alt_introns_to_transcripts

Alt intron = the next highest intron from a cluster with only 1 significant intron. Aka this intron has deltapsi < 0.1, but may represent the reciprocal to a significant intron that is significant. This script overwrites "annotation/alt_introns_195.tsv" (which was made with annotate_cryptic_introns.Rmd), adding the annotation and transcript id columns. Useful for TRIFID comparisons.

output files: 1. 31_leafcutter/alt_introns_195.tsv 2. 31_leafcutter/three_database_info_sig_junction.tsv

```
library(biomaRt)
library(tidyr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following object is masked from 'package:biomaRt':
##
##     select

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
library(ggrepel)
library(here); i_am("R/14_alt_introns_for_TRIFID.Rmd")
```

```
## here() starts at /projects/imb-pkbphil/sp/rnaseq/six_donor_trans/splicing_paper

## here() starts at /projects/imb-pkbphil/sp/rnaseq/six_donor_trans/splicing_paper
```

```
all_annot = read.delim(here("31_leafcutter", "three_database_info_all_junctions.tsv"))
sig_junctions = filter(all_annot, p.adjust < 0.05 & abs(deltapsi) > 0.1)
head(all_annot)
```

```
##   annotation  chr    start      end strand  cluster_id      deltapsi
## 1    gencode chr7 43648652 43650493      - clu_35616_- -0.0141028170
## 2    gencode chr7 43648652 43650612      - clu_35616_- -0.0408035987
## 3    gencode chr7 43648652 43665658      - clu_35616_- -0.0009734716
## 4    gencode chr7 43648652 43711400      - clu_35616_- -0.0003668545
```

```
## 5    gencode chr7 43648652 43729429      - clu_35616_- -0.0681581659
## 6    gencode chr7 43650712 43656033      - clu_35616_- -0.0034620531
##        p.adjust
## 1 2.192287e-123
## 2 2.192287e-123
## 3 2.192287e-123
## 4 2.192287e-123
## 5 2.192287e-123
## 6 2.192287e-123
##                                                                                                             trans
## 1                                                                                                         ENST00000
## 2 ENST00000446564.5,ENST00000448704.5,ENST00000451651.5,ENST00000418140.5,ENST00000431651.5,ENST0000
## 3                                                                                                          ENST0000
## 4                                                                                                          ENST0000
## 5                 ENST00000457939.1,ENST00000420441.1,ENST00000415076.6,ENST00000223336.10,ENST0000
## 6                                                                                                          ENST0000
##   min_intron_number mode_intron_number gene
## 1                 2                  2 COA1
## 2                 2                  2 COA1
## 3                 2                  2 COA1
## 4                 2                  2 COA1
## 5                 1                  1 COA1
## 6                 4                  4 COA1
##                                          biotype genes_in_cluster
## 1                                 protein_coding             COA1
## 2 nonsense_mediated_decay,protein_coding,lncRNA             COA1
## 3                        nonsense_mediated_decay             COA1
## 4                                 protein_coding             COA1
## 5        nonsense_mediated_decay,protein_coding             COA1
## 6                                 protein_coding             COA1
##   is_first_intron
## 1           FALSE
## 2           FALSE
## 3           FALSE
## 4           FALSE
## 5            TRUE
## 6           FALSE
```

```
nrow(all_annot)
```

```
## [1] 132587
```

```
head(sig_junctions)
```

Explore pairs of Diffspliced introns

```
##   annotation   chr    start      end strand  cluster_id   deltapsi
## 1    gencode chr17 17577027 17591531      - clu_19605_- -0.4296507
## 2    gencode chr11 66872159 66907821      -  clu_1700_-  0.1586862
## 3    gencode  chrX 15675778 15688661      + clu_15162_+ -0.1468799
```

```
## 4    gencode  chrX 15738352 15749971      + clu_15162_+  0.1271935
## 5    gencode  chr3 12289134 12312380      + clu_18227_+ -0.1629350
## 6    gencode  chr3 12351674 12379704      + clu_18227_+  0.1997759
##        p.adjust
## 1 4.360252e-104
## 2 8.715036e-104
## 3 5.675014e-100
## 4 5.675014e-100
## 5  5.202562e-89
## 6  5.202562e-89
##                                                                                                                                trans
## 1                 ENST00000421096.5,ENST00000580147.5,ENST00000461404.1,ENST00000255389.10,ENST0000
## 2                                                                                                                          ENST0000
## 3                                                                                                  ENST00000380333.5,ENST0000
## 4                                                                                                  ENST00000478923.1,ENST0000
## 5 ENST00000397026.7,ENST00000651735.1,ENST00000652431.1,ENST00000652098.1,ENST00000397012.7,ENST0000
## 6                                                                                                  ENST00000287820.10,ENST0000
##   min_intron_number mode_intron_number   gene
## 1                 1                  1   PEMT
## 2                 1                  1     PC
## 3                 1                  1 CA5BP1
## 4                 1                  1   CA5B
## 5                 1                  1  PPARG
## 6                 1                  1  PPARG
##                                      biotype genes_in_cluster
## 1 lncRNA,nonsense_mediated_decay,protein_coding            PEMT
## 2                                protein_coding              PC
## 3                                        lncRNA      CA5BP1,CA5B
## 4               retained_intron,protein_coding      CA5BP1,CA5B
## 5               protein_coding,retained_intron           PPARG
## 6               protein_coding,retained_intron           PPARG
##   is_first_intron
## 1            TRUE
## 2            TRUE
## 3            TRUE
## 4            TRUE
## 5            TRUE
## 6            TRUE
```

```
head(table(sig_junctions$gene))
```

```
##
##        AC002074.1        AC002467.1        AC004889.1        AC006001.3
##                 1                 1                 1                 1
##        AC008771.1 AC009879.3,ADHFE1
##                 1                 1
```

```
sig_junctions= mutate(sig_junctions, condition = if_else(deltapsi > 0, "beige", "white"))
has_gene_name =  sig_junctions[grep("^\\.$", sig_junctions$gene, invert = T),]
nrow(has_gene_name) #0 introns have no annotated gene :)
```

```
## [1] 777
```

```
table(table(has_gene_name$gene)) #this is the basic info I'm after
```

```
##
##   1   2   3   4   5
## 333 166   8   6   2
```

```
#but now use dplyr to split it on more things
```

```
head(group_by(has_gene_name, gene, condition) %>% count())
```

Introns per gene

```
## # A tibble: 6 x 3
## # Groups:   gene, condition [6]
##   gene              condition     n
##   <chr>             <chr>     <int>
## 1 AC002074.1        white         1
## 2 AC002467.1        beige         1
## 3 AC004889.1        white         1
## 4 AC006001.3        beige         1
## 5 AC008771.1        beige         1
## 6 AC009879.3,ADHFE1 white         1
```

```
introns_per_gene = group_by(has_gene_name, gene, condition) %>% count()
sum(introns_per_gene$n)
```

```
## [1] 777
```

```
genes_per_num_introns =  group_by(introns_per_gene, condition, n) %>% count(name="num_genes_with")
genes_per_num_introns
```

```
## # A tibble: 7 x 3
## # Groups:   condition, n [7]
##   condition     n num_genes_with
##   <chr>     <int>          <int>
## 1 beige         1            312
## 2 beige         2             17
## 3 beige        39              1
## 4 white         1            345
## 5 white         2             13
## 6 white         3              2
## 7 white        15              1
```

```
sum(genes_per_num_introns$num_genes_with)#647 genes*condition combos
```

```
## [1] 691
```

okay closer but I want genes that have both a white and a beige....

```
conditions_per_gene = group_by(has_gene_name, gene) %>% arrange(condition) %>%
    summarise(conditions = paste(unique(condition), collapse="&"), num_introns = n())
head(conditions_per_gene)
```

```
## # A tibble: 6 x 3
##   gene              conditions num_introns
##   <chr>             <chr>            <int>
## 1 AC002074.1        white                1
## 2 AC002467.1        beige                1
## 3 AC004889.1        white                1
## 4 AC006001.3        beige                1
## 5 AC008771.1        beige                1
## 6 AC009879.3,ADHFE1 white                1
```

```
genes_per_condition =  group_by(conditions_per_gene, conditions, num_introns) %>% count(name="num_genes
genes_per_condition
```

```
## # A tibble: 9 x 3
## # Groups:   conditions, num_introns [9]
##   conditions  num_introns num_genes_with
##   <chr>             <int>          <int>
## 1 beige                 1            152
## 2 beige                 2              3
## 3 beige&white           2            158
## 4 beige&white           3              8
## 5 beige&white           4              6
## 6 beige&white           5              2
## 7 beige&white          54              1
## 8 white                 1            181
## 9 white                 2              5
```

```
conditions_per_cluster = group_by(sig_junctions, cluster_id) %>% arrange(condition) %>%
    summarise(conditions = paste(unique(condition), collapse="&"), num_introns = n())
head(conditions_per_cluster)
```

Conditions per cluster

```
## # A tibble: 6 x 3
##   cluster_id  conditions  num_introns
##   <chr>       <chr>             <int>
## 1 clu_10104_- white                 1
## 2 clu_10181_- white                 1
## 3 clu_10209_- beige                 1
## 4 clu_10638_+ beige&white           2
## 5 clu_10654_+ beige&white           2
## 6 clu_10672_+ beige&white           2
```

```
clusters_per_condition =  group_by(conditions_per_cluster, conditions, num_introns) %>% count(name="num
clusters_per_condition
```

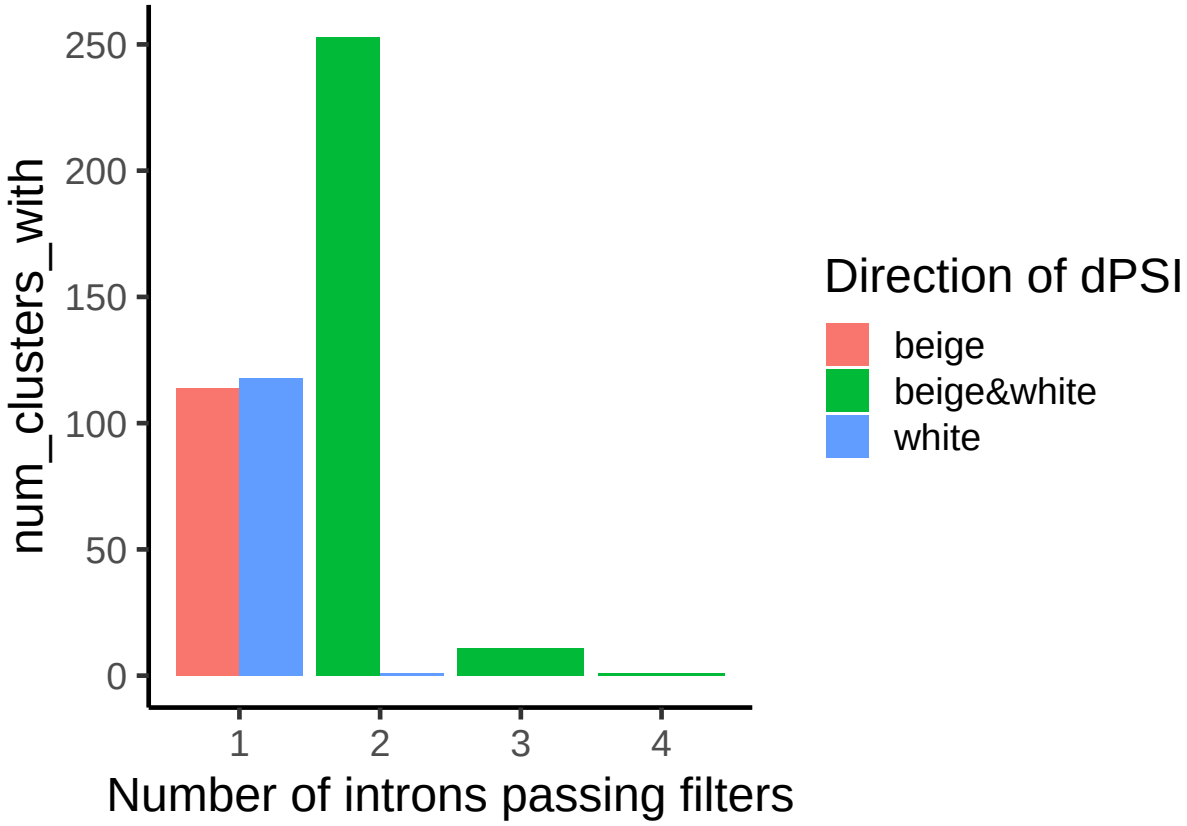
```
## # A tibble: 6 x 3
## # Groups:   conditions, num_introns [6]
##   conditions  num_introns num_clusters_with
##   <chr>             <int>             <int>
## 1 beige                 1               114
## 2 beige&white           2               253
## 3 beige&white           3                11
## 4 beige&white           4                 1
## 5 white                 1               118
## 6 white                 2                 1
```

This is much easier to represent and understand I think. CITED1 ends up in the white and beige 3 intron category

```
conditions_per_cluster[conditions_per_cluster$num_introns == 2,]
```

```
## # A tibble: 254 x 3
##    cluster_id  conditions  num_introns
##    <chr>       <chr>             <int>
##  1 clu_10638_+ beige&white           2
##  2 clu_10654_+ beige&white           2
##  3 clu_10672_+ beige&white           2
##  4 clu_1074_-  beige&white           2
##  5 clu_10840_+ beige&white           2
##  6 clu_10891_+ beige&white           2
##  7 clu_10973_+ beige&white           2
##  8 clu_10994_+ beige&white           2
##  9 clu_1114_-  beige&white           2
## 10 clu_11204_+ beige&white           2
## # i 244 more rows
```

```
ggplot(clusters_per_condition, aes(x=num_introns, fill=conditions, y=num_clusters_with)) +
    geom_bar(stat="identity", position= "dodge") + theme_classic(base_size=18) +
    labs(x="Number of introns passing filters") + scale_fill_discrete(name="Direction of dPSI")
```



As you have a significant cluster you'll have a second intron moving in the opposite direction, we're just eliminating them with the filter. So for trifid we select the next highest intron to compare?

Each cluster contains a pair of diffspliced junctions

```
sig_junctions=merge(sig_junctions, conditions_per_cluster, by="cluster_id")
nrow(sig_junctions)
```

```
## [1] 777
```

```
head(sig_junctions)
```

```
##    cluster_id annotation   chr     start       end strand   deltapsi
## 1 clu_10104_-    gencode  chr9 123401912 123403402      - -0.1076770
## 2 clu_10181_-    gencode  chr9 128266325 128267458      - -0.1130214
## 3 clu_10209_-    gencode  chr9 129108077 129110483      -  0.1074815
## 4 clu_10638_+    gencode chr12  26195951  26224293      + -0.1222677
## 5 clu_10638_+    gencode chr12  26195531  26224293      +  0.1056313
## 6 clu_10654_+    gencode chr12  27380404  27385481      + -0.1918059
##       p.adjust
## 1 1.437183e-02
## 2 8.760649e-15
## 3 2.104379e-08
```

```
## 4 1.870231e-02
## 5 1.870231e-02
## 6 1.084347e-07
##
## 1                                                                                                         ENST0000(
## 2
## 3
## 4
## 5
## 6 ENST00000395901.6,ENST00000542388.1,ENST00000311001.9,ENST00000261178.9,ENST00000457040.6,ENST0000(
##   min_intron_number mode_intron_number   gene              biotype
## 1                 1                  1 DENND1A protein_coding,lncRNA
## 2                 5                  5  GOLGA2        protein_coding
## 3                 1                  1    CRAT        protein_coding
## 4                 1                  1    SSPN        protein_coding
## 5                 1                  1    SSPN protein_coding,lncRNA
## 6                 2                  3  ARNTL2        protein_coding
##   genes_in_cluster is_first_intron condition  conditions num_introns
## 1          DENND1A            TRUE     white       white           1
## 2           GOLGA2           FALSE     white       white           1
## 3             CRAT            TRUE     beige       beige           1
## 4             SSPN            TRUE     white beige&white           2
## 5             SSPN            TRUE     beige beige&white           2
## 6           ARNTL2           FALSE     white beige&white           2
```

Select introns from significant clusters

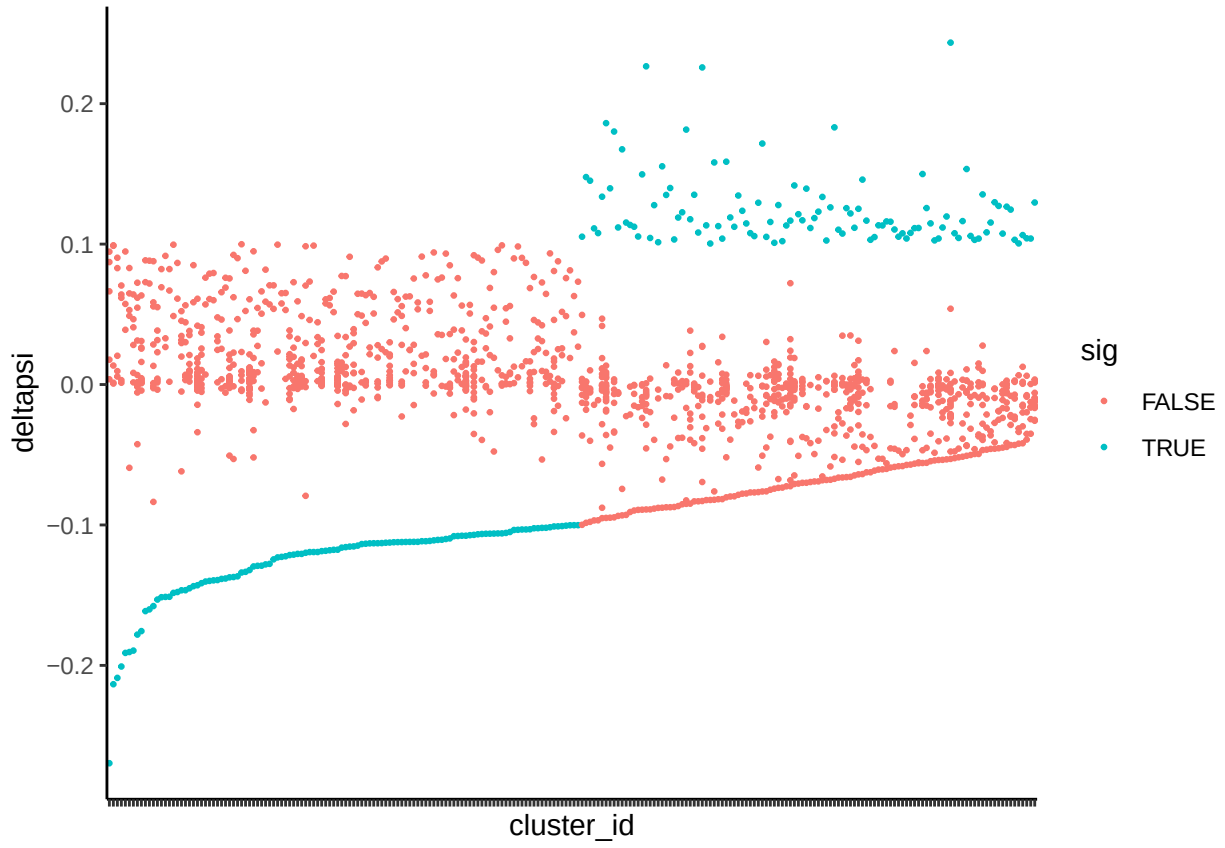
```
filt_clusters = all_annot[all_annot$cluster_id %in% sig_junctions$cluster_id,]
nrow(filt_clusters)
```

```
## [1] 2819
```

```
filt_clusters = merge(filt_clusters, select(sig_junctions, cluster_id, num_introns,conditions), by="clus
filt_clusters$cluster_id = factor(filt_clusters$cluster_id, levels=unique(filt_clusters$cluster_id[
                                      order(filt_clusters$num_introns, filt_clusters$deltapsi)]))
filt_clusters = mutate(filt_clusters, sig = p.adjust < 0.05 & abs(deltapsi) > 0.1)
```

Diagnostic plots This is an interesting diagnostic plot.
1) clusters with only 1 above-filter intron have lower deltapsi 2) ^ these also tend to have another intron just below the filter level. 3) its fun to see the three introns how they're distributed. The ones with the biggest difference have two beige introns.

```
ggplot(filter(filt_clusters,num_introns==1), aes(x=cluster_id,y=deltapsi, colour=sig)) + geom_point(size
```




```
possible_includers = filter(filt_clusters, num_introns==1 & !sig & abs(deltapsi)>0.05)
nrow(possible_includers) #236 introns
```

```
## [1] 279
```

```
length(unique(possible_includers$cluster_id))#169 of the single introns have another intron > 0.05 we c
```

```
## [1] 203
```

169 possible cluster comparisons... that could work: means we have only ~25 without a comparable intron. Its more complexity atm; but it may make sense overall to avoid comparing across categories?? And to use the leafcutter in a way that honours its concept, rather than working against it.

I don't find a problem with using the next highest, regardless of how low that dpsi is. We know its a small number with extremely low deltapsis. That can serve as a representative of the not changing transcripts.

Select alt introns

so for each of the clusters with just 1 intron; pick the intron with the maximum abs(deltpsi) to include

```
alt_introns = group_by(filt_clusters, cluster_id) %>% filter(abs(deltapsi) < 0.1 & num_introns ==1) %>%

alt_introns = select(alt_introns,colnames(all_annot) )
nrow(alt_introns)
```

```
## [1] 232
```

```
head(alt_introns)
```

```
## # A tibble: 6 x 15
## # Groups:   cluster_id [6]
##   annotation chr       start       end strand cluster_id  deltapsi p.adjust
##   <chr>      <chr>     <int>     <int> <chr>  <fct>          <dbl>    <dbl>
## 1 gencode    chr12 121534590 121536020 -      clu_25932_-   0.0946 2.31e- 6
## 2 gencode    chr2   11773745  11776086 +      clu_30968_+   0.0991 2.78e-57
## 3 gencode    chr11  58570713  58578019 -      clu_1449_-    0.0903 1.86e- 9
## 4 gencode    chr9   13140149  13147548 -      clu_9549_-    0.0705 1.22e-12
## 5 fantom_cat chr11   8717892   8717988 -      clu_1234_-    0.0947 1.74e-12
## 6 refseq     chr18  31666372  31724494 -      clu_21176_-   0.0830 1.44e- 9
## # i 7 more variables: transcript_ids <chr>, min_intron_number <int>,
## #   mode_intron_number <int>, gene <chr>, biotype <chr>,
## #   genes_in_cluster <chr>, is_first_intron <lgl>
```

```
summary(alt_introns$deltapsi)
```

```
##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -0.099900 -0.069127  0.031756  0.002647  0.075010  0.099897
```

```
write.table(alt_introns, here("31_leafcutter", "alt_introns_195.tsv"),
            sep="\t", quote = F, row.names = F)

write.table(sig_junctions, file=here("31_leafcutter", "three_database_info_sig_junctions.tsv"),
            sep="\t", quote=F, row.names = F)
nrow(sig_junctions)
```

```
## [1] 777
```