Using branchpointer for annotation of intronic human splicing branchpoints

Beth Signal

April 24, 2017

Contents

1 Introduction

The spliceosome mediates the formation of an intron lariat though inter-action between the 5' splice site and branchpoint (Will and Luhrmann, 2011). A subsequent reaction at the 3' SS then removes the intron lariat, producing a spliced RNA product. Mapping of branchpoints generally requires sequencing of the intron lariat following cDNA synthesis (Gao et al., 2008; Taggart et al., 2012). However, intron lariats are rapidly de-branched and degraded, and much less abundant than the spliced RNA, resulting in the poor recovery of elements using sequencing. Most recently, Mercer et al. (2015) employed a targeted sequencing approach to identify 59,359 branchpoints in 17.4% of annotated human gene introns. Whilst this constituted the largest annotation to date, the identification of branchpoints was restricted to highly-expressed genes with sufficient sequence coverage.

To address this limitation, and expand branchpoint annotations across the human genome, we have developed a machine-learning based model of branchpoints trained with this empirical annotation (Signal et al., 2016). This model requires only genomic sequence and exon annotations, and exhibits no discernible bias to gene type or expression, and can be applied using the R package, branchpointer. Aberrant splicing is known to lead to many human diseases (Singh and Cooper, 2012), however prediction of intronic variant effects have been typically limited to splice site alterations (McLaren et al., 2016; Wang et al., 2010). Therefore, in addition to annotation of branchpoints, branchpointer allows users to assess the effects of intronic mutations on branchpoint architecture.

Gao,K. et al. (2008) Human branch point consensus sequence is yUnAy. Nucleic Acids Res., 36, 2257–67.

McLaren,W. et al. (2016) The Ensembl Variant Effect Predictor. Genome Biol., 17, 122.

Mercer,T.R. et al. (2015) Genome-wide discovery of human splicing branchpoints. Genome Res., 25, 290–303.

Signal,B. et al. (2016) Machine-learning annotation of human splicing branchpoints. BioRxiv. doi: 10.1101/094003.

Singh,R.K. and Cooper,T.A. (2012) Pre-mRNA splicing in disease and therapeutics. Trends Mol. Med., 18, 472–482.

Taggart,A.J. et al. (2012) Large-scale mapping of branchpoints in human pre-mRNA transcripts in vivo. Nat. Struct. Mol. Biol., 19, 719–21.

Wang,K. et al. (2010) ANNOVAR: functional annotation of genetic variants from high-throughput sequencing data. Nucleic Acids Res., 38, e164.

Will,C.L. and Luhrmann,R. (2011) Spliceosome structure and function. Cold Spring Harb. Perspect. Biol., 3, a003707.

2 Preparation

2.1 Download genome annotations

Branchpointer requires a genome annotation derived from a GTF file and the fasta sequence for this genome annotation. We will be using the GENCODE annotation (http://www.gencodegenes.org/releases/current.html) as an example, although others and custom annotations can be used.

Create or move to a working directory where these files can be stored. Note that these can be large files (over 1GB) when uncompressed

```
wget ftp://ftp.sanger.ac.uk/pub/gencode/Gencode_human/release_24/gencode.v24.annotation.gtf.gz
gunzip gencode.v24.annotation.gtf.gz
```

branchpointer requires either a genome .fa file, or a BSGenome object for sequence retrieval. The genome must correspond to the gtf used – i.e. gencodev24 uses GRCh38 (p5).

Download .fa:

```
wget ftp://ftp.sanger.ac.uk/pub/gencode/Gencode_human/release_24/GRCh38.p5.genome.fa.gz
gunzip GRCh38.p5.genome.fa.gz
```

or load a *BSGenome*:

```
library(BSgenome.Hsapiens.UCSC.hg38)
genome <- BSgenome.Hsapiens.UCSC.hg38::BSgenome.Hsapiens.UCSC.hg38
```

2.2 Read in exon annotations

Start by loading branchpointer.

```
library(branchpointer)
```

readExonAnnotation will generate an exon annotation GRanges object from a gtf. We will load in the gtf downloaded from the preparation section.

```
exons <- gtfToExons("gencode.v24.annotation.gtf")
```

We will load in a small gtf from the package data for the following examples.

```
smallExons <- system.file("extdata","gencode.v24.annotation.small.gtf",
                          package = "branchpointer")
exons <- gtfToExons(smallExons)
```

```
## Warning:  closing unused connection 6 (/tmp/RtmpQz3P6K/Rinst65fe3e727744/branchpointer/extdata/gencode.
## Warning:  closing unused connection 5 (/tmp/RtmpQz3P6K/Rinst65fe3e727744/branchpointer/extdata/gencode.
```

3 Branchpoint annotations in intronic regions

3.1 Read query and calculate location attributes

Query regions must contain a branchpoint window - that is the region located at -18 to -44 from the 3' splice site. Each region given will be treated as only one query, and associated with the closest 3' exon. To cover multiple 3'exons, please provide branchpointer with separate region queries. For known regions, queries can be supplied as a table:

```
queryIntron <- system.file("extdata","intron_example.txt",
                           package = "branchpointer")
queryIntron <- readQueryFile(queryIntron,queryType = "region")

head(queryIntron)
```

```
## GRanges object with 2 ranges and 1 metadata column:
##       seqnames              ranges strand |           id
##          <Rle>           <IRanges>  <Rle> |  <character>
##   [1]    chr17 [43106534, 43106634]      - | BRCA1_intron
##   [2]    chr13 [32376570, 32376669]      + | BRCA2_intron
##   -------
##   seqinfo: 2 sequences from an unspecified genome; no seqlengths
```

Then location information can be retrieved using

```
queryIntron <- getQueryLoc(queryIntron,queryType = "region",exons = exons)
head(queryIntron)
```

```
## GRanges object with 2 ranges and 6 metadata columns:
##       seqnames              ranges strand |           id to_3prime to_5prime same_gene
##          <Rle>           <IRanges>  <Rle> |  <character> <numeric> <numeric> <logical>
##   [1]    chr17 [43106551, 43106577]      - | BRCA1_intron        18      3914      TRUE
##   [2]    chr13 [32376626, 32376652]      + | BRCA2_intron        18      1246      TRUE
##              exon_3prime       exon_5prime
##              <character>       <character>
##   [1] ENSE00003541068.1 ENSE00001888888.1
##   [2] ENSE00003461148.1 ENSE00002167182.1
##   -------
##   seqinfo: 2 sequences from an unspecified genome; no seqlengths
```

For large numbers of queries (over 500), it is recommended to use parallelisation to speed up computation.

This can be done by setting

`use_parallel=TRUE`

and supplying a cores number greater than 1 to functions with this argument.

Note that if the number of specified cores is greater than the number available, the maximum number available will be utilised

```
queryIntron <- getQueryLoc(queryIntron,queryType="region",
                           exons = exons, useParallel=TRUE, cores=4)
```

Alternatively, to generate branchpoint window region queries by exon annotations, the exon annotation file can be used:

Note that when searching for genes, transcripts, or exons, the ids used must be in the same format as in the annotation file (i.e. ENSG00000XXXXXX, ENST00000XXXXXX, ENSE00000XXXXXX). If you are unsure of an id, aliases can typically be found through ensembl (ensembl.org), or through a biomaRt query.

```
queryIntronMake <- makeRegions("ENSE00003541068", "exon_id", exons)
```

```
## Warning in if (is.na(y) | noType) {: the condition has length > 1 and only the first element
will be used
```

```
head(queryIntronMake)
```

```
## GRanges object with 6 ranges and 11 metadata columns:
##       seqnames              ranges strand |            gene_id      gene_type
##          <Rle>           <IRanges>  <Rle> |        <character>    <character>
##   [1]    chr17 [43106551, 43106577]      - | ENSG00000012048.20 protein_coding
##   [2]    chr17 [43106551, 43106577]      - | ENSG00000012048.20 protein_coding
##   [3]    chr17 [43106551, 43106577]      - | ENSG00000012048.20 protein_coding
##   [4]    chr17 [43106551, 43106577]      - | ENSG00000012048.20 protein_coding
##   [5]    chr17 [43106551, 43106577]      - | ENSG00000012048.20 protein_coding
##   [6]    chr17 [43106551, 43106577]      - | ENSG00000012048.20 protein_coding
##           transcript_id transcript_type           exon_id exon_number to_3prime to_5prime
##             <character>     <character>       <character> <character> <numeric> <numeric>
##   [1] ENST00000357654.7  protein_coding ENSE00003541068.1           4        18      3914
##   [2] ENST00000352993.7  protein_coding ENSE00003541068.1           4        18      3914
##   [3] ENST00000468300.5  protein_coding ENSE00003541068.1           4        18      3914
##   [4] ENST00000471181.6  protein_coding ENSE00003541068.1           4        18      3914
##   [5] ENST00000491747.6  protein_coding ENSE00003541068.1           4        18      3914
##   [6] ENST00000478531.5  protein_coding ENSE00003541068.1           4        18      3914
##       same_gene       exon_3prime       exon_5prime
##       <logical>       <character>       <character>
##   [1]      TRUE ENSE00003541068.1 ENSE00001888888.1
##   [2]      TRUE ENSE00003541068.1 ENSE00001888888.1
##   [3]      TRUE ENSE00003541068.1 ENSE00001888888.1
##   [4]      TRUE ENSE00003541068.1 ENSE00001888888.1
##   [5]      TRUE ENSE00003541068.1 ENSE00001888888.1
##   [6]      TRUE ENSE00003541068.1 ENSE00001888888.1
##   -------
##   seqinfo: 5 sequences from an unspecified genome; no seqlengths
```

```
#or for multiple ids
queryIntronMake <- lapply(c("ENSE00003541068", "ENSE00003461148"),
                     makeRegions, "exon_id", exons)
```

```
## Warning in if (is.na(y) | noType) {: the condition has length > 1 and only the first element
will be used
```

```
## Warning in if (is.na(y) | noType) {: the condition has length > 1 and only the first element
will be used
```

```
queryIntronMake <- do.call("c", queryIntronMake)
```

3.1.1 Using bedtools

During the prediction step, if a BSGenome object is not specified,sequences covering each site +/- 250 nt can be retrieved using bedtools. The absolute location of the bedtools binary must be provided for calls from within R. To find the location of your installed bedtools binary, using the command line type:

```
which bedtools
```

If chromosome names in the .fa genome file do not match those in the query (i.e chr1 in query, 1 in .fa), the argument rm_chr should be set to FALSE.

3.2 Predict branchpoint probabilities

Branchpoint probability scores can then be evaluated using the branchpointer model. This will generate a new GRanges object with a row for each site (of 27) in branchpoint window regions. If a SNP query type is provided (See next section), this will also perform an in silico mutation of the sequence.

We recommend use of the cutoff probability 0.5 to distinguish branchpoints and non-branchpoint sites. U2 binding energy can be used as a measurement of branchpoint strength when the probability score is above the cutoff.

All features required for the model to predict branchpoint probability are contained within the output object, along with the score and U2 binding energy.

```
branchpointPredictionsIntron <- predictBranchpoints(queryIntron, queryType = "region", BSgenome = genome)

head(branchpointPredictionsIntron)
```

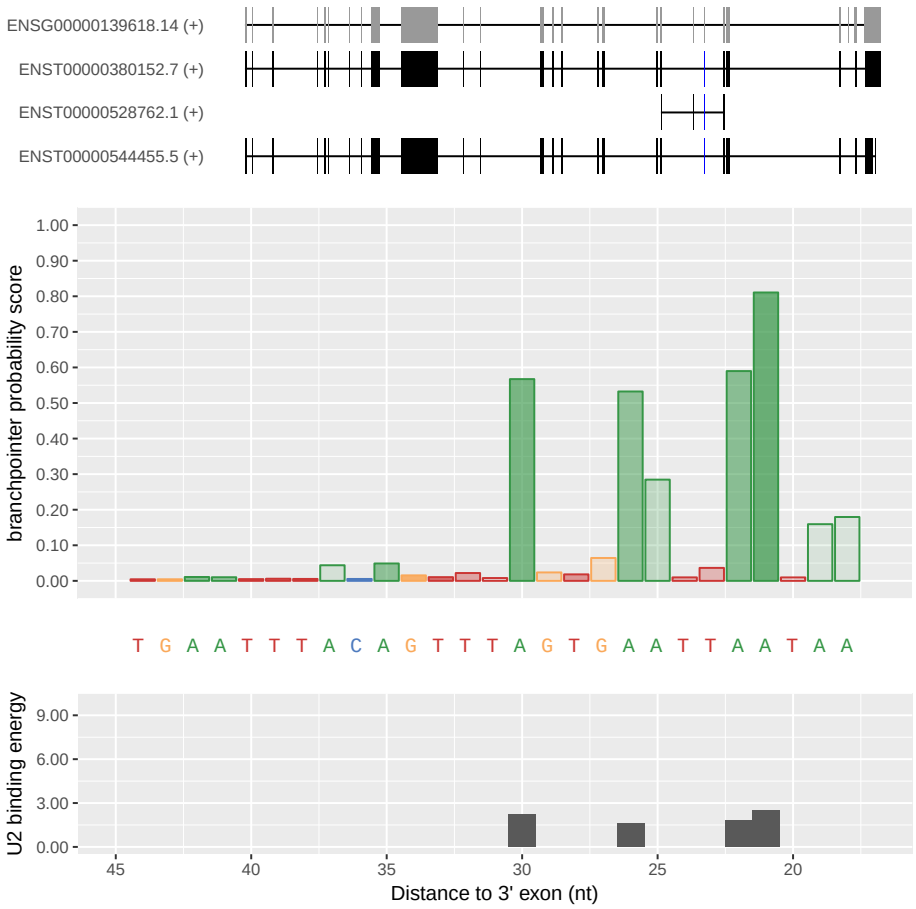
```
## GRanges object with 6 ranges and 31 metadata columns:
##       seqnames              ranges strand |           id to_3prime to_5prime same_gene
##          <Rle>           <IRanges>  <Rle> |  <character> <numeric> <numeric> <logical>
##   [1]    chr17 [43106551, 43106577]      - | BRCA1_intron        18      3914      TRUE
##   [2]    chr13 [32376626, 32376652]      + | BRCA2_intron        18      1246      TRUE
##   [3]    chr17 [43106551, 43106577]      - | BRCA1_intron        18      3914      TRUE
##   [4]    chr13 [32376626, 32376652]      + | BRCA2_intron        18      1246      TRUE
##   [5]    chr17 [43106551, 43106577]      - | BRCA1_intron        18      3914      TRUE
##   [6]    chr13 [32376626, 32376652]      + | BRCA2_intron        18      1246      TRUE
##              exon_3prime       exon_5prime
##              <character>       <character>
##   [1] ENSE00003541068.1 ENSE00001888888.1
##   [2] ENSE00003461148.1 ENSE00002167182.1
##   [3] ENSE00003541068.1 ENSE00001888888.1
##   [4] ENSE00003461148.1 ENSE00002167182.1
##   [5] ENSE00003541068.1 ENSE00001888888.1
##   [6] ENSE00003461148.1 ENSE00002167182.1
##                                                    seq      status to_3prime_point
##                                              <character> <character>       <integer>
##   [1] CTACAAAAGGAAGTAAATTAAATTGTTCTTTCTTTCTTTATAATTTATAG         REF              44
##   [2] TGCTTTTGAATTTACAGTTTAGTGAATTAATAATCCTTTTGTTTTCTTAG         REF              44
##   [3] CTACAAAAGGAAGTAAATTAAATTGTTCTTTCTTTCTTTATAATTTATAG         REF              43
##   [4] TGCTTTTGAATTTACAGTTTAGTGAATTAATAATCCTTTTGTTTTCTTAG         REF              43
##   [5] CTACAAAAGGAAGTAAATTAAATTGTTCTTTCTTTCTTTATAATTTATAG         REF              42
##   [6] TGCTTTTGAATTTACAGTTTAGTGAATTAATAATCCTTTTGTTTTCTTAG         REF              42
##       to_5prime_point test_site seq_pos0 seq_pos1 seq_pos2 seq_pos3 seq_pos4 seq_pos5
##             <numeric> <numeric> <factor> <factor> <factor> <factor> <factor> <factor>
##   [1]            3888  43106577        A        A        G        G        A        A
##   [2]            1220  32376626        T        G        A        A        T        T
##   [3]            3889  43106576        A        G        G        A        A        G
##   [4]            1221  32376627        G        A        A        T        T        T
##   [5]            3890  43106575        G        G        A        A        G        T
##   [6]            1222  32376628        A        A        T        T        T        A
##       seq_neg1 seq_neg2 seq_neg3 seq_neg4 seq_neg5 canon_hit1 canon_hit2 canon_hit3
##       <factor> <factor> <factor> <factor> <factor>  <numeric>  <numeric>  <numeric>
```

```
##   [1]        A        A        C        A        T          1          5         42
##   [2]        T        T        T        C        G          9         14         42
##   [3]        A        A        A        C        A          4         41         69
##   [4]        T        T        T        T        C          8         13         41
##   [5]        A        A        A        A        C          3         40         68
##   [6]        G        T        T        T        T          7         12         40
##       canon_hit4 canon_hit5 ppt_start ppt_run_length branchpoint_prob U2_binding_energy
##        <numeric>  <numeric> <numeric>      <numeric>        <numeric>         <numeric>
##   [1]         70         73        17             19      0.010103881               2.3
##   [2]         85         89        28             15      0.004404418               1.1
##   [3]         72         76        16             19      0.012037114               1.1
##   [4]         84         88        27             15      0.004432063               0.5
##   [5]         71         75        15             19      0.006405972               0.2
##   [6]         83         87        26             15      0.010848589               1.9
##   -------
##   seqinfo: 2 sequences from an unspecified genome; no seqlengths
```

The window scores can be plotted using plotBranchpointWindow(), with optional plots for gene and isoform structure. The main panel displays the probability scores of each site within the branchpoint window. The opacity of the bars is representative of relative U2 binding energy (darker = stronger), and the lower panel shows U2 binding energy for all sites above the provided probability cutoff.

BRCA2 intron (ENSE00002167182.1 - ENSE00003461148.1):

```
plotBranchpointWindow(queryIntron$id[2],
                      branchpointPredictionsIntron,
                      probabilityCutoff = 0.5,
                      plotMutated = FALSE,
                      plotStructure = TRUE,
                      exons = exons)
```



4 Effects of SNPs on branchpoint annotations

In addition to locating branchpoints in intronic windows, branchpointer can be used to evaluate the local effects of SNPs on branchpoints. The general workflow is the same as for annotation of intronic windows, however

`queryType="SNP"`

must be used.

4.1 Read query and calculate location attributes

Query SNPs should be located nearby a branchpoint window to have any potential effects on branchpoint architecture SNP queries can be supplied as a table formatted as follows:

```
querySNP <- system.file("extdata","SNP_example.txt", package = "branchpointer")
querySNP <- readQueryFile(querySNP,queryType="SNP")
```

Alternatively, appropriate attributes can be pulled from biomart when a list of refsnp ids is provided:

```
library(biomaRt)
mart <- useMart("ENSEMBL_MART_SNP", dataset="hsapiens_snp",host="www.ensembl.org")
querySNP <- snpToQuery(c("rs17000647","rs5031002","rs998731"), mart.snp = mart)
```

By default, all SNPs retrieved will be unstranded, and hence further processing will be done on both strands

Location information can be retrieved using:

```
querySNP <- getQueryLoc(querySNP, queryType="SNP", exons = exons, filter = FALSE)
head(querySNP)
```

```
## GRanges object with 3 ranges and 8 metadata columns:
##       seqnames              ranges strand |             id  ref_allele  alt_allele
##          <Rle>           <IRanges>  <Rle> |    <character> <character> <character>
##   [1]     chr4 [75556520, 75556520]      + | rs17000647_pos           C           A
##   [2]     chrX [67722783, 67722783]      + |  rs5031002_pos           G           A
##   [3]     chr8 [80183160, 80183160]      - |   rs998731_neg           C           T
##       to_3prime to_5prime same_gene       exon_3prime       exon_5prime
##       <numeric> <numeric> <logical>       <character>       <character>
##   [1]        21       371      TRUE ENSE00003490609.1 ENSE00001193582.3
##   [2]        44       611      TRUE ENSE00001316881.1 ENSE00003717946.1
##   [3]        22     35309      TRUE ENSE00002128654.1 ENSE00002093962.1
##   -------
##   seqinfo: 3 sequences from an unspecified genome; no seqlengths
```

Each SNP will be associated with the closest 3' exon. If SNPs are distal from branchpoint windows, the max_dist argument will remove any greater than the specified distance. Filtering prior to exon associations can speed up processing in instances where it is unknown if the majority of SNPs fall nearby branchpoint windows.

Queries can be provided as stranded or unstranded. In the case of unstranded queries, any value except "+" or "-" will cause branchpointer to run on both strands.

4.2 Predict branchpoint probabilities

Using a .fa and bedtools:

```
branchpointPredictionsSNP <- getBranchpointSequence(querySNP,
                                        queryType = "SNP",
                                        genome = "GRCh38.p5.genome.fa",
                                        bedtoolsLocation="/Apps/bedtools2/bin/bedtools")
```

Using a BSgenome:

```
#for query SNPs
branchpointPredictionsSNP <- predictBranchpoints(querySNP,
                                        queryType = "SNP",
                                        BSgenome = genome)


#to summarise effects:
querySNP <- predictionsToStats(querySNP,branchpointPredictionsSNP)
head(querySNP)
```

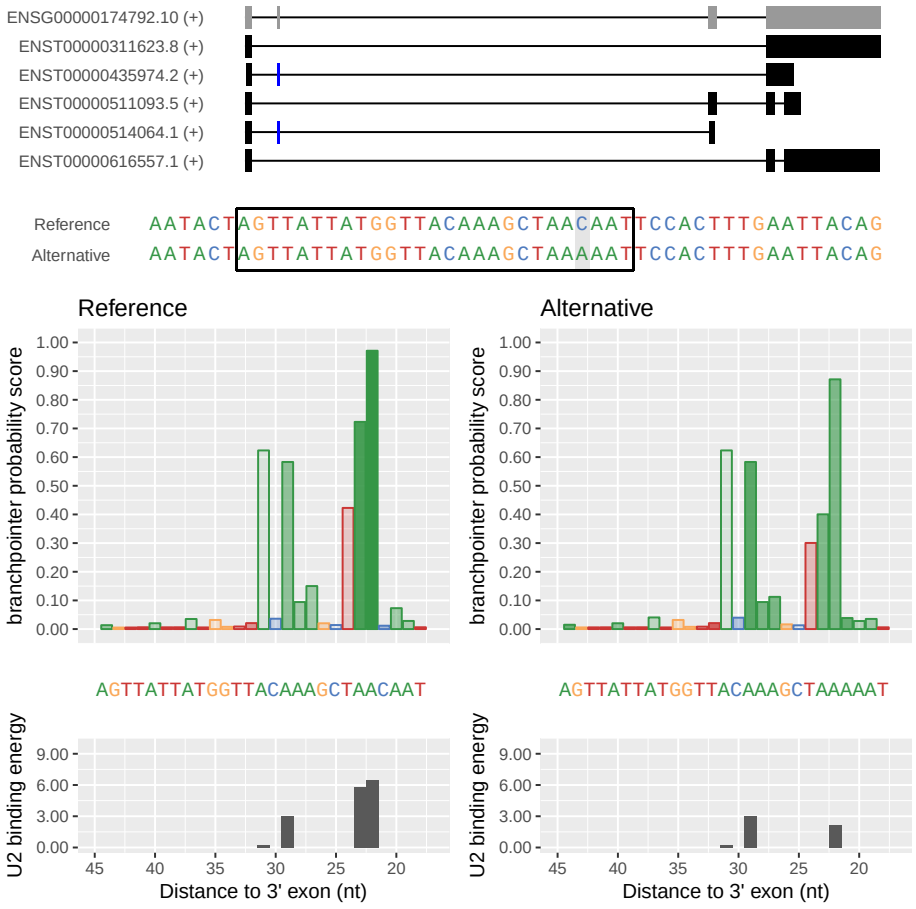
```
## GRanges object with 3 ranges and 18 metadata columns:
##       seqnames              ranges strand |             id  ref_allele  alt_allele
##          <Rle>           <IRanges>  <Rle> |    <character> <character> <character>
##   [1]     chr4 [75556520, 75556520]      + | rs17000647_pos           C           A
##   [2]     chrX [67722783, 67722783]      + |  rs5031002_pos           G           A
##   [3]     chr8 [80183160, 80183160]      - |   rs998731_neg           C           T
##       to_3prime to_5prime same_gene       exon_3prime       exon_5prime BP_num_REF
##       <numeric> <numeric> <logical>       <character>       <character>  <integer>
##   [1]        21       371      TRUE ENSE00003490609.1 ENSE00001193582.3          4
```

```
##   [2]        44       611      TRUE ENSE00001316881.1 ENSE00003717946.1          0
##   [3]        22     35309      TRUE ENSE00002128654.1 ENSE00002093962.1          1
##       BP_num_ALT deleted_n created_n dist_to_BP_REF dist_to_BP_ALT max_prob_REF
##        <integer> <integer> <integer>      <numeric>      <numeric>    <numeric>
##   [1]          3         1         0             -1             -1    0.9711006
##   [2]          1         0         1           <NA>              0    0.4562867
##   [3]          2         0         1             -4              0    0.6077347
##       max_prob_ALT max_U2_REF max_U2_ALT
##          <numeric>  <numeric>  <numeric>
##   [1]    0.8713144        6.5        3.0
##   [2]    0.6564301       <NA>        0.5
##   [3]    0.7966434        0.5        1.6
##   -------
##   seqinfo: 3 sequences from an unspecified genome; no seqlengths
```

The window scores in the reference and alternative sequences can be visualised using plotBranchpointWindow().

rs17000647 in C4orf26 intron 1.

```
plotBranchpointWindow(querySNP$id[1],
                      branchpointPredictionsSNP,
                      probabilityCutoff = 0.5,plotMutated = TRUE,
                      plotStructure = TRUE,exons = exons)
```



5 Session info

```
sessionInfo()
```

```
## R version 3.4.0 (2017-04-21)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 16.04.2 LTS
##
## Matrix products: default
## BLAS: /home/biocbuild/bbs-3.5-bioc/R/lib/libRblas.so
## LAPACK: /home/biocbuild/bbs-3.5-bioc/R/lib/libRlapack.so
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C               LC_TIME=en_US.UTF-8
##  [4] LC_COLLATE=C               LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=en_US.UTF-8       LC_NAME=C                  LC_ADDRESS=C
## [10] LC_TELEPHONE=C             LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
##  [1] splines   stats4    parallel  stats     graphics  grDevices utils     datasets
##  [9] methods   base
##
## other attached packages:
##  [1] biomaRt_2.32.0                   plyr_1.8.4
##  [3] gbm_2.1.3                        survival_2.41-3
##  [5] branchpointer_1.0.0              caret_6.0-76
##  [7] ggplot2_2.2.1                    lattice_0.20-35
##  [9] BSgenome.Hsapiens.UCSC.hg38_1.4.1 BSgenome_1.44.0
## [11] rtracklayer_1.36.0               Biostrings_2.44.0
## [13] XVector_0.16.0                   GenomicRanges_1.28.0
## [15] GenomeInfoDb_1.12.0              IRanges_2.10.0
## [17] S4Vectors_0.14.0                 BiocGenerics_0.22.0
## [19] knitr_1.15.1
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_0.12.10               Rsamtools_1.28.0           rprojroot_1.2
##  [4] digest_0.6.12              foreach_1.4.3              backports_1.0.5
##  [7] MatrixModels_0.4-1         RSQLite_1.1-2              evaluate_0.10
## [10] highr_0.6                  zlibbioc_1.22.0            lazyeval_0.2.0
## [13] data.table_1.10.4          minqa_1.2.4                SparseM_1.77
## [16] kernlab_0.9-25             car_2.1-4                  nloptr_1.0.4
## [19] Matrix_1.2-9               rmarkdown_1.4              labeling_0.3
## [22] lme4_1.1-13                BiocParallel_1.10.0        stringr_1.2.0
## [25] RCurl_1.95-4.8             munsell_0.4.3              DelayedArray_0.2.0
## [28] compiler_3.4.0             mgcv_1.8-17                htmltools_0.3.5
## [31] nnet_7.3-12                SummarizedExperiment_1.6.0 tibble_1.3.0
## [34] GenomeInfoDbData_0.99.0    codetools_0.2-15           matrixStats_0.52.2
## [37] XML_3.98-1.6               GenomicAlignments_1.12.0   MASS_7.3-47
## [40] bitops_1.0-6               ModelMetrics_1.1.0         grid_3.4.0
## [43] nlme_3.1-131               gtable_0.2.0               DBI_0.6-1
## [46] magrittr_1.5               scales_0.4.1               stringi_1.1.5
## [49] reshape2_1.4.2             cowplot_0.7.0              BiocStyle_2.4.0
## [52] iterators_1.0.8            tools_3.4.0                Biobase_2.36.0
## [55] pbkrtest_0.4-7             yaml_2.1.14                AnnotationDbi_1.38.0
```

```
## [58] colorspace_1.3-2           memoise_1.1.0              quantreg_5.33
```