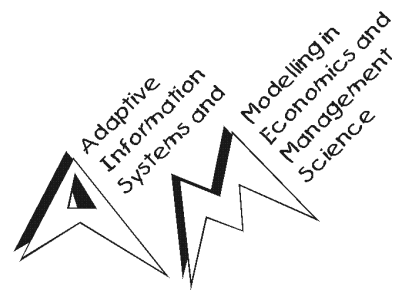


Report Series



StatDataML—An XML Format for Statistical Data

David Meyer
Friedrich Leisch
Torsten Hothorn
Kurt Hornik

Report No. 75
October 2002

Report Series



October 2002

SFB
'Adaptive Information Systems and Modelling in Economics and Management
Science'

Vienna University of Economics
and Business Administration
Augasse 2–6, 1090 Wien, Austria

in cooperation with
University of Vienna
Vienna University of Technology

<http://www.wu-wien.ac.at/am>

Papers published in this report series
are preliminary versions of journal articles
and not for quotations.

This paper was accepted for publication in:
D. Meyer, F. Leisch, T. Hothorn, and K. Hornik. StatDataML: An XML format for
statistical data. In W. Härdle and B. Rönz, editors, *Compstat 2002 —
Proceedings in Computational Statistics*, pages 545–550. Physika Verlag,
Heidelberg, Germany, 2002. ISBN 3-7908-1517-9.

This piece of research was supported by the Austrian Science Foundation
(FWF) under grant SFB#010 ('Adaptive Information Systems and Modelling in
Economics and Management Science').

Abstract

In order to circumvent common difficulties in exchanging statistical data between heterogeneous applications (format incompatibilities, technocentric data representation), we introduce an XML-based markup language for statistical data, called *StatDataML*. After comparing *StatDataML* to other data concepts, we detail the design which borrows from the language S, such that data objects are basically organized as recursive and non-recursive structures, and may also be supplemented with meta-information.

1 Introduction

Data exchange between different tools for data analysis and data manipulation is a common problem: different applications use different and often proprietary and undocumented formats for data storage. Import/export filters are often missing or insufficient, and if ever, focus on technical aspects (like storage modes and floating point specifications) in spite of supporting conceptual representation issues (like scales or representation of categorical data). The currently high costs for data exchange hence could be significantly reduced by the use of a well-defined common data exchange standard, if only because software packages would just need to provide one single mechanism.

The aim of this paper is to introduce such a data exchange standard for statistical data: the XML-based markup language *StatDataML*. The design borrows from the language S (see e.g., [Chambers, 1998](#)), such that data objects are basically organized as recursive structures (lists) and non-recursive structures (arrays), respectively.¹ Additionally, each object can have an attached list of properties (corresponding to S attributes), providing storage of meta-information.

2 Requirements on Statistical Data

Statisticians need a data format which is both flexible enough to handle all different kinds of statistical data (from time series to decision trees), and specialized enough to incorporate statistical notions like scales and factors. Such a data format should feature:

- special symbols for infinities and undefined values
- special symbols for missingness (“not available”)
- logical data
- categorical data (unordered, ordered or cyclic²)
- numeric data (in the storage modes integer, real and complex)
- character data (strings)
- date/time information
- vectors (objects with elements of the same type)
- lists (objects with—possibly different—elements of any type)

Vectors should be indexable arbitrarily—in order to build matrices or multidimensional arrays. Lists allow for complex and even recursive structures (for they can contain lists again).

Table 1 compares some software products regarding these criteria: two families of mathematical programming languages (Splus³/R⁴ and Matlab⁵/Octave⁶), statistical software (SPSS⁷, SAS⁸, Minitab⁹) and spreadsheets (Excel¹⁰, StarCalc¹¹, Gnumeric¹²). In spreadsheets and Matlab/Octave, nominal data can only be represented by strings. Arrays of arbitrary dimension are supported by Splus/R and Matlab only. Complex numbers are only supported by Splus/R and Matlab/Octave. The latter cannot handle missingness. IEEE special values are not supported by Excel, StarCalc, SPSS, SAS and Minitab.

¹See [Temple Lang & Gentleman \(2001\)](#) for a more specific approach representing S objects in XML.

²like days of a week

³For Splus see: www.insightful.com

⁴For R see: [Ihaka & Gentleman \(1996\)](#) and www.R-project.org

⁵For Matlab see: www.mathworks.com

⁶For Octave see: www.octave.org

⁷For SPSS see: www.spss.com

⁸For SAS see: www.sas.com

⁹For Minitab see: www.minitab.com

¹⁰For Excel see: www.microsoft.com

¹¹For StarCalc see: www.staroffice.com and www.openoffice.com

¹²For Gnumeric see: www.gnumeric.org

	R/Splus	Matlab	Octave	Excel/ StarCalc	Gnumeric	SPSS	SAS	Minitab
IEEE	yes	yes	yes	no	yes	no	no	no
NA	yes	no	no	yes	yes	yes	yes	yes
logical	yes	(yes)	no	yes	yes	yes	yes	yes
nominal		strings	strings	strings	strings	coding	yes	strings
unordered	yes					yes	yes	yes
ordered	yes					yes	yes	yes
cyclic	no	no	no	no	no	no	no	no
numeric	yes	yes			yes			
integer	yes	yes	no	no	no	(yes)	yes	no
real	yes	yes	yes	yes	yes	yes	yes	yes
complex	yes	yes	yes	no	no	no	no	no
character	yes	yes	yes	yes	yes	yes	yes	yes
date/time	yes	yes	yes	yes	yes	yes	yes	yes
arrays	yes	yes	no	no	no	no	no	no
matrix	yes	yes	yes	yes	yes	yes	yes	yes
lists	yes	yes	yes	no	no	no	no	no

Table 1: Data representation capabilities of different software packages

3 StatDataML

3.1 StatDataML is XML

For “statistical data” one would usually think of such things as tabular data, time series objects, responses and regressors or contingency tables. Programs that produce such data store it on disk, using either a binary format or a text format. `StatDataML` files are XML files, thus ordinary text files, with extension `.sdml`, containing several XML elements (so called *tags*), which can be formally described with a special data definition language (DTD)—see the [World Wide Web Consortium \(2000\)](#) recommendation. In the following, we will go through the rules in the `StatDataML.dtd` file.

3.2 The File Header

```
<!ELEMENT StatDataML (description?, dataset?)>
```

The top level `StatDataML` element contains one `description` and one `dataset` element, each optional. It should contain the `StatDataML` namespace:

```
<StatDataML xmlns="http://www.ci.tuwien.ac.at/StatDataML">
  ...
</StatDataML>
```

3.3 The description element

```
<!ELEMENT description (title?, source?, date?, version?,
                       comment?, creator?, class?, properties?)>
```

```
<!ELEMENT title (#PCDATA)>
<!ELEMENT source (#PCDATA)>
<!ELEMENT date (#PCDATA)>
<!ELEMENT version (#PCDATA)>
<!ELEMENT comment (#PCDATA)>
<!ELEMENT creator (#PCDATA)>
<!ELEMENT class (#PCDATA)>
```

```
<!ELEMENT properties (list)>
```

The `description` element is used to provide meta-information about a dataset that is typically not needed for computations on the data itself. It consists of eight elements: `title`, `source`, `date`, `comment`, `version`, `creator` and `class` are simple strings (PCDATA), whereas `properties` is a list element (see next section). `date` should follow the ISO 8601 format (see below). The `creator` element should contain knowledge about the creating application and the StatDataML implementation, `properties` offers a well-defined structure to save application-based meta-information, and, finally, the `class` element will contain the class name, if any. There are some discussions about meta data in statistics¹³: one could think of extending the `description` element in such that extended information with logical markup can be stored.

3.4 The dataset element

```
<!ELEMENT dataset (list | array)>
```

We define a `dataset` element either as an array or as a list. We use arrays and lists as basic “data types” in StatDataML because every data object in statistics can be decomposed into a set of arrays and lists (as in Splus/R, or like the corresponding arrays and cell-arrays in Matlab). The basic property of a list is its recursive structure, in contrast to arrays which are always non-recursive. If one thinks about data as a tree, lists would be the branches and arrays the leaves.

3.4.1 Lists

A list contains three elements: `dimension`, `properties` and `listdata`:

```
<!ELEMENT list (dimension, properties?, listdata)>
<!ELEMENT listdata (list | array)*>
```

```
<!ELEMENT dimension (dim*)>
<!ELEMENT dim (e*)>
<!ATTLIST dim size CDATA #REQUIRED>
```

The `dimension` element contains one or more `dim` tags, depending on the number of dimensions. Each of them has `size` as a required attribute, and may optionally contain up to `size` names, specified with `<e>...</e>` tags. Note that arrays, like the whole dataset, can also have additional `properties` attached, corresponding, e.g., to attributes in S. In fact, the name “properties” was chosen because the name “attribute” is already used in XML itself. The `listdata` element may either contain arrays (with the actual data), or again lists, which allows for complex and even recursive structures.

3.4.2 Arrays

```
<!ELEMENT array (dimension, properties?, (data | textdata))>
```

Arrays are blocks of data objects of the same elementary type with dimension information used for memory allocation and data access (indexing). The first two elements, `dimension` and `properties`, are identical to lists, only the `listdata` block is replaced by the `data` (or `textdata`) element which contains the data itself.

The data tag

```
<!ELEMENT data (e|ce|na)*>
<!ATTLIST data
  true CDATA "1"
  false CDATA "0"
  type (logical|nominal|numeric|character|datetime) "character"
  mode (unordered|ordered|cyclic|integer|real|complex) #IMPLIED>
```

¹³e.g., <<http://www.gla.ac.uk/External/RSS/RSScomp/metamtg.html>>

```

<!ELEMENT na EMPTY>

<!ELEMENT e (#PCDATA|posinf|neginf|nan)* >
<!ELEMENT posinf EMPTY>
<!ELEMENT neginf EMPTY>
<!ELEMENT nan EMPTY>

<!ELEMENT ce (r,i) >
<!ELEMENT r (#PCDATA|posinf|neginf|nan)* >
<!ELEMENT i (#PCDATA|posinf|neginf|nan)* >

```

If `data` is used (especially recommended for character data), then each element of the array representing an existing value is encapsulated in `<e>...</e>` (or `<ce>...</ce>` for complex numbers). For missing values, `<na/>` has to be used, empty values are just represented by `<e></e>`.

The `type` attribute specifies the statistical data type, as logical, nominal, numeric, character and date/time. The optional mode attribute allows for further specification: nominal data could be `unordered` (“factors”), `ordered` and `cyclic` (e.g., days of week), whereas numeric data could be `integer`, `real` or `complex`.

As an example, consider a character dataset formed by color names, with one value missing (after green), and one being empty (after blue). The corresponding data section would appear as follows:

```

<data type="nominal" mode="unordered">
  <e>red</e> <e>green</e> <na/> <e>blue</e> <e></e> <e>yellow<e>
</data>

```

IEEE Number Format

The implementation is responsible for the correct casts. The number format has to follow the IEEE Standard for Binary Floating Point Arithmetic (see [Institute of Electrical and Electronics Engineers, 1985](#)), which is implemented by most programming languages and system libraries. However, the IEEE special values `+Inf`, `-Inf` and `NaN` must explicitly be specified by `<posinf/>`, `<neginf/>` and `<nan/>`, respectively, to facilitate the parsing process in case the IEEE standard were not implemented. These special values could appear as follows:

```

<data type="numeric" mode="real">
  <e>1.23</e> <e><posinf/></e> <e><nan/></e> <e>2.43</e>
</data>

```

Complex Numbers

Complex numbers are enclosed in `<ce>...</ce>` tags which contain exactly one `<r>...</r>` tag (for the real part) and one `<i>...</i>` tag (for the imaginary part). Apart from that, the same rules as for `<e>...</e>` apply:

```

<data type="numeric" mode="complex">
  <ce> <r>12.4</r> <i>1</i> </ce>
</data>

```

Date and Time Information

Data of type `datetime` has to follow the ISO 8601 specification (see [International Organization for Standardization, 2000](#)). StatDataML should only make use of the complete representation in extended format:

CCYY-MM-DDThh:mm:ss±hh:mm

For example, the 12th of March 2001 at 12 hours and 53 minutes, UTC+1, would be represented as: 2001-03-12T12:53:00+01:00.

3.5 Reference Implementations

Software is available for Matlab, Octave, Splus, R, Gnumeric; a conversion tool for SPSS is currently under development.

4 Conclusion

StatDataML seems general and flexible enough to cover most of statisticians' data representation needs, although one might miss features like data inheritance and support for distributed data (e.g. via URLs or SQL queries on databases). The representation of high level objects (like time series) and statistical models are currently under investigation.

References

- Chambers, J. M. (1998). *Programming with Data: a guide to the S Language*. Springer.
- Ihaka, R. & Gentleman, R. (1996). R: A language for data analysis and graphics. *Journal of Computational and Graphical Statistics*, 5(3), 299–314.
- Institute of Electrical and Electronics Engineers (1985). *IEEE Standard 754-1985 (R 1990), Standard for Binary Floating-Point Arithmetic*.
- International Organization for Standardization (2000). *ISO 8601:2000, Data elements and interchange formats - Information Interchange - Representation of dates and times*.
- Temple Lang, D. & Gentleman, R. (2001). RXMLObjects: Reading and writing S objects in XML. S Package. <http://www.omegahat.org/RXMLObjects>.
- World Wide Web Consortium (2000). *Extensible Markup Language (XML), 1.0 (2nd Edition)*. Recommendation 6-October-2000. Edited by Tim Bray (Textuality and Netscape), Jean Paoli (Microsoft), C. M. Sperberg-McQueen (University of Illinois at Chicago and Text Encoding Initiative), and Eve Maler (Sun Microsystems, Inc. - Second Edition). Reference: <http://www.w3.org/TR/2000/REC-xml-20001006>.