Graphical Methods for Data with Multiple Levels of Nesting

José C. Pinheiro and Douglas M. Bates
Bell Laboratories *Department of Statistics*
Lucent Technologies *U. of Wisconsin–Madison*

Data with multiple nested classification factors are frequently collected in industrial experiments and in sociological studies. One of the challenges for this type of data is displaying, in a concise and efficient way, the information available at the different levels of nesting. We describe data structures and graphical methods that we developed in the S-PLUS language for representing and displaying data with multiple levels of nesting. The graphical methods are based on the powerful Trellis library, included with versions 3.3 and higher of S-PLUS, and use an object-oriented approach to display appropriate plots for different classes of multiply-nested objects. Real data examples are used to illustrate the various capabilities in the software.

Data which can be divided into disjoint groups, according to one or more classification factors, are frequently observed in many areas of application. When only one grouping factor is present, this type of data is generally called *repeated measures* or, when the observations are collected sequentially over time, *longitudinal data*. This type of data are the rule in biomedical studies, where the data are generally grouped according to patients, or subjects.

However, in many industrial and agricultural applications, as well as in sociological studies, it is common to have data grouped according to several nested classification factors. For example, in an experiment to evaluate the yield of integrated circuit manufacturing, the observations may be grouped into *lots*, which are subdivided into *wafers*, which, in turn, are further subdivided into *chips*. The three nested grouping factors could have different effects on the observed responses and displaying the data in meaningful ways becomes a challenge.

This article is about the organization and visualization of data grouped according to several nested classification factors. We describe data structures and display methods we developed in the S-PLUS language for representing and plotting this type of data. In Section 1 we describe a `groupedData` class of objects used to represent repeated measures and multiply-nested data and introduce two real-life examples of data with multiple levels of nesting. The display methods for the `groupedData` class are presented in Section 2 and illustrated with the examples of Section 1. Our conclusions and suggestions for future research are included in Section 3.

1 A `groupedData` class

A common and versatile way of organizing data in S-PLUS is as `data.frame` objects. These are in the form of rectangular arrays in which each row corresponds to an observation and each column corresponds to one of the variables being observed. We extend the `data.frame` class to the class of `groupedData` objects which are data frames with additional information about the grouping of the observations and, possibly, other special roles of some variables.

A `groupedData` object contains the data values themselves, stored as a data frame, and a formula that designates special roles for some of the variables in the data frame. The most important of the special roles is that of the *grouping factors* which divide the observations into the distinct groups of observations. The formula also designates a *response* and, when available, a *primary covariate*. It is given as

```
response ~ primary | grouping
```

where `response` is an expression for the response, `primary` is an expression for the primary Covariate, and `grouping` is an expression for the grouping factor(s). Most often these expressions are simply the name of a variable in the data frame but they can be functions of one or more variables. For example, `log(current)` would be a legitimate expression for the response if `current` is one of the variables in the data frame.

Constructing a `groupedData` object requires the data to be available as a data frame. A function to create objects of a given class is called a *constructor* for that class. The primary constructor function for a class is often given the same name as the class itself. Thus the default constructor for the `groupedData` class is the `groupedData` function. Its required arguments are a formula and a data frame. Optional arguments include `labels`, where display labels for the response and the primary covariate can be given and `units`, where the units of these variables can be given.

To illustrate the use of the `groupedData` constructor, we consider an experiment conducted at the Department of Medical Sciences, School of Veterinary Medicine, University of Wisconsin–Madison. The mean pixel intensity of the right and left lymphnodes in the axillary region obtained from CT scans of 10 dogs were recorded over a period of 14 days after intravenous application of a contrast. These observations are naturally grouped at two levels, dog and side within dog. We indicate such a nested structure of grouping factors by a `/` in the formula for the grouping factor. Assuming that the data are available as an S-PLUS data frame named `Pixel`, we can construct a `groupedData` object as follows.

```
> names(Pixel)
[1] "Dog"   "Side"   "day"   "pixel"
> Pixel <- groupeData(pixel ~ day | Dog/Side, data = Pixel,
+   labels = list(x = "Time post injection", y ="Pixel intensity"),
+   units = list(x = "(days)"))
> formula(Pixel)
pixel ~ day | Dog/Side
```

The response variable for the `Pixel` object is given by `pixel`, the primary covariate is `day`, the grouping structure is represented as `Dog/Side`, which is usually read as "`Side` within `Dog`". We refer to *levels* of grouping factors in a nested grouping structure. For the pixel intensity data, `Dog` is the first or outermost level of grouping and `Side` is the second or innermost level.

The formula in a grouped data object has the same pattern as the formula used in a call to a Trellis graphics function in S-PLUS, such as `xyplot`. This is intentional. Because such a formula is available with the data, the `plot` method for objects in the `groupedData` class can produce an informative Trellis display from the object alone. It may, in fact, be best to think of the formula stored with the data as a *display formula* for the data in the sense that it provides a meaningful default graphical display method for the data.

Although the display formula can be used to designate a response, a covariate, and the grouping structure, these may not be sufficient to describe the structure of the experiment or study completely. Many experiments impose additional structure on the data. Often times, *treatment factors* are applied to the observations and may determine the way the data should be displayed. With respect to a given grouping factor, these treatment factors can be of two types: constant over the levels of the grouping factor, in which case they will be referred as *outer* factors, or varying within the levels of the grouping factor, in which case they will called *inner* factors.

When inner and outer factors are present they can, and should, be designated when constructing a `groupedData` object. To illustrate the use of inner and outer factors with multiply-nested data, we consider an experiment conducted at the Microelectronics division of Lucent Technologies in which the intensity of current was measured at five different voltages for a set of n-channel devices of varying geometries. The purpose of the experiment was to study the effect of channel length on current variability. Measurements were taken on 6 wafers, each subdivided into 16 sites, each containing 4 devices of different channel lengths. There are three levels of nesting in these data: wafer, site within wafer, and device within site within wafer. The display formula for the `groupedData` object `Wafer` is

```
> formula(Wafer)
current ~ voltage | Wafer/Site/Device
```

The treatment factor for this experiment is the channel length, which is stored as the column `Length` of the `Wafer` object. `Length` changes within the same level of `Wafer` and `Site`, so it is inner with respect to these two grouping factors. But `Length` is constant over the measurements taken on a device and therefore it is outer with respect to `Device`. Valid declarations of the `groupedData` constructor in this case are

```
> Wafer <- groupedData(current ~ voltage | Wafer/Site/Device,
+   data = Wafer,
+   labels = list(x = "Voltage", y = "Intensity of current"),
+   units = list(x = "(V)", y = "(mA)"),
+   outer = list(Device = ~ Length))
```

or

```
> Wafer <- groupedData(current ~ voltage | Wafer/Site/Device,
. . .
+  inner = list(Wafer = ~ Length, Site = ~ Length)
```

Several methods are available for printing, summarizing, and modelling `groupedData` objects, but in this article we concentrate on plotting methods for these objects, which we present in the next section.

2 Displaying `groupedData` objects

The `groupedData` plot methods are based on the Trellis library in S-PLUS (Becker, Cleveland and Shyu, 1996). Trellis displays are a natural way of presenting repeated measures and longitudinal data, as each panel in the trellis arrangement can be associated with a different level of the single grouping factor. A software implementation for grouped data with a single grouping factor is described in Bates and Pinheiro (1997).

Trellis displays for multiply-nested data are not just a simple generalization of the single grouping factor case, as different issues, not encountered before, come into play. Some of the most important among these issues are:

- the *display level* to be used to construct the panels in the trellis display

- the *collapsing level* to be used to collapse the observations when other grouping factors are nested within the display level

- the *collapsing function* to be used to collapse the observations, when a collapsing level is declared

We illustrate these concepts through plots of the `Pixel` and `Wafer` datasets introduced in the previous section.

By default, the plot method for multiply nested `groupedData` objects will set the display and the collapse levels to the innermost level of nesting and will use the `response` and `covariate` expressions in the display formula to construct a Trellis formula. For example,

```
> plot(Pixel)
```

will produce Figure 1

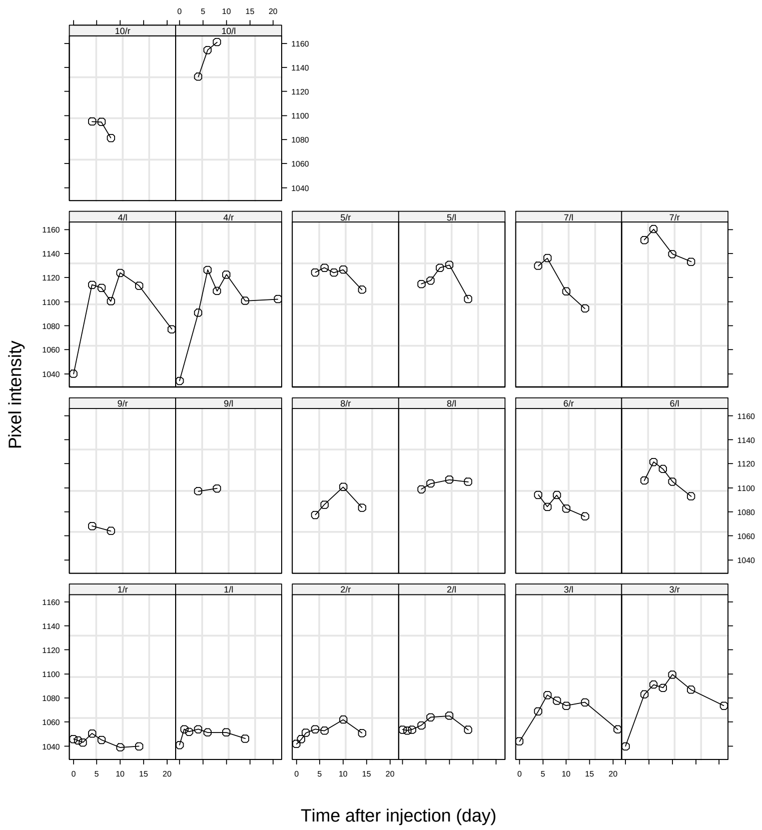

Figure 1: *Mean pixel intensity of the right and left lymphnodes in the axillary region versus time from intravenous application of a contrast in 10 dogs. Display level and collapsing level equal to* `Side` *within* `Dog`.

A more meaningful display of the `Pixel` data is obtained by setting the display level to `Dog` and keeping the collapse level at `Side` within `Dog`, as displayed in Figure 2.

```
> plot(Pixel, display = 1)
```

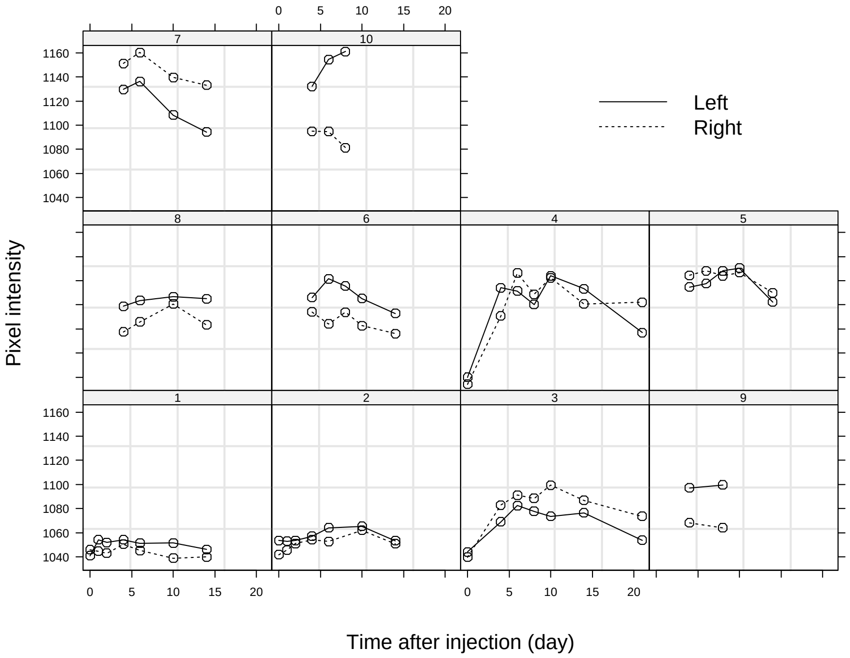

Figure 2: *Mean pixel intensity of the right and left lymphnodes in the axillary region versus time from intravenous application of a contrast in 10 dogs. Display level is equal to* `Dog` *and collapsing level is equal to* `Side` *within* `Dog`*.*

It is clear from Figure 2 that there is considerable more `Dog` to `Dog` variability than `Side` within `Dog` variability. We can explore this a further by looking at the differences between mean pixel intensities in the left and right sides within dog. This is accomplished by keeping the display level as `Dog`, changing the collapse level to `Side` within `Dog`, and specifying `diff` (difference) as the collapsing function.

```
> plot(Pixel, display = 1, collapse = 1, FUN = diff)
```

As shown in Figure 3, the differences between sides within the same dog tend to have the same sign, suggesting the existence of a `Side` within `Dog` (random) effect. This information is helpful when modelling data of this type using mixed-effects models (Pinheiro and Bates, 1995).

We now consider the `Wafer` data of Section 1. The default plot for the `Wafer` object would use one panel for each of the 384 `Wafer`/`Site`/`Device` combinations and would not be very meaningful. A better display can be obtained by setting the display level to `Site` within `Wafer`, while keeping the collapse level at the innermost level of grouping. The corresponding plot, for `Wafer` 1 only, is obtained with

```
> plot(Wafer, display = 2, subset = list(Wafer = 1))
```

and is presented in Figure 4.

We see that the intensity of current increases as the channel length decreases. There is some nonlinearity in the relationship between intensity of current and voltage, and this does not appear to vary substantially among `Sites` within `Wafer` 1. To investigate the `Wafer` to `Wafer` variation in the relation between current and voltage, we set the display and collapse levels to `Wafer` and use the argument `inner` to the plot method to preserve the information on channel length when the data are collapsed. By default, the collapsing function is set to `mean`.

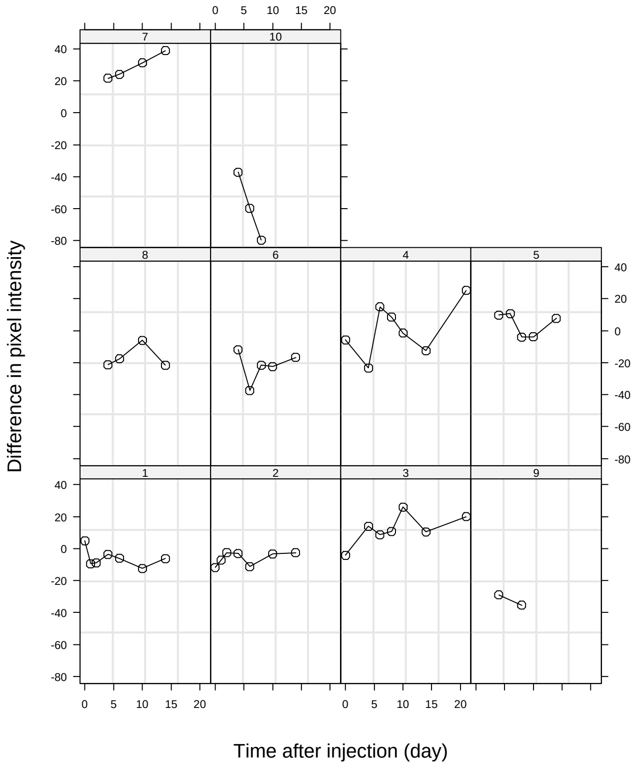

Figure 3: *Difference in mean pixel intensity between the right and left lymphnodes. Display and collapse levels equal to* `Dog` *and collapsing function equal to* `diff`.

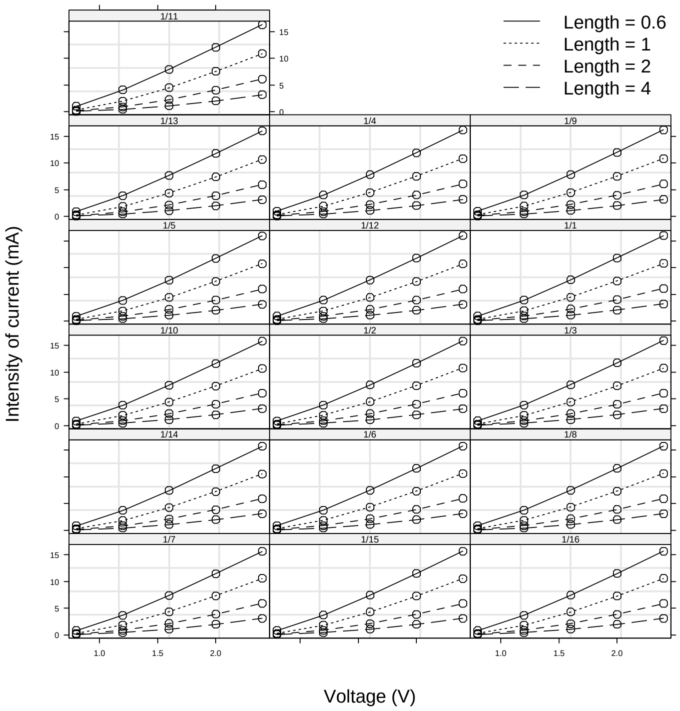

Figure 4: *Intensity of current versus voltage for four different channel lengths, for* `Wafer` *1. Display level is* `Site` *within* `Wafer`*, collapse level is* `Device` *within* `Site` *within* `Wafer`.

```
> plot(Wafer, display = 1, collapse = 1, inner = ~ Length)
```

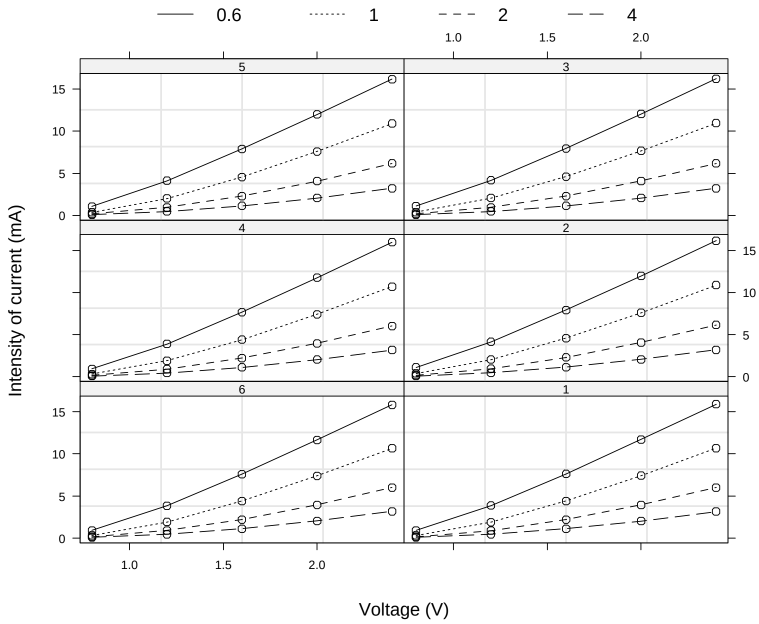

Figure 5: *Intensity of current versus voltage for four different channel lengths. Display and collapse levels equal to* `Wafer` *and collapsing function is* `mean`.

We see that the relationship between the response and the primary covariate is approximately the same across `Wafers`. The primary question of interest in this experiment is how channel length affects the variability in the intensity of current. We may graphically explore this issue by using the standard deviation as the collapsing function, keeping the display and collapse levels equal to `Wafer`.

```
> plot(Wafer, display = 1, collapse = 1, inner = ~ Length,
+      FUN = function(x) sqrt(var(x)),
+      ylab = "Standard deviation of intensity of current (mA)")
```

Figure 6 displays the resulting Trellis plot. The variability of the intensity of current increases as the channel length decreases and there is considerable differences in variability across wafers. The standard deviation of the intensity of current increases nonlinearly with voltage. All this information, obtained from the plots alone, is useful for guiding the model building process that usually follows the graphical analysis.

3 Conclusion

The Trellis library in S-PLUS, combined with object oriented programming, provides a powerful environment for displaying multiply-nested data. Some of the issues involved in visualizing multiply-nested data are not present when repeated measures or longitudinal data are plotted. These include the choice of the display level, the collapsing level, and the function used to collapse the response.

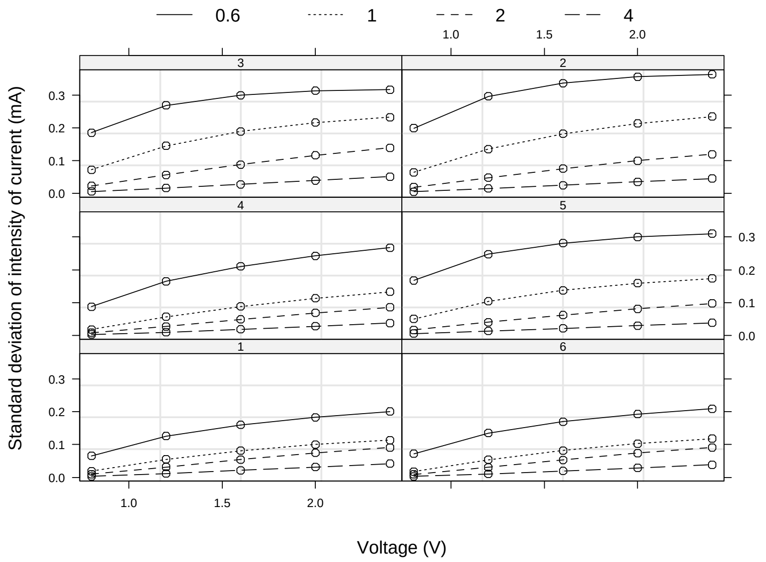

Figure 6: *Standard deviation of intensity of current versus voltage for four different channel lengths. Display and collapse level equal to* `Wafer` *and collapsing function is the standard deviation.*

We described a `groupedData` class of data structures in S-PLUS for representing grouped data with one or more nested grouping factors and presented various types of display that can be used to graph objects inheriting from this class.

Our use of the Trellis library has consisted in adapting the existing functions to provide meaningful displays of multiply-nested data. The current version of Trellis is primarily designed to handle crossed classification factors and nested factors are not handled as efficiently. One interesting research topic would be to consider modifications of the current Trellis functions to allow *true* nested-classification display capabilities. This would include, for example, the ability to dissociate the layout for the panels within different levels of an outer grouping factor. Another challenging topic would be the generalization of Trellis to simultaneously handle nested and crossed grouping factors.

References

Bates, D. M. and Pinheiro, J. C. (1997). Software design for longitudinal data, *in* T. G. Gregoire (ed.), *Modelling Longitudinal and Spatially Correlated Data: Methods, Applications and Future Directions*, Springer-Verlag, New York, pp. 37–48.

Becker, R. A., Cleveland, W. S. and Shyu, M.-J. (1996). The visual design and control of trellis graphics displays, *J. of Computational and Graphical Statistics* **5**(2): 123–156.

Pinheiro, J. C. and Bates, D. M. (1995). Model building for nonlinear mixed-effects models, *Technical Report 91*, Department of Biostatistics, University of Wisconsin – Madison.