Standardized output for putative alternative splicing; a R package as an application to combine them with microarray data

Gautier Laurent,* Dao Cong Cedric† and Yang Ueng-Cheng‡

July 11, 2003

Abstract

background A growing proportion of human genes is thought to be subject to alternative splicing. Only a very small fraction of these phenomena are still characterized. Several databases of putative alternative splicing have appeared, mainly using EST data to make their predictions.

results We defined a structured format to exchange data relative to alternative splicing. Our format is not thought to be definitive and authoritative. Our aim is to demonstrate the need for structured data in the context of alternative splicing. We implemented our ideas for the database PALSdb and in a data analysis package. We demonstrate how structured data exchange can be beneficial to the study of alternative splicing phenomena with microarrays.

conclusion The format defined can be used to export data, the cost of parsing being significantly cut. A package to combine microarrays expression data with alternative splicing information was developed and takes advantage of the features introduced. While the format presented has no pretension of being authoritative, it is believed to be flexible enough for many usages, and to provide a base for a more general format to be discussed.

Background

The multiplicity of data formats and their handling has been a common feature of research work in bioinformatics since the early developments of the field. Similar biological data can be structured different ways, with no absolute gain of performance when preferring one method to the others. The intrinsic complexity makes the organization of the data difficult. The views on the data are multiple, the data structures are changing frequently. Three main options can be considered to improve the work of researchers and facilitate queries through

---

*Center for Biological Sequence Analysis, Building 208, Technical University of Denmark, 2800 Lyngby
†National Yang-Ming University, Institute of Biochemistry, No. 155, Sec. 2, Li-Nong St., Taipei, Taiwan 11221
different data banks. One option can be to define a standard format for the data. To our knowledge, many of the efforts of standardization did not gain massive following. We believe this is due to the youth and dynamics of the field. While gaining maturity, a clearer view of what is needed and a consensus about how data can be organized will be achieved. The efforts of the Gene Ontology Consortium for genes annotation illustrate this point (it is not a data format strictu senso, but standard annotation rules and vocabulary). We also believe that a golden standard cannot be defined in many cases. Another option was introduced by the originally academic project SRS. It aimed at integrating any given data banks under one query interface. A drawback of this approach is that it requires having all the data banks installed at the same place. The maintenance cost and the hardware resources increase as updates are frequent and the data banks of significantly large size. The third option is to structure data formats and facilitate data exchange through computer programs. The XML format played an important role in the development of this option. Very complex structures be kept as they are. The markups will ensure an easy access to any given value. When the whole dataset is only accessible through a web front-end, the data displayed can lack any precise format. The back-end can be a relational database, a collection of flat files, links to other databases, links to other front-ends on the web, or a mix of some of these elements. A subset of the data is displayed dynamically by a web server in answer to a request from a user. While scientific work related to the data server can be published, the whole data are not directly available. An obvious reason for complex structures would be the installation procedure. One could hardly deploy the complete architecture locally without tremendous efforts. Moreover, when the data are constantly updated, the benefit of having a local install can be lower than the cost to have it. In a non-negligible number of cases, intellectual property related considerations prevent the unconditional distribution of the whole data. In those cases too, XML appears as an interesting option to export the result of queries. The popular databases PubMed, GENBANK and NCBI's BLAST offer XML display of their data.

The database of putative alternative splicing PALSdb[2] contains the result of matching reference sequences against the EST sequences from their respective Unigene clusters and against sequences in dbEST. Mismatching segments (i.e. gaps) are presented as supporting evidences for putative alternative splicing. For an alignment between a reference sequence and an EST sequence, when the gap is on the reference sequence this is called a type I event and when the gap is on the EST this is called a type II splicing event, as detailed in the reference for PALSdb. By extension, the matches to the reference sequence are also called supporting evidences. This approach gives a representation where three types of alternative splicing are distinguished. This point is developed below. The data can be accessed through a web front-end: http://palsdb.ym.edu.tw/. The result of a query is typically a list of reference sequences associated with their respective supporting evidences. We refer to a set { reference sequence + supporting evidences + putative splice variants } as a PALSdb entry. The individual clusters can be visualized in HTML, click-able links to external resources being provided: TrEmbl, OMIM to name few.

Alternative splicing gained a growing interest with the recent announcement

---

[1] mRNA or EST sequences

2
of the completion of the sequencing of the human genome. The number of genes in the genome is still debated, but the numbers announced trend to be smaller with time: 100,000 genes according to the Human Genome Project in 1990, then around 30,000 genes in the historical publication in Nature in 2001. A very recent rumor would give us barely 10% of genes more than *Cenorhabditis elegans*. In meanwhile, the estimated percentage of genes subject to alternative splicing raised up to 50% \(^\text{[3]}\). There is much still to discover. Microarrays have been advertised as `high-throughput’ devices, thus have been thought after to monitor alternative splicing phenomena. Only few published works report a demonstrative use in a context of alternative splicing\(^\text{[4, 5, 6]}\), but we believe them to be the few exploits preceding a wider use of the technique.

**Results**

Our aim is not to propose a definitive format, but rather to offer a structured format for data exchanges. In our approach, if a standard emerges, it will do so because of a wide acceptance by the community of users. To have a working demonstration of what can be done with a structured format in the context of microarrays data analysis will hopefully initiate a common reflexion between research groups in the field.

**Export of structured putative alternative splicing data**

It was chosen to keep the existing query interface and return results in XML. To allow a reasonable level of automation once a query has been formulated, the individual reference sequences in the result are bind in a collection of the structure detailed above. This differs from the way the PubMed website offers XML output. In this case, the individual results can be converted to XML. One interested in XML would still have to use parsers or use the separate query CGI *pmFetch*. In our case, both outputs are accessible through a unified query interface. Setting the parameter *format* to ‘xml’ is sufficient to switch from the regular HTML output to the XML. This ensures a natural correspondence between XML data fetched and an interactive consultation of the website.

The XML data gather the information contained in the click-able plot displayed in the HTML version of the *PALSdb* website. Thus, it is possible to generate a similar picture only from the XML data.

We can summarize the global structure by few key points. The result from a query is bounded by the tags:

\(<\text{ResultQuery}>\) \(</\text{ResultQuery}>\) The result itself is constituted of few practical data and a sequence of *Entries*. The few practical data are such as the query submitted itself, as it provides a convenient way to submit the same query when the database is updated and compare the results, and such as the status of the processing of the query, which provides a way to handle errors on the client side and a way to generate error messages can be easily processed on the server side.

Each element in the sequence of *Entry* is described in a structure delimited by the tags

\(<\text{Entry}>\) \(</\text{Entry}>\). The global structure for a request returning two of them would be like:

\(<\text{ResultQuery}>\)
... query submitted, status, etc...

<Entry>
  .... details for the first entry
</Entry>

<Entry>
  .... details for the second entry
</Entry>

</ResultQuery>

In the case of PALSdb, an entry can be thought of as a mRNA with the EST matches suggesting the existence of splice variants. An Entry itself is constituted by information related to the reference sequence, and by a sequence of type I and type II alternative splicing events. Each one of these events is described in a structure delimited by the tags <Alt-splice> and </Alt-splice>. The associated supporting evidences data are nested within the corresponding Alt-splice, in their own <Hit-info> / </Hit-info> structure. The general representation for an entry with two skipped exons would look like:

<Entry>
  <Reference-sequence>
    ... data for the reference sequence
  </Reference-sequence>

  <Alt_splice>
    ... first exon skipped. Two EST matches support its existence.
    <Hit-info>
      ... first match.
    </Hit-info>
  </Alt_splice>

  <Hit-info>
    ... second match.
  </Hit-info>

  <Alt_splice>
    ... second exon skipped. Only one EST match supports its existence.
    <Hit-info>
    </Hit-info>
  </Alt_splice>

</Entry>

The related data are the ID of the match, tissue information, histology information, etc... One will refer to the fully documented DTD for a comprehensive description of the features.

Having a structured output facilitates the integration of meta-data (defined as 'structured data about a resource'). A first example of application would be the embedding of data from one web page into another. This would allow web pages external to the primary source of information to offer alternative views on the data (eventually combining different sources). As shown in the
XML data could be queried from a server by another server, itself in response to a query. The structured information can be easily parsed out to be used. An application would be to offer to display with common rules graphics from different relevant websites. One could think of it as an extension of the idea of hyperlinks. This idea has been discussed in a not too distant past, but remained mainly on strictly technical forums. The Web Distributed Data exchange (WDDX) and the Simple Object Access Protocol (SOAP) technologies, both based on XML, are the main alternatives. In our opinion, the bioinformatics community probably did not pay yet the attention these efforts deserve. However encouraging comments are appearing [7]. WDDX appears like a lighter-weight protocol than SOAP and could suit our particular needs better. On the other hand, SOAP is gaining a larger popularity, but the specifications of format are still being worked on. The version 1.2 of the specifications are only dated May, 2003. We remain currently with XML to keep the structures as simple as possible in order to validate the concept and call for participation with other groups hosting databases of alternative splicing. The move to one of the standards will come at later stage.

The second example is less general, but should stimulate the reader’s imagination about applications. When building microarrays to study alternative splicing phenomena, an efficient handling of alternative splicing data is required. If the data have to be retrieved from other places, like external databases, precise definition of the storage format has to be known. When the data are accessible through web front ends, which is mostly the case for alternative splicing to our knowledge, XML allows an efficient handling of the data. A scenario would be that a user enters keywords for a pathway, a disease, or any other biological context and retrieve the set of genes with putative (or confirmed) splice variants. He would then design probes and make a custom microarray, or check if an existing oligonucleotide array can be used by matching the probes to the mRNA sequences in the database. The data structure chosen is close to the representation used in PALSdb. More particularly, the putative splice variant can be of type I, II or III. This representation suits the combination with oligonucleotide microarray data. The reference sequence is a mRNA against which probes are matched, as presented below. However, it has the drawback of being apparently incompatible with the most commonly chosen approach: the mapping of the exons to the genomic sequence. The representation of splice variants is done by linking exons by broken lines. Two examples of similar representations are shown in Figure 2.

To demonstrate the possibilities of our approach, we designed a R package to study alternative splicing phenomena. A primary concern was to link expression data from existing high-density oligonucleotide arrays to the putative alternative splicing in PALSdb. The building of custom made oligonucleotide arrays to monitor the expression of known splice variants, or to verify experimentally the existence of predicted or putative splice variants can be addressed by our package. The importance of alternative splicing in the making of high-density nucleotide arrays is still relatively minor in recently published work 8, but we forecast this to change in the near future. At least commonly found splice variants should be considered, as they are of significant biological importance.
To ensure these features we designed a class SpliceExprSet that is merely an aggregation of three classes (Figure 3). The first is the class Probes. It contains the position for probes on a given sequence, plus optional information for the probes. The second class is SpliceSites. It contains alternative splicing information related to the same sequence. The third class is exprSet. This class is widely used in the Bioconductor project to hold expression values and associated informations. In our package it contains the intensity values for the probes (described in Probes). The design allows to work very efficiently on alternative splicing and oligonucleotide arrays. The three components of the class SpliceExprSet just have to be replaced when needed by the analyst. To make things clearer we present two scenarios:

- A researcher wants to study putative alternative splicing related to alcohol dehydrogenase.
  1. He enters the keywords *alcohol dehydrogenase*, using the package to query PALSdb.
  2. The package retrieves the results in XML and builds the corresponding SpliceSites objects.
  3. Using the tools of his choice, the user designs suitable probe sets to make an ad-hoc microarray. Information related to the probe sets are stored in Probes objects.
  4. After making hybridization experiments with the newly designed microarrays, the probe expression values are stored in exprSet objects. This will happen naturally when using package of the Bioconductor project to pre-process the chips data.
  5. SpliceExprSet objects can be built from the three components SpliceSites, Probes and exprSet. And the results analyzed further using the package, as briefly outlined below.

- A researcher uses an existing chip and wants to study alternative splicing events using different sets of experiments.
  1. He obtained the SpliceSites and Probes objects corresponding to the probe sets on the chip, or did the mapping of the probes to the reference sequences himself.
  2. He makes hybridization experiments for his biological assays to study alternative splicing phenomena (or validate experimentally predicted alternative splicing). The results are stored in objects of class exprSet.
  3. Other hybridization experiments can replace the one mentioned above, or he can develop routine procedure to detect alternative splicing and do high-throughput screening. The only components to change will be the ones of class exprSet.

We applied the second scenario to perform the mapping of the probes for few Affymetrix arrays to every reference sequence considered for the build of PALSdb. The behavior of the probes being in sections of the reference putatively involved in alternative splicing were studied (Laurent Gautier, manuscript in preparation).
The general concept used to design the package, the features offered by \texttt{R} and its many packages provide a powerful environment to explore the data, by the means of graphical representations or by the means of highlighting features using statistical tools. High-level plotting functions are provided. One can combine information about a reference sequence, putative alternative splicing events, matching probes and their experimental intensities in hybridization experiments on the same figure (see Figure 4). The choice we made to represent the ESTs which support the existence of alternative splicing event is somewhat unconventional. We did not represent the alignments but the gaps in the alignments (horizontal bars in the lower part of the left panel in Figure 4). This leads to clearer plots, especially when combining with probes and expression data. Facilities allowing to work intuitively with \texttt{R} modeling capabilities are also provided. XML parsers for the DTD defined for our needs are included in the package. The PALSdb database exports metadata in the format described. \texttt{R} objects for these data can be obtained directly through a HTTP connection, as introduced in the package \texttt{annotate}\cite{9}. Exploring features of the data becomes extremely simple, as the short example presented in the Figure 5 demonstrates it. As expressed earlier, the exchange of structured meta-data was a strong element in our work. We took care to make the package independent from an exclusive linking to \texttt{PALSdb}, and the data in the XML format it exports. This feature allows the use of the package with XML files resulting from queries on \texttt{PALSdb} stored locally, or the use of the package with any other sources exporting splicing informations in the format detailed in the \texttt{DTD}. Other structured alternative splicing data can be imported in the package. One would only need to write the mapping between the structure of a new XML format to the \texttt{R} objects defined in the package \texttt{splicegear}. The package is available on the \texttt{Bioconductor} website, a complete documentation is included in the package.

\section*{Discussion}

With the growing interest in alternative splicing, having structured ways to exchange data would be beneficial to researchers. We introduce an XML-based data format and an application that takes a real advantage from this.

As mentioned earlier, the data structure proposed appears hardly compatible with the popular representation in which the exons are positioned on the genomic sequence. This is not completely the case. The XML structure presented can accommodate this: all the alternative splicing events have to be of type I, as exons can only be skipped or not. For each gene an attribute detailing the possible combinations of skipped exons, \textit{i.e.} indicate which exons coexist with each others in the same splice variant, have to be included. This particular attribute can also represent a helpful data structure for researchers with interest in the combinatorial of exons\footnote{Not yet studied to our knowledge.}. We included in the \texttt{R} package an experimental class to handle exons on genomic sequences. We also wrote a simple plotting method producing display like the ones shown in the Figure 2. We sincerely hope that the open nature of our effort will lead to a discussion about export data structure, for the benefit of the research community. We wish to share the development of the data structure with interested contributors.
Methods

The implementation of the export of the XML data structure discussed on the server side was done in Perl using mostly notably the package CGI. The PALSdb web server can be accessed at:

http://palsdb.ym.edu.tw/index2.html

The package splicegear was written in R. It makes use of other packages, namely XML, Biobase and annotate. R itself is a GNU implementation of the S language. It is freely available under the GPL license. All the third-party packages and splicegear itself are freely available with their source code under GPL-like licenses. splicegear and related material can be downloaded from its web page:

http://www.cbs.dtu.dk/laurent/download/splicegear

Authors contribution

Laurent Gautier designed the data structures, implemented the corresponding DTD, designed and implemented the R package splicegear. Cedric Dao had input on the DTD, implemented the export of query results in XML, and dealt with various issues concerning the web server. Ueng-Chen Yang supervised the development of the extensions to the PALSdb database, hosts Laurent Gautier and Cedric Dao Cong, and obtained funding for Mr. Dao. All the authors reviewed and agreed on the manuscript.

Acknowledgments

The work of Laurent Gautier is funded by a grant from the Danish Biotechnology Instrument Center. He would like to acknowledge Steen Knudsen who supervises his Ph.D. thesis and found funding for it. PALS db was supported by grant NSC91-3112-B-010-013 from National Science Council, Taiwan. Cedric Dao was supported by the same grant during his stay at YMU.

References


Figure 1: Export of data from the PALSdb server. The common use is to query the web server through a browser and receive the resulting data in HTML in the same browser (A.). (B.) presents differs from (A.) only because the results are exported in XML. They have to be processed by the user. The option (C.) shows how a third-party server can query PALSdb (C.2) to give supplementary results to a query (C.1). The XML returned (C.3) will processed by this server (to extract the data of interest or to generate a plot fitting a given theme). The end-user receives transparently the result in his browser (C.4).
Figure 2: Other representations of alternative splicing. (top) The HASDB website presents splice variants as connected exons. The representation of genes for which multiple splice variants exist can give very complex figures. (bottom) The TrEmbl website shows skipped exons as white filled boxes. A view of the exons on the genome is also available (not shown here).

Figure 3: UML representation of the classes in the package. The class SpliceExprSet is an aggregation of the classes Probes, SpliceSites and exprSet, holding information about microarray probes, splice sites and experimental intensity values for the probes respectively. Only the main attributes are represented.
Figure 4: High-level plot performed by the R package splicegear. The probes on an Affymetrix chip U95A were aligned with the reference sequence used by PALSdb for the Unigene cluster Hs.572. Probe level intensities from the Gene Logic’s Dilution dataset were used (freely available upon request at http://qolotus02.genelogic.com/datasets.nsf/). (left) The reference sequence is represented as an horizontal gray line splitting the plot in two areas. The upper area shows the probes matching the reference, while the lower area shows the gaps in aligned ESTs suggesting that exons are spliced out (orange segments). The yellow rectangles in the background indicate ‘putatively spliced out’ areas. (right) The intensity values obtained for the probes in different experiments are plotted as connected lines (here colored according to the nature of the RNA samples: ‘liver’ samples in red and ‘brain’ samples in blue).
library(splicegear)

## query PALSdb
xml <- queryPALSdb("Hs.71119", fields="ug.id")

## construct R objects from the XML returned
spsites <- buildSpliceSites(xml)[[1]]

pdat <- pData(spsites$spsite$pos$pdata)

## make the contingency table (tissue/site)
stable <- table(pdat$site, pdat$tissue)

## color scheme for the different sites
ccol <- rainbow(nlevels(pdat$site))

## plot the splicing events and EST matches gaps
plot(spsites, col.typeI=ccol[as.integer(pdat$site)])

## barplot of the tissue information
barplot(stable, col=ccol)

Figure 5: Application of the package. (top) Short example of R code to create easily a new view on the data. The XML returned by the server is converted into SpliceSites objects. A contingency table is computed by R, then the default plotting method for an object of class SpliceSites is called, then the function barplot. (bottom) The resulting plots: The gaps in the alignment of EST sequences (supporting evidences for type I splicing events) and a barplot of the tissue information from the EST hits colored according to by sites. One can clearly see that the type I putative event (colored in blue) can be found in many different tissues.