument.

If you run NetSurfP-2 locally just use the "-csv" argument and provide the resulting csv file to the pathToNetSurfP2resultFile argument.

IDR are only added to isoforms annotated as having an ORF even if other isoforms exists in the file. This means if you quantify the same isoform many times you can just run NetSurfP2 once on all isoforms and then supply the entire file to analyzeNetSurfP2.

Please note that the analyzeNetSurfP2() function will automatically only import the NetSurfP-2 results from the isoforms stored in the switchAnalyzeRlist - even if many more are stored in the result file.

Value

A column called 'idr_identified' is added to isoformFeatures containing a binary indication (yes/no) of whether a transcript contains any protein domains or not. Furthermore the data.frame 'idrAnalysis' is added to the switchAnalyzeRlist containing positional data of each IDR identified.

The data.frame added have one row per isoform and contains the columns:

- isoform_id: The name of the isoform analyzed. Matches the 'isoform_id' entry in the 'isoformFeatures' entry of the switchAnalyzeRlist
- orf_aa_start: The start coordinate given as amino acid position (of the ORF).
- orf_aa_end: The end coordinate given as amino acid position (of the ORF).
- idr_type: A text string indicating the IDR type (one of 'IDR' or 'IDR_w_binding_region').
- transcriptStart: The transcript coordinate of the start of the IDR.
- transcriptEnd: The transcript coordinate of the end of the IDR.
- idrStarExon: The exon index in which the start of the IDR is located.
- idrEndExon: The exon index in which the end of the IDR is located.
- idrStartGenomic: The genomic coordinate of the start of the IDR.
- idrEndGenomic: The genomic coordinate of the end of the IDR.

Author(s)

Kristoffer Vitting-Seerup

References

- This function: Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. Mol. Cancer Res. (2017).
- NetSurfP-2: Klausen et al: NetSurfP-2.0: improved prediction of protein structural features by integrated deep learning. BioRxiv (2018).

See Also

[createSwitchAnalyzeRlist](#)
[extractSequence](#)
[analyzeCPAT](#)
[analyzeSignalP](#)
[analyzePFAM](#)
[analyzeIUPred2A](#)
[analyzeSwitchConsequences](#)

analyzeNovelIsoformORF

Prediction of Isoform Open Reading Frames.

Description

For the subset of isoforms not already annotated with ORFs this function predicts the most likely Open Reading Frame (ORF) and the NMD sensitivity. This function is made to help annotate isoforms if you have performed (guided) de-novo isoform reconstruction (isoform deconvolution) and is supposed to be used after [addORFfromGTF](#) have been used to annotate the known transcript. If you did not do an isoform reconstruction there is no need to run this function as CDS will (read are supposed to) already be annotated by `importRdata()`.

Usage

```

analyzeNovelIsoformORF(
  ### Core arguments
  switchAnalyzeRlist,
  analysisAllIsoformsWithoutORF, # also analyse all those annoatated as without CDS in ref annottaion
  genomeObject = NULL,

  ### Advanced argument
  minORFlength = 100,
  orfMethod = 'longest.AnnotatedWhenPossible',
  PTCDistance = 50,
  startCodons = "ATG",
  stopCodons = c("TAA", "TAG", "TGA"),
  showProgress = TRUE,
  quiet = FALSE
)

```

Arguments

switchAnalyzeRlist

A switchAnalyzeRlist object.

analysisAllIsoformsWithoutORF

A logic indicating whether to also analyse isoforms annotated as having no ORF by the `addORFfromGTF` function.

genomeObject	A BSgenome object uses as reference genome (e.g. 'Hsapiens' for Homo sapiens). Only necessary if transcript sequences were not already added (via the 'isoformNtFasta' argument in importRdata() or the extractSequence function).
minORFlength	The minimum size (in nucleotides) an ORF must be to be considered (and reported). Please note that we recommend using CPAT to predict coding potential instead of this cutoff - it is simply implemented as a pre-filter, see analyzeCPAT . Default is 100 nucleotides, which >97.5% of Gencode coding isoforms in both human and mouse have.
orfMethod	A string indicating which of the 5 available ORF identification methods should be used. The methods are: <ul style="list-style-type: none"> • longest.AnnotatedWhenPossible : A merge between "longestAnnotated" and "longest" (see below). For all isoforms where CDS start positions from known isoform overlap, only these CDS starts are considered and the longest ORF is annotated (similar to "longestAnnotated"). All isoforms without any overlapping CDS start sites they will be analysed with the "longest" approach. • longest : Identifies the longest ORF in the transcript (after filtering via minORFlength). This approach is similar to what the CPAT tool uses in it's analysis of coding potential. • mostUpstream : Identifies the most upstream ORF in the transcript (after filtering via minORFlength). • longestAnnotated : Identifies the longest ORF (after filtering via minORFlength) downstream of an annotated translation start site (which are supplied via the cds argument). • mostUpstreamAnnoated : Identifies the ORF (after filtering via minORFlength) downstream of the most upstream overlapping annotated translation start site (supplied via the cds argument). Default is longest.AnnotatedWhenPossible.
PTCDistance	A numeric giving the maximal allowed premature termination codon-distance: The minimum distance (number of nucleotides) from the STOP codon to the final exon-exon junction. If the distance from the STOP to the final exon-exon junction is larger than this the isoform to be marked as NMD-sensitive. Default is 50.
startCodons	A vector of strings indicating the start codons identified in the DNA sequence. Default is 'ATG' (corresponding to the RNA-sequence AUG).
stopCodons	A vector of strings indicating the stop codons identified in the DNA sequence. Default is c("TAA", "TAG", "TGA").
showProgress	A logic indicating whether to make a progress bar (if TRUE) or not (if FALSE). Defaults is TRUE.
quiet	A logic indicating whether to avoid printing progress messages (incl. progress bar). Default is FALSE

Details

This is a specialized function which wraps analyzeORF(). First it extract ORF start sites already annotated ORFs in the switchAnalyzeRlist. Then it analyses all isoforms in the switchAnalyzeRlist

not already annotated with an ORF (note the `analysisAllIsoformsWithoutORF` argument) using [analyzeORF](#) supplying the ORF start sites to the `cds` argument.

Value

For the isoforms analysed the ORF information in the `switchAnalyzeRlist` given as input updated and returned. See the details section of the [analyzeORF](#) documentation for full description.

Author(s)

Kristoffer Vitting-Seerup

References

- This function: Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. *Mol. Cancer Res.* (2017).
- Information about NMD: Weischenfeldt J, et al: Mammalian tissues defective in nonsense-mediated mRNA decay display highly aberrant splicing patterns. *Genome Biol.* 2012, 13:R35.

See Also

[addORFfromGTF](#)

Examples

```
### Load data
data("exampleSwitchListIntermediary")

### Select random isoforms to remove ORF annotation for
exampleSwitchListIntermediary$orfAnalysis$orf_origin <- 'Annotation'
nToRemove <- 25
rowsToModify <- sample(which(!is.na(exampleSwitchListIntermediary$orfAnalysis$orfTranscriptStart)), nToRemove)

### Remove ORF annotations
colsToModify <- which(!colnames(exampleSwitchListIntermediary$orfAnalysis) %in% c('isoform_id', 'orf_origin'))
exampleSwitchListIntermediary$orfAnalysis[
  rowsToModify,
  colsToModify
] <- NA
exampleSwitchListIntermediary$orfAnalysis$orf_origin[rowsToModify] <- 'not_annotated_yet'

### Predict ORF of missing isoforms using the ORF in other isoforms
tmp <- analyzeNovelIsoformORF(
  switchAnalyzeRlist = exampleSwitchListIntermediary,
  analysisAllIsoformsWithoutORF = TRUE
)
```

analyzeORF

*Prediction of Transcript Open Reading Frame.***Description**

Please note the vast majority of users are better off using the new [addORFfromGTF](#) + [analyzeNovellIsoformORF](#) annotation combo.

This function predicts the most likely Open Reading Frame (ORF) and the NMD sensitivity of the isoforms stored in a `switchAnalyzeRlist` object. This functionality is made to help annotate isoforms if you have performed (guided) de-novo isoform reconstruction (isoform deconvolution). Else you should use the annotated CDS (CoDing Sequence) typically obtained through one of the implemented import methods (see vignette for details).

Usage

```
analyzeORF(
  ### Core arguments
  switchAnalyzeRlist,
  genomeObject = NULL,

  ### Advanced argument
  minORFlength = 100,
  orfMethod = 'longest',
  cds = NULL,
  PTCDistance = 50,
  startCodons = "ATG",
  stopCodons = c("TAA", "TAG", "TGA"),
  showProgress = TRUE,
  quiet = FALSE
)
```

Arguments

<code>switchAnalyzeRlist</code>	A <code>switchAnalyzeRlist</code> object. n
<code>genomeObject</code>	A <code>Bsgenome</code> object uses as reference genome (e.g. 'Hsapiens' for Homo sapiens). Only necessary if transcript sequences were not already added (via the 'isoformNtFasta' argument in <code>importRdata()</code> or the <code>extractSequence()</code> function).
<code>minORFlength</code>	The minimum size (in nucleotides) an ORF must be to be considered (and reported). Please note that we recommend using CPAT to predict coding potential instead of this cutoff - it is simply implemented as a pre-filter, see analyzeCPAT . Default is 100 nucleotides, which >97.5% of Gencode coding isoforms in both human and mouse have.
<code>orfMethod</code>	A string indicating which of the 5 available ORF identification methods should be used. The methods are:

- `longest` : Identifies the longest ORF in the transcript (after filtering via `minORFlength`). This approach is similar to what the CPAT tool uses in its analysis of coding potential.
- `mostUpstream` : Identifies the most upstream ORF in the transcript (after filtering via `minORFlength`).
- `longestAnnotated` : Identifies the longest ORF (after filtering via `minORFlength`) downstream of an annotated translation start site (which are supplied via the `cds` argument).
- `mostUpstreamAnnotated` : Identifies the ORF (after filtering via `minORFlength`) downstream of the most upstream overlapping annotated translation start site (supplied via the `cds` argument).
- `longest.AnnotatedWhenPossible` : A merge between "longestAnnotated" and "longest". For all isoforms where CDS start positions overlap, only these CDS starts are considered and the longest ORF is annotated (similar to "longestAnnotated"). All isoforms without any overlapping CDS start sites they will be analysed with the "longest" approach.

Default is `longest`.

<code>cds</code>	Should not be used by end user. If analysis using known CDS start sites is wanted the user should use the combination of addORFfromGTF and analyzeNovelIsoformORF instead.
<code>PTCDistance</code>	A numeric giving the maximal allowed premature termination codon-distance: The minimum distance (number of nucleotides) from the STOP codon to the final exon-exon junction. If the distance from the STOP to the final exon-exon junction is larger than this the isoform to be marked as NMD-sensitive. Default is 50.
<code>startCodons</code>	A vector of strings indicating the start codons identified in the DNA sequence. Default is 'ATG' (corresponding to the RNA-sequence AUG).
<code>stopCodons</code>	A vector of strings indicating the stop codons identified in the DNA sequence. Default is <code>c("TAA", "TAG", "TGA")</code> .
<code>showProgress</code>	A logic indicating whether to make a progress bar (if TRUE) or not (if FALSE). Defaults is TRUE.
<code>quiet</code>	A logic indicating whether to avoid printing progress messages (incl. progress bar). Default is FALSE

Details

The function uses the genomic coordinates of the transcript model to extract the nucleotide sequence of the transcript from the supplied `BSgenome` object (reference genome). The nucleotide sequence is then used to predict the most likely ORF (the method is controlled by the `orfMethod` argument, see above). If the distance from the stop position (ORF end) to the final exon-exon junction is larger than the threshold given in `PTCDistance` (and the stop position does not fall in the last exon), the stop position is considered premature and the transcript is marked as NMD (nonsense mediated decay) sensitive in accordance with literature consensus (Weischenfeldt et al (see references)).

The Gencode reference annotation used here are GencodeV19, GencodeV24, GencodeM1 and GencodeM9. For more info see Vitting-Seerup et al 2017.

Value

A switchAnalyzeRlist where:

- 1: A columns called PTC indicating the NMD sensitivity have been added to the isoformFeatures entry of the switchAnalyzeRlist.
- 2: The transcript nucleotide sequence for all analyzed isoforms (in the form of a DNASTringSet object) have been added to the switchAnalyzeRlist in the ntSequence entry.
- 3: A data.frame containing the details of the ORF analysis have been added to the switchAnalyzeRlist under the name 'orfAnalysis'.

The data.frame added have one row pr isoform and contains 11 columns:

- isoform_id: The name of the isoform analyzed. Matches the 'isoform_id' entry in the 'isoformFeatures' entry of the switchAnalyzeRlist
- orfTranscriptStart: The start position of the ORF in transcript coordinators, here defined as the position of the 'A' in the 'AUG' start motif.
- orfTranscriptEnd: The end position of the ORF in transcript coordinates, here defined as the last nucleotide before the STOP codon (meaning the stop codon is not included in these coordinates).
- orfTranscriptLength: The length of the ORF
- orfStarExon: The exon in which the start codon is
- orfEndExon: The exon in which the stop codon is
- orfStartGenomic: The start position of the ORF in genomic coordinators, here defined as the the position of the 'A' in the 'AUG' start motif.
- orfEndGenomic: The end position of the ORF in genomic coordinates, here defined as the last nucleotide before the STOP codon (meaning the stop codon is not included in these coordinates).
- stopDistanceToLastJunction: Distance from stop codon to the last exon-exon junction
- stopIndex: The index, counting from the last exon (which is 0), of which exon is the stop codon is in.
- PTC: A logic indicating whether the isoform is classified as having a Premature Termination Codon. This is defined as having a stop codon more than PTCDistance (default is 50) nt upstream of the last exon exon junction.
- orf_origin: A column indicating where the ORF annotation originates form. Possible values are "Annotation" (imported from GTF), "Predicted" (indicating they were predicted) and "not_annotated_yet" indicting ORF have not been annotated yet.

NA means no information was available aka no ORF (passing the minORFlength filter) was found.

Author(s)

Kristoffer Vitting-Seerup

References

- This function : Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. Mol. Cancer Res. (2017).
- Information about NMD : Weischenfeldt J, et al: Mammalian tissues defective in nonsense-mediated mRNA decay display highly aberrant splicing patterns. Genome Biol. 2012, 13:R35.

See Also

```
createSwitchAnalyzeRlist
preFilter
isoformSwitchTestDEXSeq
isoformSwitchTestDRIMSeq
extractSequence
analyzeCPAT
```

Examples

```
### Prepare for orf analysis
# Load example data and prefilter
data("exampleSwitchList")
exampleSwitchList <- preFilter(exampleSwitchList)

# Perform test
exampleSwitchListAnalyzed <- isoformSwitchTestDEXSeq(exampleSwitchList, dIFcutoff = 0.3) # high dIF cutoff for fast

### analyzeORF
library(BSgenome.Hsapiens.UCSC.hg19)
exampleSwitchListAnalyzed <- analyzeORF(exampleSwitchListAnalyzed, genomeObject = Hsapiens)

### Explore result
head(exampleSwitchListAnalyzed$orfAnalysis)
head(exampleSwitchListAnalyzed$isoformFeatures) # PTC column added
```

analyzePFAM

Import Result of PFAM analysis

Description

Allows for easy integration of the result of Pfaam (external sequence analysis of protein domains) in the IsoformSwitchAnalyzeR workflow. Please note that due to the 'removeNoncodingORFs' option in analyzeCPAT and analyzeCPC2 we recommend using analyzeCPC2/analyzeCPAT before using analyzePFAM, analyzeNetSurfP2 and analyzeSignalP if you have predicted the ORFs with analyzeORF.

Usage

```
analyzePFAM(  
    switchAnalyzeRlist,  
    pathToPFAMresultFile,  
    showProgress=TRUE,  
    quiet=FALSE  
)
```

Arguments

switchAnalyzeRlist	A switchAnalyzeRlist object
pathToPFAMresultFile	A string indicating the full path to the Pfam result file(s). If multiple result files were created (multiple web-server runs) just supply all the paths as a vector of strings. See details for suggestion of how to run and obtain the result of the Pfam tool.
showProgress	A logic indicating whether to make a progress bar (if TRUE) or not (if FALSE). Default is TRUE.
quiet	A logic indicating whether to avoid printing progress messages (incl. progress bar). Default is FALSE

Details

A protein domain is a part of a protein which by itself can maintain a fixed three-dimensional structure. Protein domains are found in most proteins and usually have a specific function.

The PFAM webserver is quite strict with regards to the number of sequences in the files uploaded so we suggest multiple runs each with one of the the files containing subsets. See [extractSequence](#) for info on how to split the amino acid fasta files.

Notes for how to run the external tools:

Use default parameters. If you want to use the webserver it is easily done as follows: 1) Go to <https://www.ebi.ac.uk/Tools/hmmer/search/hmmsearch> 2) Switch to the the "Upload a File" tab. 3) Upload the amino acid file (_AA) created with `extractSequence` file and add your mail address - this is important because there is currently no way of downloading the web output so you need them to send the result to your email. 4) Check Pfam is selected in the "HMM database" window. 5) Submit your job. 6) Wait till you receive the email with the result (usually quite fast). 7) Copy/paste the result part of the (ONLY what is below the line starting with "seq id") into an empty plain text document (notepad, sublimetext TextEdit or similar (not word)). 8) Save the document and supply the path to that document to `analyzePFAM()`

To run PFAM locally you should use the `pfam_scan.pl` script as described in the readme at <ftp://ftp.ebi.ac.uk/pub/databases/Pfam/Tools/> and supply the path to the result file to `analyzePFAM()`.

Protein domains are only added to isoforms annotated as having an ORF even if other isoforms exists in the file. This means if you quantify the same isoform many times you can just run `pfam` once on all isoforms and then supply the entire file to `analyzePFAM()`.

Please note that the `analyzePFAM()` function will automatically only import the Pfam results from the isoforms stored in the `switchAnalyzeRlist` - even if many more are stored in the result file.

Value

A column called 'domain_identified' is added to isoformFeatures containing a binary indication (yes/no) of whether a transcript contains any protein domains or not. Furthermore the data.frame 'domainAnalysis' is added to the switchAnalyzeRlist containing the details about domain names(s) and position for each transcript (where domain(s) were found).

The data.frame added have one row per isoform and contains the columns:

- isoform_id: The name of the isoform analyzed. Matches the 'isoform_id' entry in the 'isoformFeatures' entry of the switchAnalyzeRlist
- orf_aa_start: The start coordinate given as amino acid position (of the ORF).
- orf_aa_end: The end coordinate given as amino acid position (of the ORF).
- hmm_acc: A id which pfam have given to the domain
- hmm_name: The name of the domain
- clan: The clan which the domain belongs to
- transcriptStart: The transcript coordinate of the start of the domain.
- transcriptEnd: The transcript coordinate of the end of the domain.
- pfamStarExon: The exon index in which the start of the domain is located.
- pfamEndExon: The exon index in which the end of the domain is located.
- pfamStartGenomic: The genomic coordinate of the start of the domain.
- pfamEndGenomic: The genomic coordinate of the end of the domain.

Furthermore depending on the exact tool used (local vs web-server) additional columns are added with information such as E score and type.

Author(s)

Kristoffer Vitting-Seerup

References

- This function : Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. Mol. Cancer Res. (2017).
- Pfam : Finn et al. The Pfam protein families database. Nucleic Acids Research (2014) Database Issue 42:D222-D230

See Also

[createSwitchAnalyzeRlist](#)
[extractSequence](#)
[analyzeCPAT](#)
[analyzeSignalP](#)
[analyzeNetSurfP2](#)
[analyzeSwitchConsequences](#)

Examples

```
### Load example data (matching the result files also store in IsoformSwitchAnalyzer)
data("exampleSwitchListIntermediary")
exampleSwitchListIntermediary

### Add PFAM analysis
exampleSwitchListAnalyzed <- analyzePFAM(
  switchAnalyzeRlist = exampleSwitchListIntermediary,
  pathToPFAMresultFile = system.file("extdata/pfam_results.txt", package = "IsoformSwitchAnalyzer"),
  showProgress=FALSE
)

exampleSwitchListAnalyzed
```

analyzeSignalP	<i>Import Result of SignalP Analysis</i>
----------------	--

Description

Allows for easy integration of the result of SignalP (external sequence analysis of signal peptides) in the IsoformSwitchAnalyzer workflow. Please note that due to the 'removeNoncodingORFs' option in analyzeCPAT and analyzeCPC2 we recommend using analyzeCPC2/analyzeCPAT before using analyzeSignalP, analyzeNetSurfP2, analyzePFAM if you have predicted the ORFs with analyzeORF.

Usage

```
analyzeSignalP(
  switchAnalyzeRlist,
  pathToSignalPresultFile,
  minSignalPeptideProbability = 0.5,
  quiet=FALSE
)
```

Arguments

switchAnalyzeRlist	A switchAnalyzeRlist object
pathToSignalPresultFile	A string indicating the full path to the summary SignalP result file(s). If multiple result files were created (multiple web-server runs) just supply all the paths as a vector of strings. See details for suggestion of how to run and obtain the result of the SignalP tool.
minSignalPeptideProbability	A numeric between 0 and 1 indicating the minimum probability for calling a signal peptide. Default is 0.5
quiet	A logic indicating whether to avoid printing progress messages (incl. progress bar). Default is FALSE

Details

A signal peptide is a short peptide sequence which indicate a protein is destined towards the secretory pathway.

The SignalP web-server is less stringent than PFAM with regards to the number of sequences in the files uploaded so we suggest trying the combined fasta file first - and if that does not work try the files containing subsets. See [extractSequence](#) for info on how to split the amino acid fasta files.

Notes for how to run the external tools: If using the web-server (<http://www.cbs.dtu.dk/services/SignalP/>) SignalP should be run with the parameter "Short output (no figures)" under "Output format" and one should select the appropriate "Organism group". When using a stand-alone version SignalP should be run with the '-f summary' option. If using the web-server the results can be downloaded using the "Downloads" bottom in the top-right corner where the user should select "Prediction summary" and supply the path to the resulting file to the pathToSignalResultFile argument. If a stand-alone version was just supply the path to the summary result file.

Please note that the `analyzeSignalP()` function will automatically only import the SignalP results from the isoforms stored in the `switchAnalyzeRlist` - even if many more are stored in the result file.

Also note that `analyzeSignalP` automatically subset SignalP results to only contain predictions with an annotated cleavage site (CS pos) and "Probable protein fragment" results are also removed.

Value

A column called 'signal_peptide_identified' is added to `isoformFeatures` containing a binary indication (yes/no) of whether a transcript contains a signal peptide or not. Furthermore the data.frame 'signalPeptideAnalysis' is added to the `switchAnalyzeRlist` containing the details of the signal peptide analysis.

The data.frame added have one row pr isoform and contains 6 columns:

- `isoform_id`: The name of the isoform analyzed. Matches the 'isoform_id' entry in the 'isoformFeatures' entry of the `switchAnalyzeRlist`
- `has_signal_peptide`: A text string indicating whether there is a signal peptide or not. Can be yes or no
- `network_used`: A text string indicating whether SignalP used the Neural Network (NN) optimized for proteins with trans-membrane sections (string='TM') or proteins without trans-membrane sections (string='noTM'). Per default, SignalP 4.1 uses the NN with TM as a pre-processor to determine whether to use TM or noTM in the final prediction (if 4 or more positions are predicted to be in a transmembrane state, TM is used, otherwise SignalP-noTM). Reference: <http://www.cbs.dtu.dk/services/SignalP/instructions.php>
- `aa_removed`: A integer giving the number of amino acids removed when the signal peptide is cleaved off.
- `transcriptCleaveAfter`: The transcript position of the last nucleotide in the isoform which is removed when the signal peptide is cleaved off.
- `genomicCleaveAfter`: The genomic position of the last nucleotide in the isoform which is removed when the signal peptide is cleaved off.

Author(s)

Kristoffer Vitting-Seerup

References

- This function : Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. Mol. Cancer Res. (2017).
- SignalP : Almagro et al. SignalP 5.0 improves signal peptide predictions using deep neural networks. Nat. Biotechnol (2019).

See Also

```
createSwitchAnalyzeRlist  
extractSequence  
analyzePFAM  
analyzeNetSurfP2  
analyzeCPAT  
analyzeSwitchConsequences
```

Examples

```
### Load example data  
data("exampleSwitchListIntermediary")  
exampleSwitchListIntermediary  
  
### Add SignalP analysis  
exampleSwitchListAnalyzed <- analyzeSignalP(  
  switchAnalyzeRlist = exampleSwitchListIntermediary,  
  pathToSignalPresultFile = system.file(  
    "extdata/signalP_results.txt",  
    package = "IsoformSwitchAnalyzerR")  
)  
  
exampleSwitchListAnalyzed
```

analyzeSwitchConsequences

Analyze Consequences of Isoform Switches

Description

This function extracts all isoforms with an absolute dIF change larger than dIFcutoff from genes with a significant isoform switch (as defined by alpha). For each gene these isoforms are then analyzed for differences in the functional annotation (defined by consequencesToAnalyze) by pairwise comparing the isoforms that are used more (switching up (dIF > 0)) with the isoforms that are used less (switching down (dIF < 0)). For each comparison a small report of the analyzed features is returned.

Usage

```

analyzeSwitchConsequences(
  switchAnalyzeRlist,
  consequencesToAnalyze=c(
    'intron_retention',
    'coding_potential',
    'ORF_seq_similarity',
    'NMD_status',
    'domains_identified',
    'IDR_identified',
    'IDR_type',
    'signal_peptide_identified'
  ),
  alpha=0.05,
  dIFcutoff=0.1,
  onlySigIsoforms=FALSE,
  ntCutoff=50,
  ntFracCutoff=NULL,
  ntJCSimCutoff=0.8,
  AaCutoff=10,
  AaFracCutoff=0.5,
  AaJCSimCutoff=0.9,
  removeNonConseqSwitches=TRUE,
  showProgress=TRUE,
  quiet=FALSE
)

```

Arguments

switchAnalyzeRlist

A `switchAnalyzeRlist` object containing the result of an isoform switch analysis (such as the one provided by `isoformSwitchTestDEXSeq`) as well as additional annotation data for the isoforms.

consequencesToAnalyze

A vector of strings indicating what type of functional consequences to analyze. Do note that there is bound to be some differences between isoforms (else they would be identical and not annotated as separate isoforms). See details for full list of usable strings and their meaning. Default is `c('intron_retention','coding_potential','ORF_seq_similarity')` (corresponding to analyze: intron retention, CPAT result, ORF AA sequence similarity, NMD status, protein domains annotated and signal peptides annotated by Pfam).

alpha

The cutoff which the FDR correct p-values (q-values) must be smaller than for calling significant switches. Default is 0.05.

dIFcutoff

The cutoff which the changes in (absolute) isoform usage must be larger than before an isoform is considered switching. This cutoff can remove cases where isoforms with (very) low dIF values are deemed significant and thereby included in the downstream analysis. This cutoff is analogous to having a cutoff on \log_2

fold change in a normal differential expression analysis of genes to ensure the genes have a certain effect size. Default is 0.1 (10%).

onlySigIsoforms

A logic indicating whether to only consider significant isoforms, meaning only analyzing genes where at least two isoforms which both have significant usage changes in opposite direction (quite strict) Naturally this only works if the isoform switch test used have isoform resolution (which the build in [isoform-SwitchTestDEXSeq](#) has). If FALSE all isoforms with an absolute dIF value larger than dIFcutoff in a gene with significant switches (defined by alpha and dIFcutoff) are included in the pairwise comparison. Default is FALSE (non significant isoforms are also considered based on the logic that if one isoform changes its contribution - there must be an equivalent opposite change in usage in the other isoforms from that gene).

ntCutoff

An integer indicating the length difference (in nt) a comparison must be larger than for reporting differences when evaluating 'isoform_length', 'ORF_length', '5_utr_length', '3_utr_length', 'isoform_seq_similarity', '5_utr_seq_similarity' and '3_utr_seq_similarity'. Default is 50 (nt).

ntFracCutoff

An numeric indicating the cutoff in length difference, measured as a fraction of the length of the downregulated isoform, a comparison must be larger than for reporting differences when evaluating 'isoform_length', 'ORF_length', '5_utr_length', '3_utr_length'. For example does 0.05 mean the upregulated isoform must be 5% longer/shorter before it is reported. NULL disables the filter. Default is NULL.

ntJCSimCutoff

An numeric (between 0 and 1) indicating the cutoff on Jaccard Similarity (JCsim) (see details) between the overlap of two nucleotide (nt) sequences. If the measured JCsim is smaller than this cutoff the sequences are considered different and reported as such. This cutoff affects the result of the 'isoform_seq_similarity', '5_utr_seq_similarity' and '3_utr_seq_similarity' analysis. Default is 0.8

AaCutoff

An integer indicating the length difference (in AA) a comparison must be larger than for reporting differences when evaluating 'ORF_seq_similarity', primarily implemented to avoid differences in very short AA sequences being classified as different. Default is 10 (AA).

AaFracCutoff

An numeric indicating the cutoff of length difference of the protein domain or IDR. The difference is measured as a fraction of the longest region, a comparison must be larger than before reporting it. Only used when analyzing 'domain_length' or 'IDR_length'. For example setting AaFracCutoff = 0.5 means the short protein domain (or IDR) must be <50% of the length of the long region, before it is reported. NULL disables the filter. Default is 0.5.

AaJCSimCutoff

An numeric (between 0 and 1) indicating the cutoff on Jaccard Similarity (JCsim) (see details) between the overlap of two amino acid (AA) sequences. If the measured JCsim is smaller than this cutoff the sequences are considered different and reported as such. This cutoff affects the result of the 'ORF_seq_similarity' analysis. Default is 0.9

removeNonConseqSwitches

A logic indicating whether to, in the "switchConsequence" entry added to the switchAnalyzeRList, remove the comparison of isoforms where no consequences were found (if TRUE) or to keep them (if FALSE). Default is TRUE.

showProgress	A logic indicating whether to make a progress bar (if TRUE) or not (if FALSE). Default is TRUE.
quiet	A logic indicating whether to avoid printing progress messages (incl. progress bar). Default is FALSE

Details

Changes in isoform usage are measure as the difference in isoform fraction (dIF) values, where isoform fraction (IF) values are calculated as $\langle \text{isoform_exp} \rangle / \langle \text{gene_exp} \rangle$.

The idea is that once we know there is (at least) one isoform with a significant change in how much it is used (as defined by alpha and dIFcutoff) in a gene we take that/those isoform(s) and compare the functional annotation of this isoform to the isoform(s) with the compensatory change(s) in isoform usage (since if one isoform is use more another/others have to be used less). Here we only require that one of the isoforms in the comparison of annotation is significant (unless onlySigIsoforms=TRUE, then both must be), but all isoforms considered must have a change in isoform usage larger than dIFcutoff.

Note that sometimes we find complex switches meaning that many isoforms passes all the filters. In these cases we compare all pairwise combinations of the isoform(s) used more (positive dIF) vs the isoform(s) used less (negative dIF).

For sequences similarity analysis the two compared sequences are (globally) aligned to one another and the Jaccard similarity (JCSim) is calculated. Here JCSim is defined as the length of the aligned regions (omitting gaps) divided by the total combined unique sequence length: $JCSim = (\text{length of aligned region w.o gaps}) / ((\text{length of sequence a}) + (\text{length of sequence b}) - (\text{length of aligned region w.o gaps}))$. The pairwise alignment is done with pairwiseAlignment{Biostrings} as a Needleman-Wunsch global alignment which is guaranteed to find the optimal global alignment. The pairwise alignment is done with end gap penalties for the full sequences alignments ('isoform_seq_similarity' and 'ORF_seq_similarity') and without gap penalties for the alignment of sub-sequence ('5_utr_seq_similarity' and '3_utr_seq_similarity') by specifying type='global' and type='overlap' respectively.

If AA sequences were trimmed in the process of exporting the fasta files when using extractSequence the regions not analyzed in both isoforms will be ignored.

The arguments passed to consequencesToAnalyze must be a combination of:

- all : Test transcripts for any of the differences described below. Please note that jointly the analysis below covers all transcript feature meaning that they should be different. Furthermore note that 'class_code' will only be included if the switchAnalyzeRlist was made from Cufflinks/Cuffdiff output.
- tss : Test transcripts for whether they use different Transcription Start Site (TSS) (more than ntCutoff from each other).
- tts : Test transcripts for whether they use different Transcription Termination Site (TTS) (more than ntCutoff from each other).
- last_exon : Test whether transcripts utilizes different last exons (defined as the last exon of each transcript is non-overlapping).

- `isoform_seq_similarity` : Test whether the isoform nucleotide sequences are different (as described above). Reported as different if the measured `JCsim` is smaller than `ntJCsimCutoff` and the length difference of the aligned and combined region is larger than `ntCutoff`.
- `isoform_length` : Test transcripts for differences in isoform length. Only reported if the difference is larger than indicated by the `ntCutoff` and `ntFracCutoff`. Please note that this is a less powerful analysis than implemented in `'isoform_seq_similarity'` as two equally long sequences might be very different.
- `exon_number` : Test transcripts for differences in exon number.
- `intron_structure` : Test transcripts for differences in intron structure, e.g. usage of exon-exon junctions. This analysis corresponds to analyzing whether all introns in one isoform is also found in the other isoforms.
- `intron_retention` : Test for differences in intron retentions (and their genomic positions). Require that `analyzeIntronRetention` have been run.
- `isoform_class_code` : Test transcripts for differences in the transcript classification provide by cufflinks. For a updated list of class codes see <http://cole-trapnell-lab.github.io/cufflinks/cuffcompare/#transfrag-class-codes>.
- `coding_potential` : Test transcripts for differences in coding potential, as indicated by the CPAT or CPC2 analysis. Requires that `analyzeCPAT` or `analyzeCPC2` have been used to add external condong potential analysis to the `switchAnalyzeRlist`.
- `ORF_seq_similarity` : Test whether the amino acid sequences of the ORFs are different (as described above). Reported as different if the measured `JCsim` is smaller than `AaJCsimCutoff` and the length difference of the aligned and combined region is larger than `AaCutoff`. Requires that least one of the isoforms are annotated with a ORF either via `identifyORF` or by supplying a GTF file and setting `addAnnotatedORFs=TRUE` when creating the `switchAnalyzeRlist`.
- `ORF_genomic` : Test transcripts for differences in genomic position of the Open Reading Frames (ORF). Requires that least one of the isoforms are annotated with an ORF either via `identifyORF` or by supplying a GTF file and setting `addAnnotatedORFs=TRUE` when creating the `switchAnalyzeRlist`.
- `ORF_length` : Test transcripts for differences in length of Open Reading Frames (ORF). Note that this is a less powerful analysis than implemented in `ORF_seq_similarity` as two equally long sequences might be very different. Only reported if the difference is larger than indicated by the `ntCutoff` and `ntFracCutoff`. Requires that least one of the isoforms are annotated with a ORF either via `identifyORF` or by supplying a GTF file and setting `addAnnotatedORFs=TRUE` when creating the `switchAnalyzeRlist`.
- `5_utr_seq_similarity` : Test whether the isoform nucleotide sequences of the 5' UnTranslated Region (UTR) are different (as described above). The 5'UTR is defined as the region from the transcript start to the ORF start. Reported as different if the measured `JCsim` is smaller than `ntJCsimCutoff` and the length difference of the aligned and combined region is larger than `ntCutoff`. Requires that both the isoforms are annotated with an ORF either via `identifyORF` or by supplying a GTF file and setting `addAnnotatedORFs=TRUE` when creating the `switchAnalyzeRlist`.
- `5_utr_length` : Test transcripts for differences in the length of the 5' UnTranslated Region (UTR). The 5'UTR is defined as the region from the transcript start to the ORF start. Note that this is a less powerful analysis than implemented in `'5_utr_seq_similarity'` as two equally long

sequences might be very different. Only reported if the difference is larger than indicated by the `ntCutoff` and `ntFracCutoff`. Requires that both the isoforms are annotated with a ORF either via `identifyORF` or by supplying a GTF file and setting `addAnnotatedORFs=TRUE` when creating the `switchAnalyzeRlist`.

- `3_utr_seq_similarity` : Test whether the isoform nucleotide sequences of the 3' UnTranslated Region (UTR) are different (as described above). The 3'UTR is defined as the region from the end of the ORF to the transcript end. Reported as different if the measured `JCsim` is smaller than `ntJCsimCutoff` and the length difference of the aligned and combined region is larger than `ntCutoff`. Requires that both the isoforms are annotated with a ORF either via `identifyORF` or by supplying a GTF file and setting `addAnnotatedORFs=TRUE` when creating the `switchAnalyzeRlist`.
- `3_utr_length` : Test transcripts for differences in the length of the 3' UnTranslated Regions (UTR). The 3'UTR is defined as the region from the end of the ORF to the transcript end. Note that this is a less powerful analysis than implemented in `3_utr_seq_similarity` as two equally long sequences might be very different. Requires that `identifyORF` have been used to predict NMD sensitivity or that the ORF was imported through one of the dedicated import functions implemented in `isoformSwitchAnalyzeR`. Only reported if the difference is larger than indicated by the `ntCutoff` and `ntFracCutoff`. Requires that both the isoforms are annotated with a ORF either via `identifyORF` or by supplying a GTF file and setting `addAnnotatedORFs=TRUE` when creating the `switchAnalyzeRlist`.
- `NMD_status` : Test transcripts for differences in sensitivity to Nonsense Mediated Decay (NMD). Requires that both the isoforms have been annotated with PTC either via `identifyORF` or by supplying a GTF file and setting `addAnnotatedORFs=TRUE` when creating the `switchAnalyzeRlist`.
- `domains_identified` : Test transcripts for differences in the name and order of which domains are identified by the Pfam in the transcripts. Requires that `analyzePFAM` have been used to add external Pfam analysis to the `switchAnalyzeRlist`. Requires that both the isoforms are annotated with a ORF either via `identifyORF` or by supplying a GTF file and setting `addAnnotatedORFs=TRUE` when creating the `switchAnalyzeRlist`.
- `domain_length` : Test transcripts for differences in the length of overlapping domains of the same type (same `hmm_name`) thereby enabling analysis of protein domain truncation. Do however note that a small difference in length is will likely not truncate the protein domain. The length difference, measured in AA, must be larger than `AaCutoff` and `AaFracCutoff`. Requires that `analyzePFAM` have been used to add external Pfam analysis to the `switchAnalyzeRlist`. Requires that both the isoforms are annotated with a ORF either via `identifyORF` or by supplying a GTF file and setting `addAnnotatedORFs=TRUE` when creating the `switchAnalyzeRlist`.
- `genomic_domain_position` : Test transcripts for differences in the genomic position of the domains identified by the Pfam analysis (Will be different unless the two isoforms have the same domains at the same genomic location). Requires that `analyzePFAM` have been used to add external Pfam analysis to the `switchAnalyzeRlist`. Requires that both the isoforms are annotated with a ORF either via `identifyORF` or by supplying a GTF file and setting `addAnnotatedORFs=TRUE` when creating the `switchAnalyzeRlist` (and are thereby also affected by `removeNoncodingORFs=TRUE` in `analyzeCPAT`).
- `IDR_identified` : Test for differences in isoform IDRs. Specifically the two isoforms are analyzed for whether they contain IDRs which do not overlap in genomic coordinates. Requires

that analyzeNetSurfP2 or analyzeIUPred2A have been used to add external IDR analysis to the switchAnalyzeRlist.

- `IDR_length` : Test for differences in the length of overlapping (in genomic coordinates) IDRs. The length difference, measured in AA, must be larger than `AaCutoff` and `AaFracCutoff`. Requires that analyzeNetSurfP2 or analyzeIUPred2A have been used to add external IDR analysis to the switchAnalyzeRlist.
- `IDR_type` : Test for differences in IDR type. Specifically the two isoforms are tested for overlapping IDRs (genomic coordinates) and overlapping IDRs are compared with regards to their IDR type (IDR vs IDR w binding site). Only available if analyzeIUPred2A was used to add external IDR analysis to the switchAnalyzeRlist.
- `signal_peptide_identified` : Test transcripts for differences in whether a signal peptide was identified or not by the SignalP analysis. Requires that analyzeSignalP have been used to add external SignalP analysis to the switchAnalyzeRlist. Requires that both the isoforms are annotated with a ORF either via analyzeORF or by supplying a GTF file and setting `addAnnotatedORFs=TRUE` when creating the switchAnalyzeRlist (and are thereby also affected by `removeNoncodingORFs=TRUE` in analyzeCPAT).

Value

The supplied switchAnalyzeRlist is returned, but now annotated with the predicted functional consequences as follows. First a column called 'switchConsequencesGene' is added to isoformFeatures entry of switchAnalyzeRlist. This column containing a binary indication (TRUE/FALSE (and NA)) of whether the switching gene have predicted functional consequences or not.

Secondly the data.frame 'switchConsequence' is added to the switchAnalyzeRlist containing one row feature analyzed per comparison of isoforms pr comparison of condition. It contains 8 columns:

- `gene_ref` : A unique reference to a specific gene in a specific comparison of conditions. Enables easy handles to integrate data from all the parts of a switchAnalyzeRlist.
- `gene_id`: The id of the gene which the isoforms compared belongs to. Matches the 'gene_id' entry in the 'isoformFeatures' entry of the switchAnalyzeRlist
- `gene_name` : The gene name associated with the <gene_id>, typically a more readable one (for example p53 or BRCA1)
- `condition_1`: The first condition of the comparison. Should be thought of as the ground state - meaning the changes occur from condition_1 to condition_2. Matches the 'condition_1' entry in the 'isoformFeatures' entry of the switchAnalyzeRlist
- `condition_2`: The second condition of the comparison. Should be thought of as the changed state - meaning the changes occur from condition_1 to condition_2. Matches the 'condition_2' entry in the 'isoformFeatures' entry of the switchAnalyzeRlist
- `isoformUpregulated`: The name of the isoform which is used more in condition_2 (when compared to condition_1, positive dIF values). Matches the 'isoform_id' entry in the 'isoformFeatures' entry of the switchAnalyzeRlist
- `isoformDownregulated`: The name of the isoform which is used less in condition_2 (when compared to condition_1, negative dIF values). Matches the 'isoform_id' entry in the 'isoformFeatures' entry of the switchAnalyzeRlist
- `iso_ref_up` : A unique reference to a specific isoform in a specific comparison of conditions for the isoform switching up. Enables easy handles to integrate data from all the parts of a switchAnalyzeRlist.

- `iso_ref_down` : A unique reference to a specific isoform in a specific comparison of conditions for the isoform switching down. Enables easy handles to integrate data from all the parts of a `switchAnalyzeRlist`.
- `featureCompared`: The category of the isoform features/annotation compared in this row (see details above)
- `isoformsDifferent`: A logic (TRUE/FALSE) indicating whether the two isoforms are different with respect to the `featureCompared` (see details above)
- `switchConsequence`: If the isoforms compared are different this columns contains a short description of the features of the upregulated isoform. E.g. domain loss means that the upregulated isoforms (`isoformUpregulated`) have lost domains compare to the downregulated isoform (`isoformDownregulated`).

Author(s)

Kristoffer Vitting-Seerup

References

Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. *Mol. Cancer Res.* (2017).

See Also

[analyzeORF](#)
[analyzeCPAT](#)
[analyzePFAM](#)
[analyzeSignalP](#)
[extractConsequenceSummary](#)
[extractConsequenceEnrichment](#)
[extractConsequenceEnrichmentComparison](#)
[extractConsequenceGenomeWide](#)

Examples

```

### Prepare example data
data("exampleSwitchListAnalyzed")

# subset for fast runtime
exampleSwitchListAnalyzed <- subsetSwitchAnalyzeRlist(
  exampleSwitchListAnalyzed,
  exampleSwitchListAnalyzed$isoformFeatures$gene_id %in% sample(exampleSwitchListAnalyzed$isoformFeatures$gene_id, 10)
)

### Analyze consequences
consequencesOfInterest <- c(
  'intron_retention',
  'coding_potential',
  'NMD_status',
  'domains_identified'
)

```

```
exampleSwitchListAnalyzed <- analyzeSwitchConsequences(  
  exampleSwitchListAnalyzed,  
  consequencesToAnalyze = consequencesOfInterest,  
)  
  
### simple overview  
extractSwitchSummary(exampleSwitchListAnalyzed, filterForConsequences = FALSE)  
extractSwitchSummary(exampleSwitchListAnalyzed, filterForConsequences = TRUE)  
  
### Detailed switch overview  
consequenceSummary <- extractConsequenceSummary(  
  exampleSwitchListAnalyzed,  
  includeCombined = TRUE,  
  returnResult = TRUE,      # return data.frame with summary  
  plotGenes = TRUE         # plot summary  
)  
  
### Now switches are analyzed we can also extract the the largest/most significant switches with the extractTopSwitches  
# Extract top 2 switching genes (by q-value)  
extractTopSwitches(  
  exampleSwitchListAnalyzed,  
  filterForConsequences = TRUE,  
  n = 2,  
  extractGenes = TRUE,  
  sortByQvals = TRUE  
)  
  
# Extract top 2 switching isoforms (by q-value)  
extractTopSwitches(  
  exampleSwitchListAnalyzed,  
  filterForConsequences = TRUE,  
  n = 2,  
  extractGenes = FALSE,  
  sortByQvals = TRUE  
)  
  
# Extract top 2 switching isoforms (by dIF)  
extractTopSwitches(  
  exampleSwitchListAnalyzed,  
  filterForConsequences = TRUE,  
  n = 2,  
  extractGenes = FALSE,  
  sortByQvals = FALSE  
)  
  
### Note the function ?extractConsequenceSummary is specific made for the post analysis of switching consequences
```

CDSSet

Container for coding sequence (CDS) annotation information

Description

A container for coding sequence annotation information.

Usage

```
CDSSet(cds)
```

Arguments

`cds` A `data.frame` object containing CDS annotation. See details for required columns.

Details

This object can be generated automatically from [getCDS](#), or can be generated manually by creating a new `CDSSet` from a `data.frame` with the following columns:

`chrom`, the chromosome name (NB: chromosome names must match when running [analyzeORF](#)).
`strand`, the strand, `cdsStart`, the genomic start of the coding sequence (beware of 0/1-frame issues), and `cdsEnd`, the genomic end of the coding sequence (beware of 0/1-frame issues).

The `CDSSet` object is used with [analyzeORF](#) if annotated TSS should be analyzed.

For an example, see [getCDS](#).

Value

A `CDSSet` object.

Author(s)

Kristoffer Vitting-Seerup, Johannes Waage

References

Vitting-Seerup K, et al: spliceR: an R package for classification of alternative splicing and prediction of coding potential from RNA-seq data. *BMC Bioinformatics* 2014, 15:81.

`createSwitchAnalyzeRlist`*Create a switchAnalyzeRlist Object*

Description

Create a `switchAnalyzeRlist` containing all the information needed to do the full analysis with `IsoformSwitchAnalyzeR`.

Usage

```
createSwitchAnalyzeRlist(  
  isoformFeatures,  
  exons,  
  designMatrix,  
  isoformCountMatrix=NULL,  
  isoformRepExpression=NULL,  
  sourceId,  
  removeFusionTranscripts = TRUE  
)
```

Arguments

- `isoformFeatures` A data.frame where each row corresponds to a isoform in a specific comparison and contains all the annotation for this isoform. See details below for details.
- `exons` A GRanges object containing isoform exon structure. See details below for details.
- `designMatrix` A data.frame with the information of which samples originate from which conditions. A data.frame with two columns: `sampleID` 1 contains the sample names which matches the column names used in `isoformCountMatrix`. `condition`: which indicates which conditions the sample originate from. If sample 1-3 originate from the same condition they should all have the same string (for example 'ctrl', in this column). By adding additional columns to this `designMatrix` batch effects can be taking into account with the DRIMSeq based isoform switch test.
- `isoformCountMatrix` A data.frame with unfiltered biological (not technical) replicate isoform (estimated) counts. Must have a column called 'isoform_id' with the isoform_id that matches `isoformFeatures`. The name of the columns must match the sample names in the `designMatrix` argument and contain the estimated counts.
- `isoformRepExpression` A data.frame with unfiltered biological (not technical) replicate isoform abundances. Must have a column called 'isoform_id' with the isoform_id that matches `isoformFeatures`. The name of the columns must match the sample names in the `designMatrix` argument and contain the estimated abundances.

sourceId A character stating the origin of the data used to create the switchAnalyzeRlist.
 removeFusionTranscripts A logic indicating whether to remove genes with cross-chromosome fusion transcripts as IsoformSwitchAnalyzeR cannot handle them.

Details

For cufflinks data, use [importCufflinksFiles](#) to prepare the switchAnalyzeRlist. For other RNA-seq assemblies, either uses this constructor or the general-purpose [importRdata](#) to create the switchAnalyzeRlist - See vignette for details.

The isoformFeatures should be a data.frame where each row corresponds to a isoform in a specific comparison and contains all the annotation for this isoform. The data.frame can contain any columns supplied (enabling addition of user specified columns) but the following columns are necessary and must be provided:

- iso_ref : A unique reference to a specific isoform in a specific comparison of conditions. Mainly created to have an easy handle to integrate data from all the parts of a switchAnalyzeRlist.
- gene_ref : A unique reference to a specific gene in a specific comparison of conditions. Mainly created to have an easy handle to integrate data from all the parts of a switchAnalyzeRlist.
- isoform_id : A unique isoform id
- gene_id : A unique gene id referring to a gene at a specific genomic loci (not the same as gene_name since gene_names can refer to multiple genomic loci)
- condition_1 : Name of the first condition in the comparison
- condition_2 : Name of the second condition in the comparison
- gene_name : The gene name associated with the <gene_id>, typically a more readable one (for example p53 or BRCA1)
- gene_overall_mean : Mean expression of <gene_id> across all samples (if you create it yourself consider inter-library normalization)
- gene_value_1 : Expression of <gene_id> in condition_1 (if you create it yourself consider inter-library normalization)
- gene_value_2 : Expression of <gene_id> in condition_2 (if you create it yourself consider inter-library normalization)
- gene_stderr_1 : Standard error (of mean) of <gene_id> expression in condition_1
- gene_stderr_2 : Standard error (of mean) of <gene_id> expression in condition_2
- gene_log2_fold_change : log2 fold change of <gene_id> expression between condition_1 and condition_2
- gene_q_value : The FDR corrected (for multiple testing) p-value of the differential expression test of <gene_id>
- iso_overall_mean : Mean expression of <isoform_id> across all samples (if you create it yourself consider inter-library normalization)
- iso_value_1 : Expression of <isoform_id> in condition_1 (if you create it yourself consider inter-library normalization)
- iso_value_2 : Expression of <isoform_id> in condition_2 (if you create it yourself consider inter-library normalization)

- `iso_stderr_1` : Standard error (of mean) of `<isoform_id>` expression in `condition_1`
- `iso_stderr_2` : Standard error (of mean) of `<isoform_id>` expression in `condition_2`
- `iso_log2_fold_change` : log2 fold change of `<isoform_id>` expression between `condition_1` and `condition_2`
- `iso_q_value` : The FDR corrected (for multiple testing) p-value of the differential expression test of `<isoform_id>`
- `IF_overall` : The average `<isoform_id>` usage across all samples (given as Isoform Fraction (IF) value)
- `IF1` : The `<isoform_id>` usage in condition 1 (given as Isoform Fraction (IF) value)
- `IF2` : The `<isoform_id>` usage in condition 2 (given as Isoform Fraction (IF) value)
- `dIF` : The change in isoform usage from `condition_1` to `condition_2` (difference in IF values (dIF))
- `isoform_switch_q_value` : The q-value of the test of differential isoform usage in `<isoform_id>` between condition 1 and condition 2. Use NA if not performed. Will be overwritten by the result of `testIsoformSwitches`. If only performed at gene level use same values on isoform level.
- `gene_switch_q_value` : The q-value of the test of differential isoform usage in `<gene_id>` between condition 1 and condition 2. Use NA if not performed. Will be overwritten by the result of `testIsoformSwitches`.

The `exons` argument must be supplied with a `GenomicRange` object containing one entry per exon in each isoform. Furthermore it must also have two meta columns called `isoform_id` and `gene_id` which links it to the information in the `isoformFeatures` entry.

The `conditions` should be a `data.frame` with two columns: `condition` and `nrReplicates` giving the number of biological (not technical) replicates each condition analyzed. The strings used to conditions the conditions must match the strings used in `condition_1` and `condition_2` columns of the `isoformFeatures` entry.

Value

A list-type object `switchAnalyzeRlist` object containing all the information needed to do the full analysis with `IsoformSwitchAnalyzeR`. Note that `switchAnalyzeRlist` appears as a normal list and all the information (incl that added by all the `analyze*` functions) can be obtained using both the named entries (f.x. `myIsoSwitchList$isoformFeatures`) or indexes (f.x. `myIsoSwitchList[[1]]`).

When fully analyzed the `isoformFeatures` entry of the will furthermore contain the following columns:

- `id`: During the creation of `switchAnalyzeRlist` a unique id is constructed for each row - meaning for each isoform in each comparison. The id is constructed as 'isoComp' an acronym for 'isoform comparison', followed by `XXXXXXXX` indicating the row number
- `PTC`: A logic indicating whether the `<isoform_id>` is classified as having a Premature Termination Codon. This is defined as having a stopcodon more than `PTCDistance`(default is 50) nt upstream of the last exon exon.
- `codingPotentialValue`: containing the coding potential value predicted by CPAT.
- `codingPotential`: A logic (TRUE/FALSE) indicating whether the isoform is coding or not (based on the `codingCutoff` supplied)

- `signal_peptide_identified`: A string ('yes'/'no') indicating whether the `<isoform_id>` have a signal peptide, as predicted by SignalP.
- `domain_identified`: A string ('yes'/'no') indicating whether the `<isoform_id>` contain (at least one) protein domain, as predicted by pfam.
- `switchConsequencesGene`: A logic (TRUE/FALSE) indicating whether the `<gene_id>` contain an isoform switch with functional consequences, as predicted by `analyzeSwitchConsequences`.

Author(s)

Kristoffer Vitting-Seerup

References

Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. *Mol. Cancer Res.* (2017).

See Also

```
importRdata
importCufflinksFiles
importGTF
importIsoformExpression
```

Examples

```
### Please note
# 1) The way of importing files in the following example with
#     "system.file('pathToFile', package="IsoformSwitchAnalyzeR") is
#     specialized to access the sample data in the IsoformSwitchAnalyzeR package
#     and not something you need to do - just supply the string e.g.
#     "myAnnotation/isoformsQuantified.gtf" to the functions
# 2) importRdata directly supports import of a GTF file - just supply the
#     path (e.g. "myAnnotation/isoformsQuantified.gtf") to the isoformExonAnnotation argument

### Import quantifications
salmonQuant <- importIsoformExpression(system.file("extdata/", package="IsoformSwitchAnalyzeR"))

### Make design matrix
myDesign <- data.frame(
  sampleID = colnames(salmonQuant$abundance)[-1],
  condition = gsub('_.*', '', colnames(salmonQuant$abundance)[-1])
)

### Create switchAnalyzeRlist
aSwitchList <- importRdata(
  isoformCountMatrix = salmonQuant$counts,
  isoformRepExpression = salmonQuant$abundance,
  designMatrix = myDesign,
  isoformExonAnnotation = system.file("extdata/example.gtf.gz", package="IsoformSwitchAnalyzeR")
)
aSwitchList
```

`exampleData`*Example data for IsoformSwitchAnalyzeR*

Description

Three `switchAnalyzeRlist` corresponding to a `switchAnalyzeRlist` in different stages of an isoform switch analyzer workflow.

Usage

```
data("exampleSwitchList")  
  
data("exampleSwitchListIntermediary")  
  
data("exampleSwitchListAnalyzed")
```

Format

see `?createSwitchAnalyzeRlist` for detailed format of an `switchAnalyzeRlist`

Details

The three example `switchAnalyzeRlist` are:

- `exampleSwitchList` : Which corresponds to a newly created `switchAnalyzeRlist` such as one would get by using either of the `import*` function (such as `importCufflinksData`) or by using `createSwitchAnalyzeRlist` on your own data. Not this is a small subset to allow for fast example generation.
- `exampleSwitchListIntermediary` : Which corresponds to the `exampleSwitchList` data (see above) which have been analyzed with the `isoformSwitchAnalysisPart1` function meaning that it have been filtered, tested for isoform switches, ORF have been predicted and both nucleotide and ORF amino acid sequences have been added to the `switchAnalyzeRlist`. Not this is a small subset to allow for fast example generation.
- `exampleSwitchListAnalyzed` : Which corresponds to a subset of two of the TCGA Cancer types analyzed in Vitting-Seerup et al 2017 which have been analyzed with the full switch analysis workflow (including external sequence analysis of protein domains (via Pfam), coding potential (via CPAT) and signal peptides (via SignalP)). Note that the nucleotide and amino acid sequences normally added to the `switchAnalyzeRlist` have been removed from the `switchAnalyzeRlist` (but also that they can easily be added again with the [extractSequence](#) function).

Source

`exampleSwitchList` and `exampleSwitchListIntermediary` is a modified subset of a dataset comparing human Embryonic Stem Cells (hESC) vs induced Pluripotent Cells (iPS) and mature cells (Fibroblast) originally released with the `cummeRbund` R package. This data is only included to provide examples for usage of function. As it is modified to illustrate the package it should not be considered real and no biological conclusions should be made from it.

The exampleSwitchListAnalyzed is a subset of two of the TCGA Cancer types analyzed in Vitting-Seerup et al 2017 and are unmodified meaning results are real!

References

Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. Mol. Cancer Res. (2017).

Examples

```
### Summarize newly created switchAnalyzeRlist
data("exampleSwitchList")
summary(exampleSwitchList)
```

```
extractConsequenceEnrichment
```

Analyze data for enrichment of specific consequences

Description

This functions analyzes the genome wide enrichment of specific consequences by for each set of opposing consequences (e.g.. domain gain vs loss) analyzing the fraction of events belonging to one of them.

Usage

```
extractConsequenceEnrichment(
  switchAnalyzeRlist,
  consequencesToAnalyze = 'all',
  alpha=0.05,
  dIFcutoff = 0.1,
  countGenes = TRUE,
  analysisOppositeConsequence=FALSE,
  plot=TRUE,
  localTheme = theme_bw(base_size = 12),
  minEventsForPlotting = 10,
  returnResult=TRUE,
  returnSummary=TRUE
)
```

Arguments

switchAnalyzeRlist

A switchAnalyzeRlist object where analyzeSwitchConsequences() have been run to identify consequences of isoform switches

consequencesToAnalyze

A string indicating which consequences should be considered. See detail section of [analyzeSwitchConsequences](#) for description . Default is all consequences analyzed with analyzeSwitchConsequences.

alpha	The cutoff which the FDR correct p-values must be smaller than for calling significant switches. Default is 0.05.
dIFcutoff	The cutoff which the changes in (absolute) isoform usage must be larger than before an isoform is considered switching. This cutoff can remove cases where isoforms with (very) low dIF values are deemed significant and thereby included in the downstream analysis. This cutoff is analogous to having a cutoff on log2 fold change in a normal differential expression analysis of genes to ensure the genes have a certain effect size. Default is 0.1 (10%).
countGenes	A logic indicating whether it is the number of genes (if TRUE) or isoform switches (if FALSE) which primary result in gain/loss that are counted. Default is TRUE.
analysisOppositeConsequence	A logic indicating whether reverse the analysis meaning if "Domain gains" are analyze using default parameters setting analysisOppositeConsequence=TRUE will case the analysis to be performed on "Domain loss". The main effect is for the visual appearance of plot which will be mirrored (around the 0.5 fraction). Default is FALSE.
plot	A logic indicting whether the analysis should be plotted. If TRUE and returnResult = FALSE the ggplot2 object will be returned instead. Default is TRUE.
localTheme	General ggplot2 theme with which the plot is made, see ?ggplot2::theme for more info. Default is theme_bw(base_size = 14).
minEventsForPlotting	The minimum number of events (total gain/loss) must be present before the result is visualized. Default is 10.
returnResult	A logic indicating whether the analysis should be returned as a data.frame. If FALSE (and plot=TRUE) the ggplot2 object will be returned instead. Default is TRUE.
returnSummary	A logic indicating whether to return the statistical summary (if TRUE) or the underlying data (if FALSE). Depends on returnResult=TRUE Default is TRUE.

Details

The significance test is performed with R's build in `binom.test()` with default parameters and resulting p-values are corrected via `p.adjust()` using FDR (Benjamini-Hochberg).

Value

If `plot=TRUE` a plot summarizing the proportions is also created of switches with specific consequences is created.

If `returnResult=TRUE` a data.frame with the statistical summary for each opposing consequences in each comparison. This data.frame will have the following collumns:

- `condition_1`: Condition 1.
- `condition_2`: Condition 2.
- `conseqPair`: The set of oposite consequences consudered.

- **feature:** Description of which consequence the calculations are done from the perspective of (with the opposite mention in parentheses)
- **propOfRelevantEvents:** Proportion of total number of genes (of genes affected by the consequence described in the `conseqPair` column) being of the type described in the feature column.
- **propCiLo:** The lower boundary of the confidence interval of the `propOfRelevantEvents`.
- **propCiHi:** The high boundary of the confidence interval of the `propOfRelevantEvents`.
- **propPval:** The p-value associated with the null hypothesis that `propOfRelevantEvents` is 0.5.
- **nUp:** The number of genes with the consequence described in the feature column.
- **nDown:** The number of genes with the opposite consequence of what is described in the feature column (as described in the parentheses of the feature column).
- **propQval:** The q-values resulting when p-values are corrected via `p.adjust()` using FDR (Benjamini-Hochberg).

Author(s)

Kristoffer Vitting-Seerup

References

- Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. *Mol. Cancer Res.* (2017).
- Vitting-Seerup et al. `IsoformSwitchAnalyzeR`: Analysis of changes in genome-wide patterns of alternative splicing and its functional consequences. *Bioinformatics* (2019).

See Also

[analyzeSwitchConsequences](#)
[extractSwitchSummary](#)
[extractConsequenceEnrichmentComparison](#)
[extractConsequenceGenomeWide](#)

Examples

```
### Load exempld data
data("exampleSwitchListAnalyzed")

extractConsequenceEnrichment( exampleSwitchListAnalyzed)
```

```
extractConsequenceEnrichmentComparison
```

Compare enrichment of specific consequences between comparisons

Description

This function compares the enrichment of a consequences (f.x. domain gain) between two comparisons (ctrl vs ko1 compared to ctrl vs ko2) and reports whether there is a significant difference between the comparisons. In other words it compares the output of `extractConsequenceEnrichment`.

Usage

```
extractConsequenceEnrichmentComparison(
  switchAnalyzeRlist,
  consequencesToAnalyze = 'all',
  alpha=0.05,
  dIFcutoff = 0.1,
  countGenes = TRUE,
  analysisOppositeConsequence=FALSE,
  plot=TRUE,
  localTheme = theme_bw(base_size = 14),
  minEventsForPlotting = 10,
  returnResult=TRUE
)
```

Arguments

<code>switchAnalyzeRlist</code>	A <code>switchAnalyzeRlist</code> object where <code>analyzeSwitchConsequences()</code> have been run to identify consequences of isoform switches
<code>consequencesToAnalyze</code>	A string indicating which consequences should be considered. See details for description (note it is identical to the strings used with <code>analyzeSwitchConsequences</code>). Default is all consequences analyzed with <code>analyzeSwitchConsequences</code>
<code>alpha</code>	The cutoff which the FDR correct p-values must be smaller than for calling significant switches. Default is 0.05.
<code>dIFcutoff</code>	The cutoff which the changes in (absolute) isoform usage must be larger than before an isoform is considered switching. This cutoff can remove cases where isoforms with (very) low dIF values are deemed significant and thereby included in the downstream analysis. This cutoff is analogous to having a cutoff on log2 fold change in a normal differential expression analysis of genes to ensure the genes have a certain effect size. Default is 0.1 (10%).
<code>countGenes</code>	A logic indicating whether it is the number of genes (if TRUE) or isoform switches (if FALSE) which primary result in gain/loss that are counted. Default is TRUE.

<code>analysisOppositeConsequence</code>	A logic indicating whether reverse the analysis meaning if "Domain gains" are analyze using default parameters setting <code>analysisOppositeConsequence=TRUE</code> will case the analysis to be performed on "Domain loss". The main effect is for the visual appearance of plot which will be mirrored (around the 0.5 fraction). Default is FALSE.
<code>plot</code>	A logic indicting whether the analysis should be plotted. If TRUE and <code>returnResult = FALSE</code> the ggplot2 object will be returned instead. Default is TRUE.
<code>localTheme</code>	General ggplot2 theme with which the plot is made, see <code>?ggplot2::theme</code> for more info. Default is <code>theme_bw(base_size = 14)</code> .
<code>minEventsForPlotting</code>	The minimum number of events (total gain/loss) must be present before the result is visualized. Default is 10.
<code>returnResult</code>	A logic indicating whether the analysis should be returned as a data.frame. If FALSE (and <code>plot=TRUE</code>) the ggplot2 object will be returned instead. Default is TRUE.

Details

The significance test is performed with R's build in `prop.test()` with default parameters and resulting p-values are corrected via `p.adjust()` using FDR (Benjamini-Hochberg).

Value

If `returnResult=TRUE` a data.frame with the statistical summary for each opposing consequences in each comparison. If `plot=TRUE` a plot summarizing the proportions is also created of switches with specific consequences is created.

Author(s)

Kristoffer Vitting-Seerup

References

- Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. *Mol. Cancer Res.* (2017).
- Vitting-Seerup et al. IsoformSwitchAnalyzeR: Analysis of changes in genome-wide patterns of alternative splicing and its functional consequences. *Bioinformatics* (2019).

See Also

[analyzeSwitchConsequences](#)
[extractSwitchSummary](#)
[extractConsequenceEnrichment](#)
[extractConsequenceGenomeWide](#)

Examples

```
### Load example data
data("exampleSwitchListAnalyzed")

extractConsequenceEnrichmentComparison( exampleSwitchListAnalyzed)
```

extractConsequenceGenomeWide

Genome wide Analysis of Consequences due to isoform switching

Description

This function enables a genome wide analysis of changes in isoform usage of isoforms with a common annotation.

Specifically this function extract isoforms of interest and for each category of annotation (such as signal peptides) the global distribution of IF (measuring isoform usage) are plotted for each subset of features in that category (e.g with and without signal peptides). This enables a global analysis of isoforms with a common annotation. The annotation considered are (if added to the switchAnalyzeRlist) coding potential, intron retentions, isoform class code (Cufflinks/Cuffdiff data only), NMD status, ORFs, protein domains, signal peptide and whether switch consequences were identified.

The isoforms of interest can either be defined by isoforms form gene differentially expressed, isoform that are differential expressed or isoforms from genes with isoform switching - as controlled by featureToExtract. Please note that the [extractConsequenceEnrichment](#) function probably more relevant than using featureToExtract='isoformUsage' since it directly uses the paired information from switches.

This function offers both visualization of the result as well as analysis via summary statistics of the comparisons.

Usage

```
extractConsequenceGenomeWide(
  switchAnalyzeRlist,
  featureToExtract = 'isoformUsage',
  annotationToAnalyze = 'all',
  alpha=0.05,
  dIFcutoff = 0.1,
  log2FCcutoff = 1,
  violinPlot=TRUE,
  alphas=c(0.05, 0.001),
  localTheme=theme_bw(),
  plot=TRUE,
  returnResult=TRUE
)

extractGenomeWideAnalysis(
```

```

switchAnalyzeRlist,
featureToExtract = 'isoformUsage',
annotationToAnalyze = 'all',
alpha=0.05,
dIFcutoff = 0.1,
log2FCcutoff = 1,
violinPlot=TRUE,
alphas=c(0.05, 0.001),
localTheme=theme_bw(),
plot=TRUE,
returnResult=TRUE
)

```

Arguments

switchAnalyzeRlist

A `switchAnalyzeRlist` object containing the result of an isoform switch analysis (such as the one provided by `isoformSwitchTestDEXSeq()`) as well as additional annotation data for the isoforms.

featureToExtract

This argument, given as a string, defines the set isoforms which should be analyzed. The available options are:

- 'isoformUsage' (Default): Analyze a subset of isoforms defined by change in isoform usage (controlled by `dIFcutoff`) and the significance of the change in isoform expression (controlled by `alpha`). Please note that the [extractConsequenceEnrichment](#) function probably more relevant than using `featureToExtract='isoformUsage'` since it directly uses the paired information from switches.
- 'isoformExp' :Analyze a subset of isoforms defined by change in isoform expression (controlled by `log2FCcutoff`) and the significance of the change in isoform expression (controlled by `alpha`)
- 'geneExp' :Analyze all isoforms from a subset of genes defined by change in gene expression (controlled by `log2FCcutoff`) and the significance of the change in gene expression (controlled by `alpha`)
- 'all' : Analyze all isoforms stored in the `switchAnalyzeRlist` (note that this is highly depending on the parameter `reduceToSwitchingGenes` in [isoformSwitchTestDEXSeq](#) - which should be set to `FALSE` (default is `TRUE`) if the 'all' option should be used here).

annotationToAnalyze

A vector of strings indicating what categories of annotation to analyze. Annotation types given here but not (yet) analyzed in the `switchAnalyzeRlist` will not be plotted. See details for full list of usable strings, their meaning and dependencies. Default is 'All'.

alpha

The cutoff which the FDR correct p-values (q-values) must be smaller than for calling significant switches. Default is 0.05.

dIFcutoff

The cutoff which the changes in (absolute) isoform usage must be larger than before an isoform is considered switching. This cutoff can remove cases where

	isoforms with (very) low dIF values are deemed significant and thereby included in the downstream analysis. This cutoff is analogous to having a cutoff on log2 fold change in a normal differential expression analysis of genes to ensure the genes have a certain effect size. Default is 0.1 (10%).
log2FCcutoff	The cutoff which the changes in (absolute) isoform or gene expression must be larger than before an isoform is considered for inclusion.
violinPlot	A logical indicating whether to make a violin plots (if TRUE) or boxplots (if FALSE). Violin plots will always have added 3 black dots, one of each of the 25th, 50th (median) and 75th percentile of the data. Default is TRUE.
alphas	A numeric vector of length two giving the significance levels represented in plots. The numbers indicate the q-value cutoff for significant (*) and highly significant (***) respectively. Default 0.05 and 0.001 which should be interpret as $q < 0.05$ and $q < 0.001$ respectively). If q-values are higher than this they will be annotated as 'ns' (not significant).
localTheme	General ggplot2 theme with which the plot is made, see <code>?ggplot2::theme</code> for more info. Default is <code>theme_bw()</code> .
plot	A logic indicting whether the analysis should be plotted. If TRUE and <code>returnResult = FALSE</code> the ggplot2 object will be returned instead. Default is TRUE.
returnResult	A logical indicating whether to return a data.frame with summary statistics of the comparisons (if TRUE) or not (if FALSE). If FALSE (and <code>plot=TRUE</code>) the ggplot2 object will be returned instead. Default is TRUE.

Details

`extractGenomeWideAnalysis` is just a wrapper for `extractGenomeWideConsequenceAnalysis` included for backward comparability.

Changes in isoform usage are measure as the difference in isoform fraction (dIF) values, where isoform fraction (IF) values are calculated as $\langle \text{isoform_exp} \rangle / \langle \text{gene_exp} \rangle$.

The significance test is performed with R's build in `wilcox.test()` (aka 'Mann-Whitney-U') with default parameters and resulting p-values are corrected via `p.adjust()` using FDR (Benjamini-Hochberg).

The arguments passed to `annotationToAnalyze` must be a combination of:

- `isoform_class_code` : Divide transcripts based on differences in the transcript classification provide by cufflinks (only available for data imported from Cufflinks/Cuffdiff). For a updated list of class codes see <http://cole-trapnell-lab.github.io/cufflinks/cuffcompare/#transfrag-class-codes>.
- `coding_potential` : Divide transcripts based on differences in coding potential, as indicated by the CPAT analysis. Requires that `importCPATanalysis` have been used to add external CPAT analysis to the `switchAnalyzeRlist`.
- `intron_retention` : Divide transcripts based on presence intron retentions (and their genomic positions). Require that `analyzeIntronRetention` have been run.

- **ORF** : Divide transcripts based on whether an ORF is annotated or not. Requires that both the isoforms have been annotated with ORF either via `identifyORF` or by supplying a GTF file and setting `addAnnotatedORFs=TRUE` when creating the `switchAnalyzeRlist`.
- **NMD_status** : Divide transcripts based on differences in sensitivity to Nonsense Mediated Decay (NMD). Requires that both the isoforms have been annotated with PTC either via `identifyORF` or by supplying a GTF file and setting `addAnnotatedORFs=TRUE` when creating the `switchAnalyzeRlist`.
- **domains_identified** : Divide transcripts based on differences in the name and order of which domains are identified by the Pfam in the transcripts. Requires that `importPFAManalysis` have been used to add external Pfam analysis to the `switchAnalyzeRlist`. Requires that both the isoforms are annotated with a ORF either via `identifyORF` or by supplying a GTF file and setting `addAnnotatedORFs=TRUE` when creating the `switchAnalyzeRlist`.
- **signal_peptide_identified** : Divide transcripts based on differences in whether a signal peptide was identified or not by the SignalP analysis. Requires that `analyzeSignalP` have been used to add external SignalP analysis to the `switchAnalyzeRlist`. Requires that both the isoforms are annotated with a ORF either via `analyzeORF` or by supplying a GTF file and setting `addAnnotatedORFs=TRUE` when creating the `switchAnalyzeRlist` (and are thereby also affected by `removeNoncodingORFs=TRUE` in `analyzeCPAT`).
- **switch_consequences** : Whether the gene is involved in isoform switches with predicted consequences. Requires that `analyzeSwitchConsequences` have been used).

Value

If `plot=TRUE`: A plot of the distribution of IF values as a function of the annotation and condition compared. If `returnResult=TRUE`: A `data.frame` with the summary statistics from the comparison of the two conditions with a `Wilcox.test`.

Author(s)

Kristoffer Vitting-Seerup

References

Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. *Mol. Cancer Res.* (2017).

See Also

[isoformSwitchTestDEXSeq](#)
[isoformSwitchTestDRIMSeq](#)
[analyzeORF](#)
[analyzeAlternativeSplicing](#)
[analyzeCPAT](#)
[analyzePFAM](#)
[analyzeSignalP](#)
[analyzeSwitchConsequences](#)
[extractConsequenceEnrichment](#)
[extractConsequenceEnrichmentComparison](#)

Examples

```
### Load example data
data("exampleSwitchListAnalyzed")

### make the genome wide analysis
summaryStatistics <- extractConsequenceGenomeWide(
  switchAnalyzeRlist = exampleSwitchListAnalyzed,
  featureToExtract = 'isoformUsage', # alternatives are 'isoformExp' and 'geneExp'
  plot=TRUE,
  returnResult = TRUE
)
```

```
extractConsequenceSummary
      Analyze Switch Consequences
```

Description

This functions function summarizes the individual types of consequences for each gene or the pairwise switches and plots and/or returns a data.frame with the information

Usage

```
extractConsequenceSummary(
  switchAnalyzeRlist,
  consequencesToAnalyze='all',
  includeCombined=FALSE,
  asFractionTotal=FALSE,
  alpha=0.05,
  dIFcutoff=0.1,
  plot=TRUE,
  plotGenes=FALSE,
  simplifyLocation = TRUE,
  removeEmptyConsequences = FALSE,
  localTheme=theme_bw(),
  returnResult=FALSE
)
```

Arguments

switchAnalyzeRlist

A switchAnalyzeRlist object where analyzeSwitchConsequences() have been run to identify consequences of isoform switches

consequencesToAnalyze

A string indicating which consequences should be considered. See detail section of [analyzeSwitchConsequences](#) for description . Default is all consequences analyzed with analyzeSwitchConsequences.

includeCombined	A logic indicating whether an analysis of how many (how large a fraction) of genes have any type of functional consequence.
asFractionTotal	A logic indicating whether the consequences should be summarized calculated as numbers (if FALSE) or as a fraction of the total number of switches/genes (as indicated by plotGenes). Default is FALSE.
alpha	The cutoff which the FDR correct p-values must be smaller than for calling significant switches. Default is 0.05.
dIFcutoff	The cutoff which the changes in (absolute) isoform usage must be larger than before an isoform is considered switching. This cutoff can remove cases where isoforms with (very) low dIF values are deemed significant and thereby included in the downstream analysis. This cutoff is analogous to having a cutoff on log2 fold change in a normal differential expression analysis of genes to ensure the genes have a certain effect size. Default is 0.1 (10%).
plot	A logic indicating whether the analysis should be plotted. If TRUE and returnResult = FALSE the ggplot2 object will be returned instead. Default is TRUE.
plotGenes	A logic indicating whether to plot the number/fraction of genes with (if TRUE) or isoforms (if FALSE) involved with isoform switches with functional consequences (both filtered via alpha and dIFcutoff).
simplifyLocation	A logic indicating whether to simplify the switches involved in changes in sub-cellular localizations (due the the hundreds of possible combinations). Done by only considering where the isoform used more has a location switch to. Default is TRUE.
removeEmptyConsequences	A logic indicating whether to remove consequences analyzed but where no differences was found (those showing zero in the plot). Default is FALSE.
localTheme	General ggplot2 theme with which the plot is made, see ?ggplot2::theme for more info. Default is theme_bw().
returnResult	A logic indicating whether the summarized results should be returned as a data.frame. If FALSE (and plot=TRUE) the ggplot2 object will be returned instead. Default is FALSE.

Details

A less detailed version just summarizing the number of switches with functional consequences can be obtained by setting `filterForConsequences=TRUE` in the `extractSwitchSummary` function.

For details on the arguments passed to `consequencesToAnalyze` please see details section of [analyzeSwitchConsequences](#).

Value

If `returnResult=TRUE` a data.frame with the number (and fraction) of switches with specific consequences in each condition is returned. If `plot=TRUE` a plot summarizing the number (or fraction) of switches with specific consequences is created.

Author(s)

Kristoffer Vitting-Seerup

References

Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. Mol. Cancer Res. (2017).

See Also

[analyzeSwitchConsequences](#)
[extractConsequenceEnrichment](#)
[extractConsequenceEnrichmentComparison](#)
[extractConsequenceGenomeWide](#)

Examples

```
### Prepare example data
data("exampleSwitchListAnalyzed")

### Summarize switch consequences
consequenceSummary <- extractConsequenceSummary(
  exampleSwitchListAnalyzed,
  returnResult = TRUE,      # return data.frame with summary
  plotGenes = TRUE         # plot summary
)

dim(consequenceSummary)

subset(consequenceSummary, featureCompared=='Domains identified')
```

extractGeneExpression *Extract raw gene counts or abundances from a switchAnalyzeRlist object.*

Description

Extract replicate gene raw unnormalised counts or expression from a switchAnalyzeRlist object using all the annotation fixes employed in creating the switchAnalyzeRlist.

Usage

```
extractGeneExpression(
  switchAnalyzeRlist,
  extractCounts = TRUE,
  addGeneNames = TRUE,
  addIdsAsColumns = TRUE
)
```

Arguments

- `switchAnalyzeRlist`
A `switchAnalyzeRlist` object.
- `extractCounts` A logic to indicate whether to extract raw unnormalised counts (if TRUE, default) or expression estimates (if FALSE).
- `addGeneNames` A logic to indicate whether to add `gene_names` to the expression matrix (if TRUE, default) or not (if FALSE).
- `addIdsAsColumns`
A logic to indicate whether to add the gene identifiers to the `data.frame` as columns (if TRUE, default) or rownames (if FALSE).

Details

The count matrix obtained if `extractCounts=TRUE` is the same as would be obtained by running `tximport` with `countsFromAbundance="scaledTPM"` which are suitable both for analysis of differential expression and usage.

Value

A `data.frame` with the replicate count/abundance estimates as well as `gene_id` (and `gene_name` if `extractCounts=TRUE`)

Author(s)

Kristoffer Vitting-Seerup

References

Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. *Mol. Cancer Res.* (2017).

Examples

```
data("exampleSwitchList")

### Raw count matrix
geneCountMatrix <- extractGeneExpression(
  exampleSwitchList,
  extractCounts = TRUE
)

### Raw count matrix - with ids as rownames instead of columns
geneCountMatrix <- extractGeneExpression(
  exampleSwitchList,
  extractCounts = TRUE,
  addIdsAsColumns = FALSE
)

### Abundance matrix
geneExpresionMatrix <- extractGeneExpression(
```

```

    exampleSwitchList,
    extractCounts = FALSE
)

```

extractSequence *Extract nucleotide (and amino acid) sequence of transcripts.*

Description

This function extracts the nucleotide (NT) sequence of transcripts by extracting and concatenating the sequences of a reference genome corresponding to the genomic coordinates of the isoforms. If ORF is annotated (e.g. via `analyzeORF`) this function can furthermore translate the ORF NT sequence to Amino Acid (AA) sequence (via the `Biostrings::translate()` function where `if.fuzzy.codon='solve'` is specified). The sequences (both NT and AA) can be outputted as fasta file(s) and/or added to the `switchAnalyzeRlist`.

Usage

```

extractSequence(
  switchAnalyzeRlist,
  genomeObject = NULL,
  onlySwitchingGenes = TRUE,
  alpha = 0.05,
  dIFcutoff = 0.1,
  extractNTseq = TRUE,
  extractAAseq = TRUE,
  removeShortAAseq = TRUE,
  removeLongAAseq = FALSE,
  alsoSplitFastaFile = FALSE,
  removeORFwithStop=TRUE,
  addToSwitchAnalyzeRlist = TRUE,
  writeToFile = TRUE,
  pathToOutput = getwd(),
  outputPrefix='isoformSwitchAnalyzeR_isoform',
  forceReExtraction = FALSE,
  quiet=FALSE
)

```

Arguments

`switchAnalyzeRlist` A `switchAnalyzeRlist` object (where ORF info (predicted by [analyzeORF](#)) have been added if the amino acid sequence should be extracted).

`genomeObject` A `BSgenome` object uses as reference genome (for example `Hsapiens` for `Homo sapiens`, `Mmusculus` for mouse). Only necessary if sequences have not already been extracted.

onlySwitchingGenes	A logic indicating whether the only sequences from transcripts in genes with significant switching isoforms (as indicated by the alpha and dIFcutoff cutoff) should be extracted. Default is TRUE.
alpha	The cutoff which the FDR correct p-values must be smaller than for calling significant switches. Default is 0.05.
dIFcutoff	The cutoff which the changes in (absolute) isoform usage must be larger than before an isoform is considered switching. This cutoff can remove cases where isoforms with (very) low dIF values are deemed significant and thereby included in the downstream analysis. This cutoff is analogous to having a cutoff on log2 fold change in a normal differential expression analysis of genes to ensure the genes have a certain effect size. Default is 0.1 (10%).
extractNTseq	A logical indicating whether the nucleotide sequence of the transcripts should be extracted (necessary for CPAT analysis). Default is TRUE.
extractAAseq	A logical indicating whether the amino acid (AA) sequence of the annotated open reading frames (ORF) should be extracted (necessary for pfam and SignalP analysis). The ORF can be annotated with the analyzeORF function. Default is TRUE.
removeShortAAseq	A logical indicating whether to remove sequences based on their length. This option exist to allows for easier usage of the Pfam and SignalP web servers which both currently have restrictions on allowed sequence lengths. If enabled AA sequences are filtered to be > 5 AA. This will only affect the sequences written to the fasta file (if writeToFile=TRUE) not the sequences added to the switchAnalyzeRlist (if addToSwitchAnalyzeRlist=TRUE). Default is TRUE.
removeLongAAseq	A logical indicating whether to removesequences based on their length. This option exist to allows for easier usage of the Pfam and SignalP web servers which both currently have restrictions on allowed sequence lengths. If enabled AA sequences are filtered to be < 1000 AA. This will only affect the sequences written to the fasta file (if writeToFile=TRUE) not the sequences added to the switchAnalyzeRlist (if addToSwitchAnalyzeRlist=TRUE). Default is FALSE.
alsoSplitFastaFile	A subset of the web based analysis tools currently supported by IsoformSwitch-AnalyzeR have restrictions on the number of sequences in each submission (currently PFAM and to a less extend SignalP). To enable easy use of those web tool this parameter was implemented. By setting this parameter to TRUE a number of amino acid FASTA files will ALSO be generated each only containing the number of sequences allow (currently max 500 for some tools) thereby enabling easy analysis of the data in multiple web-based submissions. Only considered (if writeToFile=TRUE).
removeORFwithStop	A logical indicating whether ORFs containing stop codons, defined as * when the ORF nucleotide sequences is translated to the amino acid sequence, should be A) removed from the ORF annotation in the switchAnalyzeRlist and B) removed from the sequences added to the switchAnalyzeRlist and/or written to fasta files. This is only necessary if you are analyzing quantified known annotated data where you supplied a GTF file to the import function. If you have

	used analyzeORF to identify ORFs this should not have an effect. This option will have no effect if no ORFs are found. Default is TRUE.
addToSwitchAnalyzeRlist	A logical indicating whether the extracted sequences should be added to the switchAnalyzeRlist. Default is TRUE.
writeToFile	A logical indicating whether the extracted sequence(s) should be exported to (separate) fasta files (thereby enabling analysis with external software such as CPAT, Pfam and SignalP). Default is TRUE.
pathToOutput	If writeToFile is TRUE, this argument controls the path to the directory where the fasta files are exported to. Default is working directory.
outputPrefix	If writeToFile=TRUE this argument allows for a user specified prefix of the output files(s). The prefix provided here will get a suffix of '_nt.fasta' or '_AA.fasta' depending on the file type. Default is 'isoformSwitchAnalyzeR_isoform' (thereby creating the 'isoformSwitchAnalyzeR_isoform_nt.fasta' and 'isoformSwitchAnalyzeR_isoform_AA.fasta' files).
forceReExtraction	A logic indicating whether to force re-extraction of the biological sequences - else sequences already stored in the switchAnalyzeRlist will be used instead if available (because this function had already been used once). Default is FALSE
quiet	A logic indicating whether to avoid printing progress messages. Default is FALSE

Details

Changes in isoform usage are measure as the difference in isoform fraction (dIF) values, where isoform fraction (IF) values are calculated as $\langle \text{isoform_exp} \rangle / \langle \text{gene_exp} \rangle$.

The BSgenome object are loaded as separate packages. Use for example `library(BSgenome.Hsapiens.UCSC.hg19)` to load the human genome v19 - which is then loaded as the object `Hsapiens` (that should be supplied to the `genomeObject` argument). It is essential that the chromosome names of the annotation fit with the genome object. The `extractSequence` function will automatically take the most common ambiguity into account: whether to use 'chr' in front of the chromosome name (UCSC style, e.g.. 'chr1') or not (Ensembl style, e.g.. '1').

The two fasta files outputted by this function (if `writeToFile=TRUE`) can be used as input to among others:

- CPAT : The Coding-Potential Assessment Tool, which can be run either locally or via their webserver <http://lilab.research.bcm.edu/cpat/>
- Pfam : Prediction of protein domains, which can be run either locally or via their webserver <http://pfam.xfam.org/search#tabview=tab1>
- SignalP : Prediction of Signal Peptide, which can be run either locally or via their webserver <http://www.cbs.dtu.dk/services/SignalP/>

See `?analyzeCPAT`, `?analyzePFAM` or `?analyzeSignalP` (under details) for suggested ways of running these tools.

Value

If `writeToFile=TRUE` one fasta file per sequence type (controlled via `extractNTseq` and `extractAAsseq`) are written to the folder indicated by `pathToOutput`. If `alsoSplitFastaFile=TRUE` both a fasta file containing all isoforms (denoted `'_complete'` in file name) as well as a number of fasta files containing subsets of the entire file will be created. The subset fasta files will have the following indication `"subset_X_of_Y"` in the file names. If `addToSwitchAnalyzeRlist=TRUE` the sequences are added to the `switchAnalyzeRlist` as respectively `DNAStrngSet` and `AAStringSet` objects under the names `'ntSequence'` and `'aaSequence'`. The names of these sequences matches the `'isoform_id'` entry in the `'isoformFeatures'` entry of the `switchAnalyzeRlist`. The `switchAnalyzeRlist` is return no matter whether it was modified or not.

Author(s)

Kristoffer Vitting-Seerup

References

For

- This function: Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. *Mol. Cancer Res.* (2017).

See Also

[switchAnalyzeRlist](#)
[isoformSwitchTestDEXSeq](#)
[isoformSwitchTestDRIMSeq](#)
[analyzeORF](#)
[analyzeCPAT](#)
[analyzePFAM](#)
[analyzeSignalP](#)

Examples

```
### Prepare for sequence extraction
# Load example data and prefilter
data("exampleSwitchList")
exampleSwitchList <- preFilter(exampleSwitchList)

# Perform test
exampleSwitchListAnalyzed <- isoformSwitchTestDEXSeq(exampleSwitchList, dIFcutoff = 0.3) # high dIF cutoff for fas

# analyzeORF
library(BSgenome.Hsapiens.UCSC.hg19)
exampleSwitchListAnalyzed <- analyzeORF(exampleSwitchListAnalyzed, genomeObject = Hsapiens)

### Extract sequences
exampleSwitchListAnalyzed <- extractSequence(
  exampleSwitchListAnalyzed,
  genomeObject = Hsapiens,
  writeToFile=FALSE # to avoid output when running example data
```

```

)

### Explore result
head(exampleSwitchListAnalyzed$ntSequence,2)

head(exampleSwitchListAnalyzed$aaSequence,2)

```

```
extractSplicingEnrichment
```

Analyze data for enrichment of specific type of alternative splicing

Description

This functions function analyzes (the number of and) enrichment of specific splice events by for each set of opposing event (e.g.. exon skipping gain vs loss), by analyzing the fraction of events belonging to each type of consequence. Please note this summarizes the differences between the isoforms in a switch - for an overview of the total number of AS events please use [extractSplicing-Summary](#).

Usage

```

extractSplicingEnrichment(
  switchAnalyzeRlist,
  splicingToAnalyze = 'all',
  alpha = 0.05,
  dIFcutoff = 0.1,
  onlySigIsoforms = FALSE,
  countGenes = TRUE,
  plot = TRUE,
  localTheme = theme_bw(base_size = 14),
  minEventsForPlotting = 10,
  returnResult=TRUE,
  returnSummary=TRUE
)

```

Arguments

switchAnalyzeRlist	A switchAnalyzeRlist object where analyzeSwitchConsequences() have been run to identify consequences of isoform switches
splicingToAnalyze	A string indicating which consequences should be considered. See details for description. Default is all.
alpha	The cutoff which the FDR correct p-values must be smaller than for calling significant switches. Default is 0.05.

dIFcutoff	The cutoff which the changes in (absolute) isoform usage must be larger than before an isoform is considered switching. This cutoff can remove cases where isoforms with (very) low dIF values are deemed significant and thereby included in the downstream analysis. This cutoff is analogous to having a cutoff on log2 fold change in a normal differential expression analysis of genes to ensure the genes have a certain effect size. Default is 0.1 (10%).
onlySigIsoforms	A logic indicating whether to only consider significant isoforms, meaning only analyzing genes where at least two isoforms which both have significant usage changes in opposite direction (quite strict) Naturally this only works if the isoform switch test used have isoform resolution (which the build in isoform-SwitchTestDEXSeq has). If FALSE all isoforms with an absolute dIF value larger than dIFcutoff in a gene with significant switches (defined by alpha and dIFcutoff) are included in the pairwise comparison. Default is FALSE (non significant isoforms are also considered based on the logic that if one isoform changes its contribution - there must be an equivalent opposite change in usage in the other isoforms from that gene).
countGenes	A logic indicating whether it is the number of genes (if TRUE) or isoform switches (if FALSE) which primary result in gain/loss that are counted. Default is TRUE.
plot	A logic indicating whether the analysis should be plotted. If TRUE and returnResult = FALSE the ggplot2 object will be returned instead. Default is TRUE.
localTheme	General ggplot2 theme with which the plot is made, see ?ggplot2::theme for more info. Default is theme_bw(base_size = 14).
minEventsForPlotting	The minimum number of events (total gain/loss) must be present before the result is visualized. Default is 10.
returnResult	A logic indicating whether the analysis should be returned as a data.frame. If FALSE (and plot=TRUE) the ggplot2 object will be returned instead. Default is TRUE.
returnSummary	A logic indicating whether to return the statistical summary (if TRUE) or the underlying data (if FALSE). If FALSE (and plot=TRUE) the ggplot2 object will be returned instead. Default is TRUE.

Details

The classification of alternative splicing is always compared to the hypothetical pre-mRNA constructed by concatenating all exons from isoforms of the same gene.

The alternative splicing types, which can be passed to `splicingToAnalyze` must be a combination of:

- all : All of the alternative splicing types indicated below.
- IR : Intron Retention.
- A5 : Alternative 5' donor site (changes in the 5' end of the upstream exon).
- A3 : Alternative 3' acceptor site (changes in the 3' end of the downstream exon).

- ATSS : Alternative Transcription Start Site.
- ATTS : Alternative Transcription Termination Site.
- ES : Exon Skipping (EI means Exon Inclusion).
- MES : Multiple Exon Skipping. Skipping of >1 consecutive exons. (MEI means Multiple Exon Inclusion).
- MEE : Mutually Exclusive Exons.

For details of how to interpret the splice events see the details section of [analyzeAlternativeSplicing](#).

The significance test is performed with R's build in `prop.test()` with default parameters and resulting p-values are corrected via `p.adjust()` using FDR (Benjamini-Hochberg).

Value

If `plot=TRUE` a plot summarizing the proportions is also created of switches with specific consequences is created.

If `returnResult=TRUE` a data.frame with the statistical summary for each opposing consequences in each comparison. This data.frame will have the following columns:

- `condition_1`: Condition 1.
- `condition_2`: Condition 2.
- `AStype`: The type of splicing considered.
- `nUp`: The number of genes with a gain of the splicing type described in the `AStype` column.
- `nDown`: The number of genes with a loss of the splicing type described in the `AStype` column.
- `propUp`: Proportion of total number of genes (of genes with either loss or gain of the splice type described in the `AStype` column) being having a gain.
- `propUpCiLo`: The lower boundary of the confidence interval of the `propUp`.
- `propUpCiHi`: The high boundary of the confidence interval of the `propUp`.
- `propUpPval`: The p-value associated with the null hypothesis that `propUp` is 0.5.
- `propUpQval`: The q-values resulting when p-values are corrected via `p.adjust()` using FDR (Benjamini-Hochberg).

Author(s)

Kristoffer Vitting-Seerup

References

- Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. *Mol. Cancer Res.* (2017).
- Vitting-Seerup et al. IsoformSwitchAnalyzeR: Analysis of changes in genome-wide patterns of alternative splicing and its functional consequences. *Bioinformatics* (2019).

See Also

```

isoformSwitchTestDEXSeq
isoformSwitchTestDRIMSeq
analyzeAlternativeSplicing
extractSplicingSummary
extractSplicingEnrichmentComparison
extractSplicingGenomeWide

```

Examples

```

### Load example data
data("exampleSwitchListAnalyzed")

extractSplicingEnrichment( exampleSwitchListAnalyzed )

```

```

extractSplicingEnrichmentComparison

```

Compare enrichment of specific type of alternative splicing between comparisons

Description

This function compares the enrichment of alternative splicing (f.x. exon skipping) between two comparisons (ctrl vs ko1 compared to ctrl vs ko2) and reports whether there is a significant difference between the comparisons. In other words it compares the output of `extractSplicingEnrichment`.

Usage

```

extractSplicingEnrichmentComparison(
  switchAnalyzeRlist,
  splicingToAnalyze = 'all',
  alpha = 0.05,
  dIFcutoff = 0.1,
  onlySigIsoforms = FALSE,
  countGenes = TRUE,
  plot = TRUE,
  localTheme = theme_bw(base_size = 14),
  minEventsForPlotting = 10,
  returnResult=TRUE
)

```

Arguments

```

switchAnalyzeRlist

```

A `switchAnalyzeRlist` object where `analyzeSwitchConsequences()` have been run to identify consequences of isoform switches

splicingToAnalyze	A string indicating which consequences should be considered. See details for description. Default is all.
alpha	The cutoff which the FDR correct p-values must be smaller than for calling significant switches. Default is 0.05.
dIFcutoff	The cutoff which the changes in (absolute) isoform usage must be larger than before an isoform is considered switching. This cutoff can remove cases where isoforms with (very) low dIF values are deemed significant and thereby included in the downstream analysis. This cutoff is analogous to having a cutoff on log2 fold change in a normal differential expression analysis of genes to ensure the genes have a certain effect size. Default is 0.1 (10%).
onlySigIsoforms	A logic indicating whether to only consider significant isoforms, meaning only analyzing genes where at least two isoforms which both have significant usage changes in opposite direction (quite strict) Naturally this only works if the isoform switch test used have isoform resolution (which the build in isoform-SwitchTestDEXSeq has). If FALSE all isoforms with an absolute dIF value larger than dIFcutoff in a gene with significant switches (defined by alpha and dIFcutoff) are included in the pairwise comparison. Default is FALSE (non significant isoforms are also considered based on the logic that if one isoform changes its contribution - there must be an equivalent opposite change in usage in the other isoforms from that gene).
countGenes	A logic indicating whether it is the number of genes (if TRUE) or isoform switches (if FALSE) which primary result in gain/loss that are counted. Default is TRUE.
plot	A logic indicating whether the analysis should be plotted. If TRUE and returnResult = FALSE the ggplot2 object will be returned instead. Default is TRUE.
localTheme	General ggplot2 theme with which the plot is made, see <code>?ggplot2::theme</code> for more info. Default is <code>theme_bw(base_size = 14)</code> .
minEventsForPlotting	The minimum number of events (total gain/loss) must be present before the result is visualized. Default is 10.
returnResult	A logic indicating whether the analysis should be returned as a data.frame. If FALSE (and plot=TRUE) the ggplot2 object will be returned instead. Default is FALSE.

Details

The classification of alternative splicing is always compared to the hypothetical pre-mRNA constructed by concatenating all exons from isoforms of the same gene.

The alternative splicing types, which can be passed to `splicingToAnalyze` must be a combination of:

- all : All of the alternative splicing types indicated below.
- IR : Intron Retention.

- A5 : Alternative 5' donor site (changes in the 5' end of the upstream exon).
- A3 : Alternative 3' acceptor site (changes in the 3' end of the downstream exon).
- ATSS : Alternative Transcription Start Site.
- ATTS : Alternative Transcription Termination Site.
- ES : Exon Skipping.
- MES : Multiple Exon Skipping. Skipping of >1 consecutive exons.
- MEE : Mutually Exclusive Exons.

For details of how to interpret the splice events see the details section of [analyzeAlternativeSplicing](#).

The significance test is performed with R's built in `fisher.test()` with default parameters and resulting p-values are corrected via `p.adjust()` using FDR (Benjamini-Hochberg).

Value

If `returnResult=TRUE` a data.frame with the statistical summary for each opposing consequences in each comparison. If `plot=TRUE` a plot summarizing the proportions is also created of switches with specific consequences is created.

Author(s)

Kristoffer Vitting-Seerup

References

- Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. Mol. Cancer Res. (2017).
- Vitting-Seerup et al. IsoformSwitchAnalyzeR: Analysis of changes in genome-wide patterns of alternative splicing and its functional consequences. Bioinformatics (2019).

See Also

[isoformSwitchTestDEXSeq](#)
[isoformSwitchTestDRIMSeq](#)
[analyzeAlternativeSplicing](#)
[extractSplicingSummary](#)
[extractSplicingEnrichment](#)
[extractSplicingGenomeWide](#)

Examples

```
### Load example data
data("exampleSwitchListAnalyzed")

extractSplicingEnrichmentComparison( exampleSwitchListAnalyzed )
```

`extractSplicingGenomeWide`*Genome wide Analysis of alternative splicing*

Description

This function enables a genome wide analysis of changes in isoform usage of isoforms with a common annotation.

Specifically this function extract isoforms of interest and for each splicing type (such as exon skipping) the global distribution of IF (measuring isoform usage) are plotted for each subset of features in that category (e.g with exons skipping vs without exon skipping). This enables a global analysis of isoforms with a common annotation.

The isoforms of interest can either be defined by isoforms form gene differentially expressed, isoform that are differential expressed or isoforms from genes with isoform switching - as controlled by `featureToExtract`. Please note that the [extractSplicingEnrichment](#) function probably more relevant than using `featureToExtract='isoformUsage'` since it directly uses the paired information from switches.

This function offers both visualization of the result as well as analysis via summary statistics of the comparisons.

Usage

```
extractSplicingGenomeWide(  
  switchAnalyzeRlist,  
  featureToExtract = 'isoformUsage',  
  splicingToAnalyze = 'all',  
  alpha=0.05,  
  dIFcutoff = 0.1,  
  log2FCcutoff = 1,  
  violinPlot=TRUE,  
  alphas=c(0.05, 0.001),  
  localTheme=theme_bw(),  
  plot=TRUE,  
  returnResult=TRUE  
)
```

Arguments

`switchAnalyzeRlist`

A `switchAnalyzeRlist` object containing the result of an isoform switch analysis (such as the one provided by `isoformSwitchTestDEXSeq()`) as well as additional annotation data for the isoforms.

`featureToExtract`

This argument, given as a string, defines the set isoforms which should be analyzed. The available options are:

- 'isoformUsage' (Default): Analyze a subset of isoforms defined by change in isoform usage (controlled by dIFcutoff) and the significance of the change in isoform expression (controlled by alpha). Please note that the [extractSplicingEnrichment](#) function probably more relevant than using featureToExtract='isoform' since it directly uses the paired information from switches.
- 'isoformExp' :Analyze a subset of isoforms defined by change in isoform expression (controlled by log2FCcutoff) and the significance of the change in isoform expression (controlled by alpha)
- 'geneExp' :Analyze all isoforms from a subset of genes defined by change in gene expression (controlled by log2FCcutoff) and the significance of the change in gene expression (controlled by alpha)
- 'all' : Analyze all isoforms stored in the switchAnalyzeRlist (note that this is highly depending on the parameter reduceToSwitchingGenes in [isoformSwitchTestDEXSeq](#) - which should be set to FALSE (default is TRUE) if the 'all' option should be used here).

splicingToAnalyze	A string indicating which consequences should be considered. See details for description. Default is all.
alpha	The cutoff which the FDR correct p-values (q-values) must be smaller than for calling significant switches. Default is 0.05.
dIFcutoff	The cutoff which the changes in (absolute) isoform usage must be larger than before an isoform is considered switching. This cutoff can remove cases where isoforms with (very) low dIF values are deemed significant and thereby included in the downstream analysis. This cutoff is analogous to having a cutoff on log2 fold change in a normal differential expression analysis of genes to ensure the genes have a certain effect size. Default is 0.1 (10%).
log2FCcutoff	The cutoff which the changes in (absolute) isoform or gene expression must be larger than before an isoform is considered for inclusion.
violinPlot	A logical indicating whether to make a violin plots (if TRUE) or boxplots (if FALSE). Violin plots will always have added 3 black dots, one of each of the 25th, 50th (median) and 75th percentile of the data. Default is TRUE.
alphas	A numeric vector of length two giving the significance levels represented in plots. The numbers indicate the q-value cutoff for significant (*) and highly significant (***) respectively. Default 0.05 and 0.001 which should be interpret as $q < 0.05$ and $q < 0.001$ respectively). If q-values are higher than this they will be annotated as 'ns' (not significant).
localTheme	General ggplot2 theme with which the plot is made, see ?ggplot2::theme for more info. Default is theme_bw().
plot	A logic indicting whether the analysis should be plotted. If TRUE and returnResult = FALSE the ggplot2 object will be returned instead. Default is TRUE.
returnResult	A logical indicating whether to return a data.frame with summary statistics of the comparisons (if TRUE) or not (if FALSE). If FALSE (and plot=TRUE) the ggplot2 object will be returned instead. Default is TRUE.

Details

The classification of alternative splicing is always compared to the hypothetical pre-mRNA constructed by concatenating all exons from isoforms of the same gene.

The alternative splicing types, which can be passed to `splicingToAnalyze` must be a combination of:

- `all` : All of the alternative splicing types indicated below.
- `IR` : Intron Retention.
- `A5` : Alternative 5' donor site (changes in the 5' end of the upstream exon).
- `A3` : Alternative 3' acceptor site (changes in the 3' end of the downstream exon).
- `ATSS` : Alternative Transcription Start Site.
- `ATTS` : Alternative Transcription Termination Site.
- `ES` : Exon Skipping.
- `MES` : Multiple Exon Skipping. Skipping of >1 consecutive exons.
- `MEE` : Mutually Exclusive Exons.

The significance test is performed with R's built in `wilcox.test()` (aka 'Mann-Whitney-U') with default parameters and resulting p-values are corrected via `p.adjust()` using FDR (Benjamini-Hochberg).

Value

If `plot=TRUE`: A plot of the distribution of IF values as a function of the annotation and condition compared. If `returnResult=TRUE`: A data.frame with the summary statistics from the comparison of the two conditions with a `Wilcox.test`.

Author(s)

Kristoffer Vitting-Seerup

References

Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. *Mol. Cancer Res.* (2017).

See Also

[isoformSwitchTestDEXSeq](#)
[isoformSwitchTestDRIMSeq](#)
[analyzeAlternativeSplicing](#)
[extractSplicingSummary](#)
[extractSplicingEnrichment](#)
[extractSplicingEnrichmentComparison](#)

Examples

```
### Load example data
data("exampleSwitchListAnalyzed")

extractSplicingGenomeWide( exampleSwitchListAnalyzed )
```

```
extractSplicingSummary
```

Extracts alternative splicing summary

Description

This functions function summarizes the individual alternative splicing events for each gene or switches and plots and/or returns a data.frame with the information. Please note this summarizes the overall number of splicing events - for looking into differences between the isoforms in a switch please use [extractSplicingEnrichment](#).

Usage

```
extractSplicingSummary(
  switchAnalyzeRlist,
  splicingToAnalyze = 'all',
  asFractionTotal = FALSE,
  alpha = 0.05,
  dIFcutoff = 0.1,
  onlySigIsoforms = FALSE,
  plot = TRUE,
  plotGenes = FALSE,
  localTheme = theme_bw(),
  returnResult = FALSE
)
```

Arguments

switchAnalyzeRlist	A switchAnalyzeRlist object where analyzeSwitchConsequences() have been run to identify consequences of isoform switches
splicingToAnalyze	A string indicating which consequences should be considered. See details for description. Default is all.
asFractionTotal	A logic indicating whether the consequences should be summarized calculated as numbers (if FALSE) or as a fraction of the total number of switches/genes (as indicated by plotGenes). Default is FALSE.
alpha	The cutoff which the FDR correct p-values must be smaller than for calling significant switches. Default is 0.05.

dIFcutoff	The cutoff which the changes in (absolute) isoform usage must be larger than before an isoform is considered switching. This cutoff can remove cases where isoforms with (very) low dIF values are deemed significant and thereby included in the downstream analysis. This cutoff is analogous to having a cutoff on log2 fold change in a normal differential expression analysis of genes to ensure the genes have a certain effect size. Default is 0.1 (10%).
onlySigIsoforms	A logic indicating whether to only consider significant isoforms, meaning only analyzing genes where at least two isoforms which both have significant usage changes in opposite direction (quite strict) Naturally this only works if the isoform switch test used have isoform resolution (which the build in isoform-SwitchTestDEXSeq has). If FALSE all isoforms with an absolute dIF value larger than dIFcutoff in a gene with significant switches (defined by alpha and dIFcutoff) are included in the pairwise comparison. Default is FALSE (non significant isoforms are also considered based on the logic that if one isoform changes its contribution - there must be an equivalent opposite change in usage in the other isoforms from that gene).
plot	A logic indicating whether the summarized results should be plotted. If TRUE and returnResult = FALSE the ggplot2 object will be returned instead. Default is TRUE.
plotGenes	A logic indicating whether to plot the number/fraction of genes (if TRUE) or switches (if FALSE) with functional consequences should be plotted.
localTheme	General ggplot2 theme with which the plot is made, see ?ggplot2::theme for more info. Default is theme_bw().
returnResult	A logic indicating whether the summarized results should be returned as a data.frame. If FALSE (and plot=TRUE) the ggplot2 object will be returned instead. Default is TRUE.

Details

The classification of alternative splicing is always compared to the hypothetical pre-mRNA constructed by concatenating all exons from isoforms of the same gene.

The alternative splicing types, which can be passed to `splicingToAnalyze` must be a combination of:

- all : All of the alternative splicing types indicated below.
- IR : Intron Retention.
- A5 : Alternative 5' donor site (changes in the 5' end of the upstream exon).
- A3 : Alternative 3' acceptor site (changes in the 3' end of the downstream exon).
- ATSS : Alternative Transcription Start Site.
- ATTS : Alternative Transcription Termination Site.
- ES : Exon Skipping.
- MES : Multiple Exon Skipping. Skipping of >1 consecutive exons.
- MEE : Mutually Exclusive Exons.

For details of how to interpret the splice events see the details section of [analyzeAlternativeSplicing](#).

Value

If returnResult=TRUE a data.frame with the number (and fraction) of switches with specific consequences in each condition is returned. If plot=TRUE a plot summarizing the number (or fraction) of switches with specific consequences is created.

Author(s)

Kristoffer Vitting-Seerup

References

Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. Mol. Cancer Res. (2017).

See Also

[analyzeAlternativeSplicing](#)
[extractSplicingEnrichment](#)
[extractSplicingEnrichmentComparison](#)
[extractSplicingGenomeWide](#)

Examples

```
### Load example data
data("exampleSwitchListAnalyzed")

extractSplicingSummary( exampleSwitchListAnalyzed )
```

extractSwitchOverlap *Visualize Switch Overlap*

Description

This function produces two Venn diagrams respectively showing the overlap in switching isoforms and genes.

Usage

```
extractSwitchOverlap(
  switchAnalyzeRlist,
  filterForConsequences = FALSE,
  alpha = 0.05,
  dIFcutoff = 0.1,
  scaleVennIfPossible=TRUE,
```

```

    plotIsoforms = TRUE,
    plotSwitches = TRUE,
    plotGenes = TRUE
)

```

Arguments

switchAnalyzeRlist	A switchAnalyzeRlist object.
filterForConsequences	A logical indicating whether to filter for genes with functional consequences. Requires that analyzeSwitchConsequences() have been run on the switchAnalyzeRlist. The output will then be the number of significant genes and isoforms originating from genes with predicted consequences. Default is FALSE.
alpha	The cutoff which the FDR correct p-values must be smaller than for calling significant switches. Default is 0.05.
dIFcutoff	The cutoff which the changes in (absolute) isoform usage must be larger than before an isoform is considered switching. This cutoff can remove cases where isoforms with (very) low dIF values are deemed significant and thereby included in the downstream analysis. This cutoff is analogous to having a cutoff on log2 fold change in a normal differential expression analysis of genes to ensure the genes have a certain effect size. Default is 0.1 (10%).
scaleVennIfPossible	A logic indicating whether the Venn diagram should be scaled (so the circle area and overlap size reflect the number of features) if possible. Only available for 2- and 3-way Venn Diagrams. Default is TRUE.
plotIsoforms	A logic indicating whether the Venn diagram of differentially used isoforms should be plotted. Default is TRUE.
plotSwitches	A logic indicating whether the Venn diagram of identified isoform switches should be plotted. Default is TRUE.
plotGenes	A logic indicating whether the Venn diagram of genes containing differentially used isoforms should be plotted. Default is TRUE.

Value

A Venn diagram which shows the number of isoforms and genes with a isoform switch.

Author(s)

Kristoffer Vitting-Seerup

References

- Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. Mol. Cancer Res. (2017).

See Also

```

preFilter
isoformSwitchTestDEXSeq
isoformSwitchTestDRIMSeq
extractTopSwitches
extractSwitchSummary
analyzeSwitchConsequences

```

Examples

```

# Load example data and prefilter
data("exampleSwitchListAnalyzed")

extractSwitchOverlap(exampleSwitchListAnalyzed)

```

```

extractSwitchSummary  Summarize Isoform Switches test Result.

```

Description

Summarize the number of switching isoforms/genes identified.

Usage

```

extractSwitchSummary(
  switchAnalyzeRlist,
  filterForConsequences=FALSE,
  alpha=0.05,
  dIFcutoff = 0.1,
  onlySigIsoforms = FALSE,
  includeCombined=nrow(unique(switchAnalyzeRlist$isoformFeatures[,c('condition_1','condition_1')]))
)

```

Arguments

```

switchAnalyzeRlist
    A switchAnalyzeRlist object.

filterForConsequences
    A logical indicating whether to filter for genes with functional consequences. Requires that analyzeSwitchConsequences() have been run on the switchAnalyzeRlist. The output will then be the number of significant genes and isoforms originating from genes with predicted consequences. Default is FALSE.

alpha
    The cutoff which the FDR correct p-values must be smaller than for calling significant switches. Default is 0.05.

```

- dIFcutoff** The cutoff which the changes in (absolute) isoform usage must be larger than before an isoform is considered switching. This cutoff can remove cases where isoforms with (very) low dIF values are deemed significant and thereby included in the downstream analysis. This cutoff is analogous to having a cutoff on log2 fold change in a normal differential expression analysis of genes to ensure the genes have a certain effect size. Default is 0.1 (10%).
- onlySigIsoforms** A logic indicating whether to only consider significant isoforms, meaning only analyzing genes where at least two isoforms which both have significant usage changes in opposite direction (quite strict) Naturally this only works if the isoform switch test used have isoform resolution (which the build in [isoform-SwitchTestDEXSeq](#) has). If FALSE all isoforms with an absolute dIF value larger than dIFcutoff in a gene with significant switches (defined by alpha and dIFcutoff) are included in the pairwise comparison. Default is FALSE (non significant isoforms are also considered based on the logic that if one isoform changes its contribution - there must be an equivalent opposite change in usage in the other isoforms from that gene).
- includeCombined** A logic indicating whether a combined summary across all comparisons should also be made. Default is TRUE if more than 1 comparison is analyzed and FALSE if only 1 comparison is analyzed.

Value

A data.frame with the number of switches found in each comparison (as well as when all data is considered if includeCombined=TRUE)

Author(s)

Kristoffer Vitting-Seerup

References

Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. Mol. Cancer Res. (2017).

See Also

[preFilter](#)
[isoformSwitchTestDEXSeq](#)
[isoformSwitchTestDRIMSeq](#)
[extractSwitchOverlap](#)
[extractTopSwitches](#)
[analyzeSwitchConsequences](#)

Examples

```
# Load example data and prefilter  
data("exampleSwitchList")
```

```

exampleSwitchList <- preFilter(exampleSwitchList)

# Perform test
exampleSwitchListAnalyzed <- isoformSwitchTestDEXSeq(exampleSwitchList)

# extract summary of number of switching features
extractSwitchSummary(exampleSwitchListAnalyzed)

```

extractTopSwitches *Extract Top Isoform Switches.*

Description

This function allows the user extract the (top) switching genes/isoforms (with functional consequences).

Usage

```

extractTopSwitches(
  switchAnalyzeRlist,
  filterForConsequences=FALSE,
  extractGenes=TRUE,
  alpha=0.05,
  dIFcutoff = 0.1,
  n=10,
  inEachComparison=FALSE,
  sortByQvals=TRUE
)

```

Arguments

switchAnalyzeRlist	A switchAnalyzeRlist object.
extractGenes	A logic indicating whether to extract the (top) switching isoforms (if FALSE) or top switching genes (if TRUE). Default is TRUE (extract genes).
filterForConsequences	A logical indicating whether to filter for genes with functional consequences. Requires that analyzeSwitchConsequences() have been run on the switchAnalyzeRlist. Default is FALSE.
alpha	The cutoff which the FDR correct p-values must be smaller than for calling significant switches. Default is 0.05.
dIFcutoff	The cutoff which the changes in (absolute) isoform usage must be larger than before an isoform is considered switching. This cutoff can remove cases where isoforms with (very) low dIF values are deemed significant and thereby included in the downstream analysis. This cutoff is analogous to having a cutoff on log2 fold change in a normal differential expression analysis of genes to ensure the genes have a certain effect size. Default is 0.1 (10%).

n	The number of switching features (genes/isoforms) to return. Use Inf to return all significant results (NA will internally be converted to Inf for backward comparability). Default is 10.
inEachComparison	A logic indicating whether to extract top n in each comparison (if TRUE) or from the all analysis (if FALSE). Default is FALSE.
sortByQvals	A logic indicating whether the top n features are defined by smallest q-values (if sortByQvals=TRUE) or the largest changes in isoform usage (absolute dIF) which are still significant (if sortByQvals=FALSE). The dIF values for genes are considered as the total change within the gene calculated as $\text{sum}(\text{abs}(\text{dIF}))$ for each gene. If set to NA no sorting is performed. Default is TRUE (sort by p-values).

Value

A data.frame containing the top n switching genes or isoforms as controlled by the extractGenes argument, sorted by q-values or dIF values as controlled by the sortByQvals argument.

Author(s)

Kristoffer Vitting-Seerup

References

Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. Mol. Cancer Res. (2017).

See Also

[preFilter](#)
[isoformSwitchTestDEXSeq](#)
[isoformSwitchTestDRIMSeq](#)
[analyzeSwitchConsequences](#)

Examples

```
# Load example data and prefilter
data("exampleSwitchList")
exampleSwitchList <- preFilter(exampleSwitchList)

# Perform test
exampleSwitchListAnalyzed <- isoformSwitchTestDEXSeq(exampleSwitchList)

# extract summary of number of switching features
extractSwitchSummary(exampleSwitchListAnalyzed)

### Filter for functional consequences (identified via analyzeSwitchConsequences() )
data("exampleSwitchListAnalyzed")
switchingIso <- extractTopSwitches(
```

```
    exampleSwitchListAnalyzed,  
    filterForConsequences = TRUE,  
  )  
  dim(switchingIso)  
  head(switchingIso,2)
```

getCDS

Retrieve CDS information from UCSC

Description

Retrieve CDS information from a selected repository from UCSC genome browser repositories.

Usage

```
getCDS(selectedGenome, repoName)
```

Arguments

`selectedGenome` A character, giving the genome. Currently supported are "hg19" and "mm9".
`repoName` A character, giving the gene model repository. Currently supported are "ensemble", "UCSC" (knownGene), and "refseq".

Details

For other genomes and/or gene model repositories, please construct a `CDSset` directly (see [CDSset](#)).
For a full example of how to use `getCDS` in a workflow, please see [analyzeORF](#).

Value

A `CDSset` containing the annotated CDSs. For a description of the `data.frame`, see [CDSset](#).

Author(s)

Kristoffer Vitting-Seerup, Johannes Waage

References

Vitting-Seerup K, et al: spliceR: an R package for classification of alternative splicing and prediction of coding potential from RNA-seq data. BMC Bioinformatics 2014, 15:81.

Examples

```
## Not run:  
mm9UCSC <- getCDS("mm9", "UCSC")  
  
## End(Not run)
```

`importCufflinksFiles` *Import CuffDiff (Cufflinks) Data Into R*

Description

This function enables users to run Cufflinks/Cuffdiff and then afterwards import the result into R for post analysis with isoformSwitchAnalyzeR. The user just has to point IsoformSwitchAnalyzeR to some of the Cuffdiff result files. The data is then imported into R, massaged and returned as a switchAnalyzeRlist enabling a full analysis with IsoformSwitchAnalyzeR. This approach also supports post-analysis of results from Galaxy.

Usage

```
importCufflinksFiles(  
  ### Core arguments  
  pathToGTF,  
  pathToGeneDEanalysis,  
  pathToIsoformDEanalysis,  
  pathToGeneFPKMtracking,  
  pathToIsoformFPKMtracking,  
  pathToIsoformReadGroupTracking,  
  pathToSplicingAnalysis = NULL,  
  pathToReadGroups,  
  pathToRunInfo,  
  isoformNtFasta = NULL,  
  
  ### Advanced arguments  
  fixCufflinksAnnotationProblem = TRUE,  
  addIFmatrix = TRUE,  
  estimateDifferentialGeneRange = TRUE,  
  quiet = FALSE  
)
```

Arguments

<code>pathToGTF</code>	A string indicating the path to the GTF file used as input to Cuffdiff file (downloaded from e.g. galaxy). Please note this file is usually not in the same directory as the CuffDiff results.
<code>pathToGeneDEanalysis</code>	A string indicating the path to the file "gene differential expression testing" file (downloaded from e.g. galaxy).
<code>pathToIsoformDEanalysis</code>	A string indicating the path to the file "transcript differential expression testing" file (downloaded from e.g. galaxy).

<code>pathToGeneFPKMtracking</code>	A string indicating the path to the file "gene FPKM tracking" file (downloaded from e.g. galaxy).
<code>pathToIsoformReadGroupTracking</code>	A string indicating the path to the file "isoform read group tracking" file (downloaded from e.g. galaxy).
<code>pathToIsoformFPKMtracking</code>	A string indicating the path to the file "transcript FPKM tracking" file (downloaded from e.g. galaxy).
<code>pathToSplicingAnalysis</code>	A string indicating the path to the file "splicing differential expression testing" file (downloaded from e.g. galaxy).. Only needed if the splicing analysis should be added. Default is NULL (not added).
<code>pathToReadGroups</code>	A string indicating the path to the file "Read groups" file (downloaded from e.g. galaxy).
<code>pathToRunInfo</code>	A string indicating the path to the file "Run details" file (downloaded from e.g. galaxy).
<code>isoformNtFasta</code>	A (vector of) text string(s) providing the path(s) to the a fasta file containing the nucleotide sequence of all isoforms quantified. This is useful for: 1) people working with non-model organisms where extracting the sequence from a BSgenome might require extra work. 2) workflow speed-up for people who already have the fasta file (which most people running Salmon, Kallisto or RSEM for the quantification have as that is used to build the index). Please note this different from a fasta file with the sequences of the entire genome.
<code>fixCufflinksAnnotationProblem</code>	A logic indicating whether to fix the problem with Cufflinks gene symbol annotation. Please see the details for additional information. Default is TRUE.
<code>addIFmatrix</code>	A logic indicating whether to add the Isoform Fraction replicate matrix (if TRUE) or not (if FALSE). Keeping it will make testing with limma faster but will also make the <code>switchAnalyzeRlist</code> larger - so it is a trade-off for speed vs memory. For most experimental setups we expect that keeping it will be the better solution. Default is TRUE.
<code>estimateDifferentialGeneRange</code>	A logic indicating whether to make a very quick estimate of the number of genes with differential isoform usage. Please note this number should be taken as a pilot and cannot be trusted. It merely servers to indcate what could be expected if the data is analyzed with the rest of the <code>IsoformSwitchAnalyzeR</code> . See details for more information. Default is TRUE.
<code>quiet</code>	A logic indicating whether to avoid printing progress messages. Default is FALSE

Details

One problem with cufflinks is that it considers islands of overlapping transcripts - this means that sometimes multiple genes (defined by gene short name) as combined into one cufflinks gene

(XLOC_XXXXXX) and this gene is quantified and tested for differential expression. Setting `fixCufflinksAnnotationProblem` to `TRUE` will make the `import` function modify the data so that false conclusions are not made in downstream analysis. More specifically this cause the function to recalculate expression values, set gene standard error (of mean) to NA and the p-value and q-value of the differential expression analysis to 1 whereby false conclusions can be prevented.

Cuffdiff performs a statistical test for changes in alternative splicing between transcripts that utilize the same transcription start site (TSS). If evidence for alternative splicing, resulting in alternative isoforms, are found within a gene then there must per definition also be isoform switching occurring within that gene. Therefore we have implemented the `addCufflinksSwichTest` parameter which will add the FDR corrected p-value (q-value) of CuffDiffs splicing test as the gene-level evidence for isoform switching (the `gene_switch_q_value` column). By coupling this evidence with a cutoff on minimum switch size (which is measured a gene-level and controlled via `dIFcutoff`) in the downstream analysis, switches that are not negligible at gene-level will be ignored. Note that CuffDiff have a parameter (`'-min-reps-for-js-test'`) which controls how many replicates (default is 3) are needed for the test of alternative splicing is performed and that the test requires TSSs are annotated in the GTF file supplied to Cuffmerge via the `'-g/-ref-gtf'` parameter.

The `guestimate` produced by setting `estimatedDifferentialGeneRange = TRUE` is created by subsetting a lot on data (both on samples, conditions and genes) and running a fast but unreliable DTU method. The resulting number is then multiplied by a factor to caclulate back what would be expected by running the `IsoformSwitchAnalyzeR` pipeline. It should go without saying due to all these factors the acutal `guestimate` is just that - and estimate which cannot be trusted but merely indicate the expected range. It is to be expected the acutal results from running the `IsoformSwitchAnalyzeR` pipeline differs from the `guestimate` in which case the `guestimate` should not be trusted.

Value

A `switchAnalyzeRlist` containing all the gene and transcript information as well as the isoform structure. See `?switchAnalyzeRlist` for more details. If `addCufflinksSwichTest=TRUE` a `data.frame` with the result of CuffDiffs test for alternative splicing is also added to the `switchAnalyzeRlist` under the entry `'isoformSwitchAnalysis'` (only if analysis was performed).

Note

Note that since there was an error in Cufflinks/Cuffdiff's estimation of standard errors that was not corrected until cufflinks 2.2.1. This function will give a warning if the cufflinks version used is older than this. Note that it will not be possible to test for differential isoform usage (isoform switches) with data from older versions of cufflinks (because the test among other uses the standard errors).

Author(s)

Kristoffer Vitting-Seerup

References

Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. *Mol. Cancer Res.* (2017).

See Also

[createSwitchAnalyzeRlist](#)
[preFilter](#)

Examples

```
## Not run:
### Please note
# The way of importing files in the following example with
# "system.file('pathToFile', package="cummeRbund") is
# specialized way of accessing the example data in the cummeRbund package
# and not something you need to do - just supply the string e.g.
# "myAnnotation/isoformsQuantified.gtf" to the functions.

### If you want to run this example code you need the cummeRbund package. It can be installed by running the code below
if (!requireNamespace("cummeRbund", quietly = TRUE)){
  BiocManager::install("cummeRbund")
}

### Use the files from the cummeRbund example data
aSwitchList <- importCufflinksFiles(
  pathToGTF = system.file('extdata/chr1_snippet.gtf', package = "cummeRbund"),
  pathToGeneDEanalysis = system.file('extdata/gene_exp.diff', package = "cummeRbund"),
  pathToIsoformDEanalysis = system.file('extdata/isoform_exp.diff', package = "cummeRbund"),
  pathToGeneFPKMtracking = system.file('extdata/genes.fpkm_tracking', package = "cummeRbund"),
  pathToIsoformFPKMtracking = system.file('extdata/isoforms.fpkm_tracking', package = "cummeRbund"),
  pathToIsoformReadGroupTracking = system.file('extdata/isoforms.read_group_tracking', package = "cummeRbund"),
  pathToSplicingAnalysis = system.file('extdata/splicing.diff', package = "cummeRbund"),
  pathToReadGroups = system.file('extdata/read_groups.info', package = "cummeRbund"),
  pathToRunInfo = system.file('extdata/run.info', package = "cummeRbund"),
  fixCufflinksAnnotationProblem=TRUE,
  quiet=TRUE
)

### Filter with very strict cutoffs to enable short runtime
aSwitchListAnalyzed <- preFilter(
  switchAnalyzeRlist = aSwitchList,
  isoformExpressionCutoff = 10,
  IFcutoff = 0.3,
  geneExpressionCutoff = 50
)

### Test isoform switches
aSwitchListAnalyzed <- isoformSwitchTestDEXSeq(
  aSwitchListAnalyzed
)

# extract summary of number of switching features
extractSwitchSummary(aSwitchListAnalyzed)

## End(Not run)
```

importGTF

*Import Transcripts from a GTF file into R***Description**

Function for importing a (gzipped or unpacked) GTF/GFF file into R as a `switchAnalyzerList`. This approach is well suited if you just want to annotate a transcriptome and are not interested in expression. If you are interested in expression estimates it is easier to use [importRdata](#).

Usage

```
importGTF(
  ### Core arguments
  pathToGTF,
  isoformNtFasta = NULL,

  ### Advanced arguments
  extractAaSeq = FALSE,
  addAnnotatedORFs=TRUE,
  onlyConsiderFullORF=FALSE,
  removeNonConventionalChr=FALSE,
  ignoreAfterBar = TRUE,
  ignoreAfterSpace = TRUE,
  ignoreAfterPeriod=FALSE,
  removeTECgenes = TRUE,
  PTCDistance=50,
  removeFusionTranscripts = TRUE,
  removeUnstrandedTranscripts = TRUE,
  quiet=FALSE
)
```

Arguments

<code>pathToGTF</code>	Can either be: <ul style="list-style-type: none"> • 1: A string indicating the full path to the (gzipped or unpacked) GTF file which have been quantified. If supplied the exon structure and isoform annotation will be obtained from the GTF file. An example could be "myAnnotation/myGenome/isoformsQuantified.gtf") • 2: A string indicating the full path to the (gzipped or unpacked) RefSeq GFF file which have been quantified. If supplied the exon structure and isoform annotation will be obtained from the GFF file. Please note only GFF files from RefSeq downloaded from ftp://ftp.ncbi.nlm.nih.gov/genomes/ are supported (see database FAQ in vignette for more info). An example could be "RefSeq/isoformsQuantified.gff")
<code>isoformNtFasta</code>	A (vector of) text string(s) providing the path(s) to the a fasta file containing the nucleotide sequence of all isoforms quantified. This is useful for: 1) people working with non-model organisms where extracting the sequence from a

BSgenome might require extra work. 2) workflow speed-up for people who already have the fasta file (which most people running Salmon, Kallisto or RSEM for the quantification have as that is used to build the index). The file will automatically be subsetted to the isoforms found in the gtf file so additional sequences (such as decoys) does not need to be manually removed. Please note this different from a fasta file with the sequences of the entire genome.

- `extractAaSeq` A logic indicating whether the nucleotide sequence imported via `isoformNtFasta` should be translated to amino acid sequence and stored in the `switchAnalyzeList`. Requires ORFs are imported, see `addAnnotatedORFs`. Default is true if a fasta file is supplied.
- `addAnnotatedORFs` A logic indicating whether the ORF from the GTF should be added to the `switchAnalyzeRlist`. This ORF is defined as the regions annotated as 'CDS' in the 'type' column (column 3). Default is TRUE.
- `onlyConsiderFullORF` A logic indicating whether the ORFs added should only be added if they are fully annotated. Here fully annotated is defined as those that both have a annotated 'start_codon' and 'stop_codon' in the 'type' column (column 3). This argument is only considered if `onlyConsiderFullORF=TRUE`. Default is FALSE.
- `removeNonConventionalChr` A logic indicating whether non-conventional chromosomes, here defined as chromosome names containing either a '_' or a period ('.'). These regions are typically used to annotate regions that cannot be associated to a specific region (such as the human 'chr1_gl000191_random') or regions quite different due to different haplotypes (e.g. the 'chr6_cox_hap2'). Default is FALSE.
- `ignoreAfterBar` A logic indicating whether to subset the isoform ids by ignoring everything after the first bar ("|"). Useful for analysis of GENCODE files. Default is TRUE.
- `ignoreAfterSpace` A logic indicating whether to subset the isoform ids by ignoring everything after the first space (" "). Useful for analysis of gffutils generated GTF files. Default is TRUE.
- `ignoreAfterPeriod` A logic indicating whether to subset the gene/isoform is by ignoring everything after the first period ("."). Should be used with care. Default is FALSE.
- `removeTECgenes` A logic indicating whether to remove genes marked as "To be Experimentally Confirmed" (if annotation is available). The default is TRUE aka to remove them which is in line with Gencode recommendations (TEC are not in Gencode annotations). For more info about TEC see <https://www.gencodegenes.org/pages/biotypes.html>.
- `PTCDistance` Only considered if `addAnnotatedORFs=TRUE`. A numeric giving the premature termination codon-distance: The minimum distance from the annotated STOP to the final exon-exon junction, for a transcript to be marked as NMD-sensitive. Default is 50
- `removeFusionTranscripts` A logic indicating whether to remove genes with cross-chromosome fusion transcripts as `IsoformSwitchAnalyzeR` cannot handle them.

removeUnstrandedTranscripts	A logic indicating whether to remove non-stranded isoforms as the Isoform-SwitchAnalyzeR workflow cannot handle them.
quiet	A logic indicating whether to avoid printing progress messages. Default is FALSE.

Details

The GTF file must have the following 3 annotation in column 9: 'transcript_id', 'gene_id', and 'gene_name'. Furthermore if addAnnotatedORFs is to be used the 'type' column (column 3) must contain the features marked as 'CDS'. If the onlyConsiderFullORF argument should work the GTF must also have 'start_codon' and 'stop_codon' annotated in the 'type' column (column 3).

Value

A switchAnalyzeRlist containing a all the gene and transcript information as well as the transcript models. See ?switchAnalyzeRlist for more details.

If addAnnotatedORFs=TRUE a data.frame containing the details of the ORF analysis have been added to the switchAnalyzeRlist under the name 'orfAnalysis'.

The data.frame added have one row pr isoform and contains 11 columns:

- isoform_id: The name of the isoform analyzed. Matches the 'isoform_id' entry in the 'isoformFeatures' entry of the switchAnalyzeRlist
- orfTranscriptStart: The start position of the ORF in transcript Coordinates, here defined as the position of the 'A' in the 'AUG' start motif.
- orfTranscriptEnd: The end position of the ORF in transcript coordinates, here defined as the last nucleotide before the STOP codon (meaning the stop codon is not included in these coordinates).
- orfTranscriptLength: The length of the ORF
- orfStarExon: The exon in which the start codon is
- orfEndExon: The exon in which the stop codon is
- orfStartGenomic: The start position of the ORF in genomic coordinates, here defined as the the position of the 'A' in the 'AUG' start motif.
- orfEndGenomic: The end position of the ORF in genomic coordinates, here defined as the last nucleotide before the STOP codon (meaning the stop codon is not included in these coordinates).
- stopDistanceToLastJunction: Distance from stop codon to the last exon-exon junction
- stopIndex: The index, counting from the last exon (which is 0), of which exon is the stop codon is in.
- PTC: A logic indicating whether the isoform is classified as having a Premature Termination Codon. This is defined as having a stop codon more than PTCDistance (default is 50) nt upstream of the last exon exon junction.

NA means no information was available aka no ORF (passing the minORFlength filter) was found.

Author(s)

Kristoffer Vitting-Seerup

References

Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. Mol. Cancer Res. (2017).

See Also

[createSwitchAnalyzeRlist](#)
[preFilter](#)

Examples

```
# Note the way of importing files in the following example with
# "system.file('pathToFile', package="IsoformSwitchAnalyzeR") is
# specialized way of accessing the example data in the IsoformSwitchAnalyzeR package
# and not something you need to do - just supply the string e.g.
# "myAnnotation/isoformsQuantified.gtf" to the functions

aSwitchList <- importGTF(pathToGTF=system.file("extdata/example.gtf.gz", package="IsoformSwitchAnalyzeR"))
aSwitchList
```

```
importIsoformExpression
```

Import expression data from Kallisto, Salmon, RSEM or StringTie into R.

Description

A general-purpose import function which imports isoform expression data from Kallisto, Salmon, RSEM or StringTie into R. This is a wrapper for the tximport package with some extra functionalities and is meant to be used to import the data and afterwards a switchAnalyzeRlist can be created with importRdata. It is highly recommended that both the imported TxPM and counts values are used both in the creation of the switchAnalyzeRlist with importRdata (through the "isoformCountMatrix" and "isoformRepExpression" arguments). Importantly this import function also enables (and per default performs) inter-library normalization (via edgeR) of the abundance estimates. Note that the pattern argument allows import of only a subset of files. Can be used together with isoformToGeneExp() to get gene expression.

Usage

```
importIsoformExpression(
  ### Core arguments
  parentDir = NULL,
  sampleVector = NULL,
```

```

### Advanced arguments
calculateCountsFromAbundance=TRUE,
addIsoformIdAsColumn=TRUE,
interLibNormTxPM=TRUE,
normalizationMethod='TMM',
pattern='',
invertPattern=FALSE,
ignore.case=FALSE,
ignoreAfterBar = TRUE,
ignoreAfterSpace = TRUE,
ignoreAfterPeriod = FALSE,
readLength = NULL,
showProgress = TRUE,
quiet = FALSE
)

```

Arguments

parentDir	Parent directory where each quantified sample is in a sub-directory. The function will then look for files containing the (suffix) of the default files names for the quantification tools. The suffixes identified are 'abundance.tsv' for Kallisto, 'quant.sf' for Salmon, 'isoforms.results' for RSEM and '_t_data.ctab' for StringTie. This is an alternative to sampleVector (aka only one of them should be used).
sampleVector	A vector with the path to each quantification file to import. If the vector has names assigned (via the names function) these names will be used as the column name of the resulting tables. Else This is an alternative to parentDir (aka only one of them should be used). See example.
calculateCountsFromAbundance	A logic indicating whether to generate estimated counts using the estimated abundances. Recommended as it will incorporate the bias correction algorithms into the analysis. Default is TRUE.
addIsoformIdAsColumn	A logic indicating whether to add isoform id as a separate column (necessary for use with isoformSwitchAnalyzeR) or not (resulting in a data.frame ready for many other functions for exploratory data analysis (EDA) or clustering). Default is TRUE.
interLibNormTxPM	A logic indicating whether to apply an inter-library normalization (via edgeR) to the imported abundances. Recommended as it allow better comparison of abundances between samples. Will not affect the returned counts - even if calculateCountsFromAbundance=TRUE. Default is TRUE.
normalizationMethod	A string indicating the method used for the inter-library normalization. Must be one of "TMM", "RLE", "upperquartile". See ?edgeR::calcNormFactors for more details. Default is "TMM".
pattern	Only used in combination with parentDir. A character string containing a regular expression for which files to import (applied to full path). Default is "" corresponding to all. See base::grepl for more details.

<code>invertPattern</code>	Only used in combination with <code>parentDir</code> . A Logical. If TRUE only use files which do not match the <code>pattern</code> argument.
<code>ignore.case</code>	Only used in combination with <code>parentDir</code> . A logical. If TRUE case is ignored during matching with the <code>pattern</code> argument. If FALSE the matching with the <code>pattern</code> argument is case sensitive.
<code>ignoreAfterBar</code>	A logic indicating whether to subset the isoform ids by ignoring everything after the first bar (" "). Useful for analysis of GENCODE data. Default is TRUE.
<code>ignoreAfterSpace</code>	A logic indicating whether to subset the isoform ids by ignoring everything after the first space (" "). Useful for analysis of gffutils generated GTF files. Default is TRUE.
<code>ignoreAfterPeriod</code>	A logic indicating whether to subset the gene/isoform is by ignoring everything after the first period ("."). Should be used with care. Default is FALSE.
<code>readLength</code>	Only necessary when importing from StringTie. Must be the number of base pairs sequenced. e.g. if the data quantified is 75 bp paired ends the user should supply <code>readLength=75</code> .
<code>showProgress</code>	A logic indicating whether to make a progress bar (if TRUE) or not (if FALSE). Default is FALSE.
<code>quiet</code>	A logic indicating whether to avoid printing progress messages (incl. progress bar). Default is FALSE

Details

This function requires all data that should be imported is in a directory (as indicated by `parentDir`) where each quantified sample is in a separate sub-directory.

The actual import of data is done with `tximport` using `"countsFromAbundance='scaledTPM'"` to extract counts.

For Kallisto the bias estimation is enabled by adding `'-bias'` to the function call. For Salmon the bias estimation is enabled by adding `'-seqBias'` and `'-gcBias'` to the function call. For RSEM the bias estimation is enabled by adding `'-estimate-rspd'` to the function call. For StringTie the bias corrections are always enabled (and cannot be turned off by the user).

Inter library normalization is (almost always) necessary due to small changes in the RNA composition between cells and is highly recommended for all analysis of RNAseq data. For more information please refer to the edgeR user guide.

The inter-library normalization of FPKM/TxPM values is performed as a 3/4 step process: If `calculateCountsFromAbundance=TRUE` the effective counts are calculated from the abundances using the library specific effective isoform lengths, else the original counts are used. The count matrix is then subsetted to the isoforms expressed more than 1 TxPM/RPKM in more than one sample. The count matrix supplied to edgeR which calculates the normalization factors necessary. Lastly the calculated normalization factors are applied to the imported FPKM/TxPM values.

This function expects the files produced by Kallisto/Salmon/RSEM/StringTie to be called their default names (with possible custom prefix): Kallisto files are called `'abundance.tsv'`, Salmon files are called `'quant.sf'`, RSEM files are called `'isoforms.results'` and StringTie files are called `'t_data.ctab'`.

Importantly StringTie must be run with the -B option to produce the quantified file: An example could be: "StringTie -eB -G transcripts.gtf <source_file.bam>"

Value

A list containing an abundance matrix, a count matrix and a matrix with the effective lengths for each isoform quantified (rows) in each sample (col) where the first column contains the isoform_ids. The options used for import are stored under the "importOptions" entry). The abundance estimates are in the unit of Transcripts Per Million (TPM) and measuring the relative abundance of a specific transcript.

Transcripts Per Million values are abbreviated to TPM by RSEM, Kallisto and Salmon but will here referred to as TxPM to avoid confusion with the commonly used Tags Per Million (which have been around for way longer). TxPM is an equivalent to RPKM/FPKM except it has been adjusted for as all the biases being modeled by the tools used for the quantification including the fragment length distribution and sequence-specific bias as well as GC-fragment bias (this is specific to each tool and how it was run so you need to look up the specific tool). The TxPM is optimal for expression comparison of abundances since most biases will be taking into account.

Author(s)

Kristoffer Vitting-Seerup

References

Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. Mol. Cancer Res. (2017). Soneson et al. Differential analyses for RNA-seq: transcript-level estimates improve gene-level inferences. F1000Research 4, 1521 (2015). Robinson et al. A scaling normalization method for differential expression analysis of RNA-seq data. Genome Biology (2010)

See Also

[importRdata](#)
[createSwitchAnalyzeRlist](#)
[preFilter](#)

Examples

```
### Please note
# The way of importing files in the following example with
# "system.file('pathToFile', package="IsoformSwitchAnalyzeR") is
# specialized way of accessing the example data in the IsoformSwitchAnalyzeR package
# and not something you need to do - just supply the string e.g.
# parentDir = "/path/to/mySalmonQuantifications/" or
# sampleVector = c('mySalmonQuantifications/file1.sf', 'mySalmonQuantifications/file2.sf') to the function

### Import all quantifications stored in a folder
salmonQuant <- importIsoformExpression(
  parentDir = system.file("extdata/", package="IsoformSwitchAnalyzeR")
)
```

```

names(salmonQuant)
head(salmonQuant$abundance, 2)

### Import individual quantification files
myFiles <- c(
  system.file("extdata/Fibroblasts_0/quant.sf.gz", package="IsoformSwitchAnalyzeR"),
  system.file("extdata/Fibroblasts_1/quant.sf.gz", package="IsoformSwitchAnalyzeR")
)
names(myFiles) <- c('Fibroblasts_0', 'Fibroblasts_1')

salmonQuant <- importIsoformExpression(
  sampleVector = myFiles
)

names(salmonQuant)
head(salmonQuant$abundance, 2)

### Get gene expression/count from isoform expression/count
geneRepCount <- isoformToGeneExp(
  isoformRepExpression = salmonQuant$counts, # just change to "salmonQuant$abundance" to get gene abundances
  isoformGeneAnnotation = system.file("extdata/example.gtf.gz", package="IsoformSwitchAnalyzeR")
)

head(geneRepCount, 2)

```

importRdata

Create SwitchAnalyzeRlist From Standard R Objects

Description

A general-purpose interface to constructing a switchAnalyzeRlist. The data needed for this function are

- 1: An isoform count expression matrix (necessary). See [importIsoformExpression](#) for an easy way to import Salmon/Kallisto/RSEM or StringTie expression
- 2: Optional. Normalized biological replicate isoform abundances. See [importIsoformExpression](#) for an easy way to import Salmon/Kallisto/RSEM or StringTie expression
- 3: Isoform annotation (both genomic exon coordinates and which gene the isoform belongs to). This can also be supplied as the path to a GTF file from which the data is then extracted.
- 4: A design matrix indicating which samples belong to which condition

Please note that

- 1 It is possible to specify which comparisons to make using the comparisonsToMake (default is all possible pairwise of the once indicated by the design matrix).
- 2 The importRdata() also includes an extended algorithm to correct some of the annotation problems frequently occurring when doing transcript assembly via tools such as StringTie/Cufflinks (gene merging and unassigned novel isoforms). These can be controlled via the fixStringTie* arguments.

Usage

```

importRdata(
  ### Core arguments
  isoformCountMatrix,
  isoformRepExpression = NULL,
  designMatrix,
  isoformExonAnnoation,
  isoformNtFasta = NULL,
  comparisonsToMake = NULL,

  ### Advanced arguments
  addAnnotatedORFs = TRUE,
  onlyConsiderFullORF = FALSE,
  removeNonConvensionalChr = FALSE,
  ignoreAfterBar = TRUE,
  ignoreAfterSpace = TRUE,
  ignoreAfterPeriod = FALSE,
  removeTECgenes = TRUE,
  PTCDistance = 50,
  foldChangePseudoCount = 0.01,
  addIFmatrix = TRUE,
  fixStringTieAnnotationProblem = TRUE,
  fixStringTieViaOverlapInMultiGenes = TRUE,
  fixStringTieMinOverlapSize = 50,
  fixStringTieMinOverlapFrac = 0.2,
  fixStringTieMinOverlapLog2RatioToContender = 0.65,
  estimateDifferentialGeneRange = TRUE,
  showProgress = TRUE,
  quiet = FALSE
)

```

Arguments**isoformCountMatrix**

A data.frame with unfiltered independent biological (aka not technical) replicate isoform (estimated) fragment counts (see FAQ in vignette for more details) with genes as rows and samples as columns. Must have a column called 'isoform_id' with the isoform_id that matches the isoform_id column in isoformExonAnnoation. The name of the columns must match the sample names in the designMatrix argument and contain the estimated counts.

isoformRepExpression

Optional but highly recommended: A data.frame with unfiltered normalized independent biological (aka not technical) replicate isoform expression (see FAQ in vignette for more details). Ideal for supplying quantification measured in Transcripts Per Million (TxPM) or RPKM/FPKM. Must have a column called 'isoform_id' that matches the isoform_id column in isoformExonAnnoation. The name of the expression columns must match the sample names in the designMatrix argument. If not supplied RPKM values are calculated from the count matrix

and used instead.

`designMatrix` A `data.frame` with the information of which samples originate from which conditions. Must be a `data.frame` containing at least these two columns:

- Column 1: called 'sampleID'. This column contains the sample names and must match the column names used in `isoformRepExpression`.
- Column 2: called 'condition'. This column indicates with a string which conditions the sample originate from. If sample 1-3 originate from the same condition they should all have the same string (for example 'ctrl', in this column).

Additional columns can be used to describe other co-factors such as batch effects or patient ids (for paired sample analysis). For more information see discussion of cofactors in vignette.

`isoformExonAnnotation`

Can either be:

- 1: A string indicating the full path to the (gzipped or unpacked) GTF file with the annotation of the isoforms quantified. If you are using a reference-only workflow (tools such as Kallisto/Salmon/RSEM etc) this argument should point to the reference database GTF corresponding to the fasta file that you used to build the reference index. If you use a guided/de-novo transcriptome assembly approach (tools like StringTie and Cufflinks) this argument should point to the GTF file created at the "merge" stage of the workflow. Please refer to the "What Quantification Tool(s) Should I Use" section of the vignette for a more detailed description of the two different workflows. An example could be "myAnnotation/myGenome/isoformsQuantified.gtf")
- 2: A string indicating the full path to the (gzipped or unpacked) RefSeq GFF file which have been quantified. If supplied the exon structure and isoform annotation will be obtained from the GFF file. Please note only GFF files from RefSeq downloaded from <ftp://ftp.ncbi.nlm.nih.gov/genomes/> are supported (see database FAQ in vignette for more info). An example could be "RefSeq/isoformsQuantified.gff")
- 3: A `GRange` object (see `?GRanges`) containing one entry per exon per isoform with the genomic coordinates of that exon. This `GRange` should furthermore contain two meta data columns called 'isoform_id' and 'gene_id' indicating both which isoform the exon belongs to as well as which gene the isoform belongs to. The 'isoform_id' column must match the isoform ids used in the 'isoform_id' column of the `isoformRepExpression` `data.frame`. If possible we suggest that a third column called 'gene_name' with the corresponding gene names/symbols is also added. If not supplied `gene_name` will be annotated as missing (NA).

`isoformNtFasta` A (vector of) text string(s) providing the path(s) to the a fasta file containing the nucleotide sequence of all isoforms quantified. This is useful for: 1) people working with non-model organisms where extracting the sequence from a `BSgenome` might require extra work. 2) workflow speed-up for people who already have the fasta file (which most people running Salmon, Kallisto or RSEM for the quantification have as that is used to build the index). The file(s) will automatically be subsetted to the isoforms found in the expression matrix so addi-

tional sequences (such as decoys) does not need to be manually removed. Please note this different from a fasta file with the sequences of the entire genome.

comparisonsToMake

A data.frame with two columns indicating which pairwise comparisons the switchAnalyzeRlist created should contain. The two columns, called 'condition_1' and 'condition_2' indicate which conditions should be compared and the strings indicated here must match the strings in the designMatrix\$condition column. If not supplied all pairwise (unique non directional) comparisons of the conditions given in designMatrix\$condition are created. If only a subset of the supplied data is used in the comparisons the Un-used data is automatically removed.

addAnnotatedORFs

Only used if a GTF file is supplied to isoformExonAnnotation. A logic indicating whether the ORF from the GTF should be added to the switchAnalyzeRlist. This ORF is defined as the regions annotated as 'CDS' in the 'type' column (column 3). Default is TRUE.

onlyConsiderFullORF

A logic indicating whether the ORFs added should only be added if they are fully annotated. Here fully annotated is defined as those that both have a annotated 'start_codon' codon in the 'type' column (column 3). This argument exists because these CDS regions are highly problematic and does not resemble true ORFs as >50% of CDS without a stop_codon annotated contain multiple stop codons (see Vitting-Seerup et al 2017 - supplementary materials). This argument is only considered if addAnnotatedORFs=TRUE. Default is FALSE.

removeNonConventionalChr

A logic indicating whether non-conventional chromosomes, here defined as chromosome names containing either a '_' or a period ('.'). These regions are typically used to annotate regions that cannot be associated to a specific region (such as the human 'chr1_gl000191_random') or regions quite different due to different haplotypes (e.g. the 'chr6_cox_hap2'). Default is FALSE.

ignoreAfterBar

A logic indicating whether to subset the isoform ids by ignoring everything after the first bar ("|"). Useful for analysis of GENCODE data. Default is TRUE.

ignoreAfterSpace

A logic indicating whether to subset the isoform ids by ignoring everything after the first space (" "). Useful for analysis of gffutils generated GTF files. Default is TRUE.

ignoreAfterPeriod

A logic indicating whether to subset the gene/isoform is by ignoring everything after the first period ("."). Should be used with care. Default is FALSE.

removeTECgenes

A logic indicating whether to remove genes marked as "To be Experimentally Confirmed" (if annotation is available). The default is TRUE aka to remove them which is in line with Gencode recommendations (TEC are not in Gencode annotations). For more info about TEC see <https://www.gencodegenes.org/pages/biotypes.html>.

PTCDistance

Only used if a GTF file is supplied to isoformExonAnnotation and addAnnotatedORFs=TRUE. A numeric giving the premature termination codon-distance: The minimum distance from the annotated STOP to the final exon-exon junction, for a transcript to be marked as NMD-sensitive. Default is 50

- `foldChangePseudoCount`
A numeric indicating the pseudocount added to each of the average expression values before the log₂ fold change is calculated. Done to prevent log₂ fold changes of Inf or -Inf. Default is 0.01
- `addIFmatrix`
A logic indicating whether to add the Isoform Fraction replicate matrix (if TRUE) or not (if FALSE). Keeping it will make testing with `isoformSwitchTestDEXSeq` faster but will also make the `switchAnalyzeRlist` larger - so it is a trade off for speed vs memory. For most experimental setups we expect that keeping it will be the better solution. Default is TRUE.
- `fixStringTieAnnotationProblem`
A logic indicating whether to try and fix the following two annoation problems.
- 1: Fix the problem where novel isoforms are not assigned to a reference gene. This is done by assigning the `gene_name` of the parrent `gene_id`, but only when the `gene_id` is associated with a single `gene_name`.
 - 2: Fix the problem where multiple genes (as indicated by reference `gene_ids`) are merged into a single `gene_id`. This can only be done when all isoforms are assigned a `gene_name`. The `gene_id` is simply split into multiple `gene_ids` via the `gene_names`. Genes with this problem, which could not be fixed, are removed since such constallations might result in untrustworthy switches (where the isoforms are acutally from different genes).
 - 3: Fix the problem where all genes containing novel isoforms are assigned a `StringTie gene_id` instead of their original id. This is done by assigning the `gene_name` of the parrent `gene_id`, but only when the `gene_id` is associated with a single `gene_name`.
- Default is TRUE.
- `fixStringTieViaOverlapInMultiGenes`
A logic indicating whether the `IsoformSwitchAnalyzeR` should also try to assign `gene_names` to novel isoforms within `gene_ids` with multiple `gene_names` associated. This is done by comparing the genomic overlap of exons in novel isoforms to exons in known isoforms. This is only done if all of the `fixStringTieViaOverlap*` cutoffs are met and `fixStringTieAnnotationProblem=TRUE`. Default is TRUE.
- `fixStringTieMinOverlapSize`
The minimum number of nucleotide a novel isoform must overlap with a known isoform for `gene_name` transfer. This argument modulates the process described in the `fixStringTieViaOverlapInMultiGenes` argument. Default is 50.
- `fixStringTieMinOverlapFrac`
The minimum fraction of a novel isoform must overlap with a known isoform for `gene_name` transfer. This argument modulates the process described in the `fixStringTieViaOverlapInMultiGenes` argument. Default is 0.2.
- `fixStringTieMinOverlapLog2RatioToContender`
A log₂ ratio which describes how much larger the overlap between a novel isoform and a known isoform must be for `gene_name` transfer in cases where overlap with known isoforms from multiple `gene_names` occure. This is the most important argument of deciding what to do with isoform overlapping multiple genes. If increased only more certain cases are assigned at the cost of more isoforms not being assigned. If decrease more isoforms are assigned but the certainty is lower. The default is 0.65 (corresponding to approx 1.57 fold) which according to our test data is the best a balance between strict and lenient.

<code>estimateDifferentialGeneRange</code>	A logic indicating whether to make a very quick estimate of the number of genes with differential isoform usage. Please note this number should be taken as a pilot and cannot be trusted. It merely servers to indcate what could be expected if the data is analyzed with the rest of the IsoformSwitchAnalyzeR. See details for more information. Default is TRUE.
<code>showProgress</code>	A logic indicating whether to make a progress bar (if TRUE) or not (if FALSE). Default is FALSE.
<code>quiet</code>	A logic indicating whether to avoid printing progress messages (incl. progress bar). Default is FALSE

Details

For each gene in each replicate sample the expression of all isoforms belonging to that gene (as annotated in `isoformExonAnnoation`) are summed to get the gene expression. It is therefore very important that the `isoformRepExpression` is unfiltered. For each gene/isoform in each condition (as indicate by `designMatrix`) the mean and standard error (of mean (measurement), `s.e.m`) are calculated. Since all samples are considered it is very important the `isoformRepExpression` does not contain technical replicates. The comparison indicated `comparisonsToMake` (or all pairwise if not supplied) is then constructed and the mean gene and isoform expression values are then used to calculate log2 fold changes (using `foldChangePseudoCount`) and Isoform Fraction (IF) values. The whole analysis is then wrapped in a `SwitchAnalyzeRlist`.

Changes in isoform usage are measure as the difference in isoform fraction (dIF) values, where isoform fraction (IF) values are calculated as $\langle \text{isoform_exp} \rangle / \langle \text{gene_exp} \rangle$.

The gestimate produced by setting `estimateDifferentialGeneRange = TRUE` is created by subsetting a lot on data (both on samples, conditions and genes) and running a fast but unreliable DTU method. The resulting number is then multiplied by a factor to caclulate back what would be expected by running the IsoformSwitchAnalyzeR pipeline. It should go without saying due to all these factors the acutal gestimate is just that - and estimate which cannot be trusted but merely indicate the expected range. It is to be expected the acutal results from running the IsoformSwitchAnalyzeR pipeline differs from the gestimate in which case the gestimate should not be trusted.

Value

A `SwitchAnalyzeRlist` containing the data supplied stored into the `SwitchAnalyzeRlist` format (created by `createSwitchAnalyzeRlist()`). For details about the format see details of [createSwitchAnalyzeRlist](#).

If a GTF file was supplied to `isoformExonAnnoation` and `addAnnotatedORFs=TRUE` a `data.frame` containing the details of the ORF analysis have been added to the `switchAnalyzeRlist` under the name `'orfAnalysis'`. The `data.frame` added have one row pr isoform and contains 11 columns:

- `isoform_id`: The name of the isoform analyzed. Matches the `'isoform_id'` entry in the `'isoformFeatures'` entry of the `switchAnalyzeRlist`
- `orfTranscriptStart`: The start position of the ORF in transcript Coordinates, here defined as the position of the 'A' in the 'AUG' start motif.
- `orfTranscriptEnd`: The end position of the ORF in transcript coordinates, here defined as the last nucleotide before the STOP codon (meaning the stop codon is not included in these coordinates).

- orfTranscriptLength: The length of the ORF
- orfStarExon: The exon in which the start codon is
- orfEndExon: The exon in which the stop codon is
- orfStartGenomic: The start position of the ORF in genomic coordinators, here defined as the the position of the 'A' in the 'AUG' start motif.
- orfEndGenomic: The end position of the ORF in genomic coordinates, here defined as the last nucleotide before the STOP codon (meaning the stop codon is not included in these coordinates).
- stopDistanceToLastJunction: Distance from stop codon to the last exon-exon junction
- stopIndex: The index, counting from the last exon (which is 0), of which exon is the stop codon is in.
- PTC: A logic indicating whether the isoform is classified as having a Premature Termination Codon. This is defined as having a stop codon more than PTCDistance (default is 50) nt upstream of the last exon exon junction.

NA means no information was available aka no ORF (passing the minORFlength filter) was found.

Author(s)

Kristoffer Vitting-Seerup

References

Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. Mol. Cancer Res. (2017).

See Also

[createSwitchAnalyzeRlist](#)
[importIsoformExpression](#)
[preFilter](#)

Examples

```
### Please note
# 1) The way of importing files in the following example with
#     "system.file('pathToFile', package="IsoformSwitchAnalyzeR") is
#     specialized way of accessing the example data in the IsoformSwitchAnalyzeR package
#     and not something you need to do - just supply the string e.g.
#     isoformExonAnnoation = "myAnnotation/isoformsQuantified.gtf" to the functions
# 2) importRdata directly supports import of a GTF file - just supply the
#     path (e.g. "myAnnotation/isoformsQuantified.gtf") to the isoformExonAnnoation argument

### Import quantifications
salmonQuant <- importIsoformExpression(system.file("extdata/", package="IsoformSwitchAnalyzeR"))

### Make design matrix
myDesign <- data.frame(
  sampleID = colnames(salmonQuant$abundance)[-1],
```

```

    condition = gsub('_', '.', colnames(salmonQuant$abundance)[-1])
  )

  ### Create switchAnalyzeRlist
  aSwitchList <- importRdata(
    isoformCountMatrix = salmonQuant$counts,
    isoformRepExpression = salmonQuant$abundance,
    designMatrix = myDesign,
    isoformExonAnnoation = system.file("extdata/example.gtf.gz", package="IsoformSwitchAnalyzeR"),
    isoformNtFasta = system.file("extdata/example_isoform_nt.fasta.gz", package="IsoformSwitchAnalyzeR")
  )
  aSwitchList

```

importSalmonData	<i>Direct creation of a switchAnalyzeRlist from Salmon quantification</i>
------------------	---

Description

This function uses tximeta (see Love et al 2020) to import Salmon data into R. The nice thing about using tximeta is that it recognizes which transcriptome was quantified (if quantified with Salmon >= 1.1.0). If the quantified transcriptome is one of the main model organisms (Ensembl, Gencode and RefSeq for human/mouse (and Drosophila)) tximeta can automatically import the associated annoation (GTF and Fasta file) making the creation of the switchAnalyzeRlist easier. For a full list of supported transcriptomes please refer to https://bioconductor.org/packages/devel/bioc/vignettes/tximeta/inst/doc/tximeta.html#Pre-computed_checksums. If importSalmonData() does not work you can always use importIsoformExpression followed by importRdata.

Usage

```

importSalmonData(
  ### Core arguments
  salmonFileDataFrame,

  ### Advanced arguments
  comparisonsToMake=NULL,
  ignoreAfterBar = TRUE,
  ignoreAfterSpace = TRUE,
  ignoreAfterPeriod = FALSE,
  showProgress = TRUE,
  quiet = FALSE,
  ...
)

```

Arguments

salmonFileDataFrame

The data.frame created by the [prepareSalmonFileDataFrame](#) function. Alternatively it can be created manually and must be a data.frame with 3 (or more columns).

- Column 1: "files". Contains the file path to each of the quant.sf files which should be imported
- Column 2: "names". Contains the name to be used to refer to each of the quant.sf files mentioned in the "files" column.
- Column 3: "condition". Contains an indication of which samples belong to each condition. E.g if sample 1-3 originate from the same condition they should all have the same string (for example 'ctrl', in this column).

Additional columns can be used to describe other co-factors not of interesting such as batch effects or patient ids (for paired sample analysis). For more information see discussion of cofactors in vignette.

comparisonsToMake

A data.frame with two columns indicating which pairwise comparisons the switchAnalyzeRlist created should contain. The two columns, called 'condition_1' and 'condition_2' indicate which conditions should be compared and the strings indicated here must match the strings in the designMatrix\$condition column. If not supplied all pairwise (unique non directional) comparisons of the conditions given in salmonFileDataFrame\$condition are created. If only a subset of the supplied data is used in the comparisons the Un-used data is automatically removed.

ignoreAfterBar A logic indicating whether to subset the isoform ids by ignoring everything after the first bar ("|"). Useful for analysis of GENCODE data. Default is TRUE.

ignoreAfterSpace

A logic indicating whether to subset the isoform ids by ignoring everything after the first space (" "). Useful for analysis of gffutils generated GTF files. Default is TRUE.

ignoreAfterPeriod

A logic indicating whether to subset the gene/isoform is by ignoring everything after the first period ("."). Should be used with care. Default is FALSE.

showProgress A logic indicating whether to make a progress bar (if TRUE) or not (if FALSE). Default is FALSE.

quiet A logic indicating whether to avoid printing progress messages (incl. progress bar). Default is FALSE

... Other argument passed to [importRdata](#)

Details

The Tximeta R package (Love et al 2020) is used to import Salmon and associated annotation data (isoform exon structure, nucleotide sequence and coding region) into R. These are then passed to [importRdata](#) to generate the switchAnalyzeRlist object.

Value

A SwitchAnalyzeRlist containing the data supplied stored into the SwitchAnalyzeRlist format (created by [createSwitchAnalyzeRlist\(\)](#)). For details about the format see details of [createSwitchAnalyzeRlist](#).

Author(s)

Kristoffer Vitting-Seerup

References

- IsoformSwitchAnalyzerR: Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. Mol. Cancer Res. (2017)..
- Tximeta: Love et al. Tximeta: Reference sequence checksums for provenance identification in RNA-seq. PLoS Comput. Biol. (2020).

See Also

```
prepareSalmonFileDataFrame
createSwitchAnalyzeRlist
importIsoformExpression
importRdata
preFilter
```

Examples

```
### Please note
# The way of importing files in the following example with
# "system.file('pathToFile', package="IsoformSwitchAnalyzerR") is
# specialized way of accessing the example data in the IsoformSwitchAnalyzerR package
# and not something you need to do - just supply the string e.g.
# parentDir = "individual_quantifications_in_subdir/" to the functions
# path (e.g. "myAnnotation/isoformsQuantified.gtf") to the isoformExonAnnotation argument

### Prepare data.frame with quant file info
salmonDf <- prepareSalmonFileDataFrame(
  system.file("extdata/drosophila", package="IsoformSwitchAnalyzerR")
)

### Add conditions
salmonDf$condition <- c('wt','wt','ko','ko')

### Create switchAnalyzeRlist
aSwitchList <- importSalmonData(salmonDf)
```

isoformSwitchAnalysisCombined

Isoform Switch Analysis Workflow: Extract, Annotate and Visualize all Significant Isoform Switches

Description

This high-level function uses a pre-existing switchAnalyzeRlist as input. Then isoform switches are identified, annotated with ORF and intron retention. Then functional consequences are identified and isoform switch analysis plots are generated for the top n isoform switches. Lastly a plot summarizing the global effect of isoform switches with functional consequences is generated. If external analysis of protein domains (Pfam), coding potential (CPAT) or signal peptides (SignalP) should be incorporated please use the combination of isoformSwitchAnalysisPart1 and isoformSwitchAnalysisPart2 instead.

Usage

```

isoformSwitchAnalysisCombined(
  ### Core arguments
  switchAnalyzeRlist,

  ### Annotation arguments
  genomeObject = NULL,
  pathToGTF = NULL,

  ### Analysis and output arguments
  n = Inf,
  consequencesToAnalyze = c('intron_retention', 'ORF_seq_similarity', 'NMD_status'),
  pathToOutput = getwd(),
  fileType = 'pdf',
  outputPlots = TRUE,

  ### Other arguments
  quiet = FALSE
)

```

Arguments

switchAnalyzeRlist	A switchAnalyzeRlist.
genomeObject	A BSgenome object (for example Hsapiens for Homo sapiens).
pathToGTF	A string indicating the full path to the (gzipped or unpacked) GTF file which contains the the known annotation (aka from a official source) which was used to guided the transcript assembly (isoform deconvolution).
n	The number of top genes (after filtering and sorted according to sortByQvals) that should be saved to each sub-folder indicated by splitConditions, splitFunctionalConsequences. Use Inf to create all. Default is Inf (all).
consequencesToAnalyze	A vector of strings indicating what type of functional consequences to analyze. Do note that there is bound to be some differences between transcripts (else there would be identical). See details in analyzeSwitchConsequences for full list of usable strings and their meaning. Default is c('intron_retention', 'coding_potential', 'ORF_seq_similarity', 'CPAT_result', 'NMD_status', 'PFAM_domains_annotated', 'signal_peptides_annotated').
pathToOutput	A path to the folder in which the plots should be made. Default is working directory (getwd()).
fileType	A string indicating which file type is generated. Available options are 'pdf' and 'png'. Default is pdf.
outputPlots	A logic indicating whether all isoform switches as well as the summary of functional consequences should be saved in the directory specified by pathToOutput argument. Default is TRUE.

`quiet` A logic indicating whether to avoid printing progress messages (incl. progress bar). Default is FALSE

Details

This function performs the full Isoform Analysis Workflow by

1. Remove non-expressed isoforms and single-isoform genes (see [preFilter](#))
2. predict switches (only if switches is not already annotated, see [isoformSwitchTestDEXSeq](#))
3. Analyzing the isoforms for open reading frames (ORFs, see [analyzeORF](#))
4. Output fasta files containing the nucleotide and amino acid sequences which enables external sequence analysis with CPAT, Pfam and SignalP (see [extractSequence](#))
5. Predict functional consequences of switching (see [analyzeSwitchConsequences](#))
6. Output Isoform Switch Analysis plots for all genes with a significant switch (see [switchPlot](#))
7. Output a visualization of general consequences of isoform switches.

Value

This function outputs:

1. The supplied `switchAnalyzeRlist` now annotated with all the analysis described above
2. One folder per comparison of condition containing the isoform switch analysis plot of all significant isoforms.
3. A plot summarizing the overall consequences of all the isoform switches.

Author(s)

Kristoffer Vitting-Seerup

References

Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. Mol. Cancer Res. (2017).

See Also

[isoformSwitchAnalysisPart1](#)
[isoformSwitchAnalysisPart2](#)
[preFilter](#)
[isoformSwitchTestDEXSeq](#)
[isoformSwitchTestDRIMSeq](#)
[analyzeORF](#)
[extractSwitchSummary](#)
[analyzeSwitchConsequences](#)
[switchPlotTopSwitches](#)

Examples

```

data("exampleSwitchList")
exampleSwitchList

library(BSgenome.Hsapiens.UCSC.hg19)
exampleSwitchListAnalyzed <- isoformSwitchAnalysisCombined(
  switchAnalyzeRlist=exampleSwitchList,
  outputPlots = FALSE    # keeps the function from outputting the Isoform Switch AnalyzeR Plots from this example
)

exampleSwitchListAnalyzed

```

```
isoformSwitchAnalysisPart1
```

*Isoform Switch Analysis Workflow Part 1: Extract Isoform Switches
and Their Bio-sequences*

Description

This high-level function takes a pre-existing switchAnalyzeRlist as input (see importRdata). Then part 1 of the workflow is performed. Specifically it is filtered to remove low expression, isoform switches are identified via the statistical methods (unless switchTestMethod='none') and ORF are predicted if not already annotated. Lastly the function extracts the nucleotide sequence and the ORF AA sequences of the isoforms involved in isoform switches. To enable external and internal sequence analysis these sequences are both saved to the computer (as fasta files) and added to the switchAnalyzeRlist.

This function is meant to be used as part 1 of the isoform switch analysis workflow, which can be followed by the second step via isoformSwitchAnalysisPart2.

Usage

```

isoformSwitchAnalysisPart1(
  ### Core arguments
  switchAnalyzeRlist,

  ### Annotation arguments
  genomeObject = NULL,
  pathToGTF = NULL,

  ### Output arguments
  prepareForWebServers,
  pathToOutput = getwd(),
  outputSequences = TRUE,

  ### Other arguments
  quiet = FALSE
)

```

Arguments

switchAnalyzeRlist	A switchAnalyzeRlist.
genomeObject	A BSgenome object (for example Hsapiens for Homo sapiens).
pathToGTF	A string indicating the full path to the (gzipped or unpacked) GTF file which contains the the known annotation (aka from a official source) which was used to guided the transcript assembly (isoform deconvolution).
prepareForWebServers	A logical indicating whether the amino acid fasta files saved (if outputSequences=TRUE) should be prepared for the online web-services currently supported (as they have some limitations on what can submitted). See details. Default is FALSE (for backward compatibility).
pathToOutput	A path to the folder in which the plots should be made. Default is working directory (getwd()).
outputSequences	A logical indicating whether transcript nucleotide and amino acid sequences should be outputted to pathToOutput. Default is TRUE.
quiet	A logical indicating whether to avoid printing progress messages (incl. progress bar). Default is FALSE

Details

This function performs the first part of a Isoform Analysis Workflow by

1. Remove low-expressed isoforms and single-isoform genes (see [preFilter](#))
2. Identify isoform switches.
3. If no ORFs are annotated the isoforms are analyzed for open reading frames (ORFs, see [analyzeORF](#))
4. The isoform nucleotide and ORF amino acid sequences are extracted and saved to fasta files as well as added to the switchAnalyzeRlist enabling external sequence analysis with CPAT, Pfam and SignalP (see vignette for more info).

if prepareForWebServers=TRUE both the "removeLongAAseq" and "alsoSplitFastaFile" will be enabled in the extractSequence function.

Value

This function have two outputs. It returns a switchAnalyzeRlist object where information about the isoform switch test, ORF prediction and nt and aa sequences have been added. Secondly (if outputSequences=TRUE) the nucleotide and amino acid sequence of transcripts involved in switches are also save as fasta files enabling external sequence analysis.

Author(s)

Kristoffer Vitting-Seerup

References

Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. Mol. Cancer Res. (2017).

See Also

[preFilter](#)
[isoformSwitchTestDEXSeq](#)
[isoformSwitchTestDRIMSeq](#)
[analyzeORF](#)
[extractSequence](#)

Examples

```
### load example data
data("exampleSwitchList")

### Subset for quick runtime
exampleSwitchList <- subsetSwitchAnalyzeRlist(
  exampleSwitchList,
  abs(exampleSwitchList$isoformFeatures$dIF) > 0.4
)

### Show summary
summary(exampleSwitchList)

### Run Part 1
exampleSwitchList <- isoformSwitchAnalysisPart1(
  switchAnalyzeRlist=exampleSwitchList,
  prepareForWebServers = FALSE,
  outputSequences = FALSE # keeps the function from outputting the fasta files from this example
)

### Show summary
summary(exampleSwitchList)
```

isoformSwitchAnalysisPart2

*Isoform Switch Analysis Workflow Part 2: Plot All Isoform Switches
and Their Annotation*

Description

This high-level function adds the results of the external sequence analysis supplied (if any), then proceeds to analyze alternative splicing. Then functional consequences of the isoform switches are identified and isoform switch analysis plots are created for the top n isoform switches. Lastly a plot summarizing the functional consequences is created. This function is meant to be used after [isoformSwitchAnalysisPart1](#) have been used.

Usage

```

isoformSwitchAnalysisPart2(
  ### Core arguments
  switchAnalyzeRlist,

  ### External annotation arguments
  codingCutoff = NULL,
  removeNoncodingORFs,
  pathToCPATresultFile = NULL,
  pathToCPC2resultFile = NULL,
  pathToPFAMresultFile = NULL,
  pathToIUPred2AresultFile = NULL,
  pathToNetSurfP2resultFile = NULL,
  pathToSignalPresultFile = NULL,

  ### Analysis and output arguments
  n = Inf,
  consequencesToAnalyze = c(
    'intron_retention',
    'coding_potential',
    'ORF_seq_similarity',
    'NMD_status',
    'domains_identified',
    'IDR_identified',
    'IDR_type',
    'signal_peptide_identified'
  ),
  pathToOutput = getwd(),
  fileType = 'pdf',
  outputPlots = TRUE,

  ### Other arguments
  quiet = FALSE
)

```

Arguments

- switchAnalyzeRlist** The switchAnalyzeRlist object as produced by [isoformSwitchAnalysisPart1](#)
- codingCutoff** Numeric indicating the cutoff used by CPAT/CPC2 for distinguishing between coding and non-coding transcripts.
1. For CPAT: The cutoff is dependent on species analyzed. Our analysis suggest that the optimal cutoff for overlapping coding and noncoding isoforms are 0.725 for human and 0.721 for mouse - HOWEVER the suggested cut-offs from the CPAT article (see references) derived by comparing known genes to random non-coding regions of the genome is 0.364 for human and 0.44 for mouse. No default is used.

2. For CPC2: The cutoff suggested is 0.5 for all species, and this cutoff will be used if nothing is specified by the user

`removeNoncodingORFs`

A logic indicating whether to remove ORF information from the isoforms which the CPAT analysis classifies as non-coding. This can be particular useful if the isoform (and ORF) was predicted de-novo but is not recommended if ORFs was imported from a GTF file. This will affect all downstream analysis and plots as both analysis of domains and signal peptides requires that ORFs are annotated (e.g. `analyzeSwitchConsequences` will not consider the domains (potentially) found by Pfam if the ORF have been removed).

`pathToCPATresultFile`

Path to the CPAT result file. If the webserver is used please download the tab-delimited file from the bottom of the result page and give that as input, else simply supply the result file. See [analyzeCPAT](#) for details.

`pathToCPC2resultFile`

Path to the CPC2 result file. If the webserver is used please download the tab-delimited file from the bottom of the result page and give that as input, else simply supply the result file. See [analyzeCPC2](#) for details.

`pathToPFAMresultFile`

A string indicating the full path to the Pfam result file(s). If multiple result files were created (multiple web-server runs) just supply all the paths as a vector of strings. If the webserver is used you need to copy paste the result part of the mail you get into a empty plain text document (notepad, sublimetext TextEdit or similar (aka not word)) and save that. See [analyzePFAM](#) for details.

`pathToIUPred2AresultFile`

A string indicating the full path to the NetSurfP-2 result csv file. See [analyzeIUPred2A](#) for details.

`pathToNetSurfP2resultFile`

A string indicating the full path to the NetSurfP-2 result csv file. See [analyzeNetSurfP2](#) for details.

`pathToSignalPresultFile`

A string indicating the full path to the SignalP result file(s). If multiple result files were created (multiple web-server runs) just supply all the paths as a vector of strings. If using the web-server the results should be copy pasted into a empty plain text document (notepad, sublimetext TextEdit or similar (aka not word)) and save that. See [analyzeSignalP](#) for details.

`n`

The number of top genes (after filtering and sorted according to `sortByQvals`) that should be saved to each sub-folder indicated by `splitConditions`, `splitFunctionalConsequences`. Use `Inf` to create all. Default is `Inf` (all).

`consequencesToAnalyze`

A vector of strings indicating what type of functional consequences to analyze. Do note that there is bound to be some differences between transcripts (else there would be identical). See details in [analyzeSwitchConsequences](#) for full list of usable strings and their meaning. Default is `c('intron_retention', 'coding_potential', 'ORF_seq_similarity', 'ORF_AA_similarity', 'NMD_status', 'PFAM_domains_annotated', 'signal_peptides_annotated')` (corresponding to analyze: intron retention, CPAT result, ORF AA sequence similarity, NMD status, PFAM domains annotated and signal peptides annotated by Pfam).

pathToOutput	A path to the folder in which the plots should be made. Default is working directory (<code>getwd()</code>).
fileType	A string indicating which file type is generated. Available options are <code>'pdf'</code> and <code>'png'</code> . Default is <code>pdf</code> .
outputPlots	A logic indicating whether all isoform switches as well as the summary of functional consequences should be saved in the directory specified by <code>pathToOutput</code> argument. Default is <code>TRUE</code> .
quiet	A logic indicating whether to avoid printing progress messages (incl. progress bar). Default is <code>FALSE</code>

Details

This function performs the second part of a Isoform Analysis Workflow by:

1. Adding external sequence analysis (see [analyzeCPAT](#), [analyzeCPC2](#), [analyzePFAM](#) and [analyzeSignalP](#))
2. Predict functional consequences of switching (see [analyzeSwitchConsequences](#))
3. Output Isoform Switch Consequence plots for all genes where there is a significant isoform switch (see [switchPlot](#))
4. Output a visualization of general consequences of isoform switches.

Value

This function

1. Returns the supplied `switchAnalyzeRlist` now annotated with all the analysis described above
2. Generate one folder per comparison of conditions containing the isoform switch analysis plot of all genes with significant isoforms switches
3. Saves 3 plots summarizing the overall consequences of all the isoform switches.

Author(s)

Kristoffer Vitting-Seerup

References

Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. *Mol. Cancer Res.* (2017).

See Also

[analyzeCPAT](#)
[analyzeCPC2](#)
[analyzeIUPred2A](#)
[analyzeNetSurfP2](#)
[analyzePFAM](#)
[analyzeSignalP](#)

```
analyzeAlternativeSplicing
extractSwitchSummary
analyzeSwitchConsequences
switchPlotTopSwitches
```

Examples

```
### Please note
# The way of importing files in the following example with
# "system.file('pathToFile', package="IsoformSwitchAnalyzer") is
# specialized way of accessing the example data in the IsoformSwitchAnalyzer package
# and not smoothing you need to do - just supply the string e.g.
# "/path/to/externalAnalysis/toolResult.txt" pointing to the result file.

### Load example data
data("exampleSwitchListIntermediary")

### Subset for quick runtime
exampleSwitchListIntermediary <- subsetSwitchAnalyzeRlist(
  exampleSwitchListIntermediary,
  abs(exampleSwitchListIntermediary$isoformFeatures$dIF) > 0.4
)

### Run part 2
exampleSwitchListAnalyzed <- isoformSwitchAnalysisPart2(
  switchAnalyzeRlist      = exampleSwitchListIntermediary,
  pathToCPC2resultFile   = system.file("extdata/cpc2_result.txt", package = "IsoformSwitchAnalyzer"),
  pathToPFAMresultFile   = system.file("extdata/pfam_results.txt", package = "IsoformSwitchAnalyzer"),
  pathToIUPred2AresultFile = system.file("extdata/iupred2a_result.txt.gz", package = "IsoformSwitchAnalyzer"),
  pathToSignalPresultFile = system.file("extdata/signalP_results.txt", package = "IsoformSwitchAnalyzer"),
  codingCutoff            = 0.5, # since we are using CPC2
  removeNoncodinORFs     = TRUE, # Because ORF was predicted de novo
  outputPlots            = FALSE # keeps the function from outputting the plots from this example code
)
```

isoformSwitchTestDEXSeq

Statistical Test for identifying Isoform Switching via DEXSeq

Description

This function utilizes DEXSeq to test isoforms (isoform resolution) for differential isoform usage. It can furthermore also estimate corrected effect sizes (IF and dIF) in experimental setups with confounding effects (such as batches).

Usage

```
isoformSwitchTestDEXSeq(
  ### Core arguments
```

```

switchAnalyzeRlist,
alpha = 0.05,
dIFcutoff = 0.1,

### Advanced arguments
correctForConfoundingFactors=TRUE,
overwriteIFvalues=TRUE,
reduceToSwitchingGenes = TRUE,
reduceFurtherToGenesWithConsequencePotential = TRUE,
onlySigIsoforms = FALSE,
keepIsoformInAllConditions = TRUE,
showProgress = TRUE,
quiet = FALSE
)

```

Arguments

switchAnalyzeRlist
A switchAnalyzeRlist object.

alpha
The cutoff which the FDR correct p-values must be smaller than for calling significant switches. Default is 0.05.

dIFcutoff
The cutoff which the changes in (absolute) isoform usage must be larger than before an isoform is considered switching. This cutoff can remove cases where isoforms with (very) low dIF values are deemed significant and thereby included in the downstream analysis. This cutoff is analogous to having a cutoff on log2 fold change in a normal differential expression analysis of genes to ensure the genes have a certain effect size. Default is 0.1 (10%).

correctForConfoundingFactors
A logic indicating whether IsoformSwitchAnalyzeR to use limma to correct for any confounding effects (e.g. batch effects) as indicated in the design matrix (as additional columns (apart from the two default columns)). Default is TRUE.

overwriteIFvalues
A logic indicating whether to overwrite the IF and dIF stored in the switchAnalyzeRlist with the corrected IF and dIF values - if no confounding effects are present in the design matrix this will not change anything. Default is TRUE.

reduceToSwitchingGenes
A logic indicating whether the switchAnalyzeRlist should be reduced to the genes which contains at least one isoform significantly differential used (as indicated by the alpha and dIFcutoff parameters) - works on dIF values corrected for confounding effects if overwriteIFvalues=TRUE. Enabling this will make the downstream analysis a lot faster since fewer genes needs to be analyzed. Default is TRUE.

reduceFurtherToGenesWithConsequencePotential
A logic indicating whether the switchAnalyzeRlist should be reduced to the genes which have the potential to find isoform switches with predicted consequences. This argument is a more strict version of reduceToSwitchingGenes as it not only requires that at least one isoform is significantly differential used (as indicated by the alpha and dIFcutoff parameters) but also that there is an

isoform with the opposite effect size (e.g. used less if the first isoform is used more). The minimum effect size of the opposing isoform usage is also controlled by `dIFcutoff`. The existence of such an opposing isoform means a switch pair can be formed. It is these pairs that can be analyzed for functional consequences further downstream in the IsoformSwitchAnalyzeR workflow. Enabling this will make the downstream analysis a even faster (than just using `reduceToSwitchingGenes`) since fewer genes needs to be analyzed. Requires that `reduceToSwitchingGenes=TRUE` to have any effect. Default is `TRUE`.

`onlySigIsoforms`

A logic indicating whether both isoforms the pairs considered if `reduceFurtherToGenesWithConsequences` should be significantly differential used (as indicated by the `alpha` and `dIFcutoff` parameters). Default is `FALSE` (aka only one of the isoforms in a pair should be significantly differential used).

`keepIsoformInAllConditions`

A logic indicating whether the an isoform should be kept in all comparisons even if it is only deemed significant (as defined by the `alpha` and `dIFcutoff` parameters) in one comparison. This will not affect downstream runtimes only make the `switchAnalyzeRlist` use slightly more memmory (scaling with the number of conditions compared). Default is `TRUE`.

`showProgress`

A logic indicating whether to make a progress bar (if `TRUE`) or not (if `FALSE`). Defaults is `FALSE`.

`quiet`

A logic indicating whether to avoid printing progress messages (incl. progress bar). Default is `FALSE`

Details

This function uses DEXSeq to test for differential isoform usage using the replicate count matrix. This is done by for each pairwise comparison building and testing one model (building one combined model and testing each pairwise comparison from that is not supported by DEXSeq).

`isoformSwitchTestDEXSeq` also allows for estimation of effect sizes (IF and dIF) corrected for confounding effects (controlled by `correctForConfoundingFactors = TRUE`) (recommended). Confounding effects (stored as additional column(s) in the design matrix (`switchAnalyzeRlist$designMatrix`)) is done by by performing a batch correction on the isoform abundance matrix with `limma::removeBatchEffect()` and afterwards recalculate the IF matrix and summarize the IF and dIF values. These new estimates can be added to the `switchAnalyzeRlist` (overwriting the existing values) by setting `overwriteIFvalues = TRUE`.

Note that the actual testing via DEXSeq always will take confounding effects into account (a full model including all confounding effects are always made).

Value

A `switchAnalyzeRlist` where the following have been modified:

- 1: Two columns, `isoform_switch_q_value` and `gene_switch_q_value` in the `isoformFeatures` entry have overwritten with the result of the test.
- 2: A data.frame containing the details of the analysis have been added (called 'isoform-SwitchAnalysis').

The data.frame added have one row per isoform per comparison of condition and contains the following columns:

- `iso_ref` : A unique reference to a specific isoform in a specific comparison of conditions. Enables easy handles to integrate data from all the parts of a `switchAnalyzeRlist`.
- `gene_ref` : A unique reference to a specific gene in a specific comparison of conditions. Enables easy handles to integrate data from all the parts of a `switchAnalyzeRlist`.
- `isoform_id`: The name of the isoform analyzed. Matches the 'isoform_id' entry in the 'isoformFeatures' entry of the `switchAnalyzeRlist`
- `condition_1`: Condition 1 - the condition used as baseline.
- `condition_2`: Condition 2.
- `dIF`: The difference in IF values (IF2-IF1) - potentially corrected for confounding effects.
- `pvalue`: Isoform level P-values.
- `padj`: Isoform level False Discovery Rte (FDR) corrected P-values (q-values).
- `IF1`: Mean isoform fraction in condition 1 - potentially corrected for confounding effects.
- `IF2`: Mean isoform fraction in condition 2 - potentially corrected for confounding effects.

Author(s)

Kristoffer Vitting-Seerup

References

Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. *Mol. Cancer Res.* (2017). Anders et al. Detecting differential usage of exons from RNA-seq data. *Genome Research* (2012).

See Also

[preFilter](#)
[extractSwitchSummary](#)
[extractTopSwitches](#)

Examples

```
### Please note
# 1) The way of importing files in the following example with
#     "system.file('pathToFile', package="IsoformSwitchAnalyzeR") is
#     specialized way of accessing the example data in the IsoformSwitchAnalyzeR package
#     and not something you need to do - just supply the string e.g.
#     "myAnnotation/isoformsQuantified.gtf" to the functions
# 2) importRdata directly supports import of a GTF file - just supply the
#     path (e.g. "myAnnotation/isoformsQuantified.gtf") to the isoformExonAnnoation argument

### Import quantifications
salmonQuant <- importIsoformExpression(system.file("extdata/", package="IsoformSwitchAnalyzeR"))

### Make design matrix
```

```

myDesign <- data.frame(
  sampleID = colnames(salmonQuant$abundance)[-1],
  condition = gsub('_.*', '', colnames(salmonQuant$abundance)[-1])
)

### Create switchAnalyzeRlist
aSwitchList <- importRdata(
  isoformCountMatrix = salmonQuant$counts,
  isoformRepExpression = salmonQuant$abundance,
  designMatrix = myDesign,
  isoformExonAnnoation = system.file("extdata/example.gtf.gz", package="IsoformSwitchAnalyzeR"),
  showProgress = FALSE
)

### Remove lowly expressed
aSwitchListAnalyzed <- preFilter(aSwitchList)

### Test isoform swtiches
aSwitchListAnalyzed <- isoformSwitchTestDEXSeq(
  switchAnalyzeRlist = aSwitchListAnalyzed
)

# extract summary of number of switching features
extractSwitchSummary(aSwitchListAnalyzed)

```

isoformSwitchTestDRIMSeq

Statistical Test for identifying Isoform Switching via DRIMSeq.

Description

This function is an interface to an analysis with the DRIMSeq package analyzing all isoforms (isoform resolution) and conditions stored in the switchAnalyzeRlist object.

Usage

```

isoformSwitchTestDRIMSeq(
  switchAnalyzeRlist,
  alpha = 0.05,
  dIFcutoff = 0.1,
  testIntegration = 'isoform_only',
  reduceToSwitchingGenes = TRUE,
  reduceFurtherToGenesWithConsequencePotential = TRUE,
  onlySigIsoforms = FALSE,
  keepIsoformInAllConditions = TRUE,
  dmFilterArgs=list(
    min_feature_expr = 4,
    min_samps_feature_expr = min(
      switchAnalyzeRlist$conditions$nrReplicates

```

```

    )
  ),
  dmPrecisionArgs = list(),
  dmFitArgs = list(),
  dmTestArgs = list(),
  showProgress = TRUE,
  quiet = FALSE
)

```

Arguments

- switchAnalyzeRlist**
A switchAnalyzeRlist object.
- alpha**
The cutoff which the FDR correct p-values must be smaller than for calling significant switches. Default is 0.05.
- dIFcutoff**
The cutoff which the changes in (absolute) isoform usage must be larger than before an isoform is considered switching. This cutoff can remove cases where isoforms with (very) low dIF values are deemed significant and thereby included in the downstream analysis. This cutoff is analogous to having a cutoff on log2 fold change in a normal differential expression analysis of genes to ensure the genes have a certain effect size. Default is 0.1 (10%).
- testIntegration**
A string indicating how to interpret the DRIMSeq test for differential isoform usage (see also details). Since DRIMSeq both test at gene and isoform level there are multiple options. Must be one of the following:
- 'isoform_only' : Only considers the test at isoform level resolution (and ignores the gene level test). This analysis have isoform resolution (meaning exactly which isoforms are switching is known). Default
 - 'gene_only' : Only considers the test at gene level resolution (and ignores the isoform level test). This analysis have gene resolution (meaning exactly which isoforms are switching is NOT known - but the power is higher compared to isoform level analysis (probably more genes identified)).
 - 'intersect' : Only considers the cases where BOTH the gene and the isoforms are significant. This analysis have isoform resolution (meaning exactly which isoforms are switching is known) and is the conservative version of 'isoform_only' since it is also required that the gene level test for the parent gene is significant. See details.
- reduceToSwitchingGenes**
A logic indicating whether the switchAnalyzeRlist should be reduced to the genes which contains at least one isoform significantly differential used (as indicated by the alpha and dIFcutoff parameters) - works on dIF values corrected for confounding effects if overwriteIFvalues=TRUE. Enabling this will make the downstream analysis a lot faster since fewer genes needs to be analyzed. Default is TRUE.
- reduceFurtherToGenesWithConsequencePotential**
A logic indicating whether the switchAnalyzeRlist should be reduced to the genes which have the potential to find isoform switches with predicted consequences. This argument is a more strict version of reduceToSwitchingGenes

as it not only requires that at least one isoform is significantly differential used (as indicated by the `alpha` and `dIFcutoff` parameters) but also that there is an isoform with the opposite effect size (e.g. used less if the first isoform is used more). The minimum effect size of the opposing isoform usage is also controlled by `dIFcutoff`. The existence of such an opposing isoform means a switch pair can be formed. It is these pairs that can be analyzed for functional consequences further downstream in the IsoformSwitchAnalyzeR workflow. Enabling this will make the downstream analysis a even faster (than just using `reduceToSwitchingGenes`) since fewer genes needs to be analyzed. Requires that `reduceToSwitchingGenes=TRUE` to have any effect. Default is `TRUE`.

`onlySigIsoforms`

A logic indicating whether both isoforms the pairs considered if `reduceFurtherToGenesWithConsequences` should be significantly differential used (as indicated by the `alpha` and `dIFcutoff` parameters). Default is `FALSE` (aka only one of the isoforms in a pair should be significantly differential used).

`keepIsoformInAllConditions`

A logic indicating whether the an isoform should be kept in all comparisons even if it is only deemed significant (as defined by the `alpha` and `dIFcutoff` parameters) in one comparison. This will not affect downstream runtimes only make the `switchAnalyzeRlist` use slightly more memmory (scaling with the number of conditions compared). Default is `TRUE`.

`dmFilterArgs`

Offers a way to pass additional arguments to the `DRIMSeq::dmFilter()` function enabling filtering based on replicate data. Must be supplied as a named list. Default is 4 counts in at least as many libraries as there are replicates in the smallest condition

`dmPrecisionArgs`

Offers a way to pass additional arguments to the `DRIMSeq::dmPrecision()` function. Must be supplied as a named list. Please remember some parameters are shared between multiple of the `dm*()` functions so if you change a parameter for one function you might also need to change it for the other functions.

`dmFitArgs`

Offers a way to pass additional arguments to the `DRIMSeq::dmFit()` function underlying the test. Must be supplied as a named list. Please remember some parameters are shared between multiple of the `dm*()` functions so if you change a parameter for one function you might also need to change it for the other functions.

`dmTestArgs`

Offers a way to pass additional arguments to the `DRIMSeq::dmTest()` function underlying the test. Must be supplied as a named list. Please remember some parameters are shared between multiple of the `dm*()` functions so if you change a parameter for one function you might also need to change it for the other functions.

`showProgress`

A logic indicating whether to make a progress bar (if `TRUE`) or not (if `FALSE`). Defaults is `FALSE`.

`quiet`

A logic indicating whether to avoid printing progress messages (incl. progress bar). Default is `FALSE`

Details

This wrapper for DRIMSeq utilizes all data to construct one linear model (one fit) on all the data (including the potential extra covariates/batch effects indicated in the `designMatrix` entry of the supplied `switchAnalyzeRlist`). From this unified model all the pairwise test are performed (aka each unique combination of `condition_1` and `condition_2` columns of the `isoformFeatures` entry of the supplied `switchAnalyzeRlist` are tested individually). This is only suitable if a certain overlap between conditions are expected which means if you are analyzing very different conditions it is probably better to remove particular comparisons or make two separate analysis (e.g.. Brain vs Brain cancer vs liver vs liver cancer should probably be analyzed as two separate `switchAnalyzeRlists` whereas WT vs KD1 vs KD2 should be one `switchAnalyzeRlists`).

The result of the `testIntegration` (see arguments and below) is only applied to the `isoformFeatures` entry of the `switchAnalyzeRlist`. The full DRIMSeq analysis is unmodified and added to the `isoformSwitchAnalysis` entry of the `switchAnalyzeRlist`.

The `testIntegration` integration works as follows:

- `'isoform_only'` : Only the FDR adjusted P-values of the isoform level test are used. This is the default since we believe that if an isoform is significant and the effect size is large then the overall effect on the gene should be considered even if the overall gene analysis is not significant.
- `'gene_only'` : Only the FDR adjusted P-values of the gene level test are used. Isoform level data are not used.
- `'intersect'` : The FDR adjusted P-values of the isoform level test are used for cases where the gene level FDR adjusted P-values is smaller than or equal to the smallest FDR adjusted P-values of all associated isoform.

A `'union'` option is not supported due to the loss of False Discovery Rate that would lead to.

To use the `dmPrecisionArgs`, `dmFitArgs`, `dmTestArgs` arguments a named list should simply be supplied - so if you want to modify the `'prec_subset'` argument in the `dmPrecision()` function you should supply `dmPrecisionArgs=list(prec_subset=x)` where `x` is the value you want to pass to the `'prec_subset'` argument.

Please note that: 1) DRIMSeq approach depends on the filtering on the data since if too many lowly expressed transcripts are included the gene precision cannot be calculated. Therefore if you think too few genes have been tested you can try to make a more strict filtering with the `prefilter()` function. 2) DRIMSeq can be a bit slow for large comparisons (testing of many isoforms) and 0.5-1 hour per comparison is not unusual.

Value

A `switchAnalyzeRlist` where the following have been modified:

- 1: Two columns, `isoform_switch_q_value` and `gene_switch_q_value` in the `isoformFeatures` entry have been filled out summarizing the result of the above described test as affected by the `testIntegration` argument.
- 2: A `data.frame` containing the details of the analysis have been added (called `'isoform-SwitchAnalysis'`).

The `data.frame` added have one row per isoform per comparison of condition and contains the following columns:

- `iso_ref` : A unique reference to a specific isoform in a specific comparison of conditions. Enables easy handles to integrate data from all the parts of a `switchAnalyzeRlist`.
- `gene_ref` : A unique reference to a specific gene in a specific comparison of conditions. Enables easy handles to integrate data from all the parts of a `switchAnalyzeRlist`.
- `isoform_id`: The name of the isoform analyzed. Matches the 'isoform_id' entry in the 'isoformFeatures' entry of the `switchAnalyzeRlist`
- `gene_lr`: likelihood ratio statistics based on the DM model.
- `gene_df`: Degrees of freedom
- `gene_p_value`: Gene level P-values.
- `gene_q_value`: Gene level False Discovery Rte (FDR) corrected P-values (q-values).
- `iso_lr`: likelihood ratio statistics based on the BB model.
- `iso_df`: Degrees of freedom
- `iso_p_value`: Isoform level P-values.
- `iso_q_value`: Isoform level False Discovery Rte (FDR) corrected P-values (q-values).

Author(s)

Kristoffer Vitting-Seerup

References

- Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. *Mol. Cancer Res.* (2017).
- Nowicka, M., & Robinson, M. D. (2016). DRIMSeq: a Dirichlet-multinomial framework for multivariate count outcomes in genomics. *F1000Research*, 5(0), 1356. <https://doi.org/10.12688/f1000research.8900.2>

See Also

[preFilter](#)
[isoformSwitchTestDEXSeq](#)
[extractSwitchSummary](#)
[extractTopSwitches](#)
[dmPrecision](#)
[dmFit](#)
[dmTest](#)

Examples

```
### Please note
# 1) The way of importing files in the following example with
#     "system.file('pathToFile', package="IsoformSwitchAnalyzeR") is
#     specialized way of accessing the example data in the IsoformSwitchAnalyzeR package
#     and not something you need to do - just supply the string e.g.
#     "myAnnotation/isoformsQuantified.gtf" to the functions
# 2) importRdata directly supports import of a GTF file - just supply the
#     path (e.g. "myAnnotation/isoformsQuantified.gtf") to the isoformExonAnnoation argument
```

```

### Import quantifications
salmonQuant <- importIsoformExpression(system.file("extdata/", package="IsoformSwitchAnalyzerR"))

### Make design matrix
myDesign <- data.frame(
  sampleID = colnames(salmonQuant$abundance)[-1],
  condition = gsub('_.*', '', colnames(salmonQuant$abundance)[-1])
)

### Create switchAnalyzeRlist
aSwitchList <- importRdata(
  isoformCountMatrix = salmonQuant$counts,
  isoformRepExpression = salmonQuant$abundance,
  designMatrix = myDesign,
  isoformExonAnnoation = system.file("extdata/example.gtf.gz", package="IsoformSwitchAnalyzerR")
)

### Filter with very strict cutoffs to enable short runtime
aSwitchListAnalyzed <- preFilter(
  switchAnalyzeRlist = aSwitchList,
  isoformExpressionCutoff = 10,
  IFcutoff = 0.3,
  geneExpressionCutoff = 50
)
aSwitchListAnalyzed <- subsetSwitchAnalyzeRlist(
  aSwitchListAnalyzed,
  aSwitchListAnalyzed$isoformFeatures$condition_1 == 'hESC'
)

### Test isoform swtiches
aSwitchListAnalyzed <- isoformSwitchTestDRIMSeq(aSwitchListAnalyzed)

# extract summary of number of switching features
extractSwitchSummary(aSwitchListAnalyzed)

```

isoformToGeneExp

Sum transcript/isoform expression to gene get level expression.

Description

This function extract gene count/expression from isoform count/expression by for each condition summing the expression of all isoforms belonging to a specific gene. It can automatically extract the isoform:gene relationship from multiple file-types including GTF/GFF files and isoformSwitch-AnalyzeRlists

Usage

```

isoformToGeneExp(
  isoformRepExpression,
  isoformGeneAnnotation=NULL,

```

```

    quiet = FALSE
  )

```

Arguments

isoformRepExpression

A replicate isoform abundance matrix (not log-transformed) with genes as rows and samples as columns. The isoform:gene relationship can be provided by either:

- Having isoformRepExpression contain two additional columns 'isoform_id' and 'gene_id' indicating which isoforms are a part of which gene
- Using the isoformGeneAnnotation argument.

Importantly isoformRepExpression must contain isoform ids either as separate column called 'isoform_id' or as row.names. The function will figure it out by itself in what combination the annotation is supplied.

isoformGeneAnnotation

Can be either of:

- A data.frame with two columns : 'isoform_id' and 'gene_id' indicating the relationship between isoforms and parent gene. If a gene_name column is present the function checks for annoation problems commonly occurring when transcript assembly is done.
- A GRange with two meta-columns: 'isoform_id' and 'gene_id' indicating the relationship between isoforms and parent gene. If a gene_name column is present the function checks for annoation problems commonly occurring when transcript assembly is done.
- The path to a GTF file containing the annotation.
- A switchAnalyzeRlist.

quiet A logic indicating whether to avoid printing progress messages. Default is FALSE

Value

This function returns a data.frame with gene expression from all samples. The gene_ids will be given in the same way they were presented in the isoformRepExpression input (as row.names or as a separate column (gene_id))

Author(s)

Kristoffer Vitting-Seerup

References

Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. Mol. Cancer Res. (2017).

Examples

```

### Please note
# 1) The way of importing files in the following example with
#     "system.file('pathToFile', package="IsoformSwitchAnalyzeR") is
#     specialized to access the sample data in the IsoformSwitchAnalyzeR package
#     and not something you need to do - just supply the string e.g.
#     "myAnnotation/isoformsQuantified.gtf" to the functions
# 2) importRdata directly supports import of a GTF file - just supply the
#     path (e.g. "myAnnotation/isoformsQuantified.gtf") to the isoformExonAnnotation argument

### Import quantifications
salmonQuant <- importIsoformExpression(system.file("extdata/", package="IsoformSwitchAnalyzeR"))

### Summarize to gene level via GTF file
geneRepCount <- isoformToGeneExp(
  isoformRepExpression = salmonQuant$counts,
  isoformGeneAnnotation = system.file("extdata/example.gtf.gz", package="IsoformSwitchAnalyzeR")
)

### Summarize to gene level via data.frame file
# get data.frame
localAnnotation <- as.data.frame(
  mcols(
    rtracklayer::import(
      system.file("extdata/example.gtf.gz", package="IsoformSwitchAnalyzeR")
    )
  )[,c('transcript_id', 'gene_id')]
)
colnames(localAnnotation)[1] <- 'isoform_id'

geneRepCount <- isoformToGeneExp(
  isoformRepExpression = salmonQuant$counts,
  isoformGeneAnnotation = localAnnotation
)

### From switchAnalyzeRlist
# create design
myDesign <- data.frame(
  sampleID = colnames(salmonQuant$abundance)[-1],
  condition = gsub('_.*', '', colnames(salmonQuant$abundance)[-1])
)

# Create switchAnalyzeRlist
aSwitchList <- importRdata(
  isoformCountMatrix = salmonQuant$counts,
  isoformRepExpression = salmonQuant$abundance,
  designMatrix = myDesign,
  isoformExonAnnotation = system.file("extdata/example.gtf.gz", package="IsoformSwitchAnalyzeR"),
  isoformNtFasta = system.file("extdata/example_isoform_nt.fasta.gz", package="IsoformSwitchAnalyzeR")
)

```

```

)

geneRepCount <- isoformToGeneExp(
  isoformRepExpression = salmonQuant$counts,
  isoformGeneAnnotation = aSwitchList
)

# alternatively use
geneRepCount <- extractGeneExpression(
  aSwitchList,
  extractCounts = TRUE
)

```

isoformToIsoformFraction

Calculate isoform fraction from isoform abundance matrix

Description

General purpose function to calculate isoform fraction (IF) matrix from isoform abundance (and potentially gene abundance) matrix.

Usage

```

isoformToIsoformFraction(
  isoformRepExpression,
  geneRepExpression=NULL,
  isoformGeneAnnotation=NULL,
  quiet = FALSE
)

```

Arguments

isoformRepExpression

A replicate isoform abundance matrix (not log-transformed) with genes as rows and samples as columns. The isoform:gene relationship can be provided by either:

- Having isoformRepExpression contain two additional columns 'isoform_id' and 'gene_id' indicating which isoforms are a part of which gene
- Using the isoformGeneAnnotation argument.

Importantly isoformRepExpression must contain isoform ids either as separate column called 'isoform_id' or as row.names. The function will figure it out by itself in what combination the annotation is supplied.

geneRepExpression

Optional. A gene replicate abundance matrix. Must contain gene ids either as separate column called 'gene_id' or as row.names.

isoformGeneAnnotation	A data.frame or GRange with two (meta) columns: 'isoform_id' and 'gene_id' indicating the relationship between isoforms and parent gene.
quiet	A logic indicating whether to avoid printing progress messages. Default is FALSE

Details

This function calculates isoform fractions from isoform abundances. If geneRepExpression is not supplied the function automatically calculate it by itself.

Note that: 1) isoform:gene relationship can be supplied as two columns either in the isoformRepExpression or as a separate data.frame to isoformGeneAnnotation. 2) The ids in isoformRepExpression and geneRepExpression can be supplied either as row.names or as separate columns respectively called 'isoform_id' and 'gene_id'.

Value

A replicate isoform fraction matrix with layout similar to isoformRepExpression

Author(s)

Kristoffer Vitting-Seerup

References

Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. Mol. Cancer Res. (2017).

Examples

```
### Please note
# 1) The way of importing files in the following example with
#     "system.file('pathToFile', package="IsoformSwitchAnalyzeR") is
#     specialiced to access the sample data in the IsoformSwitchAnalyzeR package
#     and not somhting you need to do - just supply the string e.g.
#     "myAnnotation/isoformsQuantified.gtf" to the functions
# 2) importRdata directly supports import of a GTF file - just supply the
#     path (e.g. "myAnnotation/isoformsQuantified.gtf") to the isoformExonAnnoation argument
```

```
### Import quantifications
salmonQuant <- importIsoformExpression(system.file("extdata/", package="IsoformSwitchAnalyzeR"))
```

```
### Extract gene info
localAnnotaion <- rtracklayer::import(system.file("extdata/example.gtf.gz", package="IsoformSwitchAnalyzeR"))[,
colnames(localAnnotaion@elementMetadata)[1] <- 'isoform_id'
```

```
### Calculate isoform fractions
repIF <- isoformToIsoformFraction(
  isoformRepExpression = salmonQuant$abundance,
  isoformGeneAnnotation = localAnnotaion
```

)

preFilter

*Filtering of a switchAnalyzeRlist***Description**

This function removes genes/isoforms from a switchAnalyzeRlist with the aim of allowing faster processing time as well as more trustworthy results.

Usage

```
preFilter(
  switchAnalyzeRlist,
  geneExpressionCutoff = 1,
  isoformExpressionCutoff = 0,
  IFcutoff=0.01,
  acceptedGeneBiotype = NULL,
  acceptedIsoformClassCode = NULL,
  removeSingleIsoformGenes = TRUE,
  reduceToSwitchingGenes=FALSE,
  reduceFurtherToGenesWithConsequencePotential = FALSE,
  onlySigIsoforms = FALSE,
  keepIsoformInAllConditions=FALSE,
  alpha=0.05,
  dIFcutoff = 0.1,
  quiet=FALSE
)
```

Arguments

switchAnalyzeRlist	A switchAnalyzeRlist object.
geneExpressionCutoff	The expression cutoff (most likely in TPM/RPKM/FPKM) which the average expression in BOTH conditions must be higher than. NULL disables the filter (Not recommended). Default is 1 FPKM/TPM/RPKM.).
isoformExpressionCutoff	The expression cutoff (most likely in RPKM/FPKM) which isoforms must be expressed more than, in at least one conditions of a comparison. NULL disables the filter. Default is 0 (which removes completely unused isoforms).
IFcutoff	The cutoff on isoform usage (measured as Isoform Fraction, see details) which isoforms must be used more than in at least one conditions of a comparison. NULL disables the filter. Default is 0 (which removes non-contributing isoforms).

acceptedGeneBiotype	A vector of strings indicating which gene biotypes (data typically obtained from GTF files). Can be any biotype annotated, the most common being: "protein_coding", "lincRNA" and "antisense". Default is NULL.
acceptedIsoformClassCode	A vector of strings indicating which cufflinks class codes are accepted. Can only be used if data origins from cufflinks. For an updated list with full description see the bottom of this website: http://cole-trapnell-lab.github.io/cufflinks/cuffcompare/#tracking-transfrags-through-multiple-samples-outprefixtracking . Set to NULL to disable. Default is NULL.
removeSingleIsoformGenes	A logic indicating whether to only keep genes containing more than one isoform (in any comparison, after the other filters have been applied). Default is TRUE.
reduceToSwitchingGenes	A logic indicating whether the switchAnalyzeRlist should be reduced to the genes which contains significant switching (as indicated by the alpha and dIFcutoff parameters). Enabling this will make the downstream analysis a lot faster since fewer genes needs to be analyzed. Requires a test of isoform switches have been performed. Default is FALSE.
reduceFurtherToGenesWithConsequencePotential	A logic indicating whether the switchAnalyzeRlist should be reduced to the genes which have the potential to find isoform switches with predicted consequences. This argument is a more strict version of reduceToSwitchingGenes as it not only requires that at least one isoform is significantly differential used (as indicated by the alpha and dIFcutoff parameters) but also that there is an isoform with the opposite effect size (e.g. used less if the first isoform is used more). The minimum effect size of the opposing isoform usage is also controlled by dIFcutoff. The existence of such an opposing isoform means a switch pair can be formed. It is these pairs that can be analyzed for functional consequences further downstream in the IsoformSwitchAnalyzeR workflow. Enabling this will make the downstream analysis a even faster (than just using reduceToSwitchingGenes) since fewer genes needs to be analyzed. Requires that reduceToSwitchingGenes=TRUE to have any effect. Default is FALSE.
onlySigIsoforms	A logic indicating whether both isoforms the pairs considered if reduceFurtherToGenesWithConsequencePotential should be significantly differential used (as indicated by the alpha and dIFcutoff parameters). Default is FALSE (aka only one of the isoforms in a pair should be significantly differential used).
keepIsoformInAllConditions	A logic indicating whether the an isoform should be kept in all comparisons even if it is only passes the filters in one comparison. Default is FALSE.
alpha	The cutoff which the FDR correct p-values must be smaller than for calling significant switches. Only considered if reduceToSwitchingGenes=TRUE. Default is 0.05.
dIFcutoff	The cutoff which the changes in (absolute) isoform usage must be larger than before an isoform is considered switching. This cutoff can remove cases where isoforms with (very) low dIF values are deemed significant and thereby included in

the downstream analysis. This cutoff is analogous to having a cutoff on log2 fold change in a normal differential expression analysis of genes to ensure the genes have a certain effect size. Only considered if reduceToSwitchingGenes=TRUE. Default is 0.1 (10%).

quiet A logic indicating whether to avoid printing progress messages. Default is FALSE

Details

The filtering works by first requiring that the average isoforms/genes expression/usage across all samples is expressed above the cutoffs supplied, then the data is filtered for isoform classes and lastly for single-isoform genes.

Especially the filter for gene expression can be important since a fundamental problem with the IF values (calculated as $\langle \text{isoform_exp} \rangle / \langle \text{gene_exp} \rangle$) is when the gene expression is low it causes the IF measure to loose precision. This can easily be illustrated with the following example: Lets consider a gene with two isoforms which are expressed so they contribute to the gene expression with 73.3% and 26.7%, if we have 100 RNA-seq reads to describe these the problem is easy and we recapitulate the 73%/27% ratio. If we only have 10 reads the measurements get a little more inaccurate since the estimates now will be 70% vs 30%. If the gene is even lower expressed say 5 reads the estimates become 80%/20%. Therefore we want to filter out these genes.

Please note that for the exon entry as well as any replicate matrix entry (counts, abundances or isoform fractions) all isoforms from genes where at least one isoform passed the filters are kept.

Value

A switchAnalyzeRlist object where the genes and isoforms not passing the filters have been removed (from all annotated entries)

Author(s)

Kristoffer Vitting-Seerup

References

Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. Mol. Cancer Res. (2017).

See Also

[createSwitchAnalyzeRlist](#)
[importCufflinksFiles](#)
[importRdata](#)

Examples

```
data("exampleSwitchList")
exampleSwitchListFiltered <- preFilter(
  exampleSwitchList,
```

```

    geneExpressionCutoff = 1,
    isoformExpressionCutoff = 0,
    removeSingleIsoformGenes = TRUE
  )

```

```
prepareSalmonFileDataFrame
```

Prepare data.frame needed run [importSalmonData](#).

Description

An easy to use wrapper for creating the "salmonFileDataFrame" data.frame needed to run [importSalmonData](#).

Usage

```

prepareSalmonFileDataFrame(
  ### Core arguments
  parentDir,

  ### Advanced arguments
  pattern='',
  invertPattern=FALSE,
  ignore.case=FALSE,
  quiet = FALSE
)

```

Arguments

parentDir	Parent directory where each quantified sample is in a sub-directory. The function will then look for files containing the (suffix) of the default files names for the quantification tools. The suffixes identified are 'abundance.tsv' for Kallisto, 'quant.sf' for Salmon, 'isoforms.results' for RSEM and '_data.ctab' for StringTie. This is an alternative to sampleVector (aka only one of them should be used).
pattern	A character string containing a regular expression for which files to import (applied to full path). Default is "" corresponding to all. See <code>base::grepl</code> for more details.
invertPattern	A Logical. If TRUE only use files which do not match the pattern argument.
ignore.case	A logical. If TRUE case is ignored during matching with the pattern argument. If FALSE the matching with the pattern argument is case sensitive.
quiet	A logic indicating whether to avoid printing progress messages (incl. progress bar). Default is FALSE

Value

The data.frame with 3 columns.

- Column 1: "files". Contains the file each found in subdirectories of the parentDir directory.
- Column 2: "names". The name of the subdirectory.
- Column 3: "condition". Set to NA as the function does not attempt to guess conditions. To use [importSalmonData](#) you will need to add these manually.

Author(s)

Kristoffer Vitting-Seerup

References

Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. Mol. Cancer Res. (2017).

See Also

[importSalmonData](#)

Examples

```
### Please note
# The way of importing files in the following example with
# "system.file('pathToFile', package="IsoformSwitchAnalyzeR") is
# specialized way of accessing the example data in the IsoformSwitchAnalyzeR package
# and not something you need to do - just supply the string e.g.
# parentDir = "individual_quantifications_in_subdir/" to the functions
# path (e.g. "myAnnotation/isoformsQuantified.gtf") to the isoformExonAnnotation argument

### Prepare data.frame with quant file info
salmonDf <- prepareSalmonFileDataFrame(
  system.file("extdata/drosophila", package="IsoformSwitchAnalyzeR")
)

### Add conditions
salmonDf$condition <- c('wt','wt','ko','ko')

### Create switchAnalyzeRlist
aSwitchList <- importSalmonData(salmonDf)
```

`subsetSwitchAnalyzeRlist`*A function which subset all entries in a switchAnalyzeRlist.*

Description

This function allows the user to remove data from all entries in a switchAnalyzeRlist about isoforms that are no longer of interest. Note that it retain replicate isoforms information for all isoforms associated with genes containing isoforms in the subset (to enable correction for confounding factors when testing with isoformSwitchTestDEXSeq()).

Usage

```
subsetSwitchAnalyzeRlist(  
  switchAnalyzeRlist,  
  subset  
)
```

Arguments

switchAnalyzeRlist	A switchAnalyzeRlist object.
subset	logical expression indicating which rows in the isoformFeatures entry should be keep. The rest of the switchAnalyzeRlist is then reduced to only contain the matching information.

Value

A SwitchAnalyzeRlist only containing information about the isoforms (in their specific comparisons) indicated with TRUE in the .

Author(s)

Kristoffer Vitting-Seerup

References

Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. Mol. Cancer Res. (2017).

See Also

[createSwitchAnalyzeRlist](#)
[preFilter](#)

Examples

```
data("exampleSwitchList")

subsetSwitchAnalyzeRlist(
  switchAnalyzeRlist = exampleSwitchList,
  subset = exampleSwitchList$isoformFeatures$gene_overall_mean > 10
)
```

switchPlot

*Isoform Switch Analysis Plot***Description**

This function enables a full analysis of a specific gene containing an isoform switch (with functional consequences) by creating a composite plot visualizing 1) The isoform structure along with the concatenated annotations (including transcript classification, ORF, Coding Potential, NMD sensitivity, annotated protein domains as well as annotated signal peptides) 2) gene and isoform expression and 3) isoform usage - including the result of the isoform switch test.

Usage

```
switchPlot(
  ### Core arguments
  switchAnalyzeRlist,
  gene = NULL,
  isoform_id = NULL,
  condition1,
  condition2,

  ### Advanced arguments
  IFcutoff = 0.05,
  dIFcutoff = 0.1,
  alphas = c(0.05, 0.001),
  rescaleTranscripts = TRUE,
  reverseMinus = TRUE,
  addErrorbars = TRUE,
  logYaxis = FALSE,
  localTheme = theme_bw(base_size = 8),
  additionalArguments = list()
)
```

Arguments

switchAnalyzeRlist

A switchAnalyzeRlist object containing all the analysis to be induced (e.g. if protein domains should be visualized they should be annotated in the switchAnalyzeRlist object (via [analyzePFAM](#)))

gene	Either the gene_id or the gene name of the gene to plot, alternatively one can use the isoform_id argument to supply a vector of isoform_ids.
isoform_id	Vector of id indicating which isoforms (from the same gene) to plot, alternatively one can use the gene_id argument to plot all isoforms of a gene.
condition1	First condition of the comparison to analyze. Must match 'condition_1' in the 'isoformFeatures' entry of the switchAnalyzeRlist. Only needed if more than one comparison is analyzed.
condition2	Second condition of the comparison to analyze. Must match 'condition_2' in the 'isoformFeatures' entry of the switchAnalyzeRlist. Only needed if more than one comparison is analyzed.
IFcutoff	The cutoff used for the minimum contribution to gene expression (in at least one condition) for an isoforms must have to be plotted (measured as Isoform Fraction (IF) values). Default is 0.05 (which removes isoforms with minor contribution).
dIFcutoff	The dIF cutoff used to add usage to the transcript plot. Default is 0.1.
alphas	A numeric vector of length two giving the significance levels represented in plots. The numbers indicate the q-value cutoff for significant (*) and highly significant (***) respectively. Default 0.05 and 0.001 which should be interpret as $q < 0.05$ and $q < 0.001$ respectively). If q-values are higher than this they will be annotated as 'ns' (not significant).
rescaleTranscripts	A Logical indicating whether all the isoforms should be resealed to the square root of their original sizes. This feature is implemented because introns usually are much larger than exons making it difficult to see structural changes. This is very useful for structural visualization but the scaling might distort actual intron and exon sizes. Default is TRUE.
reverseMinus	A logic indicating whether isoforms on minus strand should be inverted so they are visualized as going from left to right instead of right to left. (Only affects minus strand isoforms). Default is TRUE
addErrorbars	A logic indicating whether error bars should be added to the expression plots to show uncertainty in estimates (recommended). By default the error-bars indicate 95% confidence intervals, see ?switchPlotGeneExp for more information and additional options (that can be passed via additionalArguments. Default is TRUE.
logYaxis	A logical indicating whether the y-axis of gene and isoform expression sub-plots should be log10 transformed. Default is FALSE.
localTheme	General ggplot2 theme with which the plot is made, see ?ggplot2::theme for more info. Default is theme_bw().
additionalArguments	A named list arguments passed to the functions switchPlotTranscript, switchPlotGeneExp, switchPlotIsoExp, and switchPlotIsoUsage which each creates a subset of the Isoform Switch Analysis Plot. This enable further customization of the plots. The name of the list entries must correspond to the corresponding argument in the sub-function.

Details

The isoform switch analysis plot is a plot contains all the information necessary to judge the importance of a gene with isoform switching, and contains information about from expression levels, switch size as well as the annotation of the isoform differences.

The gene expression, isoform expression and isoform usage plots are generated by `switchPlotGeneExp`, `switchPlotIsoExp` and `switchPlotIsoUsage` respectively. The plot of the transcript structure along with all the annotation is done with `switchPlotTranscript`. The 'Increased/decreased/unchanged usage is determined by the `dIFcutoff` and `alphas` arguments (since we require it to be both significant ($< \min(\text{alphas})$) and changing ($\text{abs}(\text{dIF}) > \text{dIFcutoff}$) before being annotated as changing.

Changes in isoform usage are measure as the difference in isoform fraction (dIF) values, where isoform fraction (IF) values are calculated as $\langle \text{isoform_exp} \rangle / \langle \text{gene_exp} \rangle$. In the transcript structure the annotation of "increased/decrease/unchanged usage" simply indicate if $|\text{dIF}| > \text{dIFcutoff}$.

The `switchPlot` contains regions "Not Annotated" if regions were not analyzed due to the limitations on EBI's website (else EBI will not accept the files). Specifically this is controlled with the "removeLongAaseq" argument of `extractSequence`.

Value

A isoform switch analysis plot

Author(s)

Kristoffer Vitting-Seerup

References

Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. *Mol. Cancer Res.* (2017).

See Also

[isoformSwitchTestDEXSeq](#)
[isoformSwitchTestDRIMSeq](#)
[switchPlotTranscript](#)
[switchPlotGeneExp](#)
[switchPlotIsoExp](#)
[switchPlotIsoUsage](#)
[switchPlotTopSwitches](#)

Examples

```
### Prepare for plotting
data("exampleSwitchListAnalyzed")

mostSwitchingGene <- extractTopSwitches(
  exampleSwitchListAnalyzed,
  filterForConsequences = TRUE,
  n = 1
```

```

)

### Make isoform Switch Analysis Plot
switchPlot(
  switchAnalyzeRlist = exampleSwitchListAnalyzed,
  gene = mostSwitchingGene$gene_id,
  condition1 = mostSwitchingGene$condition_1,
  condition2 = mostSwitchingGene$condition_2
)

```

switchPlotFeatureExp *Plots for Analyzing Expression and Isoform Usage*

Description

Together these three plots enables visualization of gene expression, isoform expression as well as isoform usage.

Usage

```

switchPlotGeneExp(
  switchAnalyzeRlist,
  gene = NULL,
  condition1 = NULL,
  condition2 = NULL,
  addErrorbars = TRUE,
  confidenceIntervalErrorbars = TRUE,
  confidenceInterval = 0.95,
  alphas = c(0.05, 0.001),
  logYaxis=FALSE,
  extendFactor = 0.05,
  localTheme = theme_bw()
)

switchPlotIsoExp(
  switchAnalyzeRlist,
  gene=NULL,
  isoform_id = NULL,
  condition1 = NULL,
  condition2 = NULL,
  IFcutoff = 0.05,
  addErrorbars = TRUE,
  confidenceIntervalErrorbars = TRUE,
  confidenceInterval = 0.95,
  alphas = c(0.05, 0.001),
  logYaxis=FALSE,
  extendFactor = 0.05,
  localTheme = theme_bw()
)

```

```

)

switchPlotIsoUsage(
  switchAnalyzeRlist,
  gene=NULL,
  isoform_id = NULL,
  condition1 = NULL,
  condition2 = NULL,
  IFcutoff = 0.05,
  addErrorbars = TRUE,
  confidenceIntervalErrorbars = TRUE,
  confidenceInterval = 0.95,
  alphas = c(0.05, 0.001),
  extendFactor = 0.05,
  localTheme = theme_bw()
)

```

Arguments

switchAnalyzeRlist	A switchAnalyzeRlist object
gene	The gene_id or the gene name of the gene to plot. If not supplied 'isoform_id' must be supplied.
isoform_id	Vector of id indicating which isoforms (from the same gene) to plot. If not supplied 'gene' must be supplied.
condition1	First condition of the comparison to analyze. Must match 'condition_1' in the 'isoformFeatures' entry of the switchAnalyzeRlist. Only needed if more than one comparison is analyzed.
condition2	Second condition of the comparison to analyze. Must match 'condition_2' in the 'isoformFeatures' entry of the switchAnalyzeRlist. Only needed if more than one comparison is analyzed.
IFcutoff	The cutoff which the Isoform Fraction (IF) value (in at least one condition) must be larger than for a isoforms to be plotted. Default is 0.05 (which removes isoforms with minor contribution).
addErrorbars	A logic indicating whether error bars should be added to the expression plots to show uncertainty in estimates (recommended). Default is TRUE.
confidenceIntervalErrorbars	A logic indicating whether error bars should be given as confidence intervals (if TRUE)(recommended) or standard error of mean (if FALSE). Default is TRUE.
confidenceInterval	The confidence level used in the confidence intervals if confidenceIntervalErrorbars is enabled. Default is 0.95 corresponding to 95% (recommended).
alphas	A numeric vector of length two giving the significance levels represented in plots. The numbers indicate the q-value cutoff for significant (*) and highly significant (***) respectively. Default 0.05 and 0.001 which should be interpret as $q < 0.05$ and $q < 0.001$ respectively). If q-values are higher than this they will be annotated as 'ns' (not significant).

logYaxis	A logical indicating whether the y-axis of the plot should be log10 transformed (a pseudocount of 1 will be added to avoid large negative values). Default is FALSE.
extendFactor	A numeric controlling the distance (as fraction of expression) between the bars indicating the expression values and the indications of significance. Default is 0.1
localTheme	General ggplot2 theme with which the plot is made, see ?ggplot2::theme for more info. Default is theme_bw().

Details

Changes in isoform usage are measured as the difference in isoform fraction (dIF) values, where isoform fraction (IF) values are calculated as $\langle \text{isoform_exp} \rangle / \langle \text{gene_exp} \rangle$.

Note that the bar indicating significance levels will only be shown if the analysis has been performed (if the q-values are not NA).

Value

- `switchPlotGeneExp` : Generates a gene expression plot which also indicates whether the gene are differentially expressed between the two conditions
- `switchPlotIsoExp` : Generates a isoform expression plot which also indicates whether the isoforms are differentially expressed between the two conditions
- `switchPlotIsoUsage` : Plots the changes in isoform usage (given by IF the values) along with the significance of the change in isoform usage of each isoform. Requires that the result of a differential isoform usage analysis have been performed (for example via [isoformSwitchTestDEXSeq](#)).

Author(s)

Kristoffer Vitting-Seerup

References

Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. *Mol. Cancer Res.* (2017).

See Also

[isoformSwitchTestDEXSeq](#)
[isoformSwitchTestDRIMSeq](#)
[switchPlotTranscript](#)
[switchPlot](#)

Examples

```
### Prepare for plotting
data("exampleSwitchListAnalyzed")

mostSwitchingGene <- extractTopSwitches(
```

```

    exampleSwitchListAnalyzed,
    filterForConsequences = TRUE,
    n = 1
)

### Plot expression
switchPlotGeneExp(
  exampleSwitchListAnalyzed,
  gene = mostSwitchingGene$gene_id,
  condition1 = mostSwitchingGene$condition_1,
  condition2 = mostSwitchingGene$condition_2
)

switchPlotIsoExp(
  exampleSwitchListAnalyzed,
  gene = mostSwitchingGene$gene_id,
  condition1 = mostSwitchingGene$condition_1,
  condition2 = mostSwitchingGene$condition_2
)

switchPlotIsoUsage(
  exampleSwitchListAnalyzed,
  gene = mostSwitchingGene$gene_id,
  condition1 = mostSwitchingGene$condition_1,
  condition2 = mostSwitchingGene$condition_2
)

```

switchPlotTopSwitches *Creating the Isoform Switch Analysis Plot for the Top Switches*

Description

This function outputs the top n (defined by n) Isoform Switch Analysis Plot (see [switchPlot](#)) for genes with significant isoform switches (as defined by α and dIF_{cutoff}) to a specific folder (controlled by `pathToOutput`). The plots are automatically sorted by decreasing significance or switch size (as controlled by `sortByQvals`). The plots can furthermore be created in sub-folders based both which conditions are compared and whether any consequences of the switch have been predicted. In summary it facilitates an easy and prioritized, (but comprehensive), manual analysis of isoform switches.

Usage

```

switchPlotTopSwitches(
  switchAnalyzeRlist,
  alpha = 0.05,
  dIFcutoff = 0.1,
  onlySigIsoforms = FALSE,
  n=10,
  sortByQvals=TRUE,

```

```

    filterForConsequences = FALSE,
    pathToOutput = getwd(),
    splitComparison=TRUE,
    splitFunctionalConsequences = TRUE,
    IFcutoff=0.05,
    fileType = "pdf",
    additionalArguments=list(),
    quiet=FALSE
)

```

Arguments

- switchAnalyzeRlist**
A switchAnalyzeRlist containing all the annotation for the isoforms.
- alpha**
The cutoff which the FDR correct p-values must be smaller than for calling significant switches. Default is 0.05.
- dIFcutoff**
The cutoff which the changes in (absolute) isoform usage must be larger than before an isoform is considered switching. This cutoff can remove cases where isoforms with (very) low dIF values are deemed significant and thereby included in the downstream analysis. This cutoff is analogous to having a cutoff on log2 fold change in a normal differential expression analysis of genes to ensure the genes have a certain effect size. Default is 0.1 (10%).
- onlySigIsoforms**
A logic indicating whether to only consider significant isoforms, meaning only analyzing genes where at least two isoforms which both have significant usage changes in opposite direction (quite strict). Naturally this only works if the isoform switch test used have isoform resolution (which the build in [isoform-SwitchTestDEXSeq](#) has). If FALSE all isoforms with an absolute dIF value larger than dIFcutoff in a gene with significant switches (defined by alpha and dIFcutoff) are included in the pairwise comparison. Default is FALSE (non significant isoforms are also considered based on the logic that if one isoform changes its contribution - there must be an equivalent opposite change in usage in the other isoforms from that gene).
- n**
The number of top genes (after filtering and sorted according to sortByQvals) that should be generated in each sub-folder indicated by splitComparison and splitFunctionalConsequences. Use Inf to create all (NA will internally be converted to Inf for backward comparability). Default is 10.
- sortByQvals**
A logic indicating whether the top n features are sorted by decreasing significance (increasing q-values) (if sortByQvals=TRUE) or decreasing switch size (absolute dIF, which are still significant as defined by alpha) (if sortByQvals=FALSE). The dIF values for genes are considered as the total change within the gene calculated as $\text{sum}(\text{abs}(\text{dIF}))$ for each gene. Default is TRUE (sort by p-values).
- filterForConsequences**
A logic indicating whether to only plot gene with predicted consequences of the isoform switch. Requires that predicted consequences have been annotated (via [analyzeSwitchConsequences](#)). Default is FALSE.

pathToOutput	A path to the folder in which the plots should be made. Default is working directory (getwd()).
splitComparison	A logic indicating whether to create a sub-folder for each comparison. If splitComparison is TRUE the sub-folders will be created else all isoform switch analyzer plots will saved in the same folder. Default is TRUE.
splitFunctionalConsequences	A logic indicating whether to create a sub-folder for those switches with predicted consequences and another sub-folder for those without. Requires that analyzeSwitchConsequences have been run. If splitComparison=TRUE the sub-folders from this argument will be created within the comparison sub-folders. Default is TRUE.
IFcutoff	The cutoff used for the minimum contribution to gene expression (in at least one condition) an isoforms must have to be plotted (measured as Isoform Fraction (IF) values). Default is 0 (which removes isoforms not contributing in any of the conditions).
fileType	A string indicating which file type is generated. Available are options are <code>\`pdf\`</code> and <code>\`png\`</code> . Default is pdf.
additionalArguments	A named list arguments passed to the <code>switchPlot</code> function which creates the individual Isoform Switch Analysis Plots. The name of the list entries must correspond to the corresponding argument in the <code>switchPlot</code> function.
quiet	A logic indicating whether to avoid printing progress messages. Default is FALSE

Details

Changes in isoform usage are measure as the difference in isoform fraction (dIF) values, where isoform fraction (IF) values are calculated as $\langle \text{isoform_exp} \rangle / \langle \text{gene_exp} \rangle$.

For a list of the top switching genes see `?extractTopSwitches`.

Value

An Isoform Switch Analysis Plot (as produce by `switchPlot`) for each of the top `n` switches in each comparison where a gene have a significant isoform switch is generated in the folder supplied by `pathToOutput`

Author(s)

Kristoffer Vitting-Seerup

References

Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. *Mol. Cancer Res.* (2017).

See Also

[switchPlot](#)
[analyzeSwitchConsequences](#)

switchPlotTranscript *Plot Transcript Structure and Annotation*

Description

This function plots the transcript structure of all (or selected) isoforms from a gene along with all the annotation added to the `switchAnalyzeRlist` including transcript classification, ORF, Coding Potential, NMD sensitivity, annotated protein domains as well as annotated signal peptides.

Usage

```
switchPlotTranscript(  
  ### Core arguments  
  switchAnalyzeRlist,  
  gene = NULL,  
  isoform_id = NULL,  
  
  ### Advanced arguments  
  rescaleTranscripts = TRUE,  
  plotXaxis = !rescaleTranscripts,  
  reverseMinus = TRUE,  
  ifMultipleIdenticalAnnotation = 'summarize',  
  annotationImportance = c('signal_peptide', 'protein_domain', 'idr'),  
  IFcutoff = 0.05,  
  rectHegith = 0.2,  
  codingWidthFactor = 2,  
  nrArrows = 20,  
  arrowSize = 0.2,  
  optimizeForCombinedPlot = FALSE,  
  condition1 = NULL,  
  condition2 = NULL,  
  dIFcutoff = 0.1,  
  alphas = c(0.05, 0.001),  
  localTheme = theme_bw()  
)
```

Arguments

`switchAnalyzeRlist`

A `switchAnalyzeRlist` object where the ORF is annotated (for example via [analyzeORF](#)).

gene	Either the gene_id or the gene name of the gene to plot, alternatively one can use the isoform_id argument to supply a vector of isoform_ids.
isoform_id	A vector of the id(s) of which isoform(s) (from the same gene) to plot, alternatively one can use the gene_id argument to plot all isoforms of a gene.
rescaleTranscripts	A Logical indicating whether all the isoforms should be rescaled to the square root of their original sizes. This feature is implemented because introns usually are much larger than exons making it difficult to see structural changes. This is very useful for structural visualization but the scaling might distort actual intron and exon sizes. Default is TRUE.
plotXaxis	A logical indicating whether x-axis should be shown. Default is the opposite of the rescaleTranscripts (meaning FALSE when rescale is TRUE and vice versa).
reverseMinus	A logic indicating whether isoforms on minus strand should be inverted so they are visualized as going from left to right instead of right to left. (Only affects minus strand isoforms). Default is TRUE
ifMultipleIdenticalAnnotation	This argument determines how to visually handle if multiple instances of the same domain is found, the options are A) 'summarize' which will assign one color to all the domains (and adding the number of domains in a bracket in the legend). B) 'number' which will add a number to each domain and give each domain a separate color. Default is 'summarize'. C) 'ignore' which will cause IsoformSwitchAnalyzeR to just plot all of them in the same color but without highlighting differences in numbers.
annotationImportance	Since some of the annotation collected potentially overlap (mainly protein domains and IDR) but only one can be visualized for a given position in the transcript this argument controls the importance of the respective annotations. This argument is used to control which annotation is shown for a given position in the transcript. Must be a vector of strings indicating the order of the annotations in decreasing importance. All annotation must be mentioned even if they have not been analyzed. Default is c('signal_peptide','protein_domain','idr') which means that if an IDR and a protein domain partially overlap the protein domain will be visualized for the overlapping region (non-overlapping regions are not affected).
IFcutoff	The cutoff used for the minimum contribution to gene expression (in at least one condition) for an isoforms must have to be plotted (measured as Isoform Fraction (IF) values). Default is 0.05 (which removes isoforms with minor contribution).
rectHegith	When drawing the transcripts what should be the size of the non-coding (and UTR) regions (if the total height of a transcript is larger than 1 they start to overlap).
codingWidthFactor	The number deciding the width of the coding regions compared to the non-coding (as a fraction of the non-coding). A number larger than 1 will result in coding regions being thicker than non-coding regions.

nrArrows	An integer controlling the number of arrows drawn in the intron of transcripts. Given as the number of arrows a hypothetical intron spanning the whole plot window should have (if you get no arrows increase this value). Default is 20.
arrowSize	The size of arrowhead drawn in the intron of transcripts. Default is 0.2
optimizeForCombinedPlot	A logic indicating whether to optimize for use with switchPlot(). Default is FALSE
condition1	First condition of the comparison to analyze must be the name used in the switchAnalyzeRlist. If specified text indicating change in isoform usage is also added to the plot.
condition2	Second condition of the comparison to analyze, must be the name used in the switchAnalyzeRlist. If specified text indicating change in isoform usage is also added to the plot.
dIFcutoff	The dIF cutoff used to add usage to the transcript plot. Only considered if both condition1 and condition2 are defined. Default is 0.1.
alphas	A numeric vector of length two giving the significance levels represented in the usage text added to the plot. The numbers indicate the q-value cutoff for significant (*) and highly significant (***) respectively. Only considered if both condition1 and condition2 are defined. Default is 0.1. Default 0.05 and 0.001 which should be interpret as $q < 0.05$ and $q < 0.001$ respectively). If q-values are higher than this they will be annotated as 'ns' (not significant).
localTheme	General ggplot2 theme with which the plot is made, see ?ggplot2::theme for more info. Default is theme_bw().

Details

This function generates a plot visualizing all the annotation for the transcripts gathered. The plot supports visualization of:

- ORF : Making the ORF part of the transcript thicker. Requires that ORF have been annotated (e.g.. via analyzeORF).
 - Coding Potential / NMD : The transcripts will be plotted in 3 categories: 'Coding', 'Non-coding' and 'NMD-sensitive'. The annotation of 'Coding' and 'Non-coding' requires the result of an external CPAT analysis have been added with analyzeCPAT. The NMD sensitivity is added by the analyzeORF.
 - Protein domains : By coloring the part of the ORF containing the protein domains. Requires the result of an external Pfam analysis have been added with analyzePFAM). Structural variants (meaning non-complete protein domains) are dindicated. If multiple of the same domain is present they are summarized as indicated by the ifMultipleIdenticalAnnotation arugment (default add "(xY)" where Y is the number of identical domains)
 - Signal Peptide : By coloring the part of the ORF containing the signal peptide. Requires the result of an external SignalP analysis have been added with analyzeSignalP).
- Transcript status : Specifically from data imported from cufflinks/cuffdiff. The status (class code) of the transcript is added in brackets after the transcript name.

Value

Returns the gg object which can then be modified or plotted in a different setting.

Author(s)

Kristoffer Vitting-Seerup

References

Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. *Mol. Cancer Res.* (2017).

See Also

[analyzeORF](#)
[analyzeCPAT](#)
[analyzePFAM](#)
[analyzeSignalP](#)

Examples

```
### Prepare for plotting
data("exampleSwitchListAnalyzed")

mostSwitchingGene <- extractTopSwitches(
  exampleSwitchListAnalyzed,
  filterForConsequences = TRUE,
  n = 1
)

### Plot transcript structure
switchPlotTranscript(exampleSwitchListAnalyzed, gene = mostSwitchingGene$gene_id)
```

Index

- * **classes**
 - CDSSet, 38
- * **datasets**
 - exampleData, 43
- addORFfromGTF, 3, 18, 20–22
- analyzeAlternativeSplicing, 5, 52, 63, 64, 66, 69, 72, 108
- analyzeCPAT, 8, 11, 15, 18, 19, 21, 24, 26, 29, 36, 52, 60, 106, 107, 140
- analyzeCPC2, 10, 10, 106, 107
- analyzeIntronRetention
 - (analyzeAlternativeSplicing), 5
- analyzeIUPred2A, 12, 18, 106, 107
- analyzeNetSurfP2, 9, 11, 15, 15, 26, 29, 106, 107
- analyzeNovelIsoformORF, 3, 5, 18, 21, 22
- analyzeORF, 4, 20, 21, 36, 38, 52, 57, 60, 78, 101, 103, 104, 137, 140
- analyzePFAM, 9, 11, 15, 18, 24, 29, 36, 52, 60, 106, 107, 128, 140
- analyzeSignalP, 10, 11, 15, 18, 26, 27, 36, 52, 60, 106, 107, 140
- analyzeSwitchConsequences, 10, 11, 15, 18, 26, 29, 29, 44, 46, 48, 52–55, 74, 75, 77, 100, 101, 106–108, 135–137
- CDSSet, 38, 78
- CDSSet-class (CDSSet), 38
- createSwitchAnalyzeRlist, 9, 11, 15, 18, 24, 26, 29, 39, 82, 86, 89, 95, 96, 98, 99, 124, 127
- dmFit, 116
- dmPrecision, 116
- dmTest, 116
- exampleData, 43
- exampleSwitchList (exampleData), 43
- exampleSwitchListAnalyzed
 - (exampleData), 43
- exampleSwitchListIntermediary
 - (exampleData), 43
- extractConsequenceEnrichment, 36, 44, 48–50, 52, 55
- extractConsequenceEnrichmentComparison, 36, 46, 47, 52, 55
- extractConsequenceGenomeWide, 36, 46, 48, 49, 55
- extractConsequenceSummary, 36, 53
- extractGeneExpression, 55
- extractGenomeWideAnalysis
 - (extractConsequenceGenomeWide), 49
- extractSequence, 9, 11, 13, 15, 16, 18, 24–26, 28, 29, 43, 57, 101, 104, 130
- extractSplicingEnrichment, 8, 61, 66–70, 72
- extractSplicingEnrichmentComparison, 8, 64, 64, 69, 72
- extractSplicingGenomeWide, 8, 64, 66, 67, 72
- extractSplicingSummary, 8, 61, 64, 66, 69, 70
- extractSwitchOverlap, 72, 75
- extractSwitchSummary, 46, 48, 74, 74, 101, 108, 111, 116
- extractTopSwitches, 74, 75, 76, 111, 116
- getCDS, 38, 78
- importCufflinksFiles, 40, 42, 79, 124
- importGTF, 42, 83
- importIsoformExpression, 42, 86, 90, 96, 97, 99
- importRdata, 40, 42, 83, 89, 90, 97–99, 124
- importSalmonData, 97, 125, 126
- isoformSwitchAnalysisCombined, 99
- isoformSwitchAnalysisPart1, 101, 102, 104, 105
- isoformSwitchAnalysisPart2, 101, 104

isoformSwitchTestDEXSeq, [24](#), [31](#), [50](#), [52](#),
[60](#), [62](#), [64–66](#), [68](#), [69](#), [71](#), [74](#), [75](#), [77](#),
[101](#), [104](#), [108](#), [116](#), [130](#), [133](#), [135](#)

isoformSwitchTestDRIMSeq, [24](#), [52](#), [60](#), [64](#),
[66](#), [69](#), [74](#), [75](#), [77](#), [101](#), [104](#), [112](#),
[130](#), [133](#)

isoformToGeneExp, [117](#)

isoformToIsoformFraction, [120](#)

preFilter, [24](#), [74](#), [75](#), [77](#), [82](#), [86](#), [89](#), [96](#), [99](#),
[101](#), [103](#), [104](#), [111](#), [116](#), [122](#), [127](#)

prepareSalmonFileDataFrame, [97](#), [99](#), [125](#)

subsetSwitchAnalyzeRlist, [127](#)

switchAnalyzeRlist, [60](#)

switchAnalyzeRlist
(createSwitchAnalyzeRlist), [39](#)

switchAnalyzeRlist-class
(createSwitchAnalyzeRlist), [39](#)

switchPlot, [101](#), [107](#), [128](#), [133](#), [134](#), [137](#)

switchPlotFeatureExp, [131](#)

switchPlotGeneExp, [130](#)

switchPlotGeneExp
(switchPlotFeatureExp), [131](#)

switchPlotIsoExp, [130](#)

switchPlotIsoExp
(switchPlotFeatureExp), [131](#)

switchPlotIsoUsage, [130](#)

switchPlotIsoUsage
(switchPlotFeatureExp), [131](#)

switchPlotTopSwitches, [101](#), [108](#), [130](#), [134](#)

switchPlotTranscript, [130](#), [133](#), [137](#)