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_- -1.189192e-02
## 2    gencode chr7 43648652 43650612      - clu_35616_- -2.327379e-02
## 3    gencode chr7 43648652 43665658      - clu_35616_-  6.207827e-03
## 4    gencode chr7 43648652 43711400      - clu_35616_- -5.406415e-06
```

```
## 5    gencode chr7 43648652 43729429      - clu_35616_- -4.443950e-02
## 6    gencode chr7 43650712 43656033      - clu_35616_- -7.926449e-03
##        p.adjust
## 1 1.007194e-106
## 2 1.007194e-106
## 3 1.007194e-106
## 4 1.007194e-106
## 5 1.007194e-106
## 6 1.007194e-106
##                                                                                                  transc
## 1                                                                                            ENST000003
## 2 ENST00000446564.5,ENST00000431651.5,ENST00000418140.5,ENST00000448704.5,ENST00000451651.5,ENST00000
## 3                                                                                             ENST00000
## 4                                                                                             ENST00000
## 5                 ENST00000223336.10,ENST00000415076.6,ENST00000420441.1,ENST00000446330.6,ENST00000
## 6                                                                                             ENST00000
##   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 is_first_intron
## 1                               protein_coding           FALSE
## 2 nonsense_mediated_decay,protein_coding,lncRNA           FALSE
## 3                      nonsense_mediated_decay           FALSE
## 4                               protein_coding           FALSE
## 5       protein_coding,nonsense_mediated_decay            TRUE
## 6                               protein_coding           FALSE
```

```
nrow(all_annot)
```

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

```
head(sig_junctions)
```

Explore pairs of Diffspliced introns

```
##   annotation   chr    start      end strand  cluster_id   deltapsi
## 1    gencode chr11 66872159 66907821      -  clu_1700_-  0.1623643
## 2    gencode chr17 17577027 17591531      - clu_19605_- -0.4096022
## 3    gencode  chrX 15675778 15688661      + clu_15162_+ -0.1750512
## 4    gencode  chrX 15738352 15749971      + clu_15162_+  0.1435707
## 5    gencode  chr3 12289134 12312380      + clu_18227_+ -0.1619036
## 6    gencode  chr3 12351674 12379704      + clu_18227_+  0.1836960
##        p.adjust
## 1 2.901103e-103
## 2 3.184596e-102
## 3  5.447502e-99
```

```
## 4  5.447502e-99
## 5  1.014028e-90
## 6  1.014028e-90
##                                                                                                                 trans
## 1                                                                                                            ENST0000
## 2                  ENST00000421096.5,ENST00000580147.5,ENST00000461404.1,ENST00000255389.10,ENST0000
## 3                                                                                          ENST00000380333.5,ENST0000
## 4                                                                                          ENST00000318636.8,ENST0000
## 5 ENST00000397012.7,ENST00000651735.1,ENST00000397026.7,ENST00000652431.1,ENST00000497594.5,ENST0000
## 6                                                                                         ENST00000477039.5,ENST00000
##   min_intron_number mode_intron_number   gene
## 1                 1                  1     PC
## 2                 1                  1   PEMT
## 3                 1                  1 CA5BP1
## 4                 1                  1   CA5B
## 5                 1                  1  PPARG
## 6                 1                  1  PPARG
##                                         biotype is_first_intron
## 1                                protein_coding            TRUE
## 2 lncRNA,nonsense_mediated_decay,protein_coding            TRUE
## 3                                        lncRNA            TRUE
## 4                 protein_coding,retained_intron            TRUE
## 5                 protein_coding,retained_intron            TRUE
## 6                 retained_intron,protein_coding            TRUE
```

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

```
##
##       AAK1     ABHD18 AC002074.1 AC002467.1 AC006001.3 AC016924.1
##          1          1          2          1          1          2
```

```
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] 693
```

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

```
##
##   1   2   3   4   5
## 289 158   9   6   1
```

```
#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 AAK1       white         1
## 2 ABHD18     beige         1
## 3 AC002074.1 beige         1
## 4 AC002074.1 white         1
## 5 AC002467.1 beige         1
## 6 AC006001.3 beige         1
```

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

```
## [1] 693
```

```
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            289
## 2 beige         2             17
## 3 beige        24              1
## 4 white         1            314
## 5 white         2              9
## 6 white         3              2
## 7 white         8              1
```

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

```
## [1] 633
```

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 AAK1       white                 1
## 2 ABHD18     beige                 1
## 3 AC002074.1 beige&white           2
## 4 AC002467.1 beige                 1
## 5 AC006001.3 beige                 1
## 6 AC016924.1 beige&white           2
```

```
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            135
## 2 beige                 2              3
## 3 beige&white           2            152
## 4 beige&white           3              9
## 5 beige&white           4              6
## 6 beige&white           5              1
## 7 beige&white          32              1
## 8 white                 1            154
## 9 white                 2              3
```

```
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_10181_- white                 1
## 2 clu_10209_- beige                 1
## 3 clu_10638_+ beige&white           2
## 4 clu_10654_+ beige&white           2
## 5 clu_10672_+ beige&white           2
## 6 clu_10690_+ beige&white           2
```

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

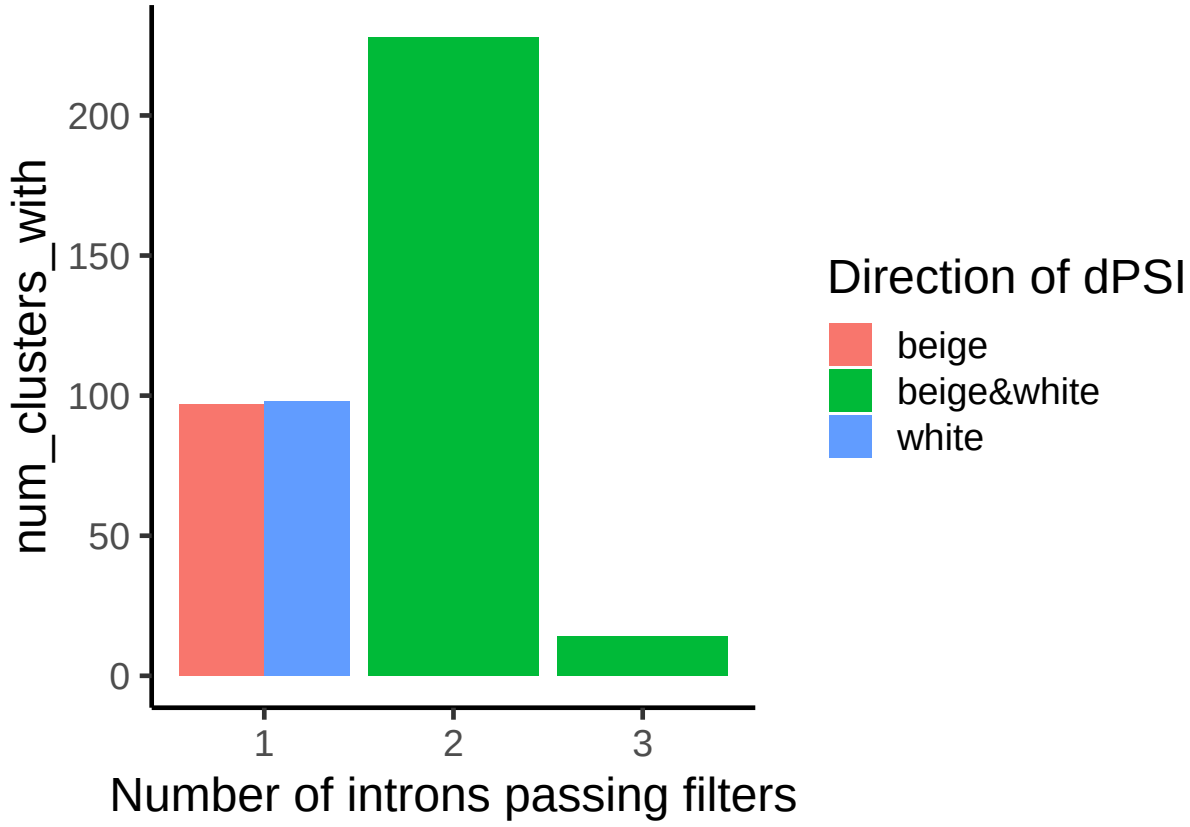
```
## # A tibble: 4 x 3
## # Groups:   conditions, num_introns [4]
##   conditions  num_introns num_clusters_with
##   <chr>             <int>              <int>
## 1 beige                 1                 97
## 2 beige&white           2                228
## 3 beige&white           3                 14
## 4 white                 1                 98
```

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: 228 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_10690_+ beige&white           2
##  5 clu_10840_+ beige&white           2
##  6 clu_10973_+ beige&white           2
##  7 clu_1114_-  beige&white           2
##  8 clu_11204_+ beige&white           2
##  9 clu_11348_+ beige&white           2
## 10 clu_11418_+ beige&white           2
## # i 218 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] 693
```

```
head(sig_junctions)
```

```
##    cluster_id annotation   chr     start       end strand   deltapsi
## 1 clu_10181_-    gencode  chr9 128266325 128267458      - -0.1125432
## 2 clu_10209_-    gencode  chr9 129108077 129110483      -  0.1072163
## 3 clu_10638_+    gencode chr12  26195951  26224293      + -0.1295000
## 4 clu_10638_+    gencode chr12  26195531  26224293      +  0.1250475
## 5 clu_10654_+ fantom_cat chr12  27382504  27385481      +  0.1596822
## 6 clu_10654_+    gencode chr12  27380404  27385481      + -0.1596822
##       p.adjust
## 1 9.378884e-15
## 2 2.288239e-08
## 3 1.945157e-02
## 4 1.945157e-02
## 5 1.104588e-07
## 6 1.104588e-07
##
## 1
## 2
## 3
## 4
## 5
## 6 ENST00000311001.9,ENST00000395901.6,ENST00000457040.6,ENST00000266503.9,ENST00000542388.1,ENST0000
##   min_intron_number mode_intron_number              gene               biotype
## 1                 5                  5            GOLGA2        protein_coding
## 2                 1                  1              CRAT        protein_coding
## 3                 1                  1              SSPN        protein_coding
## 4                 1                  1              SSPN protein_coding,lncRNA
## 5                 1                  1 ENSG00000029153.10                  <NA>
## 6                 2                  3            ARNTL2        protein_coding
##   is_first_intron condition  conditions num_introns
## 1           FALSE     white       white           1
## 2            TRUE     beige       beige           1
## 3            TRUE     white beige&white           2
## 4            TRUE     beige beige&white           2
## 5            TRUE     beige beige&white           2
## 6           FALSE     white beige&white           2
```

Select introns from significant clusters

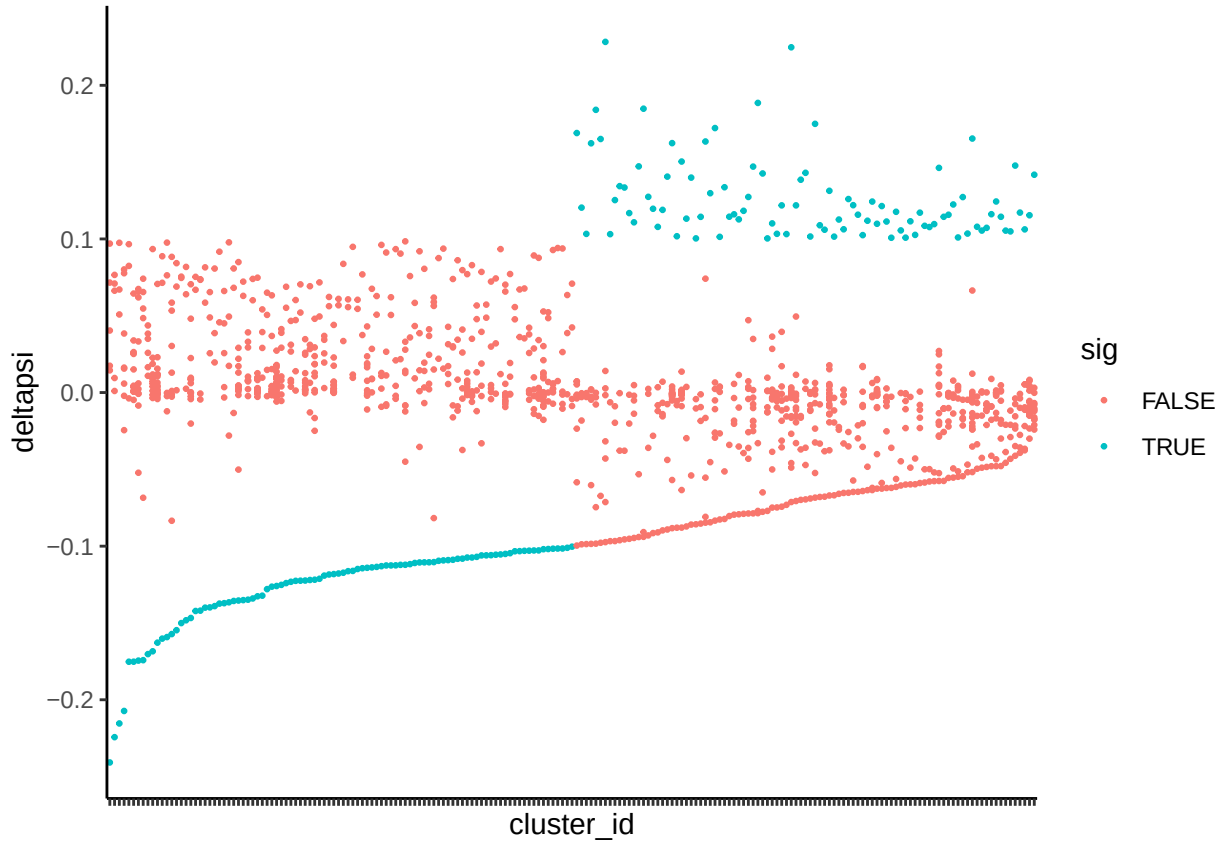
```
filt_clusters = all_annot[all_annot$cluster_id %in% sig_junctions$cluster_id,]
nrow(filt_clusters)
```

```
## [1] 2434
```

```
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) ˆ 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] 236
```

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

```
## [1] 169
```

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] 195
```

```
head(alt_introns)
```

```
## # A tibble: 6 x 14
## # Groups:   cluster_id [6]
##   annotation chr       start       end strand cluster_id  deltapsi p.adjust
##   <chr>      <chr>     <int>     <int> <chr>  <fct>          <dbl>    <dbl>
## 1 refseq     chr12 121534590 121537912 -      clu_25932_-   0.0970 1.80e- 6
## 2 gencode    chr9   13140149  13150511 -      clu_9549_-    0.0766 9.36e- 9
## 3 gencode    chr12  55979474  55986869 +      clu_10891_+   0.0975 7.75e- 4
## 4 gencode    chr3   12941865  13047478 -      clu_3320_-    0.0802 2.98e-15
## 5 gencode    chr22  17803861  17808844 -      clu_585_-     0.0966 1.49e- 8
## 6 gencode    chr12  55721494  55721689 +      clu_10876_+   0.0644 5.17e-74
## # i 6 more variables: transcript_ids <chr>, min_intron_number <int>,
## #   mode_intron_number <int>, gene <chr>, biotype <chr>, is_first_intron <lgl>
```

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

```
##       Min.    1st Qu.     Median       Mean    3rd Qu.       Max.
## -0.0996153 -0.0697084  0.0272898  0.0005775  0.0738238  0.0984361
```

```
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] 693
```