Package 'IsoformSwitchAnalyzeR'

April 12, 2018

Type Package

Title An R package to Identify, Annoatate and Visialize Isoform Switches with Functional Consequences (from RNA-seq data)

Version 1.0.0

Author Kristoffer Vitting-Seerup

Maintainer Kristoffer Vitting-Seerup <k.vitting.seerup@gmail.com>

Description IsoformSwitchAnalyzeR enabels identification and analysis of isoform switches with predicted functional consequences (such as gain/loss of protein domains etc) from quantification by Kallisto, Salmon, Cufflinks/Cuffdiff, RSEM etc.

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

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

License GPL (>= 2)

Depends R (>= 3.4), cummeRbund, spliceR

Imports methods, BSgenome, plyr, reshape2, gridExtra, Biostrings, IRanges, GenomicRanges, DRIMSeq, RColorBrewer, rtracklayer, ggplot2, VennDiagram, DBI, grDevices, graphics, stats, utils, GenomeInfoDb, grid, tximport, edgeR

Suggests knitr, BSgenome.Hsapiens.UCSC.hg19

VignetteBuilder knitr

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

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

RoxygenNote 6.0.1

NeedsCompilation no

R topics documented:

`analyzeCPAT` *Import Result of External Sequence Analysis*

Description

Allows for easy integration of the result of COAT (external sequence analysis of coding potential) in the IsoformSwitchAnalyzeR workflow. Please note that due to the 'removeNoncodinORFs' of the `analyzeCPAT` argument we recomend using `analyzeCPAT` before `analyzePFAM` and `analyzeSignalP` if you have predicted the ORFs with `analyzeORF`.

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.
removeNoncodinORFs	: A logic indicating wether 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 paramters. If the webser (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 just supply the path to the result file.

Value

Two colums 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
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
```

---

analyzeIntronRetention

Detection of Intron Retion(s)

Description

This function utilize the analysis of alternative splicing implemented in the spliceR package's spliceR function (see ?spliceR::spliceR) to detect whether an isoform contain intron retentions when compared to the hypothetical pre-RNA generated by combining all the exons within a gene. Here an intron retention is defined as when one exon of an isoform overlaps two seperate exons in the hypothetical pre-RNA.

Usage

```
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

Value

A switchAnalyzeRlist where the collumn 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 'intronRetentionAnalysis') containing the number of intron retentions as well as the genomic coordinates of these for each isoform_id, is added to the switchAnalyzeRlist.

Author(s)

Kristoffer Vitting-Seerup

References

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

See Also

switchAnalyzeRlist
isoformSwitchTest
analyzeSwitchConsequences

Examples

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

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

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

`analyzeORF` *Predictiction of Transcript Open Reading Frame.*

Description

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 though one of the implemented import methods (see vignette for details).

Usage

```
analyzeORF(
    switchAnalyzeRlist,
    genomeObject,
    minORFlength=100,
    orfMethod = "longest",
    cds = NULL,
    PTCDistance = 50,
    startCodons="ATG",
    stopCodons=c("TAA", "TAG", "TGA"),
    showProgress=TRUE,
    quiet=FALSE
)
```

Arguments

`switchAnalyzeRlist`
: A `switchAnalyzeRlist` object. n

`genomeObject`
: A `BSgenome` object uses as refrence genome (fx 'Hsapiens' for Homo sapiens).

`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 4 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 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 annoated translation start site (which are supplied via the `cds` argument).

- `mostUpstreamAnnoated` : Identifies the ORF (after filtering via minORFlength) downstream of the most unstream overlapping annoated translation start site (supplied via the `cds` argument).

Default is `longest`.

`cds` A CDSSet object containing annotated coding regions, see ?CDSSet and ?getCDS for more information. Only necessary if `orfMethod` arguments is 'longestAnnotated' or 'mostUpstreamAnnoated'.

`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

The function uses the genomic coordinats of the transcript model to extract the nucleotide sequence of the transcript from the supplied BSgenome object (refrence genome). The nucloetide sequence is then used to predict the most likely ORF (the method is controled 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 accordence with litterature consensus (Weischenfeldt et al (see refrences)).

The gencode refrence annoation used here are GencodeV19, GencodeV24, GencodeM1 and GencodeM9. For more info see Vitting-Seerup et al 2017.

Value

A `switchAnalyzeRlist` where:

- `1`: A collumns called `PTC` indicating the NMD sensitivity have been added to the `isoformFeatures` entry of the switchAnalyzeRlist.
- `2`: 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. Mathces the 'isoform_id' entry in the 'isoformFeatures' entry of the switchAnalyzeRlist
- `orfTransciptStart`: The start position of the ORF in transcript cooridnats, here defined as the position of the 'A' in the 'AUG' start motif.
- `orfTransciptEnd`: The end position of the ORF in transcript coordinats, here defined as the last nucleotide before the STOP codon (meaning the stop codon is not included in these coordinates).

- `orfTransciptLength`: 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 cooridnats, here defined as the the position of the 'A' in the 'AUG' start motif.
- `orfEndGenomic`: The end position of the ORF in genomic coordinats, 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 junciton.

NA means no information was advailable 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`
`isoformSwitchTest`
`extractSequence`
`analyzeCPAT`

Examples

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

# Perfom test
exampleSwitchListAnalyzed <- isoformSwitchTest(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 collumn added
```

analyzePFAM *Import Result of PFAM analysis*

Description

Allows for easy integration of the result of Pfam (external sequence analysis of protein domains) in the IsoformSwitchAnalyzeR workflow. Please note that due to the 'removeNoncodinORFs' of the `analyzeCPAT` argument we recommend using `analyzeCPAT` before `analyzePFAM` 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. 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

Notes for how to run the external tools:
Use default paramters. If you want to use the webserver it is easily done as follows:. 1) Go to https://www.ebi.ac.uk/Tools/hmmer/search/hmmscan 2) Switch to the the "Upload a File" tab. 3) Upload the amino acoid file (_AA) created with `extractSequence` file and add your mail adress - this is important beacue 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 recieve the email with the result (usually quite fast). 7) Copy/paste the result part of the (ONLY what is below the "====" line) 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()

If a stand-alone version was just 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()`.

Value

A colum 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 can 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 coordinat of the start of the domain.
- `pfamEndGenomic`: The genomic coordinat of the end of the domain.

Furthermore depending on the exact tool used (local vs web-server) additional collums are added with inforation 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`
`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` *Import Result of SignalP Analysis*

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 'removeNoncodinORFs' of the `analyzeCPAT` argument we recommend using `analyzeCPAT` before `analyzePFAM` and `analyzeSignalP` if you have predicted the ORFs with `analyzeORF`.

Usage

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

Arguments

`switchAnalyzeRlist`	A `switchAnalyzeRlist` object
`pathToSignalPresultFile`	A string indicating the full path to the summary SignalP result file. See `details` for suggestion of how to run and obtain the result of the SignalP tool.
`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: If using the web-server (http://www.cbs.dtu.dk/services/SignalP/) SignalP should be run with the paramter "standard" under "Output format" and "No graphics" under "Graphics output". When using a stand-alone version SignalP should be run with the '-f summary' option. If using the web-server the results should be copy pasted (from the webpage) into an empty plain text document (notepat, sublimetext TextEdit or similar (not word)) and save that to a txt file. This file is then used as input to the function. If a stand-alone version was just supply the path to the summary result file

Value

A collum called 'signal_peptide_identified' is added to `isoformFeatures` containing a binary indiaction (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 preprocessor 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). Refrence: http://www.cbs.dtu.dk/services/SignalP/instructions.php#method
- `aa_removed`: A integer giving the number of amino acids removed when the signal peptide is cleaved off.
- `transcriptClevageAfter`: The transcript position of the last nucleotide in the isoform which is removed when the signal peptide is cleaved off.
- `genomicClevageAfter`: 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` : Petersen et al. SignalP 4.0: discriminating signal peptides from transmembrane regions. Nature Methods, 8:785-786, 2011

See Also

`createSwitchAnalyzeRlist`
`extractSequence`
`analyzeSwitchConsequences`

Examples

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

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

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',
        '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 `isoformSwitchTest()`) 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 isoorms (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_s (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. Defualit 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 signicant 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 isoformSwitchTest 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 it 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 repported. NULL disables the filter. Default is NULL.

`ntJCsimCutoff` An numeric (between 0 and 1) indicating the cutoff on Jacard Similarity (JCsim) (see details) between the overlap of two nucloetide (nt) sequences. If the measured JCsim is smaller than this cutoff the sequences are considered different and repported 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 repporting 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 protein domain length difference, measured as a fraction of the longest protein domain, a comparison must be larger than before repporting it. Only used when analyzing 'domain_length'. For example does 0.5 mean the short protein domain must be >50% shorter than the long protein domain before it is repported. NULL disables the filter. Default is 0.5.

`AaJCsimCutoff` An numeric (between 0 and 1) indicating the cutoff on Jacard Distance (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 repported as such. This cutoff affect the result of the 'ORF_seq_similarity' analysis. Default is 0.9

`removeNonConseqSwitches`
: A logic indicating whether to remove the comparison of isoforms where no consequences were found (if TRUE) or to keep then (if FALSE). Defaults 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 <isoform_exp> / <gene_exp>.

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 Jacard similarity (JCsim) is calculated. Here JCsim is defined as the length of the alligned regions (omitting gaps) devided by the total combined uniuqe sequence length: JCsim = (length of alligned region w.o gaps) / ((length of sequence a) + (length of sequence b) - (length of alligned 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.

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).
- `tts` : Test transcripts for whether they use different Transcription Termination Site (TTS).
- `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 (meaning that the introns used in one isoform is a subset of the introns used in another isoform).
- 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 analysis. Requires that importCPATanalysis have been used to add external CPAT 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 alligned 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 settingaddAnnotatedORFs=TRUE when creating the switchAnalyzeRlist.
- ORF_length : Test transcripts for differences in length of Open Reading Frames (ORF). Note that this is a less powerfull 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 are different (as described above). Reported as different if the measured JCsim is smaller than ntJCsimCutoff and the length difference of the alligned 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 length of 5' UnTranslated Region (UTR), 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 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. 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 length of 3' UnTranslated Region (UTR), defined as the region from the ORF end to 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 though 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 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.
- domain_length : Test transcripts for differences in the length of domians identified in both isoforms enabling analysis of protein domain truncation. Do however note that a small difference in length is will likely not trunkate the protein domain. The length difference, measureed in AA, must be larger than AaCutoff and AaFracCutoff .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.
- genomic_domain_position : Test transcripts for differences in the genomic position of the domains identified by the Pfam analysis. 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 (and are thereby also affected by removeNoncodinORFs=TRUE in analyzeCPAT).
- 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 removeNoncodinORFs=TRUE in analyzeCPAT).

Value

The supplied switchAnalyzeRlist is returned, but now annotaed with the predicted functional consequences as follows. First a colum 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 refrence to a specific gene in a specific comaprison of conditions. Enables esay handles to integrate data from all the parts of a switchAnalyzeRlist.
- gene_id: The id of the gene which the isoforms compared belongs to. Mathces 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 though of as the ground state - meaning the changes occoure 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 though of as the changed state - meaning the changes occure 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 refrence to a specific isoform in a specific comaprison 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 refrence to a specific isoform in a specific comaprison 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`

Examples

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

# subset for fast runtime
exampleSwitchListAnalyzed <- subsetSwitchAnalyzeRlist(
```

```
    exampleSwitchListAnalyzed,
   exampleSwitchListAnalyzed$isoformFeatures$gene_id %in% sample(exampleSwitchListAnalyzed$isoformFeatures$
)

### 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 extractTo
# 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 consequen
```

`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,
    sourceId
)
```

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 collums: `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 form 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.
`sourceId`	A character stating the origin of the data used to create the switchAnalyzeRlist.

Details

For cufflinks data, use importCufflinksCummeRbund or importCufflinksFiles to prepare the switchAnalyzeRlist. For other RNA-seq assemblies, either uses this constructor or the general-purpose importRdata to create the switchAnalyzeRlist - se 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 annoation for this isoform. The data.frame can contain any colums supplied (enabling addition of user specified columns) but the following columns are nessesary and must be provided:

- `iso_ref` : A unique refrence to a specific isoform in a specific comaprison of conditions. Mainly created to have an easy handle to integrate data from all the parts of a `switchAnalyzeRlist`.
- `gene_ref` : A unique refrence to a specific gene in a specific comaprison 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_value_1` : Expression of <gene_id> in condition_1 (must be normalized)
- `gene_value_2` : Expression of <gene_id> in condition_2 (must be normalized)
- `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_value_1` : Expression of <isoform_id> in condition_1 (must be normalized)
- `iso_value_2` : Expression of <isoform_id> in condition_2 (must be normalized)
- `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>
- `IF1` : The <isoform_id> usage in condtion 1 (given as Isoform Fraction (IF) value)
- `IF2` : The <isoform_id> usage in condtion 2 (given as Isoform Fraction (IF) value)
- `dIF` : The change in isoform usage from condtion_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 condtion 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 condtion 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 pr exon in each isoform. Furthermore it must also have two meta collumns 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 collumns: condition and nrReplicates giving the number of biological (not technical) replicates each condition analyzed. The strings used to conditions the condtions 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 collumns:

- 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
readCufflinks
importCufflinksCummeRbund
importCufflinksFiles

importGTF
importIsoformExpression

Examples

```
#### Make example data
cuffDB <- prepareCuffExample()

### Replicate count matric
isoRepCount <- repCountMatrix(isoforms(cuffDB))
isoRepCount$isoform_id <- rownames(isoRepCount)

### Design matrix
designMatrix <- cummeRbund::replicates(cuffDB)[,c('rep_name','sample_name')]
colnames(designMatrix) <- c('sampleID','condition')

### Genomic Annotation
localAnnotaion <- import(system.file("extdata/chr1_snippet.gtf", package="cummeRbund"))[,c('transcript_id',
colnames(localAnnotaion@elementMetadata)[1] <- 'isoform_id'


### Take a look at the data
head(isoRepCount, 2)

designMatrix

head(localAnnotaion, 3)

### Create switchAnalyzeRlist
aSwitchList <- importRdata(
    isoformCountMatrix=isoRepCount,
    designMatrix=designMatrix,
    isoformExonAnnoation=localAnnotaion,
    showProgress=FALSE
)
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 the a full isoform switch analysis workflow (including external sequence analysis of protein domains (via Pfam), coding potential (via CPAT) and signal peptides (via SignalP)). This is the result of with the isoformSwitchAnalysisPart1 function, performing the external sequence analysis and using with the isoformSwitchAnalysisPart2 to run the rest of the pipeline. Note this is a larger subset to illustrate the combined analysis and that the nucleotide and amino acid sequences normally added to the switchAnalyzeRlist have been removed from the switchAnalyzeRlist to save space (they can easily be added again with the extractSequence function).

Source

This data is a modified subset of a dataset comparing human Embryonic Stemm Cells (hESC) vs induced Pluripotent Cells (iPS) and mature cells (Fibroblast) originally released with the cummeRbund 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 conclusions should be made from it.

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)
```

---

extractCalibrationStatus

Extract Whether P-values were Corrected.

Description

This function allows the user to extract information about whether the p-values were calibrated as described in Ferguson et al or not.

Usage

```
extractCalibrationStatus(
    switchAnalyzeRlist
)
```

Arguments

switchAnalyzeRlist	A `switchAnalyzeRlist` object which have been analyzed with isoformSwitchTest.

Value

A data.frame where it for each comparison was indicated (via a logical) whether the p-value correction was performed or not.

Author(s)

Kristoffer Vitting-Seerup

References

- Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. Mol. Cancer Res. (2017).
- Ferguson JP, Palejev D: P-value calibration fro multiple testing problems in genomics. Stat. Appl. Genet. Mol. Biol. 2014, 13:659-673.

See Also

isoformSwitchTest

Examples

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

# Perfom test
exampleSwitchListAnalyzed <- isoformSwitchTest(exampleSwitchList)

# extract whether p-value callibration was performed
extractCalibrationStatus(exampleSwitchListAnalyzed)
```

---

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,
    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 consequnces should be considered. See details for descripbtion (note it is idencital to the strings used with analyzeSwitchConsequences). 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 (callibrated) 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 indicting whether the summarized results should be plotted. 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 ggplo2 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. 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
extractSwitchSummary

Examples

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

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


# Analyze consequences
consequencesOfInterest <- c(
    'intron_retention',
    'coding_potential',
    'NMD_status',
    'domains_identified'
)
exampleSwitchListAnalyzed <- analyzeSwitchConsequences(
    exampleSwitchListAnalyzed,
    consequencesToAnalyze = consequencesOfInterest
)

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

dim(consequenceSummary)
```

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

`extractExpressionMatrix` *Extract Gene/Isoform Expression Matrix.*

Description

Extract a data.frame with (mean) gene expression, isoform expression or Isoform Fraction values for all conditions (columns) from a switchAnalyzeRlist.

Usage

```
extractExpressionMatrix(
    switchAnalyzeRlist,
    feature='isoformUsage',
    addInfo=FALSE,
    na.rm=TRUE
)
```

Arguments

`switchAnalyzeRlist`	A `switchAnalyzeRlist` object.
`feature`	The feature of which to extract the expression matrix for. Can be ether 'geneExp' for gene expression levels, 'isoformExp' for isoform expression levels or 'isoformUsage' for IF values. Default is 'isoformUsage'.
`addInfo`	A logic indicating whether annotated non-conditional data (such as gene name, PTC status etc.) should be added to the data.frame. Default is FALSE.
`na.rm`	A logic indicating whether rows with NA expression values should be removed. Default is TRUE.

Value

This function returns a data.frame where the first column is the gene/isoform id followed by the mean (if calculated by any of the `import*()` functions) expression/usage in all different conditions (one column pr condition) and if addInfo=TRUE then the additional non-conditional dependent data is added as well.

Author(s)

Kristoffer Vitting-Seerup

References

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

Examples

```
data("exampleSwitchListAnalyzed")

### Gene expression matrix
geneMatrix <- extractExpressionMatrix(exampleSwitchListAnalyzed, feature = 'geneExp')
head(geneMatrix)

# with additional info
geneMatrix <- extractExpressionMatrix(exampleSwitchListAnalyzed, feature = 'geneExp', addInfo = TRUE)
head(geneMatrix)

### Isoform Fraction value expression matrix
ifMatrix <- extractExpressionMatrix(exampleSwitchListAnalyzed, feature = 'isoformUsage')
head(ifMatrix)

# with additional info
ifMatrix <- extractExpressionMatrix(exampleSwitchListAnalyzed, feature = 'isoformUsage', addInfo = TRUE)
head(ifMatrix)
```

---

extractGenomeWideAnalysis

Genome wide Analysis of Changes in Isoform Features

Description

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

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 plottet for each subset of features in that category (e.g with and without signal peptides). This enables a gplobal analysis of isoforms with a common annotation. The annotation considdered are (if added to the switchAnalyzeRlist) coding potential, intron retentions, isoform clas code (Cufflinks/Cuffdiff data only), NMD status, ORFs, protein domains, signal oeptides 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 controled by `featureToExtract`.

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

Usage

```
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 `isoformSwitchTest()`) 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 advailable 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`)
  - `'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 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 isoformSwitchTest - 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. Defualit 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 signicant 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 (***) respecitively. 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 ggplo2 theme with which the plot is made, see ?ggplot2::theme for more info. Default is theme_bw().
plot	A logical indicating whether to generate the plot (if TRUE) not (if FALSE). 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). Default is TRUE.

Details

Changes in isoform usage are measure as the difference in isoform fraction (dIF) values, where isoform fraction (IF) values are calculated as <isoform_exp> / <gene_exp>.

The significance indicated in the plot is based on FDR corrected p-values from the wilcox.test.

The arguments passed to annotationToAnalyze must be a combination of:

- isoform_class_code : Devide transcripts based on differences in the transcript classification provide by cufflinks (only advailable 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 : Devide 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 : Devide transcripts based on pressence intron retentions (and their genomic positions). Require that analyzeIntronRetention have been run.
- ORF : Devide 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 : Devide 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 : Devide 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 : Devide 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 removeNoncodinORFs=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

isoformSwitchTest
analyzeORF
analyzeIntronRetention
analyzeCPAT
analyzePFAM
analyzeSignalP
analyzeSwitchConsequences

Examples

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

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

### Extract the summary statistics of the highly significant cases
subset(symmaryStatistics, symmaryStatistics$wilcoxQval < 0.001)
```

---

extractSequence *Extract nucloetide (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,
    onlySwitchingGenes = TRUE,
```

```
    alpha = 0.05,
    dIFcutoff = 0.1,
    extractNTseq = TRUE,
    extractAAseq = TRUE,
    filterAALength = FALSE,
    removeORFwithStop=TRUE,
    addToSwitchAnalyzeRlist = TRUE,
    writeToFile = TRUE,
    pathToOutput = getwd(),
    outputPrefix='isoformSwitchAnalyzeR_isoform',
    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).

`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 (callibrated) 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 signicant and thereby included in the downstream analysis. This cutoff is analogours 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 sequnce of the transcripts should be extracted (nessesary 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 (nessesary for pfam and SignalIP analysis). The ORF can be annotated with the `analyzeORF` function. Default is TRUE.

`filterAALength`
: A logical indicating whether to filter on the resulting sequences based on their length. This option exist to allows for easier usage of the Pfam and SignalIP web servers which both currently have restrictions on allowed sequence lengths. If enabled AA sequences are filtered to be > 5 AA and < 6001 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.

`removeORFwithStop`
: A logical indicating whether ORFs containing stop codons, definde as * when the ORF nucleotide sequences is translated to the amino acid seqeunce, should be A) removed from the ORF annotation in the switchAnalyzeRlist and B) removed from the sequences added to the switchAnalyzeRlist and/or writen to

fasta files. This is only nessesary 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' depnding on the file type. Default is 'isoformSwitchAnalyzeR_isoform' (thereby creating the 'isoformSwitchAnalyzeR_isoform_nt.fasta' and 'isoformSwitchAnalyzeR_isoform_AA.fasta' files).

`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 <isoform_exp> / <gene_exp>.

The BSGenome object are loaded as seperate 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 annoation 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, eg. 'chr1') or not (Ensembl style, eg. '1').

The two fasta files outputted by this function (if `writeToFile=TRUE`) can be used as input to amongst 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
- `SignalIP` : Prediction of Signal Peptides, which can be run either locally or via their webserver http://www.cbs.dtu.dk/services/SignalP/

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

Value

If `writeToFile=TRUE` one fasta file pr sequence type (controled via `extractNTseq` and `extractAAseq`) are written to the folder indicated by `pathToOutput`. If `addToSwitchAnalyzeRlist=TRUE` the sequences are added to the `switchAnalyzeRlist` as respectively `DNAStringSet` 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.

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).
- `CPAT` : Wang et al. CPAT: Coding-Potential Assessment Tool using an alignment-free logistic regression model. Nucleic Acids Res. 2013, 41:e74.
- `Pfam` : Finn et al. The Pfam protein families database. Nucleic Acids Research (2014) Database Issue 42:D222-D230
- `SignalP` : Petersen et al. SignalP 4.0: discriminating signal peptides from transmembrane regions. Nature Methods, 8:785-786, 2011

See Also

`switchAnalyzeRlist`
`isoformSwitchTest`
`analyzeORF`
`analyzeCPAT`
`analyzePFAM`
`analyzeSignalP`

Examples

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

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

# 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)
```

`extractSwitchSummary` *Summarize Isoform Switches test Result.*

Description

Summarize the number of switching isoforms/genes identifieds.

Usage

```
extractSwitchSummary(
    switchAnalyzeRlist,
    filterForConsequences=FALSE,
    alpha=0.05,
    dIFcutoff = 0.1,
   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 (calibrated) 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%).

`includeCombined`
: A logic indicating whether a combined summary accorss 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).
- Ferguson JP, Palejev D: P-value calibration fro multiple testing problems in genomics. Stat. Appl. Genet. Mol. Biol. 2014, 13:659-673.

See Also

preFilter
isoformSwitchTest
extractTopSwitches
analyzeSwitchConsequences

Examples

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

# Perfom test
exampleSwitchListAnalyzed <- isoformSwitchTest(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 TURE) or top swiching genes (if FALSE). Default is TRUE (extract isoforms).

`filterForConsequences`
: A logical indicating whether to filter for genes with functional consequences. Requtires that analyzeSwitchConsequences() have been run on the switchAnalyzeRlist. Default is FALSE.

`alpha`
: The cutoff which the (callibrated) fdr correct p-values must be smaller than for calling significant switches. Defualit 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 NA to return all significant results. 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 to the top n features are defined be smallest q-values (if `sortByQvals=TURE`) 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 `sum(abs(dIF))` for each gene. If set to NA no sorting is perfomed. 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).
- Ferguson JP, Palejev D: P-value calibration fro multiple testing problems in genomics. Stat. Appl. Genet. Mol. Biol. 2014, 13:659-673.

See Also

`preFilter`
`isoformSwitchTest`
`analyzeSwitchConsequences`

Examples

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

# Perfom test
exampleSwitchListAnalyzed <- isoformSwitchTest(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)
```

---

importCufflinksCummeRbund

Import CuffDiff (Cufflinks) Data Into R

Description

This function uses the SQL backend generated by `readCufflinks()` (from the 'cummeRbund' package) to extract all the data needed for a full analysis with IsoformSwitchAnalyzeR. The data is returned in a switchAnalyzeRlist object.

Usage

```
importCufflinksCummeRbund(
    cuffDB,
    fixCufflinksAnnotationProblem = TRUE,
    addCufflinksSwichTest=TRUE,
    quiet=FALSE
)
```

Arguments

`cuffDB`	A CuffSet created by cummeRbund::readCufflins(). Note that the sql database must be generated with the GTF file (meaning the gtfFile arugument must be used).
`fixCufflinksAnnotationProblem`	A logic indicating whether to fix the problem with Cufflinks gene symbol annotation. Please see the details for additional information. Default is TRUE

addCufflinksSwichTest
: A logic indicating whether to use CuffDiff's switch test (at promoter level) as the evidence for the switching occurring within the parent gene_id. See details for more information. Default is TRUE

quiet
: 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 specificly 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 occuring 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 (e.g. to 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.

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 cuffdiff's 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 amongst 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

readCufflinks
createSwitchAnalyzeRlist
importCufflinksFiles
preFilter

Examples

```
### Build test SQL-database from cummeRbund test data - typically done with readCufflinks()
library('cummeRbund')
cuffDB <- prepareCuffExample()

### Generate
aSwitchList <- importCufflinksCummeRbund(cuffDB)
aSwitchList
```

---

importCufflinksFiles *Import CuffDiff (Cufflinks) Data Into R*

Description

This function enables users to run Cufflinks/Cuffdiff on Galaxy and then afterwards import the result into R for post analysis with isoformSwitchAnalyzeR. The user just has to download (some of) the Cuffdiff result files from galaxy and input the paths to this function. The data is then imported into R, massaged and returned as a switchAnalyzeRlist enabling a full analysis with IsoformSwitchAnalyzeR.

Usage

```
importCufflinksFiles(
    pathToGTF,
    pathToGeneDEanalysis,
    pathToIsoformDEanalysis,
    pathToGeneFPKMtracking,
    pathToIsoformFPKMtracking,
    pathToIsoformReadGroupTracking,
    pathToSplicingAnalysis=NULL,
    pathToReadGroups,
    pathToRunInfo,
    fixCufflinksAnnotationProblem=TRUE,
    quiet=FALSE
)
```

Arguments

pathToGTF	A string indicating the path to the GTF file used as input to Cuffdiff file (downloaded from fx galaxy).
pathToGeneDEanalysis	A string indicating the path to the file "gene differential expression testing" file (downloaded from fx galaxy).

`pathToIsoformDEanalysis`
: A string indicating the path to the file "transcript differential expression testing" file (downloaded from fx galaxy).

`pathToGeneFPKMtracking`
: A string indicating the path to the file "gene FPKM tracking" file (downloaded from fx galaxy).

`pathToIsoformReadGroupTracking`
: A string indicating the path to the file "isoform read group tracking" file (downloaded from fx galaxy).

`pathToIsoformFPKMtracking`
: A string indicating the path to the file "transcript FPKM tracking" file (downloaded from fx galaxy).

`pathToSplicingAnalysis`
: A string indicating the path to the file "splicing differential expression testing" file (downloaded from fx galaxy).. Only needed if the splicing analysis should be added. Default is NULL (not added).

`pathToReadGroups`
: A string indicating the path to the file "Read groups" file (downloaded from fx galaxy).

`pathToRunInfo`
: A string indicating the path to the file "Run details" file (downloaded from fx galaxy).

`fixCufflinksAnnotationProblem`
: A logic indicating whether to fix the problem with Cufflinks gene symbol annotation. Please see the details for additional information. Default is TRUE.

`quiet`
: 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 specificly 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.

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 cuffdiff's 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 amongst 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
importCufflinksCummeRbund
preFilter

Examples

```
### Use the files from the cummeRbund example data
testSwitchList <- 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 = "cummeRbu
   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
)
testSwitchList
```

---

importGTF *Import Transcripts from a GTF file into R*

Description

Function for importing a GTF 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(
    pathToGTF,
    addAnnotatedORFs=FALSE,
    onlyConsiderFullORF=FALSE,
    removeNonConvensionalChr=FALSE,
    PTCDistance=50,
    quiet=FALSE
)
```

Arguments

pathToGTF	A string indicating the full path to the GTF that should be imported.
addAnnotatedORFs	A logic indicating whether the ORF from the GTF should be added to the switchAnalyzeRlist. This ORF is defined as the regions annoated as 'CDS' in the 'type' collumn (collumn 3). Default is FALSE.
onlyConsiderFullORF	A logic indicating whether the ORFs added should only be added if they are fully annotated. Here fully annoated 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 a '_'. These regions are typically used to annotate regions that cannot be assocaiated 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.
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' annoated in the 'type' column (column 3).

Value

A switchAnalyzeRlist containing a all the gene and transcript information as well as the transcipt 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. Mathces the 'isoform_id' entry in the 'isoformFeatures' entry of the switchAnalyzeRlist

- `orfTransciptStart`: The start position of the ORF in transcript cooridnats, here defined as the position of the 'A' in the 'AUG' start motif.
- `orfTransciptEnd`: The end position of the ORF in transcript coordinats, here defined as the last nucleotide before the STOP codon (meaning the stop codon is not included in these coordinates).
- `orfTransciptLength`: 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 cooridnats, here defined as the the position of the 'A' in the 'AUG' start motif.
- `orfEndGenomic`: The end position of the ORF in genomic coordinats, 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 junciton.

NA means no information was advailable 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

```
# Import gtf file from cummeRbund package

aSwitchList <- importGTF(pathToGTF=system.file("extdata/chr1_snippet.gtf", package = "cummeRbund"))
aSwitchList
```

`importIsoformExpression` *Import expression data from Kallisto, Salmon or RSEM into R.*

Description

A general-purpose import function which imports isoform expression data from Kallisto, Salmon or RSEM 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 reccomended 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 inter-library normalization (via edgeR) of the abundance estimates. Note that the pattern argument allows import of only a subset of files.

Usage

```
importIsoformExpression(
    parentDir,
    calculateCountsFromAbundance=TRUE,
    interLibNormTxPM=TRUE,
    normalizationMethod='TMM',
    pattern='',
    invertPattern=FALSE,
    ignore.case=FALSE,
    showProgress = TRUE,
    quiet = FALSE
)
```

Arguments

`parentDir`	Parrent directory where each quantified sample is in a sub-directory.
`calculateCountsFromAbundance`	A logic indicating whether to generate estimated counts using the estimated abundances. Recomended as it will incooperate the bias correction algorithms into the analysis. Default is TRUE.
`interLibNormTxPM`	A logic indicating whether to apply an inter-library normalization (via edgeR) to the imported abundances. Recomended as it allow comparison of abundances between samples. 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`	Character string containing a regular expression for which files to import (applied to full path). Defailt is "" corresponding to all. See base::grepl for more details.
`invertPattern`	Logical. If TRUE return indices or values for elements that do not match..
`ignore.case`	if FALSE, the pattern matching is case sensitive and if TRUE, case is ignored during matching.

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

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 seperate sub-directory.

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.

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

The inter-library normalization of TxPM values is performed as a three step process: First the effective counts are calculated from the abundances using the library specific effective isoform lengths. The effective counts are then suppled to edgeR which calculates the normalization factors nesseary. Lastly the calculated normalization factors are applied to the imported TxPM values.

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 measrung the relative abundance of a specic transcript.

Transcripts Per Million values are abbreviated to TPM by RSEM, Kallisto and Salmon but will here refered to as TxPM to avoid confusion with the commenly used Tags Per Million (which have been around for way longer). TxPM is an equivilent to RPKM/FPKM except it has been adjusted for as all the biases being modeled by the tools used for the quantifictation 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

`importRdata` *Create SwitchAnalyzeRlist From Standard R Objects*

Description

A general-purpose interface to constructing a switchAnalyzeRlist from Standard R objects containing expression and annoatation information. The data needed for this function are

- `1`: Normalized biological replicate isoform expression data
- `2`: 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 where the information can be found.
- `3`: A design matrix indicating which samples belong to which condition

Furthermore it's possible to specify which comparisons to make using the `comparisonsToMake` (default is all possible pairwise of the once indicated by the design matrix).

Usage

```
importRdata(
    isoformCountMatrix,
    isoformRepExpression,
    designMatrix,
    isoformExonAnnoation,
    comparisonsToMake=NULL,
    addAnnotatedORFs=FALSE,
    onlyConsiderFullORF=FALSE,
    removeNonConvensionalChr=FALSE,
    PTCDistance=50,
    foldChangePseudoCount=0.01,
    showProgress=TRUE,
    quiet=FALSE
)
```

Arguments

`isoformCountMatrix`
: A data.frame with unfiltered biological (not technical) replicate isoform (estimated) fragment counts. Must have a column called 'isoform_id' with the isoform_id that matches the isoform_id in `isoformExonAnnoation`. The name of the columns must match the sample names in the `designMatrix` argument and contain the estimated counts.

`isoformRepExpression`
: Optional but recommended: A data.frame with unfiltered normalized biological (not technical) replicate isoform expression. Ideal for supplying quantification measured in Transcripts Per Million (TxPM) or RPKM/FPKM. Must have a column called 'isoform_id' with the isoform_id. The name of the expression columns must match the sample names in the `designMatrix` argument. If not supplied RPKM values are calcualted 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 these two collums:

- 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 form 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). Additional co-factors can only be analyzed with isoformSwitchTestDRIMSeq.

isoformExonAnnoation

Can either be:

- 1: A string indicating the full path to the GTF file which have been quantified. If supplied the exon structure and isoform annotation will be obtained from the GTF file.
- 2: A GRange object (see ?GRanges) containing one entry per exon per isoform with the genomic coordinat of that isoform. This GRange should furthmore 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 columns called 'gene_name' with the corresponding gene names is also added. If not supplied gene_name will be annotated as NA.

comparisonsToMake

A data.frame 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$condition column. If not supplied all pairwise (unique nondirectional) comparisons of the conditions given in designMatrix$condition are created.

addAnnotatedORFs

Only used if a GTF file is supplied to isoformExonAnnoation. 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 FALSE.

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 onlyConsiderFullORF=TRUE. Default is FALSE.

removeNonConvensionalChr

A logic indicating whether non-conventional chromosomes, here defined as chromosome names containing a '_'. These regions are typically used to annotate regions that cannot be assocaiated 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'). This argument is only considered if a GTF file was supplied to isoformExonAnnoation. Default is FALSE.

`PTCDistance` Only used if a GTF file is supplied to `isoformExonAnnoation` 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

`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

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 <isoform_exp> / <gene_exp>.

Value

A SwitchAnalyzeRlist containing the data supplied wrap into the SwitchAnalyzeRlist format.

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. Mathces the 'isoform_id' entry in the 'isoformFeatures' entry of the switchAnalyzeRlist
- `orfTransciptStart`: The start position of the ORF in transcript cooridnats, here defined as the position of the 'A' in the 'AUG' start motif.
- `orfTransciptEnd`: The end position of the ORF in transcript coordinats, here defined as the last nucleotide before the STOP codon (meaning the stop codon is not included in these coordinates).
- `orfTransciptLength`: 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 cooridnats, here defined as the the position of the 'A' in the 'AUG' start motif.
- `orfEndGenomic`: The end position of the ORF in genomic coordinats, 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 junciton.

NA means no information was advailable 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

```
### Construct data needed from example data in cummeRbund package
### (The recomended way of analyzing Cufflinks/Cuffdiff datat is via importCufflinksCummeRbund()
### This is jus an easy way to get some example data ).
cuffDB <- prepareCuffExample()

isoRepCount <- repCountMatrix(isoforms(cuffDB))
isoRepCount$isoform_id <- rownames(isoRepCount)

### Design matrix
designMatrix <- cummeRbund::replicates(cuffDB)[,c('rep_name','sample_name')]
colnames(designMatrix) <- c('sampleID','condition')

localAnnotaion <- import(system.file("extdata/chr1_snippet.gtf", package="cummeRbund"))[,c('transcript_id',
colnames(localAnnotaion@elementMetadata)[1] <- 'isoform_id'


### Take a look at the data
head(isoRepCount)
head(designMatrix)
head(localAnnotaion)

### Create switchAnalyzeRlist
aSwitchList <- importRdata(
    isoformCountMatrix=isoRepCount,
    designMatrix=designMatrix,
    isoformExonAnnoation=localAnnotaion
)
aSwitchList
```

`isoformSwitchAnalysisCombined` *Isoform Switch Analysis Workflow: Extract, Annoate and Visualize all Significant Isoform Swtiches*

Description

This high-level function takes a CuffSet object or a pre-existing switchAnalyzeRlist as input. If the input is a CuffSet a switchFindeRList is created and else the function uses the provieded switchAnalyzeRlist.

Then isoform switches are identified, annotated with ORF and intron retion. 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 domians (Pfam), coding potential (CPAT) or signal peptides (SignalP) should be incorporated please use the combination of `isoformSwitchAnalysisPart1` and `isoformSwitchAnalysisPart2` instead.

Usage

```
isoformSwitchAnalysisCombined(
    input,
    alpha=0.05,
    dIFcutoff = 0.1,
    switchTestMethod='DRIMSeq',
    calibratePvalues=TRUE,
    n=NA,
    pathToOutput=getwd(),
    overwriteORF=FALSE,
    outputSequences=FALSE,
    genomeObject,
    orfMethod='longest',
    cds=NULL,
   consequencesToAnalyze=c('intron_retention','ORF_seq_similarity','NMD_status'),
    fileType='pdf',
    asFractionTotal=FALSE,
    outputPlots=TRUE,
    quiet=FALSE
)
```

Arguments

`input`	Either a `CuffSet` created by readCufflins() or a `switchAnalyzeRlist` object.
`alpha`	The cutoff which the (callibrated) fdr correct p-values must be smaller than for calling significant switches. Defualit 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 signicant and thereby included in the downstream analysis. This cutoff is analogours 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%).

`switchTestMethod`
: A sting indicating which statistical method should be used for testing differential isoform usage. The following options are avilable:
  - `'DRIMSeq'` : Uses the DRIMSeq package to perform the statiscal test. See isoformSwitchTestDRIMSeq. Default
  - `'IsoformSwitchAnalyzeR'` : Uses the statistical test implemented in IsoformSwitchAnalyzeR. See isoformSwitchTest.
  - `'none'` : No statistical test is performed. Should only be used if a test have already been performed and should not be overwritten (e.g when importing cuffdiff data).

`calibratePvalues`
: Only applicable if `switchTestMethod='IsoformSwitchAnalyzeR'`.A logic indicating whether the p-values should be calibrated accordin to the methods in Ferguson et al (2014), this will only be done if the p-value distribution is very skewed (if the estimated sigma^2 values is < 0.9) as suggested by the authors. See details for more information. Defualt is TRUE.

`n`
: The number of top genes (after filtering and sorted according to `sortByQvals`) that should be saved to each subfolder indicated by `splitConditions, splitFunctionalConsequen` Use NA to create all. Default is NA (all).

`pathToOutput`
: A path to the folder in which the plots should be made. Default is working directory (getwd()).

`overwriteORF`
: A logical indicating whether to overwrite the ORF analysis already stored in the supplied switchAnalyzeRlist. Default is FALSE.

`outputSequences`
: A logic indicating whether transcript nucleotide and amino acid sequences should be outputtet to outputDestination. Default is TRUE.

`genomeObject`
: A `BSgenome` object (for example Hsapiens for Homo sapiens).

`orfMethod`
: A string indicating which of the 3 ORF identification methods should be used. The methods are:
  - `longest` : Identifies the longest ORF in the transcript. This approach is similar to what the CPAT tool uses in it's analysis of coding potential
  - `longestAnnotated` : Identifies the longest ORF downstream of an annotated translation start site (supplied via the `cds` argument)
  - `mostUpstreamAnnoated` : Identifies the ORF downstream of the most upstream overlapping annotated translation start site (supplied via the `cds` argument)

  Default is `longest`.

`cds`
: A CDSSet object containing annotated coding regions, see ?CDSSet and ?getCDS for more information. Only nessesary if \'orfType\' arguments is \'longestAnnotated\' or \'mostUpstreamAnnoated\'.

`consequencesToAnalyze`
: A vector of strings indicating what type of functional consequences to analyze. Note 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','ORF_seq_similarity','NMD_status') (corresponding to analyze: intron retention, ORF AA sequence similarity and NMD status).

`fileType`
: A string indicating which file type is generated. Available options are \'pdf\' and \'png\'. Default is pdf.

`asFractionTotal`
: A logic indicating whether the number of consequences should be calculated as numbers (if FALSE) or as a fraction of the total number of switches in the plot summarizing general consequences of all the isoform swithces. Default is FALSE.

`outputPlots`
: A logic indicating whether all isoform switches as well as the summary of functional consequences should be outputted 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. If necessary import the data (see importCufflinksCummeRbund)
2. Remove non-expressed isoforms and single-isoform genes (see preFilter)
3. predict swithces (only if switches is not already annotated, see isoformSwitchTest)
4. Analyzing the isoforms for open reading frames (ORFs, see analyzeORF)
5. Output fasta files containing the nucleotide and amino acid sequences which enables external sequence analysis with CPAT, Pfam and SignalP (see extractSequence)
6. Predict functional consequences of switching (see analyzeSwitchConsequences)
7. Ouput Isoform Switch Analysis plots for all genes with a signicant switch (see switchPlot)
8. Ouput 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 off 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`
`isoformSwitchTest`
`analyzeORF`
`extractSwitchSummary`
`analyzeSwitchConsequences`
`switchPlotTopSwitches`

Examples

```
data("exampleSwitchList")
exampleSwitchList

library(BSgenome.Hsapiens.UCSC.hg19)
exampleSwitchListAnalyzed <- isoformSwitchAnalysisCombined(
    input=exampleSwitchList,
    genomeObject = Hsapiens,
    dIFcutoff = 0.4,         # Set high for short runtime in example data
   outputSequences = FALSE, # keeps the function from outputting the fasta files from this example
   outputPlots = FALSE     # keeps the function from outputting the Isoform Switch AnalyzeR Plots from this ex
)

exampleSwitchListAnalyzed
```

---

isoformSwitchAnalysisPart1

Isoform Switch Analysis Workflow Part 1: Extract Isoform Switches and Their Sequences

---

Description

This high-level function takes either a CuffSet object or a pre-existing switchAnalyzeRlist as input. If the input is a CuffSet object a switchAnalyzeRlist is created or else the function uses the provieded switchAnalyzeRlist. Then isoform switches are predicted (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. Thes sequences are both saved to external files and added to the switchAnalyzeRlist to enable external and internal sequence analysis respectively.

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(
    input,
    alpha = 0.05,
    dIFcutoff = 0.1,
    switchTestMethod='DRIMSeq',
    calibratePvalues=TRUE,
    orfMethod = "longest",
    genomeObject,
    cds = NULL,
    pathToOutput = getwd(),
    outputSequences = TRUE,
    overwriteORF=FALSE,
    quiet=FALSE
)
```

Arguments

`input`	Either a `CuffSet` created by readCufflins() or a `switchAnalyzeRlist`.
`alpha`	The cutoff which the (calibrated) fdr correct p-values must be smaller than for calling significant switches. Defualit 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%).
`switchTestMethod`	A sting indicating which statistical method should be used for testing differential isoform usage. The following options are avilable: • `'DRIMSeq'` : Uses the DRIMSeq package to perform the statiscal test. See isoformSwitchTestDRIMSeq. Default • `'IsoformSwitchAnalyzeR'` : Uses the statistical test implemented in IsoformSwitchAnalyzeR. See isoformSwitchTest. • `'none'` : No statistical test is performed. Should only be used if a test have already been performed and should not be overwritten (e.g when importing cuffdiff data).
`calibratePvalues`	Only applicable if `switchTestMethod='IsoformSwitchAnalyzeR'`.A logic indicating whether the p-values should be calibrated accordin to the methods in Ferguson et al (2014), this will only be done if the p-value distribution is very skewed (if the estimated sigma^2 values is < 0.9) as suggested by the authors. See details for more information. Defualt is TRUE.
`orfMethod`	A string indicating which of the 4 ORF identification methods should be used. The methods are: • `longest` : Identifies the longest ORF in the transcript. This approach is similar to what the CPAT tool uses in it's analysis of coding potential • `longestAnnotated` : Identifies the longest ORF downstream of an annoated translation start site (supplied via the `cds` argument) • `mostUpstreamAnnoated` : Identifies the ORF downstream of the most unstream overlapping annoated translation start site (supplied via the `cds` argument) Default is `longest`.
`genomeObject`	A `BSgenome` object (for example Hsapiens for Homo sapiens).
`pathToOutput`	A path to the folder in which the plots should be made. Default is working directory (getwd()).
`cds`	A CDSSet object containing annoated coding regions, see ?CDSSet and ?getCDS for more information. Only necessary if \'orfType\' arguments is \'longestAnnotated\' or \'mostUpstreamAnnoated\'.
`overwriteORF`	A logical indicating whether to overwrite the ORF analysis already stored in the supplied switchAnalyzeRlist. Default is FALSE.
`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. If necessary import the data (see importCufflinksCummeRbund) and create the switchAnalyzeRlist.
2. Remove non-expressed isoforms and single-isoform genes (see preFilter)
3. Predict isoform switches unless `switchTestMethod` is set to 'none'.
4. If no ORFs are annotated the isoforms are analyzed for open reading frames (ORFs, see analyzeORF)
5. 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).

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` is 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
isoformSwitchTest
analyzeORF
extractSequence

Examples

```
data("exampleSwitchList")
exampleSwitchList

library(BSgenome.Hsapiens.UCSC.hg19)
exampleSwitchList <- isoformSwitchAnalysisPart1(
    input=exampleSwitchList,
    genomeObject = Hsapiens,
    dIFcutoff = 0.4,        # Set high for short runtime in example data
   outputSequences = FALSE # keeps the function from outputting the fasta files from this example
)

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 extrenal sequence analysis supplied (if any), then proceeds to annotate intron ration. 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(
    switchAnalyzeRlist,
    alpha=0.05,
    dIFcutoff = 0.1,
    n=NA,
    codingCutoff,
    removeNoncodinORFs,
    pathToCPATresultFile=NULL,
    pathToPFAMresultFile=NULL,
    pathToSignalPresultFile=NULL,
   consequencesToAnalyze=c('intron_retention','coding_potential','ORF_seq_similarity','NMD_statu
    pathToOutput=getwd(),
    fileType='pdf',
    asFractionTotal=FALSE,
    outputPlots=TRUE,
    quiet=FALSE
)
```

Arguments

`switchAnalyzeRlist`	A `switchAnalyzeRlist` object as produced by createSwitchAnalyzeRlist
`alpha`	The cutoff which the (callibrated) fdr correct p-values must be smaller than for calling significant switches. Defualit 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 signicant and thereby included in the downstream analysis. This cutoff is analogours 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 top genes (after filtering and sorted according to `sortByQvals`) that should be saved to each subfolder indicated by `splitConditions, splitFunctionalConsequen` Use NA to create all. Default is NA (all).
`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.

`removeNoncodinORFs`
: A logic indicating wether 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 dowload the tab-delimited file from the bottom of the result page and give that as input, else simply supply the result file.

`pathToPFAMresultFile`
: Path to the PFAM result file. 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 to a file. This file is then used as input to the function.

`pathToSignalPresultFile`
: Path to the summary SignalP result file. 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 to a file. This file is then used as input to the function.

`consequencesToAnalyze`
: A vector of strings indicating what type of functional consequences to analyze. Do note that there is bound to be some differences beteen 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_similar
(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 (getwd()).

`fileType`
: A string indicating which file type is generated. Available options are \'pdf\' and \'png\'. Default is pdf.

`asFractionTotal`
: A logic indicating whether the number of consequences should be calculated as numbers (if FALSE) or as a fraction of the total number of switches in the plot summarizing general consequences of all the isoform swithces. Default is FALSE.

`outputPlots`
: A logic indicating whether all isoform switches as well as the summary of functional consequences should be ouputted 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 second part of a Isoform Analysis Workflow by:

1. Adding external sequence analysis (see analyzeCPAT, 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 ouputs

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 genes with significant isoforms switches
3. A plot summarizing the overall consequences off all the isoform switchces.

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
analyzePFAM
analyzeSignalP
extractSwitchSummary
analyzeSwitchConsequences
switchPlotTopSwitches

Examples

```
data("exampleSwitchListIntermediary")
exampleSwitchListAnalyzed <- isoformSwitchAnalysisPart2(
    switchAnalyzeRlist       = exampleSwitchListIntermediary,
    dIFcutoff                = 0.4,   # Set high for short runtime in example data
   pathToCPATresultFile    = system.file("extdata/cpat_results.txt", package = "IsoformSwitchAnalyzeR"),
   pathToPFAMresultFile    = system.file("extdata/pfam_results.txt", package = "IsoformSwitchAnalyzeR"),
   pathToSignalPresultFile  = system.file("extdata/signalP_results.txt", package = "IsoformSwitchAnalyzeR"),
    codingCutoff             = 0.364,
    removeNoncodinORFs       = TRUE,  # Because ORF was predicted de novo
   outputPlots           = FALSE  # keeps the function from outputting the plots from this example
)
```

`isoformSwitchTest` *Statistical Test of Isoform Switching.*

Description

This function performs a statistical test (see details) for each isoforms (isoform resolution) and conditions stored in the `switchAnalyzeRlist` object.

Usage

```
isoformSwitchTest(
    switchAnalyzeRlist,
    alpha=0.05,
    dIFcutoff = 0.1,
    reduceToSwitchingGenes = TRUE,
    calibratePvalues=TRUE,
    showProgress=FALSE,
    quiet=FALSE
)
```

Arguments

`switchAnalyzeRlist`
: A `switchAnalyzeRlist` object.

`alpha`
: The cutoff which the (calibrated) fdr correct p-values must be smaller than for calling significant switches. Defuault 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%).

`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. Default is TRUE.

`calibratePvalues`
: A logic indicating whether the p-values should be calibrated according to the methods in Ferguson et al (2014), this will only be done if the p-value distribution is very skewed (if the estimated sigma^2 values is < 0.9) as suggested by the authors. See details for more information. 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

Changes in isoform usage are measure as the difference in isoform fraction (dIF) values, where isoform fraction (IF) values are calculated as <isoform_exp> / <gene_exp>.

The idea behind test implemented in `isoformSwitchTest` can be explained as a three step process:

- `1` : Use the uncertainty (e.g. variance) of gene and isoform expression estimats (obtained via biological replicates) to estimate the uncertainty of the isoform fraction (e.g. the variance of the IF values).
- `2` : Use the uncertainty of the IF estimate to statistically test the validity of the changes in isoform usage between conditions.
- `3` : (calibrate p-values and) correct for multiple testing

The procedure implemented here to esitmating the uncertainty of IF values (e.g. the variance of the IF values) is only robust when the gene expression is not close to zero. When the gene expression is to close to zero the variance estimate becomes untrustworthy. To prevent this we have implemented a hardcoded filter that only allows estimation of the IF variance (and thereby testing of an isoform) if the 95% confidence intervals (CI) of the gene expression (in both conditions) is larger than zero. The 95% CI is calculated via the t-distribution thereby taking sample size into account.

The p-value calibration, if enabled, is only performed if the sigma^2 (sigma squared) estimated from the top 50% expressed data is smaller than 0.9 in accordance with the guidelines suggested by the author.

For full description please see the methods section of Vitting-Seerup et al 2017.

Value

A `switchAnalyzeRlist` where the following have been modified:

- `1`: Two collumns, `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.
- `2`: A `data.frame` containing the details of the analysis have been added (called 'isoformSwitchAnalysis').

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

- `iso_ref` : A unique refrence to a specific isoform in a specific comaprison of conditions. Enables easy handles to integrate data from all the parts of a `switchAnalyzeRlist`.
- `gene_ref` : A unique refrence to a specific gene in a specific comaprison of conditions. Enables esay 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_id`: The id of the gene which the isoform_id belongs to. Matches the 'gene_id' entry in the 'isoformFeatures' entry of the switchAnalyzeRlist
- `condition_1`: The first condition of the comparison. Should be though of as the ground state - meaning the changes occure 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 though of as the changed state - meaning the changes occure from condition_1 to condition_2. Matches the 'condition_2' entry in the 'isoformFeatures' entry of the switchAnalyzeRlist
- nrReplicates_1: The number of biological replicates in condition_1. Matches the 'nrReplicates_1' entry in the 'isoformFeatures' entry of the switchAnalyzeRlist
- nrReplicates_2: The number of biological replicates in condition_2. Matches the 'nrReplicates_2' entry in the 'isoformFeatures' entry of the switchAnalyzeRlist
- IF1: The isoform fraction value (IF) of the isoform_id in condition_1. Matches the 'IF1' entry in the 'isoformFeatures' entry of the switchAnalyzeRlist
- IF2: The isoform fraction value (IF) of the isoform_id in condition_2. Matches the 'IF2' entry in the 'isoformFeatures' entry of the switchAnalyzeRlist
- IF_var_1: The variance of IF1 measured from the nrReplicates_1 biological replicates. Matches the 'IF_var_1' entry in the 'isoformFeatures' entry of the switchAnalyzeRlist
- IF_var_2: The variance of IF2 measured from the nrReplicates_2 biological replicates. Matches the 'IF_var_2' entry in the 'isoformFeatures' entry of the switchAnalyzeRlist
- dIF: The difference in the IF values measured as IF2-IF1 meaning that postive values means the isoform is used more in condition_2 and vice versa. Matches the 'dIF' entry in the 'isoformFeatures' entry of the switchAnalyzeRlist
- dIF_std_err: The standard error of the dIF value.
- t_statistics: The test t-statistics. See Vitting-Seerup et al 2017 for details.
- deg_free: The degrees of freedome used to calculate the p_value of the statistical test. See Vitting-Seerup et al 2017 for details.
- p_value: The raw p-value from the statistical test implemented in IsoformSwitchAnalyzeR. See Vitting-Seerup et al 2017 for details.
- calibrated_p_values: The transformed/callibrated p-values. See Ferguson et al 2014 for more details. Will only be calculated if calibratePvalues=TRUE and the sigma^2 (sigma squared) estimated from the top 50% expressed data is smaller than 0.9 in accordance with the guidelines suggested by the author. If NA the callibarion was not performed.
- iso_switch_q_value: The FDR corrected p-values. If the p-value callibration was perfomed these are naturally the FDR corrected callibrated p-values.
- gene_switch_q_value: The gene-level evidence for an isoform swich occuring. Defined as the smallest iso_switch_q_value of the isoforms belonging to the gene_id.

Author(s)

Kristoffer Vitting-Seerup

References

- Vitting-Seerup et al. The Landscape of Isoform Switches in Human Cancers. Mol. Cancer Res. (2017).
- Ferguson JP, Palejev D: P-value calibration fro multiple testing problems in genomics. Stat. Appl. Genet. Mol. Biol. 2014, 13:659-673.

See Also

preFilter
extractSwitchSummary
extractTopSwitches

Examples

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

# Perfom test
exampleSwitchListAnalyzed <- isoformSwitchTest(exampleSwitchList)

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

# extract whether p-value callibration was performed
extractCalibrationStatus(exampleSwitchListAnalyzed)
```

---

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,
    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 (calibrated) fdr correct p-values must be smaller than for calling significant switches. Defuault 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 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 (probablt 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 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. 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 smalles 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 multipe of the dm*() functions so if you cange a paramter for one function you migth 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 multipe of the dm*() functions so if you cange a paramter for one function you migth 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 multipe of the dm*() functions so if you cange a paramter for one function you migth 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 combiantion 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 condtions it is probably better to remove particular comparisons or make two seperate analysis (eg. Brain vs Brain cancer vs liver vs liver cancer should probably be analyzed as two seperate switchAnalyzeRlists whereas WT vs KD1 vs KD2 should be one switchAnalyzeRlists). Note that this is not a problem if the test is performed with `isoformSwitchTest()`.

The result of the testIntegration (see arugments 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 belive 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 leve FDR adjusted P-values is smaller than or eual 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 to many lowly expressed transcripts are included the gene precition cannot be calculcated. Therefore if you think to 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 unnormal.

Value

A `switchAnalyzeRlist` where the following have been modified:

- 1: Two collumns, `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 'isoformSwitchAnalysis').

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

- `iso_ref` : A unique refrence to a specific isoform in a specific comaprison of conditions. Enables easy handles to integrate data from all the parts of a `switchAnalyzeRlist`.

- `gene_ref` : A unique refrence to a specific gene in a specific comaprison of conditions. Enables esay 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

See Also

`preFilter`
`extractSwitchSummary`
`extractTopSwitches`
`dmPrecision`
`dmFit`
`dmTest`

Examples

```
### Example data
cuffDB <- prepareCuffExample()
isoRepCount <- repCountMatrix(isoforms(cuffDB))
isoRepCount$isoform_id <- rownames(isoRepCount)

designMatrix <- cummeRbund::replicates(cuffDB)[,c('rep_name','sample_name')]
colnames(designMatrix) <- c('sampleID','condition')

localAnnotaion <- import(system.file("extdata/chr1_snippet.gtf", package="cummeRbund"))[,c('transcript_id',
colnames(localAnnotaion@elementMetadata)[1] <- 'isoform_id'

### Make swtichAnalayzeRlist
switchAnalyzeRlist <- importRdata(
    isoformCountMatrix=isoRepCount,
    designMatrix=designMatrix,
    isoformExonAnnoation=localAnnotaion
)
```

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

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

# extract summary of number of switching features
extractSwitchSummary(switchAnalyzeRlistAnalyzed)
```

`isoformToGeneExp` *Summarize isoform expression to gene level expression.*

Description

Extract a data.frame with (mean) gene expression, isoform expression or Isoform Fraction values for all conditions (columns) from a switchAnalyzeRlist.

Usage

```
isoformToGeneExp(
    isoRepExpWithGeneId,
    showProgress = TRUE,
    quiet = FALSE
)
```

Arguments

`isoRepExpWithGeneId`	A isoform expression matrix which in addition to expression columns contain two colmns 'isoform_id' and 'gene_id' indicating which isoforms are a part of which gene.
`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

This is just specialized function that is a lot faster than general pourpose functions such as ddplyr.

Value

This function returns a data.frame where the first column is the gene id followed by the gene expression in all samples.

Author(s)

Kristoffer Vitting-Seerup

References

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

Examples

```
### Construct data needed from example data in cummeRbund package
### (The recomended way of analyzing Cufflinks/Cuffdiff datat is via importCufflinksCummeRbund
### This is jus an easy way to get some example data ).
cuffDB <- prepareCuffExample()

isoRepCount <- repCountMatrix(isoforms(cuffDB))
isoRepCount$isoform_id <- rownames(isoRepCount)

### add gene info
localAnnotaion <- import(system.file("extdata/chr1_snippet.gtf", package="cummeRbund"))[,c('transcript_id',
colnames(localAnnotaion@elementMetadata)[1] <- 'isoform_id'

isoRepCount$gene_id <- localAnnotaion$gene_id[match(
    isoRepCount$isoform_id , localAnnotaion$isoform_id
)]

### Summarize to gene level
geneRepCount <- isoformToGeneExp(isoRepCount)
```

---

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. The filtering is applied to each isoform in each comparison analyzed (unless keepIsoformInAllConditions is set to TRUE).

Usage

```
preFilter(
    switchAnalyzeRlist,
    geneExpressionCutoff = 1,
    isoformExpressionCutoff = 0,
    IFcutoff=0.01,
    geneCutoffInBothConditions=TRUE,
    acceptedIsoformClassCode = NULL,
    removeSingleIsoformGenes = TRUE,
    reduceToSwitchingGenes=FALSE,
    keepIsoformInAllConditions=FALSE,
```

```
    alpha=0.05,
    dIFcutoff = 0.1,
    quiet=FALSE
)
```

Arguments

`switchAnalyzeRlist`
: A `switchAnalyzeRlist` object.

`geneExpressionCutoff`
: The expression cutoff (most likely in RPKM/FPKM) which genes must be expressed more than, in at least one conditions of a comparison. NULL disables the filter. Default is 1 (in both conditions since default of `geneCutoffInBothConditions` is `TRUE`).

`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).

`geneCutoffInBothConditions`
: A logic indicating whether to requiring the gene expression cutoff (given by `geneExpressionCutoff`) in both conditions. This can be usefull since isoform fraction (IF) values are untrustworthy when the gene expression is to low, see `isoformSwitchTest` under details for more information. Default is TRUE.

`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 http://cole-trapnell-lab.github.io/cufflinks/cuffcompare/#transfrag-class-cod 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.

`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. Defalt is FALSE.

`alpha`
: The cutoff which the (calibrated) fdr correct p-values must be smaller than for calling significant switches. Only considered if `reduceToSwitchingGenes=TRUE`. Defuault 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 isoforms/genes is expressed above the cutoffs supplied in at least one condition of a comparison, 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 <isoform_exp> / <gene_exp>) 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 - which is decent. 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%.

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`
`importCufflinksCummeRbund`
`importCufflinksFiles`

Examples

```
data("exampleSwitchList")
exampleSwitchListFiltered <- preFilter(exampleSwitchList, geneExpressionCutoff = 1, isoformExpressionCutoff
```

`prepareCuffExample` *Prepare the Cufflinks example data*

Description

Prepare the Cufflinks example data set.

Usage

```
prepareCuffExample()
```

Details

This helper function prepares the Cufflinks example dataset, including the example GTF-file.

Value

A CuffSet object.

Author(s)

Kristoffer Vitting-Seerup, Johannes Waage

References

Vitting-Seerup K , Porse BT, Sandelin A, Waage J. (2014) spliceR: an R package for classification of alternative splicing and prediction of coding potential from RNA-seq data. BMC Bioinformatics 15:81.

Examples

```
#Load cufflinks example data
cuffDB <- prepareCuffExample()
```

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

Description

This function allows the user to remove data from all entereis in a switchAnalyzeRlist about isoforms that are no longer of interest.

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) indeicated 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

---

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(
    switchAnalyzeRlist = NULL,
    gene = NULL,
    isoform_id = NULL,
    condition1,
    condition2,
    IFcutoff=0,
    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 incuded (e.g. if protein domains should be visualized they should be annotaed 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 (which removes non-expressed isoforms).
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 usefull 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 (recomended). 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 ggplo2 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 subfunction.

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`.

Changes in isoform usage are measure as the difference in isoform fraction (dIF) values, where isoform fraction (IF) values are calculated as <isoform_exp> / <gene_exp>.

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

`isoformSwitchTest`
`switchPlotTranscript`
`switchPlotGeneExp`
`switchPlotIsoExp`
`switchPlotIsoUsage`
`switchPlotTopSwitches`

Examples

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

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

### Make isoform Switch Analysis Plot
switchPlot(exampleSwitchListAnalyzed, gene = mostSwitchingGene$gene_id, condition1 = 'hESC', condition2 = 'F
```

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 = NULL,
    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 = NULL,
    gene=NULL,
    isoform_id = NULL,
    condition1 = NULL,
    condition2 = NULL,
    IFcutoff=0,
    addErrorbars = TRUE,
    confidenceIntervalErrorbars = TRUE,
    confidenceInterval = 0.95,
    alphas = c(0.05, 0.001),
    logYaxis=FALSE,
    extendFactor = 0.05,
    localTheme = theme_bw()
)

switchPlotIsoUsage(
    switchAnalyzeRlist = NULL,
    gene=NULL,
    isoform_id = NULL,
    condition1 = NULL,
    condition2 = NULL,
    IFcutoff=0,
    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 (which removes non-expressed isoforms).
addErrorbars	A logic indicating whether error bars should be added to the expression plots to show uncertainty in estimates (recomended). 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 (***) respecitively. 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 avid 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 ggplo2 theme with which the plot is made, see ?ggplot2::theme for more info. Default is theme_bw().

Details

Changes in isoform usage are measure as the difference in isoform fraction (dIF) values, where isoform fraction (IF) values are calculated as <isoform_exp> / <gene_exp>.

Note that the bar indicating significance levels will only be shown if the analysis have 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 isoformSwitchTest).

Author(s)

Kristoffer Vitting-Seerup

References

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

See Also

isoformSwitchTest
switchPlotTranscript
switchPlot

Examples

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

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

### Plot expression
switchPlotGeneExp(exampleSwitchListAnalyzed, gene = mostSwitchingGene$gene_id, condition1 = 'hESC', conditio
switchPlotIsoExp(exampleSwitchListAnalyzed, gene = mostSwitchingGene$gene_id, condition1 = 'hESC', condition
switchPlotIsoUsage(exampleSwitchListAnalyzed, gene = mostSwitchingGene$gene_id, condition1 = 'hESC', conditi
```

---

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 `alpha` and `dIFcutoff`) to a specific folder (controlled by `pathToOutput`. The plots are automatically sorted by decreasing significance or switche size (as controled 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,
    n=10,
    sortByQvals=TRUE,
    filterForConsequences = FALSE,
    pathToOutput = getwd(),
    splitComparison=TRUE,
    splitFunctionalConsequences = TRUE,
    IFcutoff=0,
    fileType = "pdf",
    additionalArguments=list(),
    quiet=FALSE
)
```

Arguments

switchAnalyzeRlist	A switchAnalyzeRlist containing all the annotation for the isoforms.
alpha	The cutoff which the (callibrated) fdr correct p-values must be smaller than for calling significant switches. Defualit 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 top genes (after filtering and sorted according to sortByQvals) that should be generated in each subfolder indicated by splitComparison, splitFunctionalConse Use NA to create all. Default is 10.
sortByQvals	A logic indicating whether to the top n features are sorted by decreasing significance (increasing q-values) (if sortByQvals=TURE) 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 sum(abs(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. Defalut 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 subfolder for each comparison. If splitComparison is TRUE the subfolders 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 subfolder for those switches with predicted consequences and another subfolder for those without. Requires that an-

alyzeSwitchConsequences have been run. If `splitComparison=TRUE` the subfolders from this argument will be created within the comparison subfolders. 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 \'pdf\' and \'png\'. Default is pdf.

`additionalArguments`
A named list arguments passed to the `switchPlot` function which creates the individual Isoform Switch Analysis Plots. The name of the list entries must correspond to the corresponding argument in the `switchPlot` 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 <isoform_exp> / <gene_exp>.

For a list of the top swiching 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 signicant 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 Annoation*

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(
    switchAnalyzeRlist = NULL,
    gene = NULL,
    isoform_id = NULL,
    rescaleTranscripts = TRUE,
    plotXaxis = !rescaleTranscripts,
    reverseMinus = TRUE,
    ifMultipleIdenticalAnnotation = "summarize",
    rectHegith = 0.2,
    codingWidthFactor = 2,
    nrArrows = 20,
    arrowSize = 0.2,
    optimizeForCombinedPlot = FALSE,
    condition1 = NULL,
    condition2 = NULL,
    localTheme = theme_bw(),
    plot = TRUE
)
```

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 usefull 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 seperate color. Default is \'summarize\'.

`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. Only needed if `optimizeForCombinedPlot=TRUE` and more than one comparisons is analyzed.

`condition2` Second condition of the comparison to analyze, must be the name used in the switchAnalyzeRlist. Only needed if `optimizeForCombinedPlot=TRUE` and more than one comparisons is analyzed.

`localTheme` General ggplot2 theme with which the plot is made, see `?ggplot2::theme` for more info. Default is theme_bw().

`plot` A Logical indicating whether the final plot should be plotted (TRUE) or returned (FALSE). Default is TRUE.

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 (fx. 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`).
- `Signal Peptide` : By coloring the part of the ORF containing the signal peptide. Requires the result of an external SignalIP analysis have been added with `analyzeSignalP)`.

  `Transcript status` : Specificly from data imported from cufflinks/cuffdiff. The status (class code) of the transcript is added in brackets after the transcript name.

Value

- If `plot=TRUE` : Plots the visualization described in the `details` section
- If `plot=FALSE` : 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