Flexible, platform-independent, open-source data structures and conversion mechanisms for paleolimnological data

DUNNINGTON, Dewey

Department of Earth and Environmental Science, Acadia University, Wolfville, Nova Scotia B4P 2R6, Canada

Introduction

Paleolimnological data by nature is complex and multidimensional, which makes the storage, visualization, and manipulation of such data inherently challenging. Dedicated programs such as C2, Tilla, the R package 'rioja', and the R package 'analogue' have attempted to ameliorate these issues, but several recurring problems exist:

1. The storage format can only be opened by a single application. If the raw data used is stored in a format that can only be opened by a single program (e.g. Microsoft Excel, Tilla, SigmaPlot), multiple copies of the data may have to be stored which increases the chance of accidental error occuring in the process of conversion and make keeping an updated copy of the data in all programs used more difficult.
2. The storage format is restrictive and does not allow the storage of error, non-detect, or other non-numeric data alongside quantitative observations. Generally the format in which paleolimnilogical data are stored makes it difficult to store non-detect values, multiple values for a single depth, or error alongside the measured value for a parameter. For many cases (e.g. species assembledge with no error assessed) this does not present a problem, but for bulk geochemical studies of lake sediment, error is essential and reporting this data is made difficult by the inability to store error alongside the measured value.

3. The application is platform specific and is not available for Windows, Mac OSX, and/or Linux. Until recently, Windows was considered the platform for scientific software, but due to the adoption of software packages such as R (R Core Team 2013) and Python that operate across platforms, this is no longer the case. Using applications that are only available for a single platform restricts the ability to reproduce and/or redistribute results.

This poster is an attempt to articulate open-source, flexible, and platform-independent data structures for paleolimnological data such that results can be reproduced and redistributed with minimal restrictions. This paper also attempts to provide examples of how these formats can be manipulated for the purposes of storage, analysis, and visualization, such that paleolimnologists can utilize readily available data manipulation tools to easily and flexibly create effective visualizations and robust numerical analysis.

qualifiers				tags	
core	depth	parameter	value	sd	n
MAJ-1	0	Ca	1884.7	452.4	3

Figure 1. Data values can have qualifiers, or elements that give meaning/ context to the value, and tags, which store extra information about the particular value. Here the tags are numeric, but notes or qualitative information can also be stored.

Summary

The four data structures discussed in this poster all have advantages and disadvantages in terms of usefulness for storage, visualization, and analysis. As has been demonstarated in the R implementation of these data structures, conversion between these structures is not always possible, especially between the summarised and non-summarised variants of the list and matrix structures. In general, the more useful the data structure, the less information it is able to store. As more advanced statistical treatment of data becomes easier, it may become adventageous to store such data in such a way that no data is lost (values list) since conversion to more useful data structures will always be possible. Of the platforms for data manipulation covered in this poster, R is by far the most user-friendly. The graphical user interface of Excel may be tempting but is inefficient for manipulation of more than a few parameters. Python offers solutions to conversion between data structures, but does not offer out of the box creation of publication-quality diagrams.

Values list

core	depth	parameter	param.value
MAJ-1	0	Ca	1623
MAJ-1	0	Ca	1624
MAJ-1	0	Ca	2407
MAJ-1	1	Ca	1418

...

Choice: handling of non-detects, type of error

```
melted %>%
  group_by(core, parameter, depth) %>%
  summarise(value=mean(param.value),
            sd=sd(param.value),
            n=length(param.value))
```

Data loss: replicates are summarised

Summarised values list

core	depth	parameter	value	sd	n
MAJ-1	0	Ca	1884.7	452.4	3
MAJ-1	0	Ti	2369.7	401.1	3
MAJ-1	0	V	78.3	9.2	3
MAJ-1	1	Ca	1418.0	NA	1
MAJ-1	1	Ti	2409.0	NA	1
MAJ-1	1	V	70.0	NA	1

...

```
original_data %>%
  melt(id.vars=c("depth", "core"),
       variable.name = "parameter",
       value.name = "param.value")
```

Note: These R commands use the **dplyr** (Wickham and Francois 2016) and **reshape2** (Wickham 2016) packages.

```
library(dplyr)
library(reshape2)
```

Choice: value, sd, or n?

```
melted_summarised %>%
  dcast(core + depth ~ parameter,
  value.var="value")
```

Data loss: only one value type preserved

Values Matrix

core	depth	Ca	Ti	V
MAJ-1	0	1623	2104	73
MAJ-1	0	1624	2174	73
MAJ-1	0	2407	2831	89
MAJ-1	1	1418	2409	70
MAJ-1	2	1550	2376	70
MAJ-1	3	1448	2485	64
MAJ-1	4	1247	2414	57
MAJ-1	5	1463	1869	78
MAJ-1	5	1269	1834	71
MAJ-1	5	1505	1989	94

Lab

handling of non-detects, value, sd, or n?

```
original_data %>%
  group_by(core, depth) %>%
  summarise(Ca=mean(Ca), Ti=mean(Ti),
            V=mean(V))
```

Data loss: replicates are summarised

Summarised values matrix

depth	Ca	Ti	V
0	1884.7	2369.7	78.3
1	1418.0	2409.0	70.0
2	1550.0	2376.0	70.0
3	1448.0	2485.0	64.0
4	1247.0	2414.0	57.0
5	1412.3	1897.3	81.0

References

Grimm, E. 2002. Tilla*Graph. Available from http://www.ncdc.noaa.gov/paleo/tiliafaq.html.
Juggins, S. 2011. C2. Available from https://www.staff.ncl.ac.uk/stephen.juggins/software/C2Home.htm.
Juggins, S. 2015. rioja: Analysis of Quaternary Science Data. R Foundation for Statistical Computing. Available from https://cran.r-project.org/package=rioja.
R Core Team. 2013. R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing, Vienna, Austria. Available from http://www.R-project.org/.
Sarkar, D. 2015. lattice: Trellis Graphics for R. Available from https://cran.r-project.org/package=lattice
Simpson, G.L., and Oksanen, J. 2016. analogue: Analogue and Weighted Averaging Methods for Palaeoecology. Available from https://cran.r-project.org/package=analogue
Wickham, H. 2016. reshape2: Flexibly Reshape Data - A Reboot of the Reshape Package. Available from https://cran.r-project.org/package=reshape2
Wickham, H., Chang, W., and RStudio. 2016. ggplot2: An Implementation of the Grammar of Graphics. Available from https://cran.r-project.org/package=ggplot2
Wickham, H., and Francois, R. 2016. dplyr: A Grammar of Data Manipulation. Available from https://cran.r-project.org/package=dplyr

Case study: plotting functions

ggplot

The *ggplot* package (Wickham et al. 2016) takes data in a summarised values list and easily deals with replicates, non-detects, multiple cores, and multiple parameters. Ouput is available in image and vector formats for easy customization of the plot in vector drawing programs. Paleoecological data are not easily visualized using ggplot because rotated titles and variable plot widths are not possible. Because data are input in a non-lossy format, plotting options are inredibly flexible with regard to the handling of multiple tags and and multiple qualifiers.

```
ggplot(melted_summarised, aes(x=value, y=depth)) +
  geom_point() + geom_path(aes(lty=core)) +
  geom_errorbarh(aes(xmin=value-sd,xmax=value+sd),height=0.1)+
  scale_y_reverse() + theme_bw() +
  facet_wrap(~parameter, scales = "free_x")
```

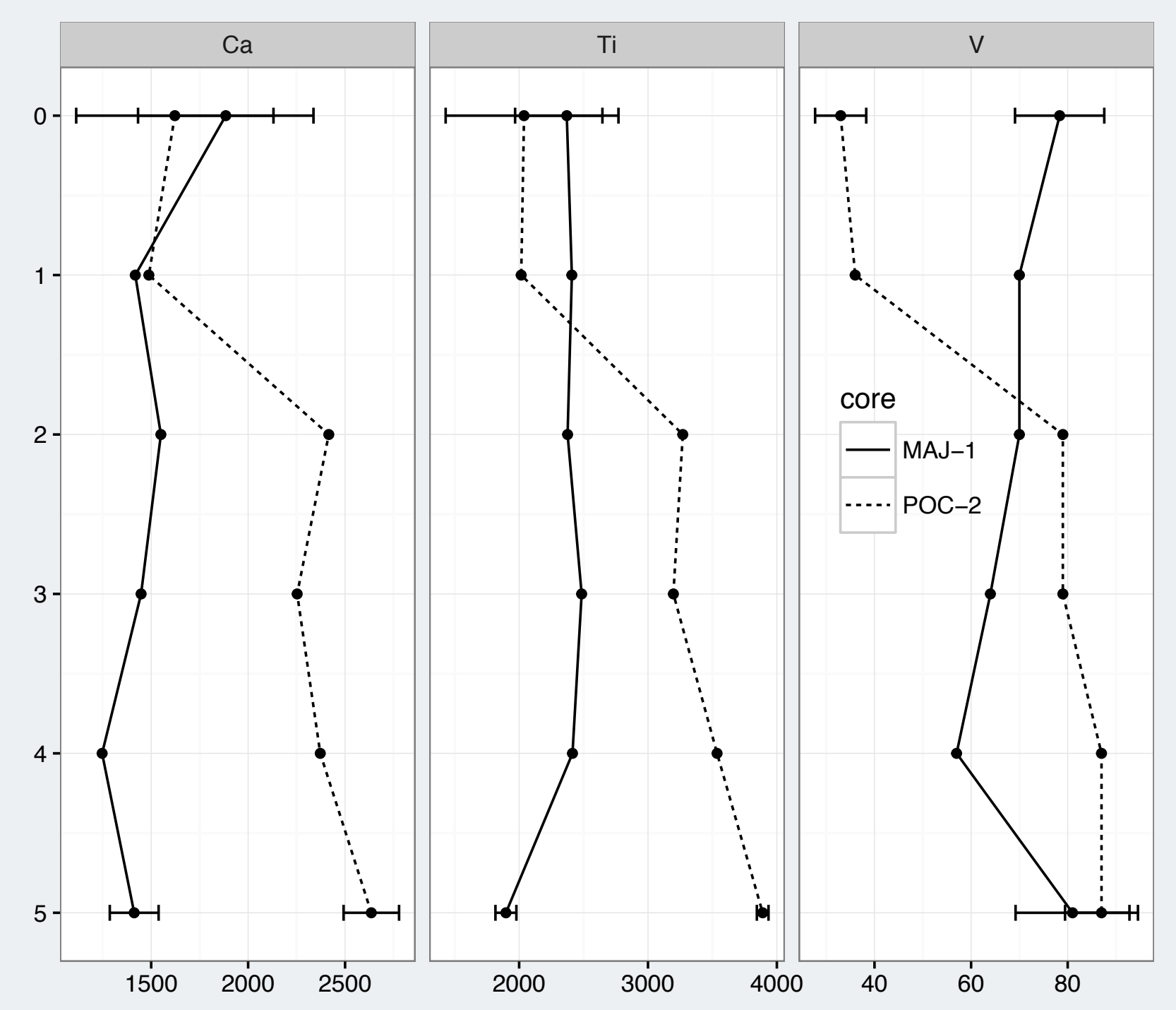

Figure 2. Example output of *ggplot*.

strat.plot

The *strat.plot* function in the package *rioja* (Juggins 2015) takes data in a summarised values matrix and is optimised for plotting paleoecological data. The function has a large number of parameters for modifying plot margins, font sizes, rotation of titles, adding dendrograms, and much more. Because data are input in a lossy format, *strat.plot* can only plot one core without consideration of error (without considerable hacking). This method uses base R as its plotting backend.

```
strat.plot(vals_matrix_maj1[c("Ca", "Ti", "V")],
           yvar = vals_matrix_maj1$depth,
           y.rev = TRUE)
```

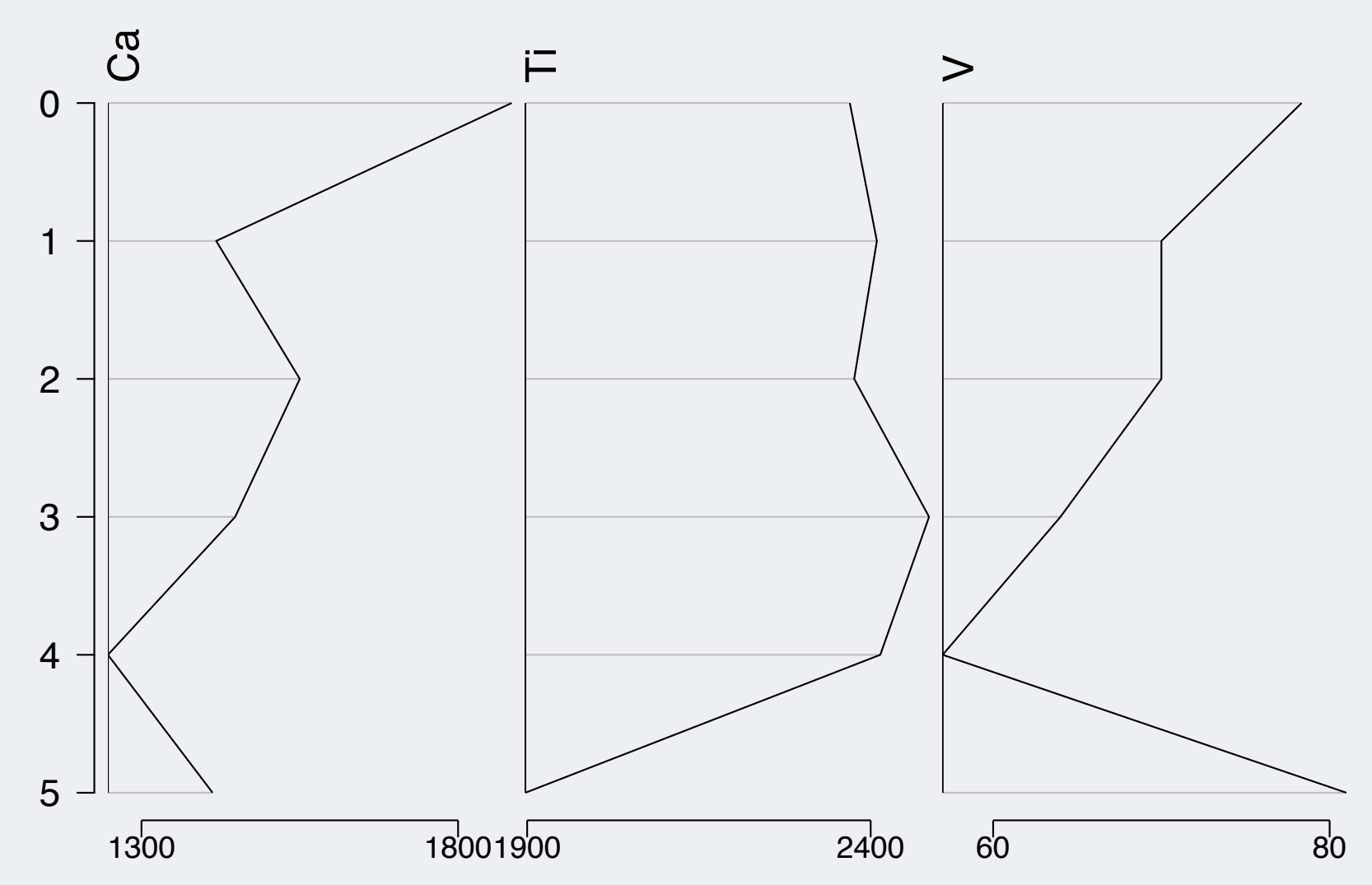

Figure 3. Example output of *strat.plot*.

Stratiplot

The *Stratiplot* function in the package *analogue* (Simpson and Oksanen 2016) takes data in a summarised values matrix and is also optimised for plotting paleoecological data. The function has a large number of parameters for modifying plot parameters. Because data are input in a lossy format, *Stratiplot* can only plot one core without consideration of error, although modification is probably possible. This method uses the *lattice* package (Sarkar 2015) as a plotting backend.

```
Stratiplot(vals_matrix_maj1[c("Ca", "Ti", "V")],
           y = vals_matrix_maj1$depth,
           type=c("p", "l"), varTypes = "absolute",
           ylab = "depth")
```

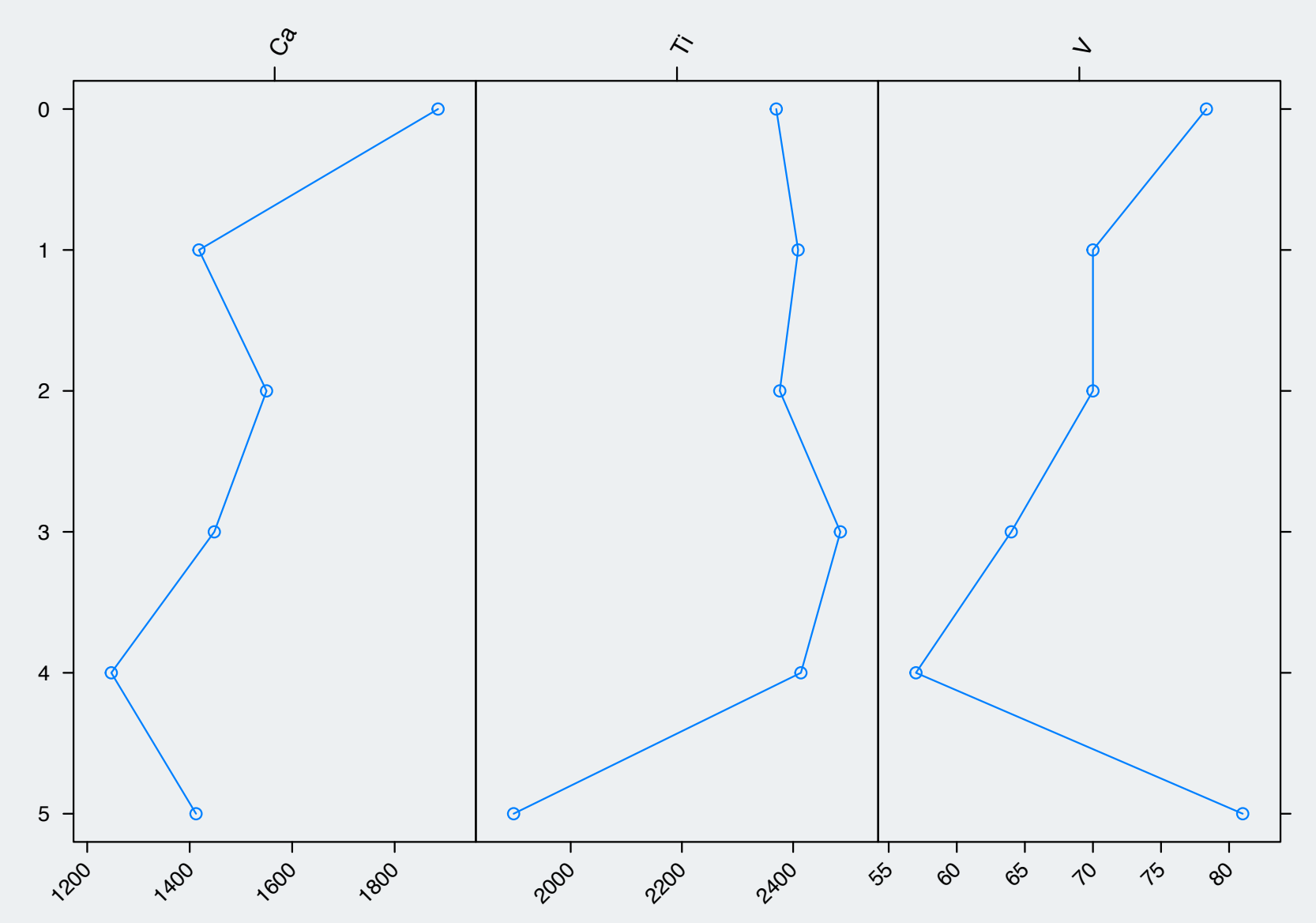

Figure 4. Example output of *Stratiplot*.