Visualizing Complex Data With Embedded Plots

Garrett GROLEMUND and Hadley WICKHAM

This article describes a class of graphs, embedded plots, that are particularly useful for analyzing large and complex datasets. Embedded plots organize a collection of graphs into a larger graphic, which can display more complex relationships than would otherwise be possible. This arrangement provides additional axes, prevents overplotting, and allows for multiple levels of visual summarization. Embedded plots also preprocess complex data into a form suitable for the human cognitive system, which can facilitate comprehension. We illustrate the usefulness of embedded plots with a case study, discuss the practical and cognitive advantages of embedded plots, and demonstrate how to implement embedded plots as a general class within visualization software, something currently unavailable. This article has supplementary material online.

Key Words: Exploratory data analysis; Graphical methods; Massive datasets.

1. INTRODUCTION

Complex datasets are difficult to analyze because they contain multiple relationships that must be understood with respect to each other. Visualization can help, but it is hard to visualize more than two or three relationships at once in a static graph. Moreover, complex datasets tend to be large, which creates problems of overplotting in traditional graphs. We present a class of graphs called embedded plots that are ideal for visualizing large, complex datasets.

Embedded plots can be generalized as graphics that embed subplots within a set of axes. Figure 1 shows three examples of this type of plot: William Cleveland’s subcycle plot, which embeds 12 line charts within a larger graph; a glyphmap that embeds polar charts within a map; and a binned graph that uses small bar charts to describe the content of two-dimensional bins. Each subplot is a self-contained plot when viewed on its own (or would be if it contained the appropriate axis, labels, and legend). A subplot can share the axis system of the plot in which it is embedded, but more often subplots will use their

Garrett Grolemund is Statistician, RStudio, Boston, MA 02210 (E-mail: garrett@rstudio.com). Hadley Wickham is Adjunct Professor, Department of Statistics, Rice University, Houston, TX 77005 (E-mail: hadley@rice.com).

© 2015 American Statistical Association, Institute of Mathematical Statistics, and Interface Foundation of North America
Journal of Computational and Graphical Statistics, Volume 24, Number 1, Pages 26–43
DOI: 10.1080/10618600.2014.896808
Color versions of one or more of the figures in the article can be found online at www.tandfonline.com/rjcg
Figure 1. Three examples of embedded subplots. (a) (upper left) A subcycle plot that shows how the monthly subfrequency components of CO2 change at Mauna Loa Observatory have fluctuated in relation to the overall seasonal trend from 1959 and 1990. Recreated from Cleveland (1994), page 187. (b) (upper right) A glyphmap of temperature fluctuations in the western hemisphere over a six-year period. Each glyph is a polar chart with $r = \text{temperature}$ and $\theta = \text{date}$. (c) (lower left) A binned plot of the diamonds dataset from the ggplot2 software package. Each bar graph describes the points contained in a 2D region of the graph. When these data are presented in its raw form, the accumulation of points hides patterns in the data, (d) (lower right).

own coordinate system to display additional information. For example, Figure 1(b) embeds polar graphs in a cartesian coordinate system.

Embedded plots are not new; they have a rich pedigree and a growing future. Charles Minard was embedding glyphs and other plots in maps in 1862 (Minard 1862). William Cleveland, one of the first innovators of computer based graphics, introduced subcycle plots in 1982 (Cleveland and Terpenning 1982). A year later, Jacques Bertin featured embedded plots on 21 pages of Semiologie of Graphics (1983), a seminal work in the academic study of
visualization. More recently, glyphmaps have been developed as a tool for tracking climate data (Hobbs et al. 2010; Wickham et al. 2012), and binned graphics like Figure 1(c) have emerged as a promising candidate for solving the problem of overplotting when visualizing big data.

Interest in embedded plots has led to many specific types of subplots, each designed to be compared to other subplots arranged in a graph or table. Purposefully designed subplots include glyphs (Anderson 1957), trees and castles (Kleiner and Hartigan 1981), Chernoff faces (Chernoff 1973), stardates (Lanzenberger, Miksch, and Pohl 2003), micromaps (Carr and Pierson 1996), and icons (Pickett and Grinstein 1988). These subplots usually contain an implicit set of axes and are organized in a larger, explicit frame. Other popular embedded plots combine common graphs into a table of subplots. These embedded plots include scatterplot matrices (Chambers 1983), conditioned choropleth maps (Carr, White, and MacEachren 2005), trellis plots (Sarkar 2008), and facetted plots (Wilkinson and Wills 2005). In these embedded plots, each subplot contains an explicit set of axes and the subplots are organized into a larger implicit frame—usually a visual table, which can be thought of as a cartesian graph that has categorical $x$ and $y$ variables.

The two-tiered structure of embedded plots makes them well suited for solving a number of data analysis problems, as seen in Figure 1. First, embedded plots make it easy to visualize interaction effects because they display two sets of relationships, one contained in the major axes of the plot and one contained in the minor axes of the subplots. Second, embedded plots provide an intuitive way to organize spatiotemporal data and other multidimensional data. Finally, embedded plots mitigate the problem of overplotting.

However, as useful as embedded plots are, they are difficult to make. Currently, programs that can make embedded plots focus on a specific type of subplot, such as glyphs (Gribov, Unwin, and Hoffman 2006) or scatterplot matrices (Sarkar 2008). This limits the ways that embedded plots can be used to explore data and present findings. In this article, we discuss the advantages of embedded plots and describe how embedded plots can be implemented as a general class of graphs in data analysis software.

The remainder of this article proceeds as follows:

Section 2 begins with a case study that presents the usefulness of embedded plots. We explore the Afghan War Diary data, made available by the WikiLeaks organization. The dataset is large and complex: 76,000 + observations organized by location and time. The case study shows how embedded plots can be used in practice to reveal patterns that cannot be seen in single level graphs.

Section 3 examines why embedded plots are useful for exploring complex data. Embedded plots have two practical advantages: they provide extra axes and a high degree of customizability. More importantly, embedded plots facilitate insight by organizing complex data into a form suitable for the human cognitive system.

Section 4 discusses how generalized embedded plots can be implemented in data analysis software. We present a very customizable implementation of embedded plots that uses the layered grammar of graphics (Wickham 2010) and the ggplot2 package (Wickham 2009) in R. Incorporating embedded plots into the grammar of graphics yields a new insight about graphics: they have an inherently hierarchical structure.

Section 5 concludes by offering general principles to guide the use of embedded plots.
2. CASE STUDY: ANALYZING COMPLEX DATA

The Afghan War Diary dataset, made available by the WikiLeaks organization at http://www.wikileaks.org/wiki/Afghan_War_Diary_2004-2010, is large, complex, and intriguing because it provides insights into an ongoing military conflict. The dataset was collected by the U.S. military and contains information about military events that occurred in or around Afghanistan between 2004 and 2010. Among other variables, the dataset records the number of injuries and deaths that resulted from each event. These casualty statistics are collected for four groups: enemy forces (enemies), coalition forces (friendly), Afghanistan police and security forces (host), and civilians (civilians). The dataset is large enough (76,000 observations) that overplotting becomes a concern when visualizing the data. The dataset is complex in that it contains a spatiotemporal component: each observation is labeled by longitude, latitude, and date.

The U.S. military engagement in Afghanistan has received criticism for the number of civilian casualties associated with the war. Civilians comprise almost a quarter of all casualties recorded in the diary, and civilians have more casualties (12,871) than coalition (8,397) and Afghan (12,184) forces. Civilians have nearly half as many casualties as enemy forces (24,233). We wish to see if these ratios vary by location. Are civilian casualties noticeably high everywhere the war has been fought, or just for certain locations, such as urban centers, where military action occurs in close proximity to a large number of civilians?

The size of the Afghan War Diary makes it difficult to visualize this information. When plotted as a point map, individual casualties obscure one another, a phenomenon known as overplotting, Figure 2(a). A heat map avoids overplotting, but cannot show casualties by type, Figure 2(b). We only see that the majority of casualties occur in the southern region of Afghanistan between Kabul and Kandahar. To examine casualties by type, we would have to create four separate heat maps, each with a different subset of the data. We turn to embedded plots for a simpler solution. In Figure 2(c), we replace each tile in the heat map with a bar graph of casualties by type. This embedded plot reveals similar information as the heat map, but it also displays the ratio of casualties for each area. We can further adjust the embedded plot to show the conditional distribution of casualties for each region, Figure 2(d). This technique makes regional patterns more clear and would not make sense for a heat map or contour plot. We also add a bounding box to each subplot to make the individual subplots more distinct.

The plots show that civilian casualties often surpass coalition and host casualties, and sometimes enemy casualties. Near Kabul, civilian casualties seem to surpass all other types of casualties put together. The visualizations suggest that high civilian casualty rates occur throughout Afghanistan and not just near population centers like Kabul, although high civilian casualty rates also occur there as well.

3. BENEFITS OF EMBEDDED PLOTS

Embedded plots give an analyst a second set of axes to work with, which creates practical advantages not available in nonembedded plots. However, this system increases
the complexity of the graph, which can diminish understanding and prevent insight. An analyst may ask, “Is this a reasonable thing to do?” Next, we review the practical advantages of embedded plots as well as cognitive science findings that suggest that embedded plots can be simple and easy to understand.

### 3.1 Practical Advantages of Embedded Subplots

Embedded plots offer two advantages over nonembedded plots: they provide an additional set of coordinate axes and an additional level of summarization, which an analyst can customize. An analyst can display four separate variables with the major x, major y, minor x, and minor y axes of an embedded plot. Additional variables can also be included with colors, shapes, sizes, etc. This makes embedded plots useful for displaying complex, multivariate data like spatiotemporal data. Visualizing spatiotemporal data usually requires four or more dimensions: two for spatial coordinates, a third for the passage of time, and a fourth for the quantity of interest. Embedded plots like Figure 1(b) can organize these dimensions in a way that is easily interpreted and that makes both spatial and temporal patterns obvious.
Embedded plots also offer increased ways to summarize data within an image. Summarization is a common tactic for visualizing large datasets without overplotting. For example, heat maps and density contours divide data into separate groups and summarize each group with a single number, which is then visualized. Analysts can use subplots to summarize data with an image, and thus retain more information about the group. For example, the bar charts in Figure 2(c) display four measurements in the same space that a heat map tile uses to display one measurement. Moreover, an analyst can choose how much information is displayed by a subplot. By selecting the type of subplot to use, an analyst can choose between no summarization, partial summarization, and complete summarization (Figure 3).

Embedded plots are particularly useful for displaying interaction effects in big datasets because subplots can provide enough summarization to avoid overplotting and enough information to display a relationship. This effect is especially well illustrated by Figures 1(c) and 1(d). Both figures describe the same 20,000 data points. When data are viewed as a colored scatterplot, the points occlude each other and underlying patterns are hidden (Figure 1(d)). When data are viewed with embedded bar charts, a relationship appears between price, carat, and color: for any value of carat, better colored diamonds occur more often in the higher price ranges than the low ones. The embedded subplots in Figure 1(c) would not suffer from overplotting even if the dataset was enlarged to 100,000, a million, or a trillion points.

3.2 COGNITIVE ADVANTAGES OF EMBEDDED SUBPLOTS

Principles of cognitive science suggest that embedded plots are a useful way to present complex information. Cognitive science provides a well-understood model that can explain why a statistical visualization succeeds or fails. In this model, the human mind processes novel information with the working memory, which has a fairly small processing capacity (Miller 1956; Cowan 2000). Every learning task, such as comprehending a statistical graph, imposes a cognitive load on the working memory. Learning will not occur if the cognitive load exceeds the available capacity of the working memory (Sweller 1988).

According to the cognitive load model, adding extra axes and information to a graph would make the graph less comprehensible, because complexity increases cognitive load.
In contrast, an analyst can facilitate learning if he or she decreases the cognitive load required to understand a graph. Cognitive scientists have documented many ways to decrease cognitive load and facilitate learning. See Sweller (2003) for a complete list and review of the supporting literature. Embedded plots seem to implement three of these methods: visualization, automation, and isolation.

3.2.1 Visualization. Visual information imposes a smaller cognitive load on the working memory than nonvisual information. Studies consistently show that the working memory can only handle four novel objects at a time, whether verbal or visual (Cowan 2000). However, each piece of visual information can be an image that contains multiple features. A study by Luck and Vogel (1997) demonstrated that four visual objects that each contain four pieces of information can be processed by the working memory as easily as four visual objects that each contain only one piece of information. This ability gives the working memory a higher bandwidth for visual information than for verbal information. It also suggests that replacing the elements of a graph with subplots may not significantly increase the cognitive load needed to understand the graph.

3.2.2 Automation. Cognitive load decreases when information is presented in a familiar format. The mind uses a cognitive structure known as a schema to process new information. The schema directs attention during information processing and identifies relationships between data points and previous knowledge. Literature on schemas are extensive. See Neisser (1976) and Rumelhart (1980) for highlights. When the mind frequently uses a particular schema, the schema becomes automated (Shiffrin and Schneider 1977; Schneider and Shiffrin 1977). When this happens, information related to the schema can be processed with less and less conscious effort. Kotovsky and Simon (1985) demonstrated that automated processing decreases cognitive load to such an extent that information can be processed 16 times faster than with nonautomated schemas.

A common example of automated processing is reading written text. For young children, reading is a laborious process that involves identifying letters, assigning sounds to them, associating these sounds with words and then meanings. However, by the time children become adults, these tasks are done unconsciously and reading proceeds automatically. Reading graphs is a similar skill that benefits from automated processing. For analysts familiar with data visualization, the subplots of embedded plots should impose little additional cognitive load when the analyst views the graph.

3.2.3 Isolation. Sometimes the cognitive load of very complex information cannot be decreased enough to allow comprehension. When this occurs, a person can use a divide and conquer strategy: he or she can break the information into small isolated parts and processes each part separately. For full comprehension to occur, the individual must next recall the separate parts from his or her memory and identify the connections between them (Sweller, Ayres, and Kalyuga 2011). This is possible because the working memory can recall information at little to no cognitive cost once the information has been learned (Sweller 2003). As a result, the mind can build a deep understanding of highly interactive data by iterating between processing small subsets of data and then recalling these subsets from the long-term memory to compare against each other and new information. Embedded
plots facilitate this process by dividing a dataset into isolated subplots and visualizing the relationship between the subplots.

3.2.4 Summary. In summary, analysts have a strong reason to believe that embedded plots will be interpretable to themselves and their audience. Embedded plots present information visually, with an intuitive organization and a familiar presentation. As a result, they minimize the cognitive load needed to comprehend and interpret graphs. Embedded plots may even allow users to comprehend complex relationships that would remain incomprehensible in other formats.

4. IMPLEMENTING EMBEDDED PLOTS WITHIN THE GRAMMAR OF GRAPHICS

Embedded graphs are useful, but difficult to make. Particular types of software exist to make particular types of embedded plots. For example, interactive glyph plots can be made with gaugain (Gribov, Unwin, and Hoffman 2006), conditioned choropleth maps can be made with CCmaps (Carr et al. 2002), faceted plots can be made with the ggplot2 R package (Wickham 2009), and trellis plots can be made with the lattice R package (Sarkar 2008). Scatterplot matrices can be created with the GGally R package (Schloerke et al. 2011) as well as with base R (R Development Core Team 2010). However, an analyst cannot rely on one software package to make the whole spectrum of embedded plots. This makes it difficult to rapidly iterate between different types of subplots, which hinders data exploration.

In this section, we describe a way to implement embedded plots as a general class of graphs in graphing software. Our implementation is built on the layered grammar of graphics and reveals a conceptual insight about graphics: graphs are hierarchical, or recursive, in structure. We demonstrate this method with ggsubplot, an R package written by the authors. ggsubplot extends ggplot2 to create embedded plots in R. However, our method can be used to extend any graphing software that is itself built upon the grammar of graphics. The ggsubplot package is available from cran.r-project.org, and can be installed within R, like any R package. The development page for ggsubplot is hosted openly at http://github.com/garrettgman/ggsubplot.

4.1 THE LAYERED GRAMMAR OF GRAPHICS

The layered grammar of graphics is a conceptual framework for understanding and creating visual graphics. The grammar was proposed by Wickham (2010) and builds on ideas from Wilkinson and Wills (2005) and Bertin (1983). The layered grammar organizes each graph into a collection of visual elements and a set of rules that describe how the appearance of these elements should be mapped to a dataset. The grammar enables a deeper understanding of how graphics function and relate to one another and allows more concise, elegant programming. This approach to graphics has become widely popular: ggplot2, an implementation of the grammar of graphics in R, has been cited over 800 times in scholarly journals and supports an online community of 4500 members. The grammar
Figure 4. Every plot is built around a geom, a visual object that is used to represent data points within the graph.

creates efficiencies and insights by replacing a descriptive taxonomy of charts with a set of general rules that can be used to make almost any type of graphic.

The layered grammar of graphics centers around two concepts: geoms and mappings. A geom is a visual element in a graph that represents an observation or group of observations in a dataset. For example, the points in a scatterplot are a type of geom because each point represents an observation in a dataset. Other types of geoms include the bars in a bar chart, the lines in a line chart, boxplots, etc. To make a graph with the grammar of graphics, you first select a type of geom to represent your data. For example, with the ggplot2 package, you can select a point geom to create a scatterplot, Figure 4.

library(ggplot2)

ggplot(example_data) +
  geom_point(mapping = aes(x = month, y = co2))

Each type of geom has its own visual characteristics (called aesthetics). These visual aesthetics can be altered in meaningful ways to display the values of an underlying dataset. For example, the color of a point can be used to display the gender of an observation in the dataset. Two of the most important aesthetics are a geom’s location along the x-axis and y-axis. The grammar of graphics calls the rules used to map aesthetics to variables in a dataset mappings. To create a mapping with ggplot2, set the mapping argument of a geom (Figure 5). This argument takes a list of aesthetics paired with variable names created with the aes function (Wickham 2009).

ggplot(example_data) +
  geom_point(mapping = aes(x = month, y = co2, shape = season))
Figure 5. An analyst can visualize relationships by mapping data values to an aesthetic property of the geoms, like shape.

Geoms and mappings work together to describe a complete graph. A geom transforms data points into visual elements and the mappings transform relationships between data points into visual relationships between geoms. Geoms and mappings also provide a framework for building generalized graphs. Once a graph has been specified with a combination of geom and mappings, the specification can be used on any dataset (Figure 6). The layered grammar of graphics also describes other elements of a graph, such as stats, position adjustments, and scales; but these are not necessary to the discussion that follows.

Embedded graphics fit seamlessly with the grammar of graphics if we recognize that a subplot can be a geom (and that every geom is a subplot). Cleveland’s subcycle plot demonstrates the equivalence between subplots and geoms (Figure 7). The plot visualizes changes in the seasonal trend for atmospheric CO₂ concentrations as measured at the Mauna Loa Observatory in Hawaii from 1959 to 1990 (Cleveland 1994). Seasonal trend

| month | co2 | season |
|-------|-----|--------|
| Jan   | -0.06 | Winter |
| Feb   | 0.60  | Winter |
| March | 1.31  | Spring |
| April | 2.42  | Spring |
| May   | 2.92  | Spring |
| June  | 2.30  | Summer |
| July  | 0.84  | Summer |
| Aug   | -1.19 | Summer |
| Sept  | -2.97 | Fall   |
| Oct   | -3.21 | Fall   |
| Nov   | -2.03 | Fall   |
| Dec   | -0.95 | Winter |

map to: | map to: | map to: | map to: |
| x location | y location | shape |

mappings

Figure 6. The same graph specification can be used to create a new graph with a new dataset.
Seasonal frequency components of Mauna Lau carbon dioxide time series between 1959 and 1990, by month

Figure 7. Cleveland’s subcycle plot can be decomposed into 12 subplots arranged as a scatterplot. The subplots behave as a rectangle point geom with an internal drawing aesthetic.
components were calculated for every month between 1959 and 1990. Trend components are organized by month along the $x$-axis. Within each month, trend components are arranged by year. This gives the cycle plot its embedded structure. Each group of readings from a particular month can be read as a standalone plot once the appropriate axes are added back in (see Figure 7(b)). In the graph, each subplot contains an $x$ position, a $y$ position, and a drawing of a line graph.

If we remove the internal drawings of the line graphs, as in Figure 7(c), what remains is a scatterplot whose points are rectangular. This demonstrates that subplots are equivalent to a rectangle geom, but contain a specialized aesthetic: the internal drawing of a graph. This aesthetic can be mapped to the underlying dataset with a complete graph specification, which will be used to draw the subplot.

To recreate the subcycle plot with the grammar of graphics, we use a new geom provided by ggsubplot, the *subplot*. This geom takes a group mapping, which is used to divide data points between the subplots, and a subplot mapping, which is used to draw the interior of each subplot. The subplot mapping should be a second graph specification. The specification will be applied individually to each group of points represented by a subplot.

```r
library(ggsubplot)

ggplot(carbon) +
  geom_subplot(aes(x = month,
                  y = mean(seasonal, na.rm = TRUE),
                  group = month,
                  subplot = geom_line(aes(x = year,
                                            y = seasonal - mean(seasonal)))))
```

In summary, a subplot is a type of geom with its own set of aesthetics. One of these aesthetics is an internal drawing of a graph. The appearance of this aesthetic is controlled by a graph specification, which creates a mapping between the data and the aesthetic. Although it may seem trivial, the equivalence between subplots and geoms operates in the opposite direction as well. Each geometric object is itself a type of subplot when viewed in isolation. This is easy to see with boxplots and bar graphs, but for many geoms the resulting subplot is so uninteresting that it may go unrecognized (see Figure 8).

This implementation makes subplots very easy to use when exploring data. For example, an analyst can easily move from a traditional geom to a subplot geom (Figures 9(a) and 9(b)).

```r

ggplot(nasa) +
  geom_point(aes(x = surftemp, y = temperature))

ggplot(nasa) + map_americas +
  geom_subplot(aes(x = long,
                   y = lat,
                   group = id,
                   subplot = geom_point(aes(x = surftemp,
                                              y = temperature))))
```
Figure 8. Every individual geom is a self-contained plot when paired with a set of axes. Such plots may not be very interesting, as is the case with point geoms.
Figure 9. An analyst can easily iterate between embedded plots while exploring data. Add subplots by changing a plot’s geom, switch subplots by changing the subplot mapping of the geom, and switch major axes by changing the $x$ and $y$ mappings of the geom.

Or an analyst can move from one type of subplot to another (Figure 9(c)). Here, we use the star geom, also included in the ggssubplot package.

```r
ggplot(nasa) +
  map_americas +
  geom_subplot(aes(x = long,
                   y = lat,
                   group = id,
                   subplot = geom_star(aes(r = fahrenheit,
                                           angle = date,
                                           fill = mean(fahrenheit)),
                                           r.zero = FALSE)))
```
Finally, an analyst can search for interaction effects by changing the $x$ and $y$ mappings of the subplot geom (Figure 9(d)).

```r
ggplot(nasa) +
  geom_subplot(aes(x = max(surftemp),
                 y = min(surftemp),
                 group = id,
                 subplot = geom_star(aes(r = surftemp,
                                         angle = date,
                                         fill = mean(fahrenheit)),
                                     r.zero = FALSE)))
```

Subplots suggest several smaller extensions for graphing software. For example, it is easier to compare subplots if they contain a reference box or line, like the subplots in Cleveland’s subcycle graph. It is also convenient to control the dimensions of subplots with width and height parameters. Subplots also require a programmer to pay special attention to how mappings are drawn for the $x$ and $y$ aesthetics of a subplot. These mappings will use not a single data value, but a group of data values to determine the location of each subplot (a similar situation occurs when placing a boxplot geom). We review these minor details and offer further suggestions for implementing subplots in Appendix A, which is included in the supplementary materials of this article.

5. CONCLUSION

Embedded plots offer a useful way to visualize complex data. They can display four or more dimensions in an intuitive manner and can also control for overplotting. This makes embedded plots a particularly useful tool for exploring interaction effects and analyzing large, multidimensional datasets. Embedded plots have been endorsed by prominent figures throughout the history of statistical graphics (if you consider use an endorsement). However, they are difficult to create with current software packages.

This article shows that embedded plots can be implemented as a general class of graphs in the layered grammar of graphics. Subplots function the same way as geoms; they provide a visual element whose appearance can be mapped to the values in an underlying dataset. Because of this, embedded plots are easily accommodated by software that uses the layered grammar of graphics. We demonstrate this implementation with the `ggsubplot` package, which extends the `ggplot2` package written for the R programming language.

This implementation is based on the observation that a description of a graph is itself an aesthetic mapping, because an aesthetic mapping describes how to use values in a dataset to construct a visual element. Taken further, this observation suggests that graphs are hierarchical, or recursive in nature. One can imagine an embedded plot whose subplots each contain subsubplots, and so on. While it is possible to make such graphs, they are likely a bad idea. We suggest in this article that a graph succeeds when it reduces the cognitive load an observer must expend to understand information in a dataset. Embedded plots are able to present increased information at a reduced cognitive load because they use traits of the human mind. However, this arrangement is unlikely to offset the cognitive load
that would be incurred by doubling or quadrupling the number of relationships displayed in an embedded plot.

Even two-tiered embedded plots flirt with incomprehensibility. Subplots increase the complexity of a visual. They make it easy to create overwhelming, cluttered, and uninterpretable graphs (although they can create simple graphs as well). As a result, we do not recommend embedded plots for every situation. We suggest the following guidelines for the effective use of embedded plots.

1. Do not use embedded plots when a simpler graph will suffice.

2. Give subplots just the elements necessary to convey the main idea of a graphic. Additional elements become distracting more quickly with embedded plots than with nonembedded plots.

3. Use subplots to highlight structure and pattern, not small details like individual values. Subplots are necessarily smaller than a full graph, which makes it harder to accurately perceive details in a subplot (in accordance with Weber’s law). Subplots are fine for estimation and approximate arithmetic, which the mind seems to perform visually at the cognitive level anyways (Dehaene et al. 1999); however, precise calculations require clear labels and numerical values. If detailed inspection is required, a subplot can and should be drawn by itself at full size.

These suggestions are meant to improve, and not prevent, the use of embedded plots. Embedded plots require good judgment in their use, but this is true of all graphs. Every graph should tell a clear story if it is to be useful, and embedded plots will often tell a more clear story than a simple graph plagued by overplotting or too few dimensions. As the examples in Section 1 illustrate, embedded plots can be powerfully useful in many contexts.

SUPPLEMENTARY MATERIALS

Appendix Appendix.pdf, A demonstration of the ggsubplot package that discusses how to best handle geoms, stats, aesthetic mappings, parameters, position adjustments, and reference objects when extending the grammar of graphics to include embedded plots. (PDF file)

R code: 0-clean.r, 1-figures.r, 2-example.r, R scripts that load and clean the data used in this article’s examples, make the figures that appear in this article, and recreate the code examples in the appendix. (R scripts)

Data casualties-by-region.RData, seasons.RData, Data files used to create Figure 3 and Figure 8. (RData files)

[Received January 2013. Revised December 2013.]

REFERENCES

Anderson, E. (1957), “A Semigraphical Method for the Analysis of Complex Problems,” Proceedings of the National Academy of Sciences of the United States of America, 43, 923. [28]
Bertin, J. (1983), *Semiology of Graphics*, Madison, WI: University of Wisconsin Press. [33]

Carr, D. B., Chen, J., Bell, B. S., Pickle, L., and Zhang, Y. (2002), “Interactive Linked Micromap Plots and Dynamically Conditioned Choropleth Maps,” in *Proceedings of the 2002 Annual National Conference on Digital Government Research*, Digital Government Society of North America, pp. 1–7. [33]

Carr, D. B., and Pierson, S. M. (1996), “Emphasizing Statistical Summaries and Showing Spatial Context With Micromaps,” *Statistical Computing & Graphics Newsletter*, 7, 16–23. [28]

Carr, D. B., White, D., and MacEachren, A. M. (2005), “Conditioned Choropleth Maps and Hypothesis Generation,” *Annals of the Association of American Geographers*, 95, 32–53. [28]

Chambers, J. (1983), *Graphical Methods for Data Analysis*, New York: Springer. [28]

Cleveland, W. (1994), *Elements of Graphing Data*, Summit, NJ: Hobart Press. [35]

Cleveland, W., and Terpenning, I. (1982), “Graphical Methods for Seasonal Adjustment,” *Journal of the American Statistical Association*, 77, 52–62. [27]

Cowan, N. (2000), “The Magical Number 4 in Short-Term Memory: A Reconsideration of Mental Storage Capacity,” *Behavioral and Brain Sciences*, 24, 87–114. [31,32]

Dehaene, S., Spelke, E., Pinel, P., Stanescu, R., and Tsivkin, S. (1999), “Sources of Mathematical Thinking: Behavioral and Brain-Imaging Evidence,” *Science*, 284, 970–974. [41]

Gribov, A., Unwin, A., and Hoffman, H. (2006), “About Glyphs and Small Multiples: Gauguin and the Expo,” *Statistical Computing and Graphics Newsletter*, 17, 14–17. [28,33]

Hobbs, J., Wickham, H., Hofmann, H., and Cook, D. (2010), “Glaciers Melt as Mountains Warm: A Graphical Case Study,” *Computational Statistics*, 25, 569–586. [28]

Kleiner, B., and Hartigan, J. (1981), “Representing Points in Many Dimensions by Trees and Castles,” *Journal of the American Statistical Association*, 76, 260–269. [28]

Kotovsky, J., and Simon, H. (1985), “Why are Some Problems Hard? Evidence From Tower of Hanoi,” *Cognitive Psychology*, 17, 248–294. [32]

Lanzenberger, M., Miksch, S., and Pohl, M. (2003), “Stardinates-Visualizing Highly Structured Data,” in *Proceedings of the Seventh International Conference on Information Visualization*, IEEE, pp. 47–52. [28]

Luck, S., and Vogel, E. (1997), “The Capacity of Visual Working Memory for Features and Conjunctions,” *Nature*, 390, 279–280. [32]

Miller, G. (1956), “The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information,” *Psychological Review*, 63, 81–97. [31]

Minard, C. (1862), *Des Tableaux Graphiques et des Cartes Figuratives, par M. Minard*, Paris, France: impr. de Thunot. [27]

Neisser, U. (1976), *Cognition and Reality: Principles and Implications of Cognitive Psychology*, New York: WH Freeman/Times Books/Henry Holt & Co. [32]

Pickett, R., and Grinstein, G. (1988), “Iconographic Displays for Visualizing Multidimensional Data,” in *Proceedings of the 1988 IEEE Conference on Systems, Man and Cybernetics*, IEEE Press, Piscataway, NJ (Vol. 1), pp. 514–519. [28]

R Development Core Team (2010), *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria. [33]

Rumelhart, D. (1980), “Schemata: The Building Blocks of Cognition,” in *Theoretical Issues in Reading Comprehension: Perspectives From Cognitive Psychology, Linguistics, Artificial Intelligence, and Education*, eds. Spiro, R., Bruce, B., and Brewer, W., Hillsdale, NJ: Lawrence Erlbaum. [32]

Sarkar, D. (2008), *Lattice: Multivariate Data Visualization with R*, New York: Springer Verlag. [28,33]

Schloerke, B., Crowley, J., Cook, D., Hofmann, H., and Wickham, H. (2011), “GGally: Extension to ggplot2,” [http://cran.r-project.org](http://cran.r-project.org) . [33]
Schneider, W., and Shiffrin, R. (1977), “Controlled and Automatic Human Information Processing: I. Detection, Search, and Attention,” *Psychological Review*, 84, 1–66. [32]

Shiffrin, R., and Schneider, W. (1977), “Controlled and Automatic Human Information Processing: II. Perceptual Learning, Automatic Attending and a General Theory,” *Psychological Review*, 84, 127–190. [32]

——— (1988), “Cognitive Load During Problem Solving: Effects on Learning,” *Cognitive Science*, 12, 257–285. [31]

Sweller, J. (2003), “Evolution of Human Cognitive Architecture,” *Psychology of Learning and Motivation*, 43, 215–266. [32]

Sweller, J., Ayres, P., and Kalyuga, S. (2011), *Cognitive Load Theory*, New York: Springer Verlag. [32]

Wickham, H. (2009), *ggplot2: Elegant Graphics for Data Analysis*, New York: Springer. [28,33,34]

——— (2010), “A Layered Grammar of Graphics,” *Journal of Computational and Graphical Statistics*, 19, 3–28. [28,33]

Wickham, H., Hofmann, H., Wickham, C., and Cook, D. (2012), “Glyph-Maps for Visually Exploring Temporal Patterns in Climate Data and Models,” *Environmetrics*, 23, 382–393. [28]

Wilkinson, L., and Wills, G. (2005), *The Grammar of Graphics*, New York: Springer Verlag. [28,33]