Visualizing complex data with embedded plots

Garrett Grolemund and Hadley Wickham

August 13, 2012

Abstract

We describe a class of graphs, embedded plots, that are particularly useful for analyzing large and complex data sets. Embedded plots organize a collection of graphs into a larger graphic. This arrangement allows more complex relationships to be visualized with static graphs than would be otherwise possible. Embedded plots provide additional axes, prevent overplotting, provide multiple levels of summarization, and facilitate understanding. Complex data overwhelms the human cognitive system, which prevents comprehension. Embedded plots preprocess complex data into a form more suitable for the human cognitive system through visualization, isolation, and automation. 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.

1 Introduction

Analyzing large, complex data is difficult. Complex data strains the human cognitive system, which can prevent comprehension. Visualizations can help, but it is difficult to visualize more than two or three dimensions at once in a static graph. We present a class of graphs, embedded plots, that are ideal for visualizing complex data.

Embedded plots can be generalized as graphics that embed subplots within a set of axes. Figure 1 shows three graphs that represent this type of plot: William Cleveland's subcycle plots, glyphmaps, and the binned graphics that are emerging from big data visualization efforts. When viewed on its own, each subplot is a self contained plot (or would be if it contained the appropriate axis, labels, and legend). The axes of the subplot do not have to be the same as the axes that the subplot is positioned on. In fact, the subplot can use an entirely different coordinate system than the higher level plot. For example, Figure 1.b. embeds polar graphs in a cartesian coordinate system.

Embedded plots have a rich pedigree and a growing future. Subcycle plots were devised by William Cleveland [Cleveland and Terpenning, 1982], one of the leading innovators in computer based graphics. Glyphs and other plots have been embedded in maps since Charles Minard [Minard, 1862]. Such maps figure prominently in Bertin's *Semiologie of Graphics* (1983), a seminal work in the academic study of visualization. Embedded maps comprise 21 pages of the text. More recently, glyphmaps have been developed as a tool for tracking climate and climate change data [Wickham et al., Submitted, Hobbs et al., 2010]. The binned graphics of Figure 1.c are a promising candidate for solving the problem of

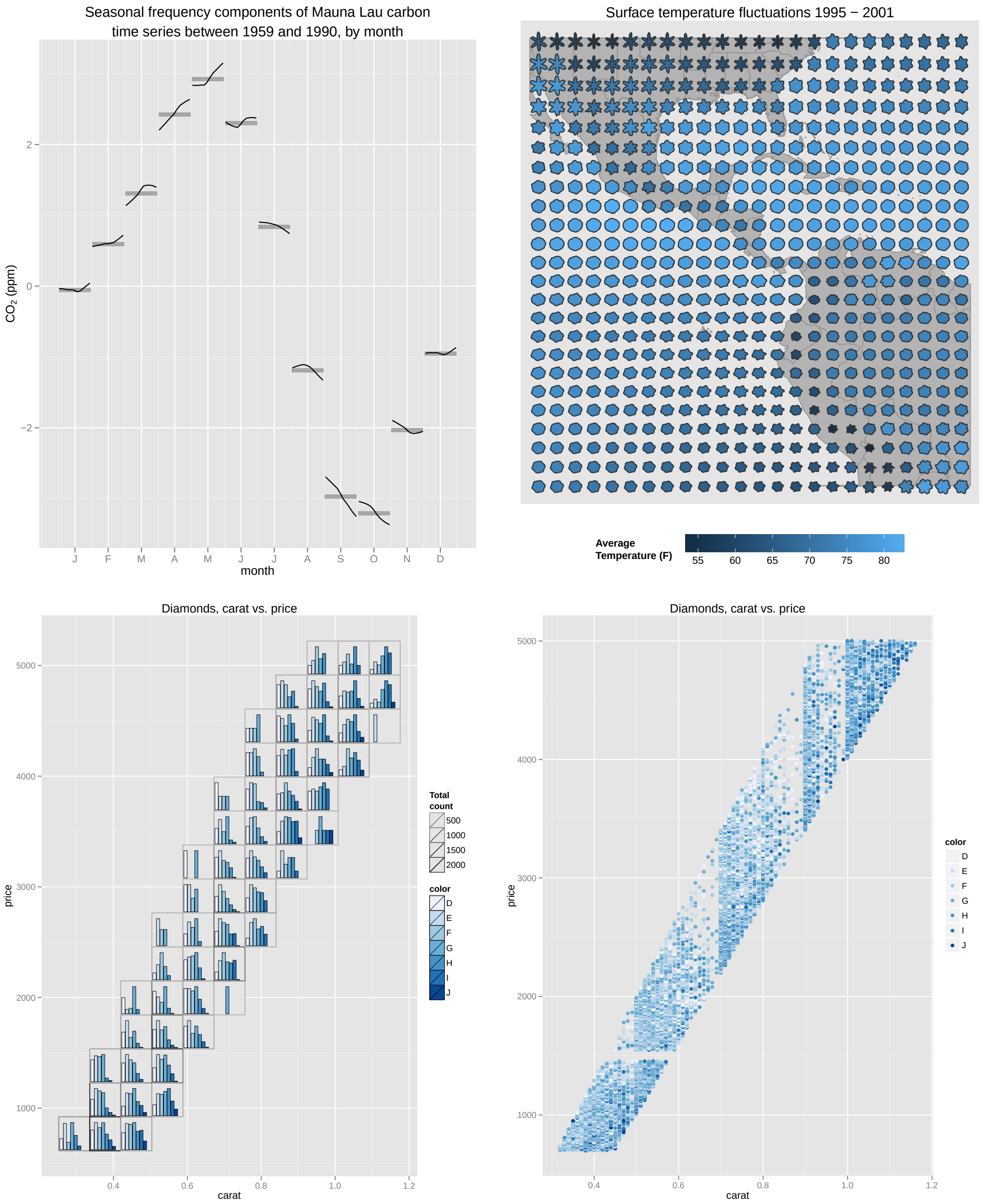

Figure 1: Three examples of graphs that use embedded subplots. **A.** (*upper left*) A subcycle plot of CO_2 measurements taken on Mauna Lau, Hawaii between 1959 and 1990. Recreated from Cleveland [1994], page 187. Observations are grouped by month. **B.** (*upper right*) A glyphmap of temperature fluctuations in the western hemisphere over a six year period. Each glyph is a polar chart with $r = temperature$ and $\theta = date$ these charts are organized on a cartesian plane with $x = longitude$ and $y = latitude$. **C.** (*lower left*) A binned plot of the diamonds data set from the `ggplot2` software package. Subplots are used to show patterns in diamond colors without overplotting. When this data is presented in its raw form, the accumulation of points hides patterns in the data (*lower right*).

overplotting when visualizing big data. Other types of embedded plots are widely used as well. Glyphs [Anderson, 1957], trees and castles [Kleiner and Hartigan, 1981], chernoff faces [Chernoff, 1973], stardinates [Lanzenberger et al., 2003], icons [Pickett and Grinstein, 1988] and others have been developed as types of subplots that can be compared to each other. Scatterplot matrices [Chambers, 1983], trellises [Sarkar, 2008] and facets [Wilkinson and Wills, 2005] are popular types of embedded graphics that arrange subplots into a table. We generalise all of these graphs into a larger class of plots, embedded plots, because they all share a two tier structure. The first tier is the overall graph or visual itself, the second tier is the collection of subplots that appear within the graph.

The two tiered structure of embedded graphs makes them well suited for solving a number of data analysis problems. The examples in Figure 1 illustrate three areas where embedded graphics are particularly useful. First, embedded graphics make it easy to visualize interaction effects. For example, Figure 1.a shows that the direction of long term change in CO_2 levels at Mauna Lau observatory in Hawaii varies by month in relation to seasonal patterns in CO_2 concentrations. Embedded graphics also provide an intuitive way to organize spatio-temporal data. Visualizing spatio-temporal 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. The glyphmap in Figure 1.b organizes these dimensions in a way that is easily interpreted and that makes both spatial and temporal patterns obvious. Finally, embedded graphics solve the problem of overplotting. Figure 1.c. represents almost 20,000 observations. When this data is plotted as a colored scatterplot, the accumulation of points obscures the underlying relationship between carat, color, and price. The use of binned subplots makes the relationship visible again. Yet embedded plots provide more than just practical advantages.

Embedded plots amplify the abilities of the human cognitive system by presenting complex information in a way that is particularly easy to process. Complex data is data that includes multiple simultaneous relationships between its elements. At the cognitive level, complex data overwhelms the capacity of the working memory Sweller [1994]. Repeated studies have shown that it is difficult to comprehend, use, and teach complex data.[1] Moreover, success in understanding complex data depends heavily on how the data is presented Mayer [2009]. Embedded plots present data in a way that exploits several known mechanisms for facilitating the processing of complex data. As a result, embedded plots may allow viewers to comprehend information that they would not grasp in other formats.

As useful as embedded plots are, it is difficult to make them. Currently, programs that can make embedded plots focus on a specific type of subplot, such as glyphs [Gribov et al., 2006] or scatterplot matrices [Sarkar, 2008]. This limits the customizability and usefulness of embedded plots. 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 paper 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 data set 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 can not be seen in single level graphs.

Section 3 examines why embedded plots are useful tools for finding and communicat-

[1] See Sweller et al. [2011] for an overview.

ing information found in large data sets. At the practical level, embedded plots have two advantages: they provide two extra axes and a high degree of customizability. More importantly, however, embedded plots exploit several cognitive mechanisms for attending to and processing information. This allows embedded plots to present complex information without becoming muddled or indecipherable.

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 offereing general principles to guide the use of embedded plots.

2 Case Study: Analyzing complex data

The Afghan War Diary data, 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 data set was collected by the US military and contains information about military events that occurred in or around Afghanistan between 2004 and 2010. Among other variables, the data set 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 data set is large enough (76,000 observations) that overplotting becomes a concern when visualizing the data. The data set is complex in that it contains a spatio-temporal component: each observation is labelled by longitude, latitude, and date. Our analysis will focus on two topics: the ratio of civilian casualties to combatant casualties and the escalation (or de-escalation) of hostilities since 2004 as measured by total casualties. We will calculate total casualties based only on the number of wounded and killed in each group. The Afghan War Diary does not have complete information on the number of people captured or missing across all four groups.

2.1 Civilian casualties

Operation Enduring Freedom, the US led military engagement in Afghanistan, has received international criticism for the high number of civilian casualties associated with the war. The Afghan War Diary seems to justify this criticism. Civilians comprise almost a quarter of all casualties recorded in the diary, and civilians have suffered 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 can not 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.

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 suggests that alarmingly high civilian casualty rates occur throughout Afghnaistan and not just near population centers like Kabul, although high civilian casualty rates also occur there as well.

2.2 Frequency of hostilities

Operation Enduring Freedom has also been criticised for lasting longer than any previous American war without showing signs of abatement. We would like to look for signs of abatement in the total number of casualties by region. If the total number of casualties in a region has decreased over time, this may suggest that the region has become pacified, a sign of progress.

Events in the Afghan War Diary are labelled according to the region in which they occurred: the capital, the north, the east, the west, or the south and unknown locations, which mostly have lattitude and longitude positions in Pakistan. These labels allow us to visualize how the war has progressed in different areas over time, Figure 3.a. However, we can only see the change in time with this plot. Embedded line plots allow us to see variation in space and time simultaneously, Figure 3.b. We again plot the conditional distributions to better see the pattern in each region, Figure 3.c. We can also use the background color of each subplot to display the total number of casualties per region. This is the information we would normally lose by looking at conditional distributions instead of marginal distributions. We see that casualties peaked in most locations around 2007, but have been on the rise again in the most recent years.

Although the embedded plot “increases” the complexity of Figure 3.a by adding two new dimensions (latittude and longitude) and over 100 new lines, it actually makes it easier to see the spatio-temporal relationship. The viewer no longer has to expend mental energy thinking about which line in Figure 3.a corresponds to which part of the country.

3 Benefits of embedded plots

Embedded subplots expand the power of static graphics. Adding a second tier of information in the form of subplots creates practical advantages not available with non-embedded plots. This second tier may at first seem counterproductive: embedded subplots increase the complexity of the graph, which can obstruct comprehension. However, embedded subplots present information in a way that minimizes the cognitive load a viewer must expend to understand the graph. This makes embedded subplots unusually comprehensible. Below,

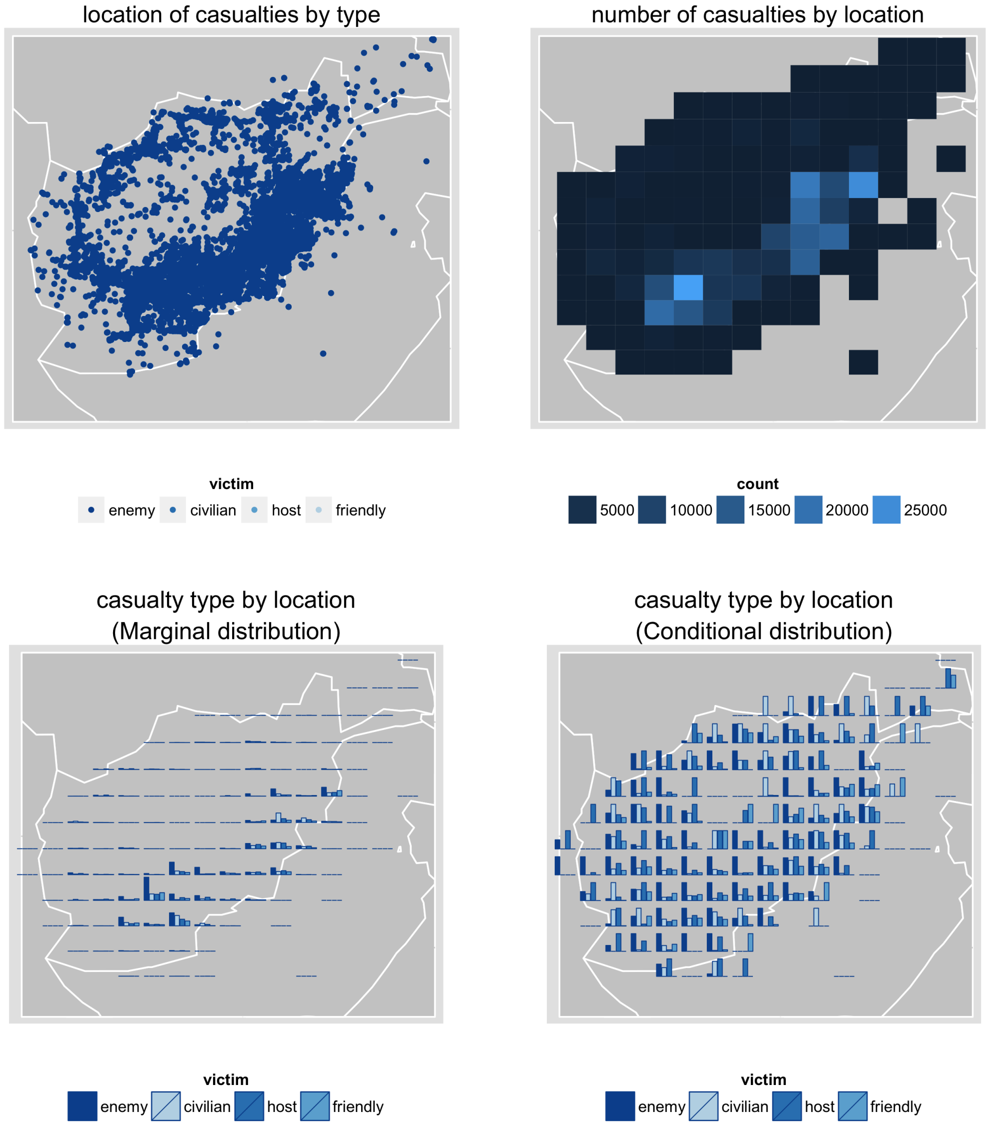

Figure 2: **A.** (*upper left*) Relative rates of casualties by area in Afghanistan between 2004 and 2010. Raw data can not be visualized due to overplotting. **B.** (*upper right*) A heat map shows casualty counts, but not relative rates by group. **C.** (*lower left*) Embedded bar charts reveal that there have been more civilian than combatant casualties around Kabul, the capital of Afghanistan. Marginal bar charts reveal similar information as a heat map, but also display rates by group. **D.** (*lower right*) Conditional bar charts make regional rates the more obvious; they show that inordinate civilian casualties is not unique to the capital city.

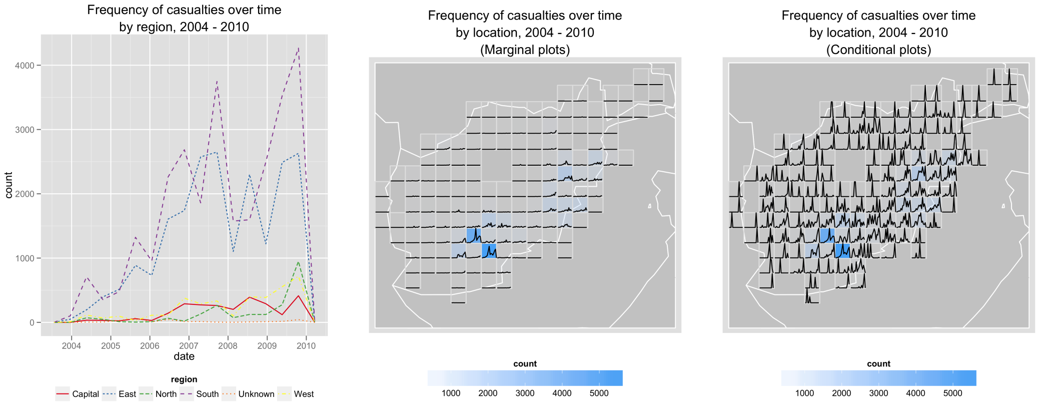

Figure 3: Casualty frequencies between 2004 and 2010 by region. The embedded graphics show that the heaviest fighting has been confined to the southern and eastern regions of Afghanistan. The most casualties have occurred around Kandahar. Many regions seem peaceful since 2008. However, casualties have increased recently throughout southeast Afghanistan.

we review the practical advantages of embedded subplots as well as the cognitive science findings that suggest that embedded subplots can be simple and easy to understand.

3.1 Practical advantages of embedded subplots

Embedded graphics provide two advantages over non-embedded graphics: they allow customizeable summarization and provide additional x and y axes. Each of these advantages can be used in a variety of ways.

Common strategies for overplotting, such as heat maps and contour maps, summarize data into a single number and then attempt to visualize that number. In contrast, subplots summarize information into an image, which can carry more information than a lone number. For example, the bar charts in Figure 2.c display multiple measurements in the same space as a heatmap tile, which only displays one. By summarizing with an image, subplots allow users to choose between no summarization, partial summarization and complete summarization, Figure 4. Distracting data can be removed, but enough information can be retained to display complex relationships.

The choice of a subplot also allows the user to control effects of overplotting. For example, Figure 1.c summarizes more than 20,000 data points. When this data is viewed as a colored scatterplot, points occlude each other and underlying patterns are hidden, Figure 1.d. The use of embedded subplots avoids overplotting and shows a relationship 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 data set was enlarged to 100,000, a million, or even a trillion points.

Embedded subplots also provide a second practical advantage: they supply an additional set of axes to plot data on, the minor x and y axes of the subplots. These two new dimensions allow complex relationships to be visualized. Four separate variables can be assigned between the major x, major y, minor x, and minor y axes. Additional variables can

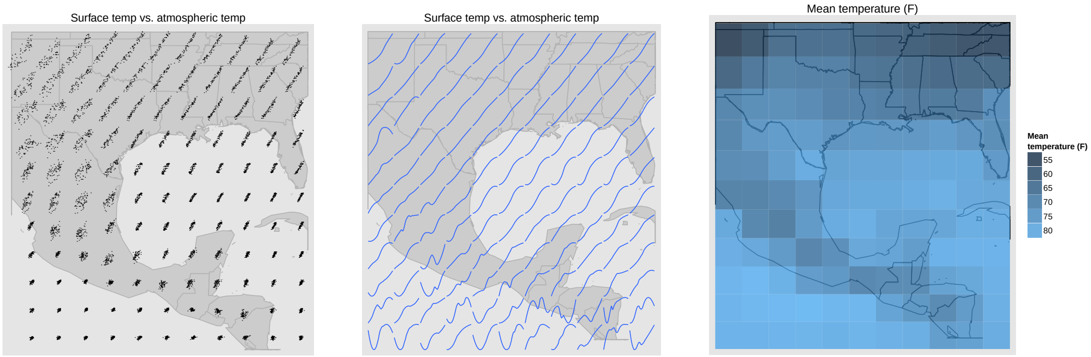
Surface temp vs. atmospheric temp
Surface temp vs. atmospheric temp
Mean temperature (F)
Mean temperature (F)
55
60
65
70
75
80

Figure 4: Users can control the amount of summarization that occurs in an embedded plot. When scatterplots are used for subplots, no summarization occurs (*left*). Line graphs provide partial summarization (*center*). Heat maps provide complete summarization, within each bin data is reduced to a single number (*right*). This may not always be desirable.

be included with colors, shapes, sizes, etc. The usefulness of this approach is most easily seen with spatio-temporal data. Spatio-temporal data usually requires at least four dimensions to be visualized: two dimensions for spatial coordinates, a third dimension for time, and a fourth for the quantity of interest. Four variables can quickly clutter a non-embedded plot, but an embedded plot can visualize them with just the major and minor x and y axes. The extra axes can be used in a similar way to visualize any high dimensional relationship, such as interaction effects and conditional effects. The two level system of axes can also be used to organize data first by groupwise characteristics, then by individual characteristics.

3.2 Cognitive advantages of embedded subplots

These practical advantages come at the expense of making a graph more complicated. Embedded plots add an extra layer of information that a viewer must process before comprehensions can occur. However, when used appropriately, embedded plots may not be much more difficult to understand than non-embedded plots. Furthermore, embedded plots may allow users to understand information that would be incomprehensible in other formats. This is because embedded plots organize information in a way that lowers the cognitive load required to process the information.

Cognitive load is the mental energy required to convert information into knowledge, understanding, and insights within the working memory. Cognitive science has long known that the working memory has a fairly small processing capacity [Miller, 1956, Cowan, 2000]. In 1988, John Sweller demonstrated that learning fails to occur when the cognitive load of a task exceeds the capacity available in the working memory [Sweller, 1988]. This insight, the basis of Cognitive Load Theory, has led to a series of successful education principles that work by reducing the cognitive load required during learning tasks (such as reading a graph).[2]

Cognitive load theory explains why it is easy to make a graph confusing by including distracting information or multiple variables. Each variable increases the cognitive load

[2] See Sweller [2003] for an extensive list of these principles and the supporting literature

required to interpret the graph by adding new information that the viewer must process. Current estimates suggest that the average person has trouble processing more than four pieces of novel information at once [Cowan, 2000]. Complex data affects the working memory in the same way as a complicated graph. Each relationship and interaction increases cognitive load and threatens to overwhelm the working memory [Sweller, 1994]. When this happens, comprehension will not occur. However, mechanisms exist that can decrease the cognitive load associated with learning tasks. These mechanisms allow more information to be processed than would otherwise be possible. Embedded plots automatically employ three such mechanisms: visualization, isolation, and automation.

Visualization is an extremely powerful information processing tool. All graphics rely on it, but embedded plots use it in a specific way to present information with a decreased cognitive load. Mounting evidence suggests that thinking is a primarily visual activity. The mind uses visual simulations to process verbal information, such as the orientation of words in a sentence [Stanfield and Zwaan, 2001] and the relationship between words [Zwaan and Yaxley, 2003]. The mind also relies on visual simulations to compare numbers [Moyer and Landauer, 1967, Dehaene, 1997], to perform approximate arithmetic [Dehaene et al., 1999, Walsh, 2003], and to make logical deductions [Lakoff and Nunez, 2000]. The human adaptation to vision is reflected in our working memory system, which treats visual and verbal information differently [Baddeley et al., 1974]. The working memory can only handle four novel objects, whether verbal or visual. However, each piece of visual information can be an image that contains multiple features. A study by Luck et al. [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. If we compare one tile of a heat map to one subplot, we see that embedded plots exploit this bandwidth more effectively than other graphs. Both the tile and the subplot are a single visual object. The tile has one feature (a color). The subplot has four (four bar lengths). Luck et al. [1997]'s study suggests that the subplot should require little (if any) more cognitive load to be processed by the working memory than the tile. The subplot, however, conveys four times as much information.

Embedded plots also organize information in a way that further decreases cognitive load. The working memory must expend considerable cognitive energy to process new information, but almost no energy to recall and use previously acquired information [Sweller, 2003]. When the complexity of a data set exceeds the capacity of the working memory, the mind can proceed by dividing the data set into small pieces and processing each separately. This is akin to rote learning. It does not create full understanding; connections between the separate pieces go unnoticed and unexamined. However, once each piece is processed, it becomes part of the long term memory where it can be recalled at little to no cognitive cost. Further processing can then occur until full understanding is attained. [Sweller et al., 2011] call this the isolating elements effect. The mind can build a deep understanding of highly interactive (i.e., complex) data by iterating between processing small subsets of data and then recalling these subsets from the LTM to compare against each other and new information.

Embedded plots usually display information that can not be understood without an approach that isolates elements; these plots usually deal with at least four interacting dimensions (major x, major y, minor x, minor y). Such data will always demand a heavy cognitive load for comprehension. However, embedded plots make this load manageable by dividing the data into isolated elements (subplots) and visualizing the interactions between these elements (the overall graph). This arrangement allows the mind to use its strongest information processing channel, visualization, to perform the isolating elements processing algorithm. As a result, embedded plots are a particularly efficient way to present information that has four to six interacting dimensions in a static graph.

Embedded plots also benefit from a third cognitive mechanism: automation. To process new information, the mind uses a cognitive structure known as a schema. The schema directs attention during information processing and identifies relationships between data points and previous knowledge.[3] When the mind frequently uses a particular schema, it becomes *automated* [Schneider and Shiffrin, 1977, Shiffrin and Schneider, 1977]. When this happens, information related to the schema can be processed with less and less conscious effort. KotovskyJ R and Simon [1985] demonstrated that automated processing decreases cognitive load to such an extent that information can be processes 16 times faster than with non-automated 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 second example of automated processing.

Embedded plots rely on graph reading skills to convey information: each subplot is a new graph. For analysts familiar with reading graphs, embedded plots allow information to be processed automatically, which results in quicker processing and reduced cognitive load. Embedded plots provide twice the opportunity for automation when subplots are embedded in a map. Reading data off a map and associating it with spatial coordinates is an activity commonly practiced by analysts and non-analysts alike. Information may be automatically read off these graphs at both the plot level and the subplot level. Embedded plots will not offer the benefits of automation to everyone, though. Occasionally, we hear anecdotal reports of people who can not easily read graphs. Until a person learns to read statistical graphs, conscious effort will be required to interpret embedded plots, but this will be true for other types of graphs as well.

In summary, embedded plots display more information than other static graphs, but remain easily interpretable. They 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. This is an attractive feature: it allows embedded plots to display complex relationships that would not otherwise appear in static graphs. More fundamentally, embedded plots may allow users to comprehend complex relationships that would remain incomprehensible in other formats. Embedded plots are not a panacea for all big data situations: it is possible to abuse embedded plots, as we describe in Section 5. Also embedded plots can not effectively visualize relationships that involve more than six dimensions. However, embedded plots provide a way to present one or two additional dimensions in a static graphic; this creates increased opportunities for exploring and understanding large, complex data.

[3] Literature on schemas are extensive. See Neisser [1976] Rumelhart [1980], etc. for highlights

4 Implementing embedded plots with 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 et al., 2006]. Facetted graphs can be made with the ggplot2 [Wickham, 2009] and lattice [Sarkar, 2008] packages in R. Scatterplot matrices can be created with the GGally package [Schloerke et al., 2011] as well as with base R [R Development Core Team, 2010]. However, these programs do not allow users to customize which type of subplot to use in an embeded plot. This customization is one of the chief advantages of embedded plots. Different types of subplots reveal different types of relationships and provide different levels of summarization. In this section, we describe how to create software that can produce any type of embedded plot. Our implementation is built on the layered grammar of graphics and reveals a conceptual insight about graphics: graphs are hierarchical, or recursive, in structure. The implementation of embedded subplots described in this section is available for the R programming language through ggsubplot. ggsubplot is a software package written by the authors that implements embedded plots within the grammar of graphics paradigm. ggsubplot is written in the R programming language and extends the ggplot2 package. The ggsubplot package is available from http://github.com/garrettgman/ggsubplot.

Embedded plots can be easily implemented in software built on the layered grammar of graphics, 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 data set. 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 200 times in scholarly journals and supports an online community of 2500 members. The grammar 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 whose appearance can vary in relation to an underlying data set. For example, the points in a scatterplot are a type of geom. Their locations (and sometimes their sizes and colors) reflect values in the underlying data set. Other types of geoms include the bars in a bar chart, the lines in a line chart, boxplots, et cetera. 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 data set. For example, the color of a point can be used to display the gender of an observation in the data set. Two of the most important aesthetics are a geom's position along the x axis and y axis. The grammar of graphics calls the rules used to map aesthetics to variables in a data set *mappings*. Geoms and mappings provide a useful framework for building generalized graphs.

Embedded graphics fit seamlessly with the grammar of graphics if we recognize that a plot can be a geom (and that every geom is a plot). Embedded subplots share the useful characteristics of a geom. They can visually represent data within a graph, and they possess aesthetics that can be mapped to a data set's values. Subplots have two primary aesthetics: their position in the cartesian plane and their internal drawing of a graph. This second aesthetic makes subplots appear more complicated than other geoms, but they function in the same way.

Cleveland's subcycle plot demonstrates the equivalence between subplots and geoms, Figure 5. The plot visualizes atmosperic CO_2 concentrations as measured at the Mauna Loa Observatory in Hawaii from 1959 to 1990 [Cleveland, 1994]. This data was some of the earliest to suggest the presence of man made global climate change. CO_2 readings are organized by month along the x axis. Within each month, CO_2 readings 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 stand alone plot once the appropriate axes are added back in, see Figure 5.b. In the subcycle plot, 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 5.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 aestetic can be mapped to the underlying data set with a graph specification.

It may seem exotic to equate a description of a graph with a mapping between a visual feature and data, but this follows a basic tenet of the grammar of graphics: a graph is an abstract mapping from data to visualization:

> We can construct a graphic that can be applied to multiple datasets. Data are what turns an abstract graphic into a concrete graphic. [Wickham, 2010]

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. Note that the internal drawing of a subplot may or may not contain axes, a grid, a legend, etc. just as a regular graph may or may not contain these elements.

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 6.

4.1 Advanced implementation

The grammar of graphics does more than describe the components of a graph, it defines how these components can be combined to make useful images. Implementing subplots as a geom requires specific considerations when combining subplots with the technical details of the grammar of graphics. These details include stats, position adjustments, and reference objects, such as coordinate axes. In this section, we discuss these considerations and illustrate them with code from `ggsubplot`.

Geom

`ggsubplot` introduces two new geoms that draw embedded subplots. These geoms will serve as examples for the technical considerations in the remainder of this section. `geom_subplot` uses a *group* aesthetic to assign data to subplots and then positions each subplot based on

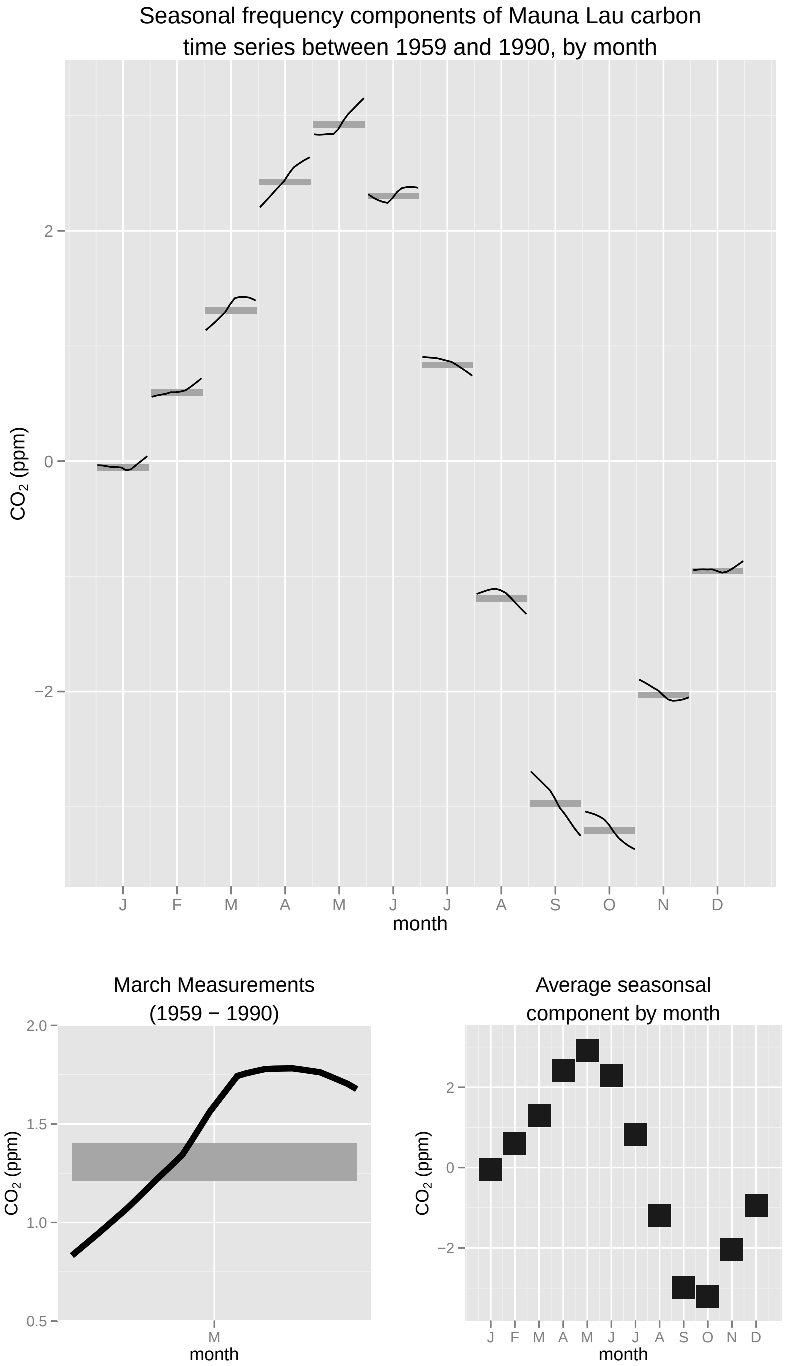

Figure 5: Cleveland's subcycle plot can be decomposed into twelve subplots arranged as a scatterplot. The subplots behave as a rectangle geom with an internal drawing aesthetic.

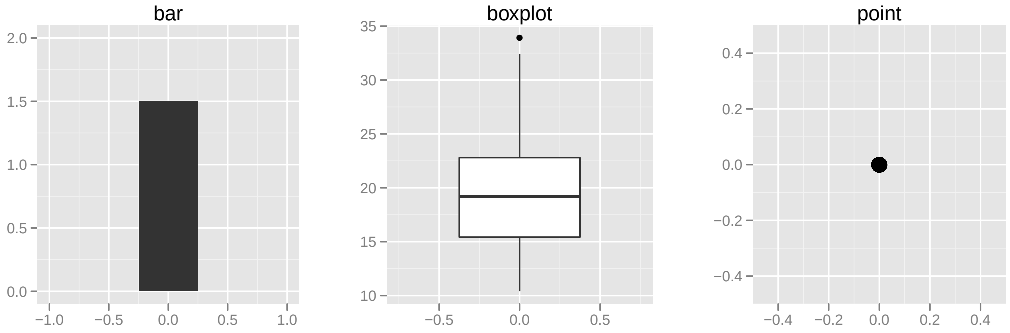

Figure 6: Every individual geom is a self contained plot when paired with a set of axes. Such plots may be not be very interesting, as is the case with point geoms.

a summary of its data points. `geom_subplot2d` bins the surface of a plot into a two dimensional grid and then represents each bin with a subplot. The use of these geoms is demonstrated in the example code below, which was used to create Figure 1.b and Figure 1.c.

```
## Figure 1.b, a glyphmap
ggplot(nasa) + map_americas +
  geom_subplot(aes(long, lat, group = id,
  subplot = geom_star(aes(r = surftemp, angle = date, fill =
mean(surftemp)),
  r.zero = FALSE, alpha = 0.75))) +
  coord_map()

## Figure 1.c, a binned plot
ggplot(ordinary.diamonds) +
  geom_subplot2d(aes(carat, price,
    subplot = geom_bar(aes(color, fill = color), position =
"dodge")),
    bins = c(10, 14), y_scale = free, height.adjust = 0.8,
    width.adjust = 0.8, ref = ref_box(aes(color = length(color))))
+
  scale_color_gradient("Total\ncount", low = "grey70", high =
"black")
```

Stats

A *stat* is any function that summarizes a group of data values into a smaller set of information. Stats and mappings form a two step processes whenever a single geom is used to display multiple data points. First, the stat summarizes the data points into summary level information. Then a mapping keys the aesthetics of the geom to the summary level information. For example, a boxplot geom uses a stat to calculate Tukey's five number summary for a group of data points. Then the numbers are used to determine the location of each part of the boxplot. Specific geoms are usually associated with specific stats. Box plots always use a five number summary, histograms always use a bin and count procedure. These patterns allow users to largely ignore stats; software can automatically implement the correct stat once a geom is chosen. When a user does decide to specify a stat, they are usually constrained to choose from a prepackaged set of stat functions.

Embedded subplots also rely on stats, but subplots require more freedom in the choice of stat than is offered in current implementations of the grammar of graphics. Each subplot must map a group of data points to a single location on the x and y axes of the large plot. The way a user chooses to do this is likely to change from graph to graph. In Figure 1.a., the y position for each subplot is mapped to the mean value of CO_2 for the observations in the subplot. In Figure 1.b., both the x and y positions of each subplot are mapped to the common longitude and lattitude of the observations within the subplot. In Figure 1.c., the x and y positions are mapped to the midpoint of the 2D bin that each group of observations has been assigned to. This variety prevents the subplot from relying on a set of prepackaged stats. Instead, users need the same freedom to create a stat as they have to create a mapping.

`geom_subplot` provides this freedom by having the mapping directly serve as a stat. If a mapping involves subsetting or a function that returns a single value (such as a mean), it will perform its own summarizing. The user just needs to ensure that the mapping is applied separately to each group used in the graph. Otherwise, the mapping will be applied to the entire underlying data set at once and each geom will be keyed to the same information, for example, the mean of the entire data set. `ggsubplot` manages these requirements with the `ply_aes` function. `ply_aes` takes a `ggplot2` layer object and modifies it so that the layer's mappings are applied groupwise according to the layer's $group$ aesthetic. `ply_aes` enforces summarization by subsetting the output of each mapping to just its first value. A warning message is given if the mapping would have otherwise returned multiple values. `geom_subplot` automatically uses `ply_aes`.

This arrangement provides a new insight into the relationship between mappings and stats. Mappings and stats perform the same function but are keyed to different levels in the hierarchy of information: individual level and group level. This parallelism is made clear in embedded plots. When we consider any single subplot in Cleveland's subcycle plot (Figure 1.b), the mean concentration of CO_2 is a groupwise statistic, (i.e., a stat) that summarizes an entire group of data. When our attention shifts to the higher level plot(Figure 1.c), the mean concentration of CO_2 becomes an aesthetic mapping of the subplots.

`ply_aes` can also be used with non-subplot layers. It behaves in the same way, turning individual mappings into groupwise mappings (i.e, stat + mapping). This technique replaces each group of geoms with a single geom that displays group level information. Figure 7 shows how this technique can remarkably reduce overplotting to reveal structure.

Position adjustments

Many embedded plots will require nontraditional choices for a position adjustment. Each layer of a graphic contains a position adjustment that determines how to plot graphical elements that interfere with one another. Wilkinson and Wills [2005] refers to this concept as a collision modifier. Position adjustments are often implicitly set to identity, which means that elements will be plotted on top of one another if they overlap. Alternatively,

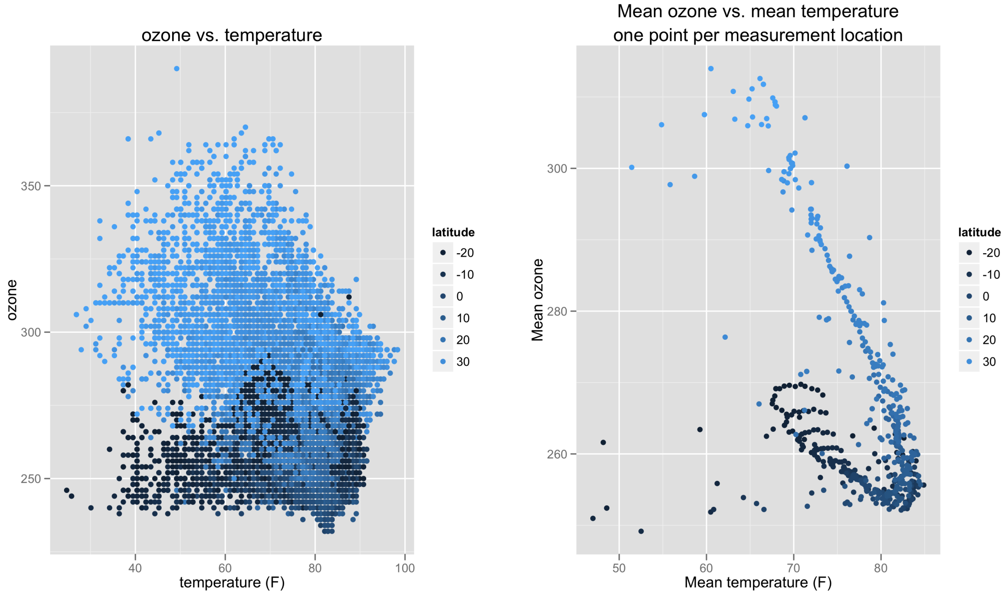

Figure 7: `ply_aes` offers a new strategy for overplotting. Groups of geoms are combined into single geoms that display summary information. This approach reveals that $mean(ozone)$ has a different linear relationship with temperature in the southern hemisphere than it does in the north.

overlapping elements can be adjusted to appear above each other (stacking), next to each other (dodging), in random nearby locations (jittering) or in other places. These solutions are inefficient for a large subset of embedded graphs.

Embedded graphics such as Figure 1.b and 1.c use the position of the subplot to signal which observations are included in the subplot. In Figure 1.b, every observation with a certain longitude and latitude is mapped to the subplot positioned at that longitude and latitude. In Figure 1.c, every observation that falls within a 2D bin is mapped to the subplot positioned at that bin. Even Figure 1.a uses position along the x axis to signal which observations are included in which line graph. This arrangement does not appear in every embedded graphic, but it can be useful. Traditional position adjustments such as stacking, dodging, and jittering disrupt this relationship. We suggest a new position adjustment that preserves the relationship between position and group membership: when two subplots overlap one another they can be merged into a single subplot.

Programming a merge adjustment is more complicated than programming traditional position adjustments. The merge adjustment will affect stat values because it alters group membership. Therefore, it must be computed early in the building process for graphs. The merge adjustment also presents a second difficulty: how do we define which subplots should be merged? `ggsubplot` combines each clique of overlapping subplots into a single subplot positioned at the mean location of the clique. This works well when graphs are relatively sparse, but can remove an undesirable amount of visual real estate when graphs are dense, see Figure 8. Clustering methods may provide a more useful approach to identifying graphs to be merged. Future versions of `ggsubplot` will explore this approach, but the most useful ways of merging are likely to arise from observing the actual application of embedded plots in data analysis. As an alternative to merging, users who wish to avoid overlaps can grid the subplots within a graph with `geom_subplot2d`. This is not a position adjustment, but a way of assigning the group aesthetic. However, gridding guarantees that membership will be mapped to position without any overlaps.

Reference objects

Reference objects are any object that is added to each subplot to provide a standard of comparison across subplots. The axes of most graphs are one of the most commonly used type of reference object. However, axes are difficult to read at the small scales used in subplots. Boxes and lines can also allow comparison and scale better to the smaller sizes of subplots. `ggsubplot` creates these objects with a reference parameter in the subplot layer, see Figure 9.

These reference objects allow viewers to judge the position of geoms inside the subplot and to make comparisons against the position of geoms in other subplots. To allow accurate comparisons, the dimensions of reference objects do not vary across subplots. They are fixed to the dimensions of the subplot. However, other features of the reference object can vary to provide additional information about a subplot. For example, the fill, color, and transparency of a reference object can display group level information about the data in a subplot. The `ggsubplot` reference parameter allows users to set these aesthetics with the functions `ref_box`, `ref_vline` and `ref_hline`, see Figure 9. By default, `ref_box` displays with a grey background and white border. This matches the color scheme of `ggplot2`'s default background, while still delineating the dimensions of the subplot. Reference objects provide a quick way to compare across subplots. However, if users require a precise judgement they should still plot the subplot in its own graph with a pair of axes.

4.2 Implications for the structure of graphics

Embedded plots demonstrate that graphics have a hierarchical, recursive structure. Graphs summarize information into an image, and images can themselves be summarized into a larger graph. This structure parallels a common pattern found in both human thought and data analysis, and suggests that graphs obey universal rules of data representation.

Many mental processes involve classifying, grouping, and aggregating. This is how we make sense of sensory information, and it is how we build data into information. At the cognitive level, the mind combines information in a number of ways. For example chunking, which extends our attentional resources, and building schemas, which assigns meaning to data. Even the sensory data that we collect is progressively aggregated and summarized as it travels through the neural network of our brain.

These mental processes guide the data collection process. As a result, they shape data that has been collected by, cleaned by, or manipulated by people. Measurements often do not directly collect information of interest. This information is built by grouping together similar observations and applying some aggregating function to the data. For example, a researcher may observe a subject's height, hair length, body shape and clothing choices and then aggregate them into a conclusion about the subject's gender. This pattern of grouping and summarizing is also an important strategy for dealing with variation, both random and systematic. The average of a groups of observations is less affected by random variation than a single observation.

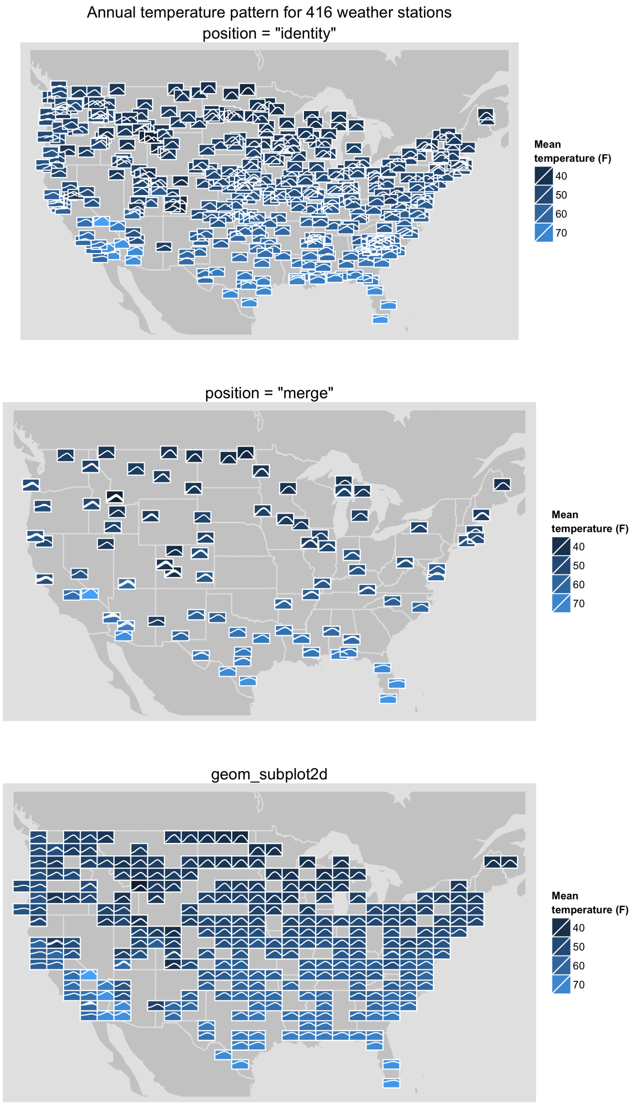

Figure 8: The postion of a subplot is often related to which points the subplot shows. Position = merge and `geom_subplot2d` provide two ways to avoid overlapping subplots without disrupting this relationship.

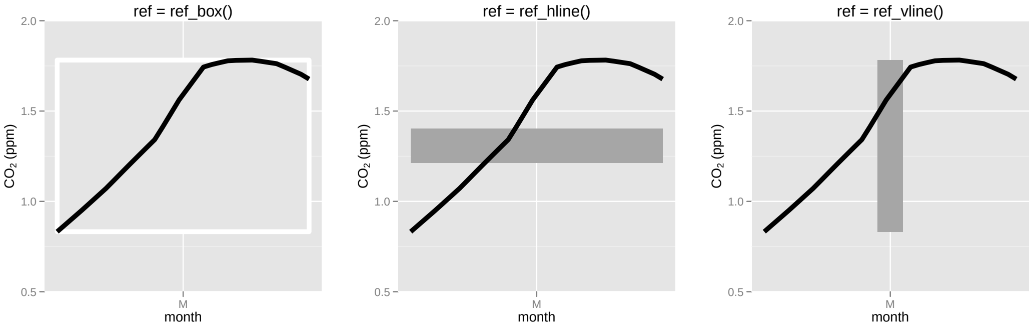

Figure 9: Reference objects allow comparison across subplots and can be more easily read at small scales than coordinate axes. In `ggsubplot`, users can add one of three types of reference objects to subplots by adding `ref = ref_box()`, `ref = ref_hline()`, or `ref = ref_vline()` to `geom_subplot` and `geom_subplot2d` calls.

Data built this way often has a hierarchical structure. A set of observations can be summarized in a table of counts. A set of counts can be summarized by a single average. Averages from different populations can be compared against each other and so on. Moreover, a hierarchical structure can be built from any multivariable data set. A variable can be selected for sorting the data into groups. Groups can be made from all observations that share the same value of a discrete variable or all observations that fall into the same range of a categorical variable. Each group can be summarized in a variety of ways with a variety of functions to provide a higher level data set. Correlation betwen summary measures and grouping variables reveals structure in the data and hints at possible causal relationships. This process underlies the split-apply-combine strategy for data analysis that has been embedded into `R` as the `plyr` package [Wickham, 2011].

Mathematical functions are one way to summarize a group of data points. Graphics are another. Graphics organize data points into a meaningful summary that can be processed by the mind. This summary is the visual image of the graph. A graph resembles a cognitive schema because it puts various facts in relationship with one another and suggests meaning. A graph resembles a chunk because it can be more easily attended to, remembered, and analyzed than its constituent components. Graphs even have an efficiency advantage over mathematical functions at the cognitive level because they provide visual input. Embedded plots show that just as numerical summaries of data can themselves be summarized, graphical summaries of data points can be organized into higher level graphs.

Implementing embedded plots suggests that components of the graphics of grammar may originate from the universal, hierarchical structure of information. Aesthetic mappings create mappings between individual data points and visual aesthetics. Stats create mappings from group level information to visual aesthetics. They accomplish this by aggregating and summarizing the individual data points within a group of data. In otherwords, stats and aesthetics perform the same function but are keyed to different levels in the hierarchy of a data set. This parallelism is made clear in embedded plots. When we consider any single subplot in Cleveland's subcycle plot, the mean concentration of CO_2 is a groupwise statistic, (i.e., a stat) that summarizes an entire group of data. When our attention shifts to the higher level plot, the mean concentration of CO_2 becomes an aesthetic mapping of the subplots.

Graphics, data, and human modes of information processing all utilize hierarchical structures of information. This universality suggests that a univeral language of data representation may also exist. This idea is further attested to by the way each of these domains groups data points and summarizes these groups to obtain higher level data points. Identifying the rules of this universal language may make it easier to teach and practice data visualization and data anaylsis.

5 Conclusion

Embedded plots are a powerful visualization tool for many data analysis tasks. Because embedded plots organize multiple dimensions of data into a static two dimensional graph, they can provide insights not found in other types of graphics. Because embedded plots present complex data in a cognitive friendly way, they facilitate understanding that could not occur otherwise. Despite presenting more complex data, embedded plots are not much more complicated than other graphs. Embedded plots are a particularly useful data analysis tool when exploring spatio-temporal data and big data, which is subject to overplotting. Embedded plots also aid the exploration of interaction effects and second order relationships.

Subplots function the same way as geoms in the layered grammar of graphics. They provide a visual element whose appearance can be mapped to the values in an underlying data set. Because of this, embedded plots are easily accomodated by the layered grammar of graphics. This paper demonstrates how methods for embedded plots can be programmed into software built on the layered grammar of graphics, such as `ggplot2`.

Extending the grammar of graphics to account for embedded plots also reveals a conceptual insight about graphics. Graphics have a hierarchical, or recursive, structure where plots can be organized into higher level plots. This ability to organize individual data points by group according to group-level summaries is a potentially useful feature of graphics that has not been well developed in statistical graphics. However, it does parallel principles of infoVis that recommend "Overview first, filter, zoom for details" [Shneiderman, 1996].

As useful as embedded plots can be, we do not recommend embedded plots for every situation. Subplots increase the complexity of a visual. They make it easy to create overwhelming, cluttered and uninterprettable graphs. We recommend 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 graphics than with simpler graphics.
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 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]. But 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 judgement 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 overploting or too few dimensions. As the examples in Section 1 illustrate, embedded plots can be powerfully useful in many contexts.

References

E. Anderson. A semigraphical method for the analysis of complex problems. *Proceedings of the National Academy of Sciences of the United States of America*, 43(10):923, 1957.

A.D. Baddeley, G.J. Hitch, et al. Working memory. *The psychology of learning and motivation*, 8:47–89, 1974.

J. Bertin. *Semiology of Graphics*. University of Wisconsin Press, Madison, WI, 1983.

J.M. Chambers. *Graphical methods for data analysis*. 1983.

H. Chernoff. The use of faces to represent points in k-dimensional space graphically. *Journal of the American Statistical Association*, pages 361–368, 1973.

W.S. Cleveland. *Elements of graphing data*. Hobart Press, Summit, New Jersey, 1994.

W.S. Cleveland and I.J. Terpenning. Graphical methods for seasonal adjustment. *Journal of the American Statistical Association*, pages 52–62, 1982.

N. Cowan. The magical number 4 in short-term memory: A reconsideration of mental storage capacity. *Behavioral and brain sciences*, 24(01):87–114, 2000.

S. Dehaene. *The number sense: How the mind creates mathematics*. Oxford University Press, 1997.

S. Dehaene, E. Spelke, P. Pinel, R. Stanescu, and S. Tsivkin. Sources of mathematical thinking: Behavioral and brain-imaging evidence. *Science*, 284(5416):970–974, 1999.

A. Gribov, A. Unwin, and H. Hoffman. About glyphs and small multiples: Gauguin and the expo. *Statistical Computing and Graphics Newsletter*, 17:14–17, 2006.

J. Hobbs, H. Wickham, H. Hofmann, and D. Cook. Glaciers melt as mountains warm: A graphical case study. *Computational Statistics*, 25(4):569–586, 2010.

B. Kleiner and J.A. Hartigan. Representing points in many dimensions by trees and castles. *Journal of the American Statistical Association*, pages 260–269, 1981.

K. KotovskyJ R and HA Simon. Why are some problems hard? evidence from tower of hanoi. *Cognitive psychology*, 17(2):248–294, 1985.

G. Lakoff and R. Nunez. *Where mathematics comes from: How the embodied mind brings mathematics into being*. Basic Books, 2000.

M. Lanzenberger, S. Miksch, and M. Pohl. The stardinates-visualizing highly structured data. In *Information Visualization, 2003. IV 2003. Proceedings. Seventh International Conference on*, pages 47–52. IEEE, 2003.

S.J. Luck, E.K. Vogel, et al. The capacity of visual working memory for features and conjunctions. *Nature*, 390(6657):279–280, 1997.

R.E. Mayer. *Multimedia learning*. Cambridge Univ Press, New York, NY, 2nd edition, 2009.

G. A. Miller. The magical number seven, plus or minus two: some limits on our capacity for processing information. *Psychological review*, 63(2):81, 1956.

C.J. Minard. *Des Tableaux graphiques et des cartes figuratives, par M. Minard,...* impr. de Thunot, Paris, France, 1862.

R.S. Moyer and T.K. Landauer. Time required for judgements of numerical inequality. *Nature*, 1967.

U. Neisser. *Cognition and reality: Principles and implications of cognitive psychology.* WH Freeman/Times Books/Henry Holt & Co, 1976.

R.M. Pickett and G.G. Grinstein. Iconographic displays for visualizing multidimensional data. In *Proc. IEEE Conf. on Systems, Man and Cybernetics, IEEE Press, Piscataway, NJ*, volume 514, page 519, 1988.

R Development Core Team. R*: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2010. URL `http://www.R-project.org`.

D.E. Rumelhart. Schemata: The building blocks of cognition. In R. J. Spiro, B. C. Bruce, and W. F. Brewer, editors, *Theoretical issues in reading comprehension: Perspectives from cognitive psychology, linguistics, artificial intelligence, and education*. Lawrence Erlbaum, 1980.

D. Sarkar. *Lattice: multivariate data visualization with R*. Springer Verlag, 2008.

B. Schloerke, J. Crowley, D. Cook, H. Hofmann, and H. Wickham. Ggally: Extension to ggplot2. http://cran.r-project.org, 2011.

W. Schneider and R.M. Shiffrin. Controlled and automatic human information processing: I. detection, search, and attention. *Psychological review*, 84(1):1, 1977.

R.M. Shiffrin and W. Schneider. Controlled and automatic human information processing: Ii. perceptual learning, automatic attending and a general theory. *Psychological review*, 84(2):127, 1977.

B. Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *Visual Languages, 1996. Proceedings., IEEE Symposium on*, pages 336–343. IEEE, 1996.

R.A. Stanfield and R.A. Zwaan. The effect of implied orientation derived from verbal context on picture recognition. *Psychological science*, 12(2):153–156, 2001.

J. Sweller. Cognitive load during problem solving: Effects on learning. *Cognitive science*, 12 (2):257–285, 1988.

J. Sweller. Cognitive load theory, learning difficulty, and instructional design. *Learning and instruction*, 4(4):295–312, 1994.

J. Sweller. Evolution of human cognitive architecture. *Psychology of Learning and Motivation*, 43:215–266, 2003.

J. Sweller, P. Ayres, and S. Kalyuga. *Cognitive load theory*, volume 1. Springer Verlag, 2011.

V. Walsh. A theory of magnitude: common cortical metrics of time, space and quantity. *Trends in cognitive sciences*, 7(11):483–488, 2003.

H. Wickham. A layered grammar of graphics. *Journal of Computational and Graphical Statistics*, 19(1):3–28, 2010.

Hadley Wickham. `ggplot2`*: Elegant Graphics for Data Analysis*. Springer New York, 2009.

Hadley Wickham. The split-apply-combine strategy for data analysis. *Journal of Statistical Software*, 40(1):1–29, 2011.

Hadley Wickham, Heike Hofmann, Charlotte Wickham, and Diane Cook. Glyph-maps for visually exploring temporal patterns in climate data and models. *Environmetrics*, Submitted.

L. Wilkinson and G. Wills. *The grammar of graphics*. Springer Verlag, 2005.

R.A. Zwaan and R.H. Yaxley. Spatial iconicity affects semantic relatedness judgments. *Psychonomic Bulletin & Review*, 10(4):954–958, 2003.