Splicegear package

October 29, 2019

Introducing splicegear

Microarrays have become an established technique for the analysis of gene expression. With the advances in numerical processing of the data, the lowering of the costs and well defined experimental protocols, the reliability of data analysis has increased. The possiblity to study alternative splicing using microarrays has appeared very recently in experimental publications. The word is now 'this is possible', and it has been reported for spotted oligonucleotides arrays, for printed arrays and for arrays made by the *Affymetrix* company. However little has been done to quantify how well the technique performs, how the existing oligonucleotide data could have been influenced by alternative phenomenon and how it could contribute to discover novel splice variants. This package defines classes to handle probe expression values in an alternative splicing context.

The class `SpliceExprSet` combines information about putative splice sites on a sequence with probes matching the sequence and corresponding probe intensities. It it constituted of three attributes, `eset`, `probes` and `spliceSites`, the first being of class `ExpressionSet` the second of class `Probes` and the third of class `SpliceSites`. The idea behind the clear separation between splice site positions, probes and probe intensities is the dynamic nature of their relationship. While the sequence for the probe is fixed, the target sequences can vary slightly and the position or presence of splice sites as well (especially when working with putative splice sites).

The view chosen is appropriate for linking alternative splicing information with microarray data, but is admitedly not very standard on the alternative splicing side. Many representations of alternative splicing show exons as boxes and draw broken lines between segments to show the possible splice variants. The model chosen is not completely incompatible with this view. We present how to use this representation within the package in one of the last sections below.

The package is loaded by a simple call to

```
library(splicegear)
```

Plotting methods

Plotting methods are defined for the classes `SpliceSites` and `SpliceExprSet`.

Exact matching of the reference sequences used by the database of putative splice sites against the probes of *Affymetrix* `U95A` chips were performed. The expression values from the *GeneLogic* 'Dilution' dataset (RNA extracted from cells from the central nervous system and from the liver) were observed.

```
> data(spsites)
> print(spsites)

Alternative splicing sites (SpliceSites):
        seq is 2496 bp long (warning: sequence not included).
        13 type I splice site(s)
An object of class 'AnnotatedDataFrame': none
        2 type II splice site(s)
An object of class 'AnnotatedDataFrame': none
        0 type III splice site(s)

> plot(spsites)
```

Import data

The package has facilities to parse XML structures in a defined XML format (see `http://palsdb.ym.edu.tw/index2.html`) Motivations were discussed in length in a once submitted manuscript. To summarize, we hoped to initiate efficient data exchange for alternative splicing, in the spirit of the WDDX and SOAP formats. The DTD we introduce is more a call for discussion by interested parties. It is not fully compliant with WDDX nor SOAP, althought it may come in the future.

The XML can be stored in a file. In **R**, one can make an object of class *xml* very easily:

```
> library(XML)
> filename <- system.file("extdata", "example.xml", package="splicegear")
> xml <- xmlTreeParse(filename, asTree=TRUE)
```

Further details concerning XML handling can be found in the documentation for the package XML.

The XML structure can be converted to a list of *SpliceSites*:

```
> spsites <- buildSpliceSites(xml, verbose=FALSE)
> length(spsites)

[1] 7
```

```
> show(spsites[1:2])

$Hs.4
Alternative splicing sites (SpliceSites):
        seq is 4080 bp long (warning: sequence not included).
        22 type I splice site(s)
An object of class 'AnnotatedDataFrame'
  rowNames: 1 2 ... 22 (22 total)
  varLabels: tissue histology site
  varMetadata: labelDescription
        1 type II splice site(s)
An object of class 'AnnotatedDataFrame'
  rowNames: 1
  varLabels: tissue histology site
  varMetadata: labelDescription
        0 type III splice site(s)

$Hs.1219
Alternative splicing sites (SpliceSites):
        seq is 2576 bp long (warning: sequence not included).
        50 type I splice site(s)
An object of class 'AnnotatedDataFrame'
  rowNames: 1 2 ... 50 (50 total)
  varLabels: tissue histology site
  varMetadata: labelDescription
        2 type II splice site(s)
An object of class 'AnnotatedDataFrame'
  rowNames: 1 2
  varLabels: tissue histology site
  varMetadata: labelDescription
        0 type III splice site(s)
```

The package is currently able to connect to the 'database of putative alternative splicing' *PALSdb*. The typical way to obtain data from the web is composed of two steps.

1. query a web site and obtain XML in return

2. build **R** objects from the XML

```
xml <- queryPALSdb("alcohol")
spsites <- buildSpliceSites(xml, verbose=FALSE)
```

The class *SpliceSites*

The class *SpliceSites* is a little complex. One should refer to the relevant help file for details. We only introduce here a detailed example of can be performed.

```
> ## build SpliceSites
> library(XML)
> filename <- system.file("extdata", "example.xml", package="splicegear")
> xml <- xmlTreeParse(filename, asTree=TRUE)
> spsites <- buildSpliceSites(xml, verbose=FALSE)
> ## subset the second object in the list
> my.spsites <- spsites[[2]]

> plot(my.spsites)
```
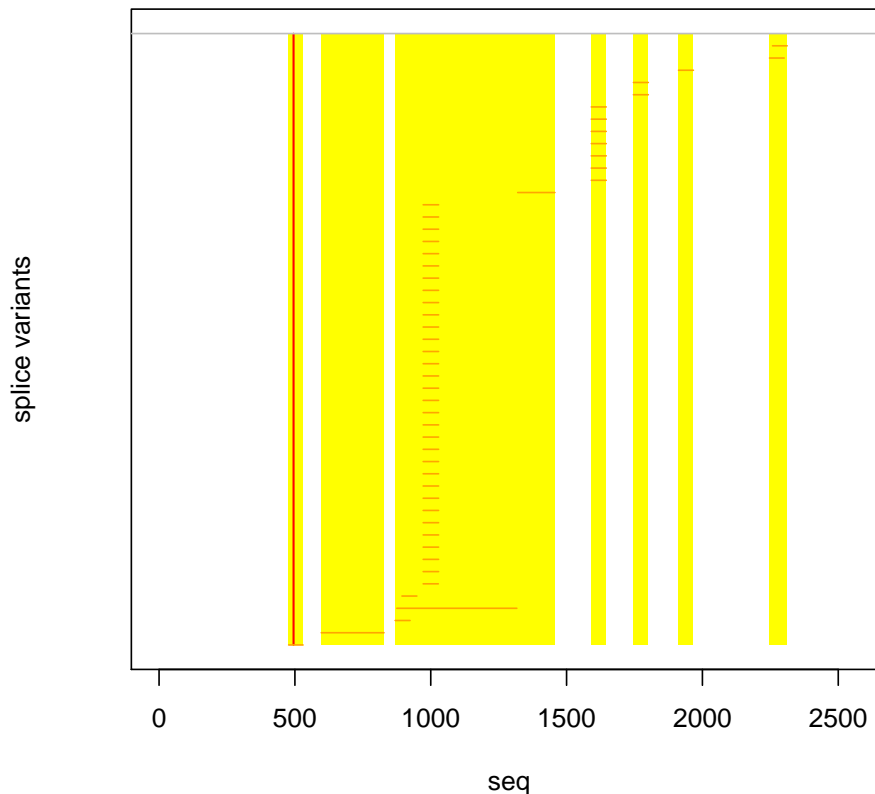

As shown, for most of the putative splice site several ESTs are supporting evidences. One might want to see the tissue distribution of the matches

Data in a *data.frame*

The *data.frame* has a very important role in the S language. A large number of function are designed around this data structure. We provide a way to link the class `SpliceExprSet` with this data structure. The function casting to *data.frame* is named `as.data.frame.SpliceExprSet`. Using the S3 dispatch system, a call to `as.data.frame` with a first argument of class *SpliceExprSet* should be enough.

```
> data(spliceset)
> dataf <- as.data.frame(spliceset)
> colnames(dataf)

 [1] "begin"                             "end"
 [3] "isintypeI"                         "isintypeII"
 [5] "Experiment.ID"                     "Material"
 [7] "Nominal.Liver.RNA.mass..micrograms" "Nominal.SNB.19.RNA.mass..micrograms"
 [9] "Scanner"                           "Liver.Dilution"
[11] "Cell.line.Dilution"                "Full.Mixture"
[13] "exprs"

> lm.panel <- function(x, y, ...) {
+                                    points(x,y,...)
+                                    p.lm <- lm(y~x); abline(p.lm)
+                                 }
>
> ## to plot probe intensity values conditioned by the position of the probes on
> ## the mRNA:
> ## (commented out to avoid a warning)
> ##coplot(log(exprs) ~ Material | begin, data=dataf, panel=lm.panel)
>
```

Further explanations about formulas and models in **S-plus** and **R** can be found easily elsewhere.

Genomic view

A popular representation of splice variants shows exons as boxes, linked by broken lines to show which exons are skipped and which ones are not for the splice variants **??**. In this context, type II and type III splice variants are not relevant: each exon is only likely to be skipped. The package features an experimental class that extends the class *SpliceExprSet* and gives compatibility with this representation.

```
> ## a 10 bp window
> seq.length <- as.integer(10)
> ## positions of the exons
> spsiteIpos <- matrix(c(1, 3.5, 5, 9, 3, 4, 8, 10), nc=2)
> ## known variants
> variants <- list(a=c(1,2,3,4), b=c(1,2,3), c=c(1,3,4))
> ##
> n.exons <- nrow(spsiteIpos)
> spvar <- new("SpliceSitesGenomic", spsiteIpos=spsiteIpos,
+          variants=variants, seq.length=seq.length)
```
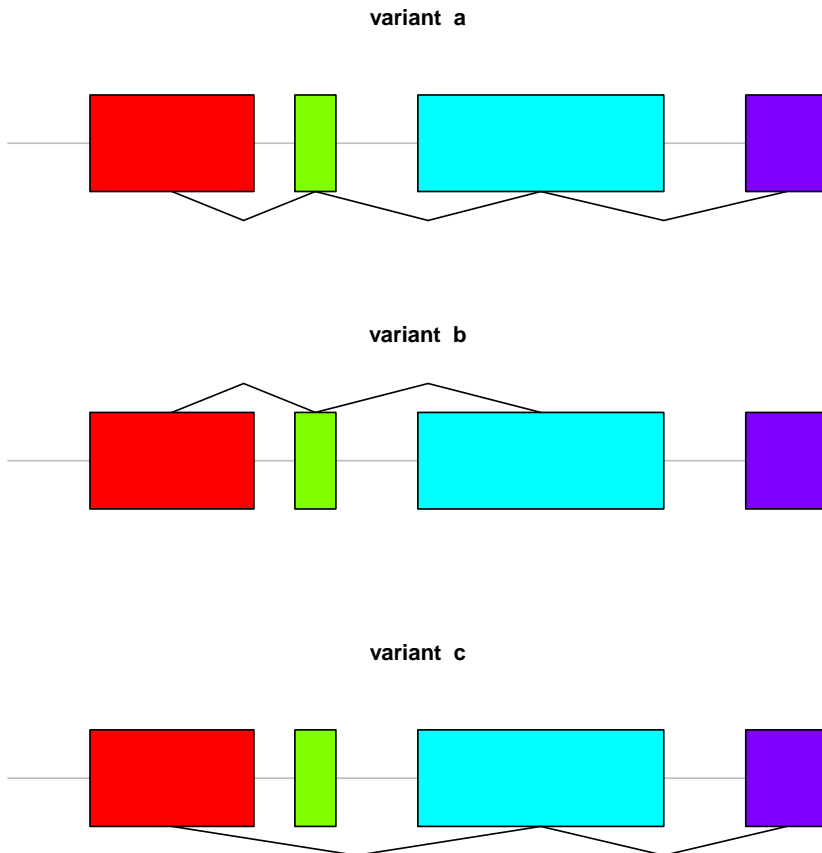
A plotting method (not unlike the display that can be found on the TrEmbl website) is provided.

```
> par(mfrow = c(3,1), mar = c(3.1, 2.1, 2.1, 1.1))
> plot(spvar, split=TRUE, col.exon=rainbow(n.exons))
```

Combining alternative splicing information with probes intensities

The class *Probes* stores information relative to probes matching a reference sequence, primarily their position on the a reference sequence. Additional data concerning the probes can be stored in the slot *info*.

The class *SpliceExprSet* is mainly an aggregation of an instance of class *SpliceSites*, an instance of *Probes* and an instance of class *ExpressionSet*.

The typical procedure is to build *SpliceSites* and their correspondings *Probes* for a type of chip[1]. Experimental hybridizations for the probes are stored in *ExpressionSet* objects. This design facilitates the use of the package in different contexts. For example, once the mapping of the probes has been performed and the data relative to alternative splicing have been acquired, the analyst can re-use these and combine them with data from different hybridization experiments. It also becomes possible to distribute mappings and splice variants information for certain types of chips. This was performed for *Affymetrix* chips and data packages providing the information will be distributed shortly.

---

[1]See the functions `queryPALSdb`, `buildSpliceSites` and the Bioconductor package *matchprobes*.