Show Me: Automatic Presentation for Visual Analysis

Jock D. Mackinlay, Pat Hanrahan, and Chris Stolte

Abstract—This paper describes Show Me, an integrated set of user interface commands and defaults that incorporate automatic presentation into a commercial visual analysis system called Tableau. A key aspect of Tableau is VizQL, a language for specifying views, which is used by Show Me to extend automatic presentation to the generation of tables of views (commonly called small multiple displays). A key research issue for the commercial application of automatic presentation is the user experience, which must support the flow of visual analysis. User experience has not been the focus of previous research on automatic presentation. The Show Me user experience includes the automatic selection of mark types, a command to add a single field to a view, and a pair of commands to build views for multiple fields. Although the use of these defaults and commands is optional, user interface logs indicate that Show Me is used by commercial users.

Index Terms—Automatic presentation, visual analysis, graphic design, best practices, data visualization, small multiples.

1 Introduction

Visual analysis is a powerful method for analyzing data. A major stumbling block to widespread adoption is that most people are not trained in the graphical design principles needed to construct graphical presentations that support their reasoning process or communicate their analytical results to others. All analysts have knowledge of their problem domain, most have an understanding of the data that can help them with their problem, some have experience with data analysis tools, but few have the skills to design effective graphical presentations of information. Analysts would rather work on their tasks than become skilled at graphic design. The resulting status quo is that most people use spreadsheets such as Excel to analyze their data. They struggle with tables of numbers that are slow to read and do not show patterns or trends. When they want to present their data visually in reports or presentations, they also struggle with visual design issues. What people need are visual analysis systems that automatically present data using the best practices of graphic design.

The Holy Grail for research on automatic presentation is an artificial assistant that uses graphic design knowledge to create graphical encodings of data that present appropriate information effectively. However, the design of effective graphical presentations also requires knowledge of the task, which is almost always inaccessible inside the user's head. Users typically start visual analysis with vague tasks in mind, which are refined and transformed as they see graphical views of data. Unfortunately, inferring such tasks is a difficult open problem.

Deferring this open problem, we explore the use of automatic presentation in a visual analysis system where users search for graphical presentations that address their tasks, and the system supports their search with user interface commands and defaults that are based on automatic presentation functionality. The key research issue is the user experience, which has not been the focus of previous research on automatic presentation. Users will avoid user interface commands that they do not understand, and will change defaults that are poor choices. Consider new users. They typically have little experience with visual analysis. To get started, new users require automatic presentation functionality that is intuitive and predictable. Skilled users have related requirements. They require a system that attends to the graphic details so that they can stay in the flow of visual analysis.

In this paper, we describe Show Me, an integrated set of user interface commands and defaults that incorporate automatic presentation into a commercial visual analysis system called Tableau. A key aspect of Tableau is an algebraic specification language called VizQL, which describes both the structure of a view and the queries used to populate that structure with data [9]. The Show Me functionality takes advantage of VizQL to automatically present data as a table of views (commonly called small multiple displays).

The Show Me user experience has been designed to be used by both new and skilled users. The **Automatic Marks** default uses the row and column structure of a VizQL expression to default the mark type of a view. The **Add to Sheet** command uses the properties of fields and the VizQL specification of views to add a single field to a view using best practices of graphic design. Finally, the **Show Me** and **Show Me Alternatives** commands use VizQL to build views "from scratch" for a set of fields, including small multiple displays. User interface logs from commercial users indicate that they use these user interface commands and defaults. We also describe how the Show Me functionality supported additional Tableau improvements. We close with a discussion of research directions.

2 Related work

2.1 VizQL

The VizQL specification language was originally used to develop a research system called Polaris [9], which has been commercialized as Tableau. VizQL is based upon Bertin [1] and the algebra that was used in APT [7] with improvements to the algebra and the ability to compile into database queries. Wilkinson also uses an algebraic approach in *The Grammar of Graphics* to specify a wide range of view types [11]. In principle, the Show Me user experience could be developed with almost any of these methods for specifying graphs and charts. However, the key technical benefit of the VizQL algebra is that it clearly describes the row and column structure of small multiple views of data.

An informal feel for VizQL can be gleaned from the Tableau user interface shown in Fig. 1. Tableau users specify VizQL expressions by dragging field instances from the data window on the left to regions called *shelves* on the right. The data window organizes the fields into *dimensions*, which are typically categorical fields, and *measures*, which are typically quantitative fields. The interface uses blue and green to identify dimensions and measures. Measures are typically aggregated at some level of detail. The fields on the shelves collectively specify a VizQL expression. In particular, the

- *Jock D. Mackinlay is with Tableau Software*
  E-Mail: jmackinlay@tableausoftware.com
- *Pat Hanrahan is with Stanford University and Tableau Software*
  E-Mail: hanrahan@cs.stanford.edu
- *Chris Stolte is with Tableau Software*
  E-Mail: cstolte@tableausoftware.com

Manuscript received 31 March 2007; accepted 1 August 2007; posted online 27 October 2007.

basic structure of a view is specified by the order of fields on the Row and Column shelves, which appear above the view in Fig. 1. For example, the order of categorical fields on the Column shelves specifies the nesting of a small multiples table. The cells of the table are called *panes* because they can hold graphical views. The Show Me functionality focuses on the specific fields on the Row and Column shelves that specify the axes and headers of the table panes.

After the Row and Column shelves, the key shelves for Show Me are the ones that specify how data can be encoded with the graphical properties of marks. For example, in Fig. 1, the field Market is on the Color shelf. It specifies the assignment of color to markets. It also specifies that the measures should be aggregated at the market level of detail. The Level of Detail shelf also specifies aggregations even though it does not specify a graphical encoding. Collectively, we refer to these shelves as *Encoding Shelves*. Different mark types can handle different encodings. Fig. 1 shows the encodings for line marks. Point marks have a Shape shelf instead of the Path shelf.

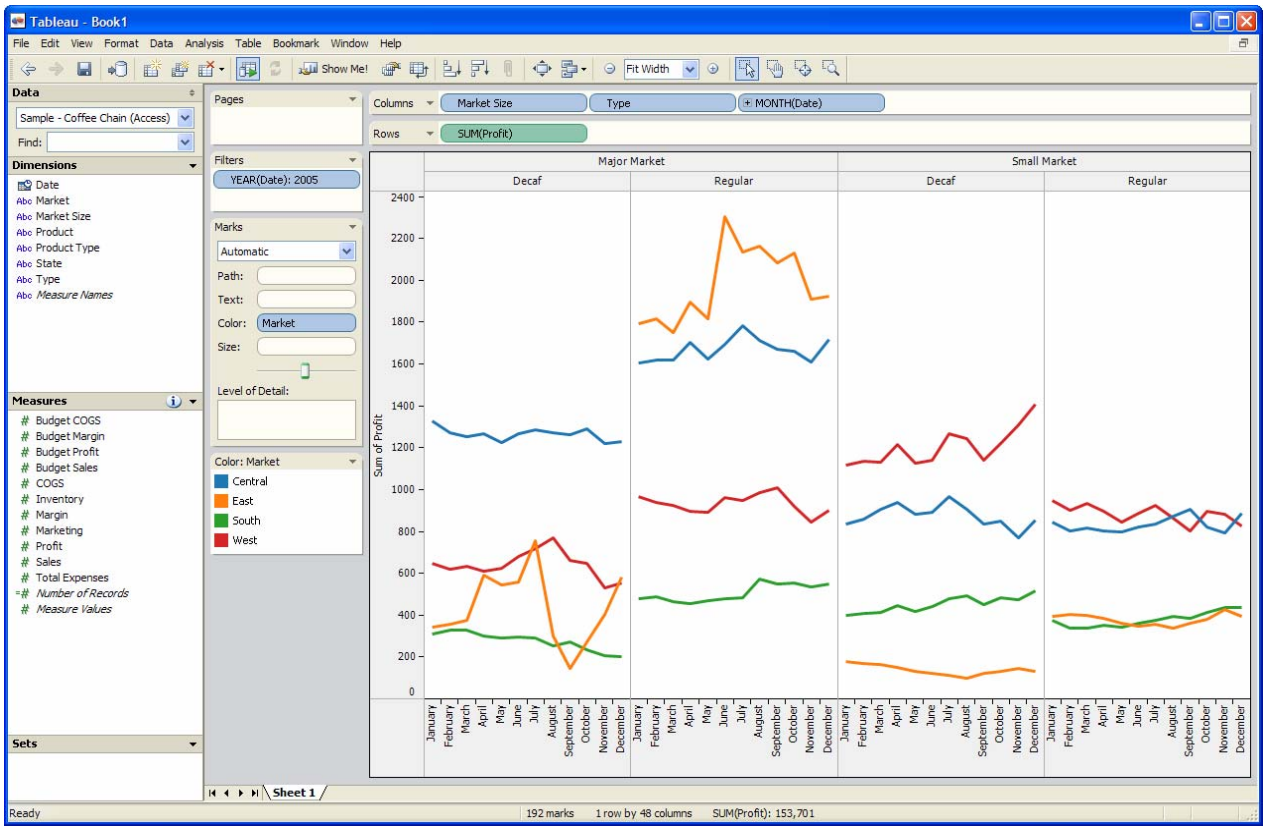


Fig. 1: The Tableau User Interface.

2.2 Automatic Presentation

Mackinlay's APT system is the foundational work on the automatic design of graphical presentations of relational information [7]. He codified Bertin's semiology of graphics as algebraic operators that were used to search for effective presentations of information [1]. Casner extended this work by comparing design alternatives via a measure of the work required to read presentations depending on the task [2]. Roth et. al. added additional types of presentations and made the resulting presentations interactive [5] [8].

APT's architecture focused on how to communicate graphically rather than what to say. Feiner's thesis from the same period as APT is the foundational work on extending computational linguistics into the area of graphical presentation [3]. His work has been extended by Zhou [12]. Computational linguistics research can decide what to say by modelling the communication with the user, which is important for developing an effective artificial assistant.

Visual data mining uses statistical algorithms to decide what to show [6]. However, effective data mining often requires considerable expertise in statistics, which is beyond the skill set of most visual analysis users.

This paper makes two key contributions to the research on automatic presentation: the generation of small multiple views and the design of a user experience for automatic presentation functionality.

Small multiple views are quite useful for presenting data that has many dimensions and measures. However, previous work on automatic presentation focused primarily on single views of data or a sequence of views. The work described here includes novel heuristics that support the automatic presentation of small multiple views.

By deferring the artificial assistant goal implicit in the research on computational linguistics and visual data mining, the work described here explores new ground in the user experience for automatic presentation, particularly in supporting the flow of visual analysis. The closest work is probably that of Roth et. al. [5][8]. However, their publications do not discuss the user experience aspects of their automatic presentation work.

2.3 Chart Type Menus

A common user interface technique for helping users present their data graphically is a menu of chart types. For example, Excel supports the following workflow:

1. Select the data to be presented
2. Invoke a menu of chart types
3. Select a chart type

Since there are many different ways to organize charts, a menu of chart types can grow large. For example, the Excel 2003 chart menu has over 70 individual items. Selecting items from a long list can interrupt the flow of visual analysis. Furthermore, this user experience focuses on the chart type. Users typically know their task. They often know their data. However, they may not know the chart type that will address their task.

This paper addresses these issues by using automatic presentation techniques to focus on the selected data rather than the list of chart types. In particular, the **Show Me Alternatives** menu described below contains only 14 items, which automatically build charts that have appropriate organization for the selected data. Furthermore, items are only active when they can build appropriate charts, which supports the flow of visual analysis.

2.4 Graphic Design Knowledge

We recommend three sources for knowledge of graphic design for effective presentation of data. Bertin's *Semiology of Graphics* is an essential resource for the systematic design of graphical presentations [1]. Tufte has many excellent books that teach graphic design with a wide range of examples, including his latest book *Beautiful Evidence* [10]. Finally, Few's book *Show Me the Numbers* is specifically targeted to business users [4].

3 Automatic Marks

Defaults are an excellent way to incorporate automatic presentation into a visual analysis system because they are easily used by both new and skilled users. For example, Tableau includes an **Automatic Marks** default mode that selects a mark type given the data fields used to specify a pane. For example, Fig. 1 shows Tableau automatically selecting a line mark for a pane because it is based on a temporal field. Line charts are best practice for temporal data. **Automatic Marks** helps new users follow this convention. Skilled users have the option of changing the mark type with a dropdown menu when that is appropriate.

Automatic Marks rules are based on the properties of the data fields that specify the axes and headers of the table panes. Tableau fields currently support three data properties:

Data type: text, date, date&time, numeric, or boolean
Data role: measure or dimension
Data interpretation: discrete or continuous

Automatic Marks takes advantage of the Tableau data model, which includes the following classifications:

C = Categorical (discrete and dimension)
    Cdate = Categorical date (date or date&time)
Q = Quantitative (continuous)
    Qd = Quantitative dependent (measure)
    Qi = Quantitative independent or Qdate (dimension)

Of particular interest is the classification of Q fields as independent or dependent. A dependent variable is intended to be a function of an independent variable. By convention in datacubes used in data warehouses, independent variables are called dimensions and dependent variables are called measures. Measures. e.g. Sum(Sales), are typically aggregated before display (and thus calculated) and are hence dependent on the choice of fields used as a dimension. This is an important and novel aspect of Tableau's data model.

Automatic Marks uses the rightmost fields in the Row and Column shelves to determine the pane types and mark types as shown in Table 1, which omits specific cases that have the same mapping as more general cases.

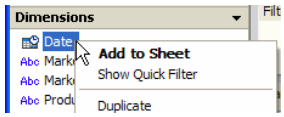
These rules result in view types that would typically be chosen for these types of fields by someone who is trained in the best ways of producing charts and graphs. **Automatic Marks** can also evolve as Tableau is extended to have richer data properties. For example, Tableau supports a polygon mark type that can be added to the **Automatic Marks** by adding data types that represent space, such as longitude and latitude.

Table 1: Automatic marks rules

Pane Type		Mark Type	View Type
Field	Field		
C	C	Text	Cross-tab
Qd	C	Bar	Bar view
Qd	Cdate	Line	Line view
Qd	Qd	Shape	Scatter plot
Qi	C	Gantt	Gantt view
Qi	Qd	Line	Line view
Qi	Qi	Shape	Scatter plot

4 Adding a Single Field to a View

This section describes **Add to Sheet**, the Show Me command that adds a single field to a view. **Add to Sheet** is invoked from the data window via a context menu:



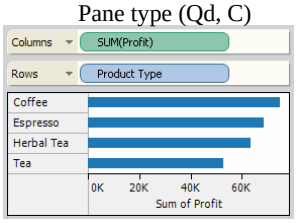
We placed this command at the top of the menu so that it will be easily found by new users. New users often explore context menus to learn an application. The bold font is a Microsoft convention indicating that the command can be invoked with double-clicks, which supports rapid view creation by skilled users.

Add to Sheet is based on various heuristics, as described in the remainder of this section.

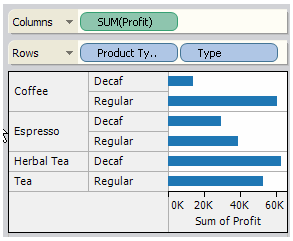
4.1 Affinity

Tableau and VizQL support the generation of small multiple displays, which can be very effective for data sets that have many fields. The affinity heuristic supports the generation of effective small multiple displays by adding fields next to related fields. For example, the following images describe **Add to Sheet** behavior starting from a bar view (Qd, C) and adding either a C field or Qd field:

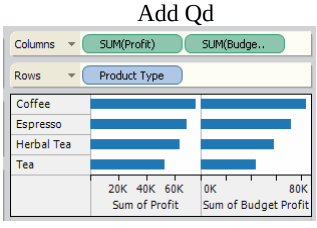
Pane type (Qd, C)



Add C



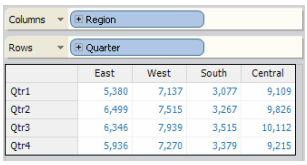
Add Qd

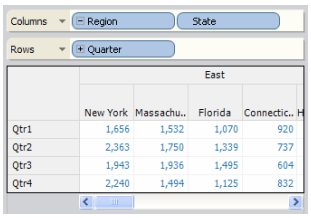


The added field has an affinity for a field of the same type. Categorical fields are added on the right to create a "break-down"

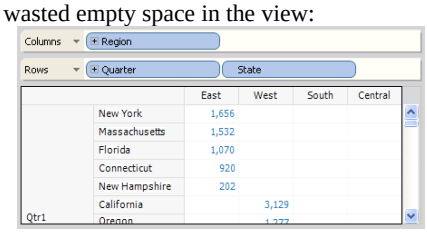
experience. When the Type field is added to the bar chart, it breaks the SUM(Coffee) into Decaf and Regular.

The affinity heuristic also applies when adding a hierarchical categorical (C) to small multiple views that have categorical fields on both row and column (C, C). For example, a data warehouse may support a hierarchical dimension that combines states into market regions. Starting with the cross-tab of Regions by Quarter, the affinity heuristic adds State next to Region rather than next to Quarter:





Adding State next to Quarter would have created a breakdown that is much less effective because each State belongs to a specific Region, thus creating wasted empty space in the view:



4.2 Encoding Requirements

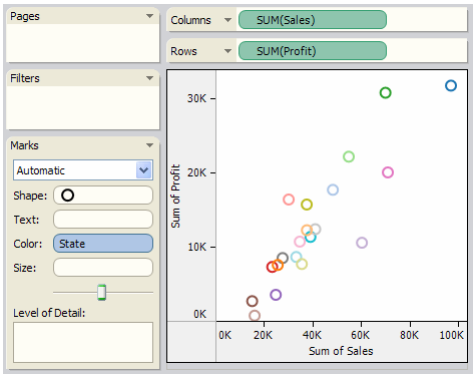
Encodings such as Color, Size, and Shape are also targets for the **Add to Sheet** command. However, using these encodings effectively is difficult. **Add to Sheet** uses best practices when it adds fields. For example, Tableau has a palette of 20 distinguishable colors for categorical fields. It also has a set of 10 shapes that are designed to overlap effectively. Which shelf is used depends on the number of categories in the data. For example, a categorical field with 15 members can be placed on the Color shelf but not the Shape shelf. Other complications include the priorities of the encodings (whether to use color or shape if they are both possible), which encodings are preferred for quantitative vs. categorical variables, and the number of simultaneous encodings to use.

4.3 Multiple Fields

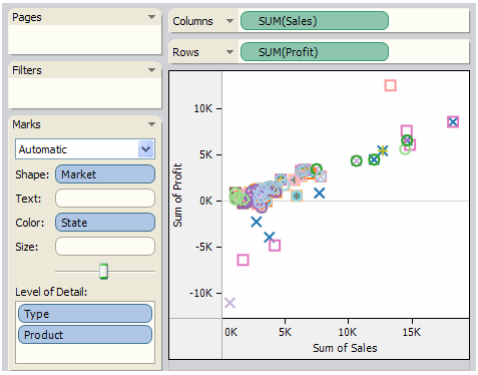
Users will not use automatic presentation commands like **Add to Sheet** unless they work effectively most of the time. In particular, **Add to Sheet** must work for small multiple displays that have many fields on the shelves. The affinity heuristic meets this multiple field requirement for (C, Q) pane types such as bar and line views because the Row and Column shelves can accept multiple fields and still produce effective presentations. However, cross-tabs (C, C) and scatter plots (Q, Q) require additional heuristics to handle multiple fields, particularly when a Q field is being added. The multiple field heuristics differ for categorical and quantitative fields.

4.3.1 Categorical: Level of detail

The heuristic for adding categorical fields to views that have multiple fields is to consider the following shelves in order: Shape, Color, and Level of Detail. Although Shape and Color only accept single fields, Level of Detail accepts multiple fields. This heuristic is particularly effective for scatter plots when combined with the encoding requirements. For example, one of Tableau's sample datasets has a State field with 20 states. Since Tableau only supports a shape encoding with 10 shapes it cannot go on the Shape shelf. However, State can go on the Color shelf because Tableau supports 20 distinct colors for categorical fields:



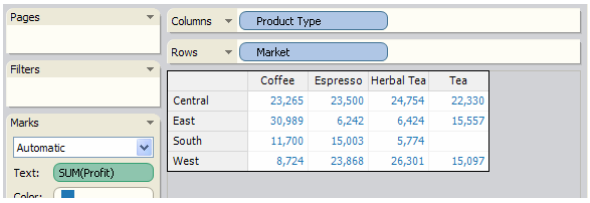
The data set also contains a Market field with 4 members. **Add to Sheet** adds Market to the Shape shelf:

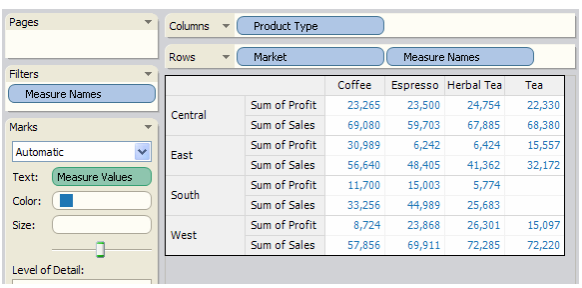


Any additional fields will be added to the Level of Detail shelf.

4.3.2 Quantitative: Measure Values

The Level of Detail shelf is not a solution for multiple quantitative fields because the quantitative fields on this shelf do not appear in the view. (They only show up in tooltip text.) In this case, we use special fields in Tableau called Measure Names and Measure Values, which are similar to Wilkinson's "blend" operator in that they combine measures together [11]. This heuristic is particularly effective for cross-tab views where the color and size of text is a poor choice for encoding quantitative fields. For example, adding SUM(Sales) to a cross-tab that breaks down SUM(Profit) by Product Type and Market, combines the measures on the text shelf:





4.4 Starting from an Empty View

The **Add to Sheet** heuristics described above handle adding multiple fields in many arrangements including small multiple displays, allowing **Add to Sheet** to generate views that are typically well-designed. The final issue is how the skilled user can start from an empty sheet and use **Add to Sheet** to generate a broad range of different types of views. The solution is to use the order of addition of the first view fields onto the sheet to specify a wide range of view types:

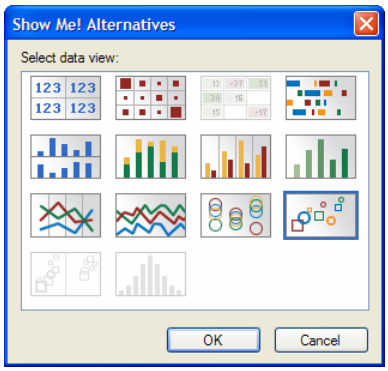
Add Order	View Type
Categorical	Cross-tab (C, C)
Quantitative, Date	Line view (Q, C)
Quantitative, Categorical	Bar view (Q, C)
Quantitative, Quantitative	Scatter view (Q, C)

The next section describes automatic presentation commands that can be used to generate additional view types.

5 Building Views From Scratch For Multiple Fields

In contrast to **Add to Sheet**, the automatic presentation commands described in this section accept multiple fields as input, which greatly increases the space of possible designs, particularly when the views are built from scratch. The user experience was carefully designed to go on the Tableau toolbar to help new users when they begin with an empty sheet. Although the design space is complex, the user interface had to be simple. The commands are called **Show Me Alternatives** and **Show Me**, which are respectively a dialog of commands that automatically build views from scratch and a button that is a shortcut to the default choice for the dialog.

The **Show Me Alternatives** dialog is a stable grid of choices with tooltips that describe the conditions for a choice to be active. The icons have saturated colors when the command is active:



The stable grid allows users who have learned the grid to invoke commands by position, which supports the flow of visual analysis.

5.1 The Show Me Default Ranking

Each **Show Me Alternatives** command builds a particular type of view. Each command has an associated condition and some of the commands have a ranking. The **Show Me Alternatives** dialog defaults to the highest ranked command whose condition is met (1 is the lowest rank). The **Show Me** command is a shortcut to this default. The rankings have been crafted to default **Show Me** to designs that embody best practices [1][4][10] and are familiar to most people. In general, the ranking prefers views that encode data graphically, which is why text tables have the lowest rank. Higher ranked commands typically have more specialized conditions that must be met for that particular view type to be effective. This section describes the commands that are ranked. The remaining commands are described in the next section.

Some of the ranked commands have a condition for the rank in addition to a condition for the command. The command condition determines whether the user can explicitly select the command. The rank condition is only used for the default computation. The command does not have a rank when the rank condition is not satisfied.

Text Tables: at least 1 field, rank 1

Text tables have the lowest rank because their primary utility is to look up specific values. The higher ranked commands present views that encode data graphically, which support other tasks such as comparison. Although text tables have a low rank, their condition is easily met. The text table command can handle a large number of fields and will always be available as a default for **Show Me Alternatives.** Heat maps are a related command that is not ranked.

Aligned Bars: at least 1 Q, rank 2

Bars are effective for comparing values because the human visual system is good at comparing bar lengths, particularly when they are aligned. Aligned Bars are a common default when the input includes a quantitative field unless the input also includes a date field or two quantitative fields. However, aligned bars can involve a lot of scrolling when multiple categorical fields are shown. The next command handles this case.

Stacked Bars: at least 2 C, at least 1 Q, rank 3 with at least 3 C

Stacked bars require less scrolling than aligned bars when there are multiple categorical fields in the view. There are two additional bar commands that are not ranked.

Discrete Lines: at least 1 Cdate, at least 1 Q, rank 4

A line view is a better default than a bar view when the input includes a date field because it is more effective for showing trends. This command treats the date field discretely. There is an unranked line command that treats the date field continuously. The primary difference is how the axes are drawn. We judged that the discrete version was more appropriate for the typical Tableau user.

Scatter Plots: between 2 and 4 Q, rank 5 for 2 Q

Scatter plots are very effective for comparing two values, particularly with categorical fields on the shape shelf. However, they grow less effective when the input has more than two quantitative fields. Lower ranked commands will be used as the default when the input has more than four quantitative fields.

Gantt Charts: at least 1 C, at least 1 Qi, 1 to 2 Q, rank 6

Although specialized, Gantts are effective for showing duration with respect to a quantitative independent. In Tableau, Gantts are also an effective way to show the distribution of a measure. Although highly ranked, the conditions for this command are rarely satisfied.

5.2 Additional Alternatives

The **Show Me Alternatives** dialog includes commands that do not have a ranking, which means they will never be the default. Skilled users can use these commands to build views in one step. When new users invoke a **Show Me Alternatives** command, they see how the fields are placed on shelves to specify that type of view, which can help them learn how to use Tableau.

Heat Map: at least 1 C, between 1 to 2 Q, no rank

Heat maps are effective for large tables that do not have room for text marks. However, they can only handle two quantitative fields. We could have assigned a rank to this command, but it would have required an expensive query to the database to identify when the data would generate a large table. See the implementation section below for a discussion of this performance issue.

Highlight Table: at least 1 C, at least 1 Q, no rank

Highlight tables are a text table with a color encoding that fills the table cell. They don't have a rank because the fill can make it hard to read the text.

Side-by-side Bars: at least 1 C, at least 1 Q, no rank

Side-by-side bars are useful for comparing similar measures. Aligned bars are the better default because aligned bars are easier to compare visually.

Measure Bars: at least 1 C, at least 2 Q, no rank

Measure bars put a quantitative field on color. Aligned bars are the better default because they are more common and easier to interpret.

Continuous Lines: at least 1 Qi, at least 1 Q, no rank

Line views are very useful for trending questions. This version treats the date field as continuous. The discrete case is a better default for the typical Tableau user.

Circle Charts: at least 1 C, at least 1 Q, no rank

Similar to a bar view except circle marks are plotted. Unlike bar marks, circle marks do not suggest that the axis has an origin. They are useful for comparing the relative positions of marks. They don't have a rank because they are more specialized than bar views.

Scatter Matrix: between 3 and 6 Q, no rank

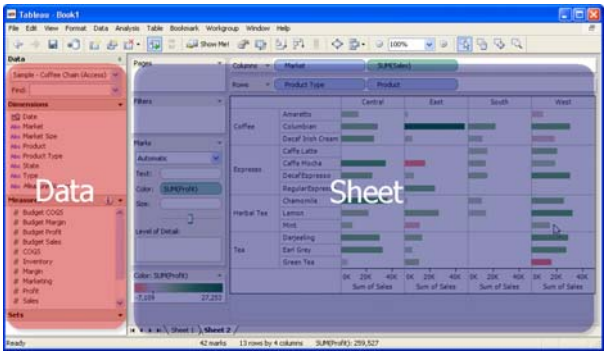
Scatter matrix is a powerful technique used by the statistics community. It is too specialized to be ranked for the typical commercial user.

Histogram: single 1 Q that supports a binning calculation, no rank

Histograms are another powerful technique used by the statistics community. It is too specialized to be ranked for the typical commercial user.

5.3 The Show Me User Interface Flow

A major design goal for **Show Me** and **Show Me Alternatives** was to develop a user interface that new users could understand and skilled users could fit into their workflow, which is why we worked hard to codify best practices of graphic design into a small set of alternatives and to include the default ranking in the user experience. However, our design goal was challenged by the fact that Tableau has two places where the input to commands is naturally specified: 1) the list of fields associated with a data source window and 2) the list of fields that are currently on the sheet:



Two sets of commands for these two inputs would have been confusing to both new and skilled users.

Our solution was to have one set of commands that prefer the input from the data window. The key innovation was to clear the selection from the data window when the input is processed. The result is a natural user interface flow from the data to the view. For example, a new user can use **Show Me** defaults to build a view for a set of fields selected in the data window and then use **Show Me Alternatives** to look at other views of the same data.

6 Implementation

Show Me Alternatives is implemented on top of **Automatic Marks** and **Add to Sheet**, including their codification of graphic design best practices. **Show Me Alternatives**, however, can embody additional best practices because it deals with multiple fields at the same time. For example, **Automatic Marks** and **Add to Sheet** will always build a line chart when the view includes a date field. However, line charts suggest that the data is changing continuously. When the data does not vary continuously, the user can select a bar choice from **Show Me Alternatives** to automatically build a view that shows the discrete nature of the data. As the Tableau type system expands, we will be able to add such views to the Show Me defaults.

Show Me Alternatives can also set formatting parameters when it builds views. For example, the Heatmap alternative sets the cross-tab to square cells and uses square marks that almost fill the cells. The result maximizes the variation in size encodings and creates a regular grid that is easy to read. Skilled users appreciate the automation of formatting best practices.

Finally, performance is a key implementation issue when automatic presentation is added to a visual analysis system that works with databases. Tableau has users with very large databases who are willing to wait minutes for database queries to run so that they can see a graphical view of their valuable data. However, users do not want interactive experiences that include such pauses.

Our solution to the performance issue is to only query the database when the user asks the automatic presentation system to build views. For example, **Automatic Marks** can query the database because the user is asking it to add a field to a view. **Show Me Alternatives**, however, should not query the database when it is calculating the **Show Me** default because that calculation is used to enable the **Show Me** toolbar button in Tableau's basic user interface loop. On the other hand, queries can be tolerated after a user chooses a **Show Me Alternative** because they expect a view to be built and will be willing to wait for the query to complete. **Show Me Alternatives** also takes advantage of a result cache in Tableau to reduce the number of queries that it generates.

7 Evaluation

7.1 Informal Feedback

The Show Me functionality has stood the test of time. It was introduced into Tableau v1.5 in October 2005. During the last year and a half, we have gotten generally positive feedback about Show Me from Tableau employees, new users, and skilled users. Tableau has thousands of users who generate a steady stream of email about Tableau, particularly when they have something negative to report. Because the feedback on Show Me has been primarily positive, the functionality has not changed significantly since it was introduced, even though many new features have been added to the application (we are now at Tableau v3.0). In particular, **Show Me** and **Show Me Alternatives** have remained on the toolbar, a scarce resource, despite competition from new features.

Tableau employees report two uses of Show Me: testing and training new users.

For testing, they use Show Me to build views during random testing where speed and variation is more important than precise control of the view. The **Add to Sheet** double-clicks, in particular, can be used to build views quickly. However, during visual analysis when control is more important, these skilled users find Show Me to be less useful because it is hard to predict what view will be built by the automatic presentation functionality. Predictability is likely to be a general problem for the usability of automatic presentation functionality.

For training, Tableau employees use **Show Me** and **Show Me Alternatives** to show new users the range of views that can be specified in Tableau. However, showing the result does not teach users the steps to build views. Training quickly moves on to the more direct ways to build views in Tableau.

7.2 User Interface Logs

Tableau is working to develop formal measures of usability to augment and validate the informal user feedback that we already collect. We are particularly interested in real visual analysis activity rather than activity involving sample tasks or feature testing. However, the Tableau user community is typically working with operational data when they do real visual analysis with Tableau, which makes them protective about their user interface activity. Nevertheless, such data could be very useful because it describes real visual analysis activity in a commercial setting.

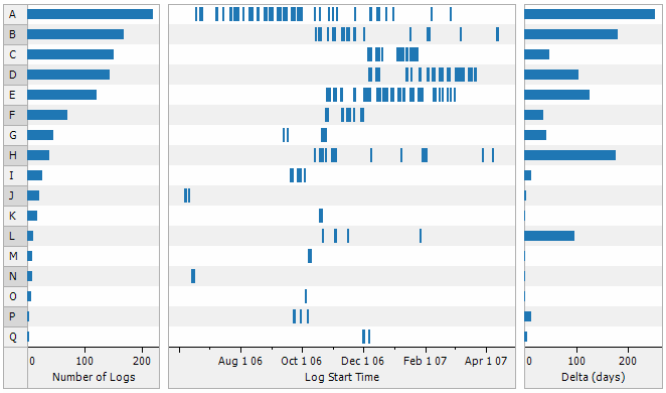
As a first step to developing such metrics, a user interface logging mechanism was added to Tableau v2.0. This mechanism collects logs about Tableau user interface activity and stores them on the user's machine. The goal is to develop trusted relationships with users and ask them to send us these logs. Given this goal, the UI logs have been carefully designed to be anonymous so that users will be willing to send them to us. In fact, they are much more anonymous than the data that is typically used in a research paper. For example, we can't even tell how many users have contributed logs because the mechanism for generating anonymous user ids can change the id generated for a specific person as the setup on a machine changes.

To date, Tableau UI logs have been solicited from two groups: v3.0 beta testers and 17 customers who were judged to be very active Tableau users by their sales contacts. Both groups are generally skilled Tableau users. We estimate that we have received logs from approximately 10 users with 17 user ids. The typical Tableau user is a professional adult working with corporate data. The ~10 users are probably fairly typical except for their willingness to share their UI logs.

Although the number of users is modest, the amount of data collected is considerable:

Number of user ids	17
Number of log files	1062
Date range	5/30/2006 to 4/28/2007

The following diagram describes these 1062 log files organized by user id:



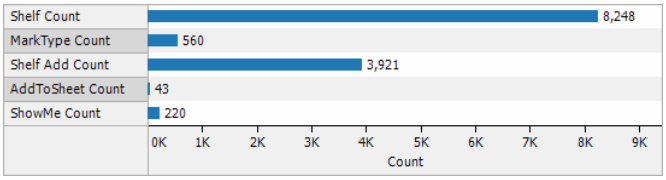
The diagram is sorted by the number of logs for each user id. The left section shows the number of logs with id A at top having 218 logs. The middle section plots the logs against a time axis. The right section describes the delta in days between the first and last log file for each user id. The data for the user ids at the top of this diagram are probably from skilled users given the number and delta of the logs for each user id. The user ids at the bottom of the diagram may or may not be data from skilled users. They might be from new users who tried Tableau on the machine of a skilled user or logs from skilled users who had their user id change during this period.

An interesting finding from this diagram is the pattern of visual analysis activity, which is uneven. Visual analysis appears to be episodic, with clear gaps in the middle history section. Furthermore, episodes of visual analysis can be intense. For example, C has a relatively large number of log files in a small delta of days.

7.3 Show Me Activity for Skilled Users

Although the user interface log data described in the last section is limited in what it can tell us about the visual analysis activity of users, it can shed light on a key question about the Show Me functionality described in this paper: Is Show Me used by skilled users? After all, the lack of negative feedback discussed above could be caused by the lack of Show Me use.

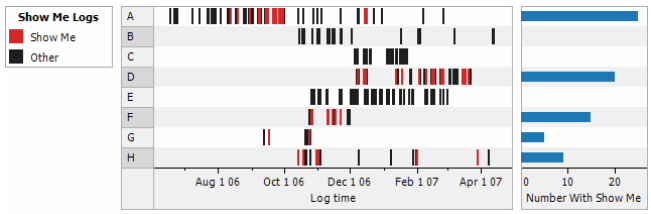
We can address this question by counting the log entries associated with the Show Me functionality and comparing the counts to some standard. A reasonable standard to use for the **Automatic Marks** default is the count of shelf activity, because Tableau builds a new view every time the shelf contents changes, which represents an opportunity for users to set the mark type explicitly when they don't like the automatic default. On the other hand, a reasonable standard for the commands **Add to Sheet**, **Show Me**, and **Show Me Alternatives** is the count of the direct addition of fields to Shelves because these commands also add fields to shelves. These counts are shown in the following diagram:



One weakness with this diagram is that the log files do not differentiate between **Show Me** and **Show Me Alternatives**. These commands are implemented with the same code and the log entry is generated when the command is successfully executed.

We draw three conclusions from this diagram. First, the **Automatic Marks** default is generally correct with 8,248 shelf changes but only 560 mark changes for a 6.8% error rate. Second, the **Add to Sheet** command is not being used by skilled users (43 automatic adds for 3,921 direct adds 1.1%). This finding supports the informal feedback from Tableau employees about their behavior during testing and visual analysis. Third, **Show Me** and **Show Me Alternatives** are being used modestly by skilled users (220 commands for 3,921 shelf adds 5.6%). Since these commands are typically used to add multiple fields to the shelves, their actual impact on the visual analysis may be higher than this percentage would suggest.

There is additional data that supports the third conclusion. The following diagram colors the Show Me logs for the history of log file creation for the user ids with a large number of log files:



Users ids B, C, and E did not use the **Show Me** and **Show Me Alternatives** commands. The reason for this is impossible to determine given this log data. However, the other user ids did use this command, which means they used these commands more

frequently that the percentage calculated above. Furthermore, the use was spread out over time, which suggests that the Show Me commands are a regular part of the visual analysis activity of some skilled users.

8 Using Show Me to Develop New Features

Show Me was developed to help users build effective views. After the successful introduction of Show Me, we discovered that we could use Show Me to develop two new Tableau features: 1) Table Calculations and 2) sending data from another application to Tableau.

Table Calculations allow users to specify running totals and other common analysis calculations. A Table Calculation is done locally rather than in the database. Since the calculation is done locally, the fields must be included in the view specification. We used the Show Me functionality to smooth the user experience when specifying Table Calculations by automatically adding fields to the view that were used in the specification of the calculation.

The second feature developed using Show Me involved Hyperion, a company that sells a popular data warehouse product. Tableau communicates with the Hyperion data warehouse, which includes an Excel-based user interface. The original connection between Tableau and this user interface involved sending data from Tableau to Excel, which is relatively easy since the data ends up as a table of values. However, sending data from Excel to Tableau involves many more design choices. We used the **Show Me Alternatives** to smooth the user experience by offering a choice of view type.

9 Conclusions and Future Work

The Show Me functionality described in this paper embodies best practices of graphic design so that users can concentrate on their visual analysis. Good defaulting creates a smooth user experience where users can concentrate on their task rather than the user interface to set the properties of their view. The Show Me functionality has been well received by Tableau's large user community.

Show Me can be extended by expanding the type system. For example, Tableau supports thematic map visualizations with marks over background map images. **Show Me Alternatives** could be extended to include maps by extending the type system to include longitude and latitude. The object-oriented architecture of **Show Me Alternatives** supports the addition of such rules.

Another direction to take Show Me is to incorporate best practices for transforming one view type into another. The **Show Me Alternatives** commands described in this paper build views "from scratch," which means that they will build the same view for the same set of fields without considering the current placement of fields on shelves. Building the same view creates a predictable user experience that makes the commands easier to understand. However, the automatic presentation system cannot know the placement of fields that is most appropriate for a given task. Given an existing view, a "transforming" command would move fields to generate the requested view type with minimal disruption to the existing field placement. This would be an expert version of Show Me.

In conclusion, this paper describes automatic presentation functionality that has been incorporated into a commercial visual analysis system. The key research issues were the automatic presentation of small multiple views, which are effective for multi-dimensional data, and the user experience for automatic presentation. Feedback clearly indicates that the Show Me user experience does not disrupt the flow of visual analysis. In fact, it might enhance it. However, the Show Me commands are used much less frequently by skilled users than more direct ways to build views of data. Our experience suggests that before you add automatic presentation, you should focus on making the user experience effective.

Acknowledgements

We acknowledge the great team at Tableau Software and thank Polle Zellweger and the anonymous reviewers for their helpful suggestions.

References

[1] J. Bertin, *Semiology of Graphics*. The University of Wisconsin Press. 1983.

[2] S. Casner. A task-analytic approach to the automated design of graphic presentations. *ACM Trans. on Graphics.* 10(2):111-151. Apr. 1991.

[3] S. Feiner. APEX: An experiment in the automated creation of pictorial explanations, *IEEE Computer Graphics and Applications* 5(11):29-37. Nov. 1985.

[4] S. Few. *Show Me the Numbers*. Analytics Press, Oakland, CA. 2004.

[5] J. Goldstein, S. F. Roth, J. Kolojejchick, and J. Mattis. A framework for knowledge-based interactive data exploration. *Journal of Visual Languages and Computing*, 5(4):339-363. Dec. 1994.

[6] D. Keim. Information visualization and visual data mining. *IEEE Trans. on Visualization and Computer Graphics* 8(1):1-8. Jan. 2002.

[7] J. Mackinlay. Automating the design of graphical presentations of relational information. *ACM Trans. on Graphics*, 5(2):110-141. Apr. 1986.

[8] S. Roth, J. Kolojejchick, J. Mattis, and J. Goldstein. Interactive graphic design using automatic presentation knowledge. In *Proceedings of the SIGCHI* (Boston, 1994), 112-117.

[9] C. Stolte, D. Tang, P. Hanrahan. Polaris: A system for query, analysis, and visualization of multidimensional relational databases. *IEEE Trans. on Visualization and Computer Graphics*, 8(1):52-65. Jan.-Mar. 2002.

[10] E. Tufte. *Beautiful Evidence*. Graphics Press LLC, Cheshire, Connecticut. 2006.

[11] L. Wilkinson. *The Grammar of Graphics*. Springer. 2005.

[12] M. X. Zhou. *Automated Generation of Visual Discourse*. Ph. D Thesis, Columbia University, S. Feiner advisor. 1999.