DataGraph

3.2

A SIMPLE AND POWERFUL GRAPHING PROGRAM

David Adalsteinsson
Table of Contents

Introduction ................................................................................................................................. 1
Structure......................................................................................................................................... 2
    Data table .................................................................................................................................. 2
    Layout & Drawing Commands ................................................................................................. 3
    Graph .......................................................................................................................................... 4
    Global Variables ...................................................................................................................... 4
    Navigating the UI ..................................................................................................................... 5
Data Table ...................................................................................................................................... 7
    Organizing your data ................................................................................................................. 7
    Column Types ............................................................................................................................ 8
    Editing table entries ................................................................................................................. 11
Importing/Pasting Data ............................................................................................................... 14
    Interpreting data ...................................................................................................................... 14
    Paste Special ............................................................................................................................ 15
    Import Special .......................................................................................................................... 15
    File formats ................................................................................................................................ 18
Layout ........................................................................................................................................... 19
    Style .......................................................................................................................................... 19
    Canvas ........................................................................................................................................ 20
    Axis ............................................................................................................................................. 20
Drawing Command Elements .................................................................................................... 24
    Selecting Data Columns ........................................................................................................... 24
    Color .......................................................................................................................................... 25
    Font Style .................................................................................................................................. 25
    Mask .......................................................................................................................................... 26
    Pop up slider .............................................................................................................................. 27
    Text labels ................................................................................................................................. 27
    Tokens ......................................................................................................................................... 28
    Displaying numbers .................................................................................................................. 29
    Fill Style ..................................................................................................................................... 29
    Axis selector ............................................................................................................................... 29
    Exclude/Hide ............................................................................................................................. 30
Drawing Commands ................................................................................................................... 31
    Plot ............................................................................................................................................ 31
    Points ......................................................................................................................................... 32
    Bars .......................................................................................................................................... 33
    Pivot ........................................................................................................................................... 34
    Fit ................................................................................................................................................ 35
    Histogram ................................................................................................................................. 37
    Box .............................................................................................................................................. 38
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Range</td>
<td>38</td>
</tr>
<tr>
<td>Label</td>
<td>39</td>
</tr>
<tr>
<td>Text</td>
<td>40</td>
</tr>
<tr>
<td>Graphic</td>
<td>40</td>
</tr>
<tr>
<td>Legend</td>
<td>41</td>
</tr>
<tr>
<td>Magnify</td>
<td>42</td>
</tr>
<tr>
<td>Function</td>
<td>43</td>
</tr>
<tr>
<td>Graphs</td>
<td>44</td>
</tr>
<tr>
<td>Combining graphs</td>
<td>44</td>
</tr>
<tr>
<td>Export</td>
<td>47</td>
</tr>
<tr>
<td>Copy</td>
<td>47</td>
</tr>
<tr>
<td>Export to a file</td>
<td>47</td>
</tr>
<tr>
<td>Exporting multiple graphs</td>
<td>47</td>
</tr>
<tr>
<td>Movies</td>
<td>47</td>
</tr>
<tr>
<td>Command Line Utility</td>
<td>49</td>
</tr>
</tbody>
</table>
Introduction

DataGraph is a feature rich drawing environment for numerical data and analytic functions. DataGraph can also do a variety of data analysis such as fitting (regression) histograms and pivoting. It can be used for anyone from a middle school student learning about graphs and trends to a researcher analyzing millions of data points and creating graphs for publications.

DataGraph combines data and graphs into one file. The data is stored in a table as columns, and each column can have any number of rows. You can view this as a spreadsheet, but also as a database with each row being a record.

DataGraph’s focus is a two dimensional representation of data. In addition, you can compute statistics and fit the data using analytic expressions. DataGraph allows you to combine plots, scatter plots, fits, analytic functions, bar graphs, stock charts and box plots in the same graph. You can attach labels, lines, add graphical elements, highlight regions and magnify regions to point out details. All of this is done interactively and updates automatically as you change data or settings.

DataGraph uses default rules to create a consistent, clean and professional looking graph. You can adjust every aspect of the graph and overwrite the default rules as needed. Using color schemes and font styles, DataGraph makes it easy to create consistent graphs and change the look and feel in very few clicks.

DataGraph is not just a drawing program. It contains a number of exploratory data analysis functions. You can map columns by using analytic expressions to combine columns, compute maxima, accumulate rows etc. You can compute statistics and fit data using linear and non-linear functions. You can compute histograms and box plots to drill into your data set and explore patterns and trends. You can change variables using sliders.

What DataGraph does not do

DataGraph focuses on accurate and clear presentation of data. The goal is to create images for presentations, publications and also to just understand your data better.

DataGraph does not have pie charts, and there is a reason for that. Typically a bar graph does a better job to visualize categorical data. Pie charts make sense in a few cases, typically in presentations, and a dedicated tool like KeyNote is better here. Another “missing feature” is 3D graphs. This does not fit into the two dimensional focus of DataGraph, and a tool like DataTank would be better in this case. Faux 3D, such as extruded bars or shadows, is inconsistent with the focus on exact data representation and graph cleanliness. Again, use KeyNote when this need arises.

DataGraph has a very active user community that doesn’t hesitate to suggest new features and point out things that they feel should be improved. A number of features have been added based on this feedback, and additional actions and drawing methods are constantly being added. Who knows, even pie charts might show up in a later version, but consistent with the “DataGraph way.” You can download the DataGraph beta from the web site, and stay on the bleeding edge.

This document

In addition to this document, DataGraph also has an on-line manual that you can access from the Help menu. The on-line manual is more exhaustive and covers features that are not included in the paper manual. DataGraph also contains many tool tips that are displayed when then mouse hovers over a menu or field.
The screen shot above is the first thing you see when you open DataGraph to create a new file. There are three parts to this window: (1) the **data table** is on the left, (2) the **layout and drawing commands** are on the top right, and (3) the **graph** based on these commands is displayed in the bottom right.

### Data table

The data for the graph is stored in the data table on the left side of the window. The first column is simply called # and contains the row number. This can not be removed but you can change its name. The x and y columns are number columns. You can directly type values into the Data Table. For example, if you type 4.24 DataGraph will understand it. If the number is not valid such as 3.3.2, the entry will show up in red.

If you click on the definitions icon, which is located in the toolbar and the bottom left corner, the column definitions list opens and you see the structure of the data table, as shown in the following graphic. The column definitions list appears to the left of the data table when there is room on the screen. If not, the list appears over the data table but will disappear when the definitions icon is clicked a second time.

Every column in the data table has an entry in the list. Each column also has a specific type, indicated by its icon. For example, in the screen shot on the right there are three columns defined, #, x, y, label. The “label” column is a text column. This is primarily used for labeling data rows but can also be used in drawings.

You can rename and add additional columns by clicking on the buttons above the column definition list. To delete columns, you can select the column in the table and hit delete or select the entry in the list and hit delete.

You can also convert columns between types by using the small gear menu to the right of the name, as
shown to the right.
For example, if you enter labels into a numerical column all of the entries will be red. In this case, you should convert the numerical column to a text column. That also gives you different sorting options.

The expression column allows you to evaluate an expression based on the data in other columns. In this column, you specify the expression that should be evaluated for each row. As shown in the following example, columns are referred to by their name, so the expression “x+y” means that for each row of the “mapping” column the entries in the x and y columns are added together.

There are several additional column types. For example, the “date” column is for calendar dates. As part of the column definition, you can specify what the format should be for each date entry. All of the column types are explained in more detail in the following chapter on the Data Table. That chapter also explains how to import data, edit individual cells, sort, organize and more.

Layout & Drawing Commands
DataGraph does not present you with a choice of predefined graph types but rather gives you a canvas where you can add drawing elements and combine them. A plot, a label, or a function curve on a graph are all listed in the DataGraph window.

The overall layout and drawing style for a graph is controlled using the style, canvas, and axis settings, which are always the top three entries above the drawing commands. These three entries control the axis ranges, space around the drawing area, and the size of the graph. The Layout chapter explains these entries in more detail.

Below the axis setting is a list of the drawing commands that specify what is drawn and how it is drawn. For example, if you click the Plot icon in the toolbar, DataGraph adds a plot command to the list. Once you have selected the data it will immediately appear in the graph. Clicking on the icon a second time will add a second command, independent of the first, but drawn on the same canvas. DataGraph will adjust the ranges to fit both plots together.
A drawing command refers to the data by columns. You select columns using a pop-up menu (see left) or by dragging the column definition onto the menu. You can also drag a selection from one menu to another by holding down the command key when you click on the menu. When you pick a different data column, or change values in a column, the graph is updated immediately. You can rename columns and reorder them without breaking the connection.

You adjust the drawing style, such as line width, marker size and type, inside each drawing command. Each drawing command shows the most frequently accessed settings at the top, and they are always visible. By clicking on the small disclosure triangle in the top left corner, you reveal the remaining settings. This gives you full control over every aspect of the graphical representation.

To delete a command, you select it and hit the delete key. You can select it by clicking on the icon, or click and drag a selection rectangle to select multiple commands.

There are a large number of drawing commands. The chapter entitled “Drawing Command Elements” explains aspects of the drawing commands that are common to all of them (i.e., how to specify a mask). The “Drawing Commands” chapter explains each individual drawing command in more detail.

Graph
The lower right corner shows the result from the drawing commands. By default the size of the graph changes when you resize the window or adjust the split views. By default the size is set to “automatic”. This means that the screen resolution is used to figure out the display scaling; thus, when you print a graph, it will print out the same size as it appears on the screen.

When you want to create graphs for publication you typically want to specify the size of the graph exactly. This is controlled using the canvas. If you switch the size to Specified, as shown here, you get two additional user interface settings. One sets the size in pixels, the other controls the viewing resolution. You can use the units in, cm or mm to specify the size. No units means you are specifying the size in pixels.

Additional Graphs
DataGraph has a single data table, but you can have a number of graphs in the same file. This is done by clicking on the Add Graph button in the toolbar, or through the “New Graph” sub-menu in the File menu. When you add the second graph to a file, DataGraph adds a list of graphs to the top of the window, and you click on the thumbnail to edit a graph. All of the graphs update automatically, so when you change the data you will see an update in the thumbnail.

Global Variables
In the lower left corner of the window, below the table definitions are the global variables. The global variables allow you to define numbers and text to which columns and fields can refer. Any numerical field in DataGraph is really an expression so a field that controls the line width can be an expression that contains a global variable and any expression. One of the variables can be specified as an animation parameter (default name is t) and you can have DataGraph animate this on the screen or save it into a QuickTime movie.
For example, you can create a Function command and specify the function to be \( \sin(2\pi b(x-t))x^a(1-x)^a \), where \( a \) and \( b \) are defined as shown below.

Note that if the variables \( a \) or \( b \) are not defined, DataGraph will not understand the expression and display it in red.

When you click the play button to the right of the slider, DataGraph runs the animation on the screen. If you click the QuickTime button above the play button you can specify the frame rate, format and name for the QuickTime movie.

Variables can also be used in columns. Both the standard numerical column, as well as expression columns. This is explained further in the next chapter about the data table. Variables can be used in text fields, such as legends, titles, labels etc. This is explained further in the Layout chapter.

Navigating the UI

The drawing commands contain all of the settings inside the view. There are a lot of settings available. Many settings are by default set to “Nothing” or “Automatic”, but you can adjust practically every aspect of the visual representation. To keep visual clutter down, only the currently active setting is visible at any given time.

As an example, consider how tick marks are specified. The tick marks are set using the \texttt{axis} commands. After clicking on the small disclosure triangle in the top left corner of the axis command, the tick mark settings are shown (third entry down). By default the rule is set to “Automatic” and DataGraph picks tick marks and formats labels to give a good visual appearance.

The tick mark menu contains several more settings. When you pick a different entry (e.g., Categories option), additional user interface elements are displayed to the right of the menu to adjust that option further.

Other options such as “Angles” or “Labels” will have different UI settings. DataGraph compacts the user interface like this in several places. It is recommended that you spend some time exploring the menus a bit to get more and more familiar with the many options that are available.
**Loupe Tool**

When a graph gets very busy it is sometimes necessary to zoom in on the detail. There are several ways to do this in DataGraph. You can zoom in by changing the axis ranges, you can use the Magnify command to draw an inset, or you can use the Loupe tool in the tool bar. You can also activate the loupe tool by using the keyboard shortcut, ⌘ M.

When you move the mouse over the graph, you see a magnified view, and can change the source and destination sizes by using the scroll wheel/gesture.
Data Table

The data table gives you access to individual rows in a column. You can enter in new values, delete rows/columns, sort data, find data, and even have DataGraph read back the numbers to you. The structure of the data table is defined in a list which is displayed by clicking on the Definitions icon or by double clicking on one of the column headers.

Each column from the data table has an entry in this list. In this entry, you can see the data type, the name of the column, and any settings that apply to either how the values are treated or displayed in the table. To create an empty column, click on one of the buttons above the list or select it from the Other menu.

The four most frequently used column types are: numbers, text, expressions, and date. These are explained briefly below and in more detail later in the chapter.

• **Number**: For standard numbers like 3.2, 1e-5 etc. Also, for expressions like 3/4.
• **Text**: Used for labels, names etc. The Values menu allows you to assign numerical values to the labels.
• **Expression**: A mathematical expression for each row. This allows you to combine columns, accumulate rows etc.
• **Date**: Convert calendar dates into numbers that can be used in a graph or for statistics.

The rest of this chapter is organized as follows. The first section deals with how to organize the column list (i.e., how to reorder, change column types, create sub-tables or Groups, and what is displayed in the data table). The second section describes the four most common column types in more detail. The third section describes how you work with the data table.

Organizing your data

Most of the table adjustments are done in the column list. What is displayed in the table depends completely on this list.

**Selecting**

Selecting columns is similar to how you select drawing commands. If you click on the icon of the column entry, the border changes to blue to indicate that this entry is selected. Another way is to click outside the command and drag a selection rectangle to select multiple entries.

You can also add to the selection by holding down the shift key when you select. If you click outside of a column, or select a command the column is deselected. Now actions like cut, copy and delete will act on the selection and not just this column. To delete the selection, just hit the delete key. If you delete a column accidentally, just select undo.

**Dragging**

You can drag columns around to reorder them in the list. If you hold down the option key, this clones the current column. This is one way to quickly create a column with the same settings. You can also drag columns between DataGraph files to copy them. As a
shortcut, you can drag the column onto a column menu in a drawing command to select that particular column.

**Groups/Sub tables**

When a table becomes complex, or you have multiple logically separate clusters of columns, you can use the group structure to keep them together and make it easier to select columns in a drawing command. In order to create a group, select the columns that you want to put together in a group and click on the Group icon. You can also drag entries in and out of the group.

When you use groups to set up sub-tables, the column selectors in the drawing commands will show the groups as sub-menus.

Since pop up menus cannot start inside a sub-menu, the column menu will show you the content of the same group at the top level.

In the example above, the “First experiment” and “Second experiment” both have two entries, and if you don’t have a column selected inside either of them the groups will show up as two sub-menus. In the example above, amplitude from the first experiment is selected. The parent group is therefore displayed at the top level, but with small triangles indicating that this is inside a group. The group name is also displayed in gray.

By default, a group has a different background in the data table to separate it visually. In the column definitions list, you can change this background color, or specify no color at all. This is done from the small action menu (gear menu) to the right of the group name.

**Text blocks**

Allows you to add comments. Create a comment block by clicking on the icon next to the group icon. You can drag this comment around, but need to click on one of the four corners to select the box.

**Column Types**

The number, text, date and expression are by far the most frequently used column types. The Other menu has additional types, which will come in handy as you start using DataGraph for more advanced actions such as extracting residual from function fits etc. This section however focuses on how the most common columns are defined. The small disclosure triangle reveals the settings for each type. For some like the number column, the settings are for optional formatting, for other columns such as date and expression, these settings are crucial.

**Number**

The number column is a standard numerical column. Typically you enter in numbers here of the form 3.41, 3.5e3 etc, but you can also type in expressions like 5/2, sqrt(5), 5e, ... By using global variables (explained later) you can for example define mm =
0.001, cm = 0.01, m = 1 so you can enter in 5m, 44cm, 22mm and this will be understood as 5^m = 1, 44*0.01 = 0.44 and 22*0.001 = 0.022. If DataGraph can not understand the expression it will display the content in red.

**Text**
The text column is intended for labels. When you import data, DataGraph tries to determine whether or not the data is text. If you end up with a column where all the entries are red, this is most likely because you are using a number column. Convert this column to a Text to avoid this.

Drawing commands that require a numerical value will show a text column in red (see below); however, it is possible to define a mapping from the text value to a numerical value.

There are several ways to define this mapping. For example, Sorted # will assign A to be 1, B to be 2 etc. Sequence # means that the value is assigned based on when the label first appears. For example, if the rows are C,A,C,B the values are 1,2,1,3.

The “Map” command allows you to specify the mapping exactly by assigning values to each label. For example, take a text column with the entries C,A,C,B. When you select “Map” from the “Values” pull down menu, a mapping table appears. This table is initially empty and you have to specify the value for each label. To simplify that task, you can ask DataGraph to populate the left column in the mapping table with the unique entries from the data table. First, select “Make sure all entries are mapped” from the gear menu. After that, all the unique entries from the data table are automatically populated in the “String” column, as shown to the right. Finally, you can type the numerical value for each String. In this example, the label “A” is mapped to the numerical value “100”. The gear menu has additional menu items to simplify this process and automatically assign values, sort entries etc.

When a text column has a mapping, it can be used where ever DataGraph expects a numerical column. For example, you can use this to map grades into a numerical score for computing averages.

**Date**
This column type is intended for calendar dates. Instead of trying to guess that a numerical column is a date and require you to use a particular format for of dates, DataGraph asks you to make that explicit by converting the column into a date column and specify the format.

For example, if you have five numbers in a standard numerical column of the form month/day year, these entries are not understood as numbers and will show up in red. To convert this column to a date use the gear menu.

When the column is converted, DataGraph makes a best guess at the date format. You can also specify the format inside the detailed setting for this column.

The action menu to the right of the name will show you formats that it guessed based on the content of the column, but you can specify this format manually. For expert users, you can specify a format string for formats that are not already supported. Any dates that can not be understood are displayed in red.

**Converting**
Sometimes you have a column that doesn’t have the proper type. Maybe you pasted in a column, or began with a standard numerical column and want to convert it into a label column. Instead of creating a new column, you can convert the type of the existing column using the gear menu, as shown below. When
you convert a number column to a label or date, the entries stay exactly the same.

Expression

A standard spreadsheet typically handles expressions by referring to individual cells of the form A5, A10, B4 where A,B,C... indicates the column and the number indicates the row. This gives you full flexibility but there are problems when you have a lot of rows and it is sometimes hard to know what each cell does. In DataGraph, this type of computation is done with the expression column. The expression column refers to other columns by name and that expression is then evaluated for each row. For example, if you have columns named “x” and “y” and use the expression “y-2x”, this expression is evaluated at each row where x and y are specified. You can move the columns around without invalidating the expression, but if you change the name of the column you need to change the expression.

As you add additional rows to x and y, the expression column is updated. The expression equation can include all of the standard operators +, -, /, * as well as the power operator ^ such as x^2, square roots sqrt(x) and parenthesis. In addition a large number of functions are defined such as cos, sin, tan, acos (inverse cos) etc. Note that cos and sin are in terms of radians. The constants π, pi and e are defined, so you can for example compute the cosine of a column x that is given in radians by using cos(x*π/180).

The number of rows in the expression column is determined from the number of rows in the columns that are used in the expression. Global variables and the # column do not have a length, so when you do not have a column in the expression you need to specify how many rows should be computed. This is demonstrated in the following example. In the “first” expression, the length of the column would be based on the length of “x”. In the “second” expression, an additional cell appears that requires you to set the length of the column.

Expressions can also refer to rows in other columns. This is done with syntax similar to functions. For example, x(1) is the first row in the x column. Since # is the row number, x(#-1) means the value of the previous row. This means that x(#) is the same as x. This allows you to enter the function x(y) where the y column will specify the row of the x column.

Expressions and sub-scripts do not allow you to refer to other rows in the same column. That is if ‘first’ is the name of the expression column, the expression cannot be x+first(#-1). This is the standard trick used to implement an incremental sum of values in x in a spreadsheet. This is done differently in DataGraph, and that mechanism allows you to do a number of other interesting effects. This is done with a the syntax column.property, where column is the name of the column and property is what you want to extract from that column. For example, the isum property is the incremental sum of a column, as shown below for the column x and using the expression x.isum.
There are a number of other properties. Some are single values such as x.max, x.min, x.average, x.median, x.range, x.sigma, x.s where x is the name of the column. So for example if you want to normalize numbers in a column x, you can use the expression (x-x.min)/x.range or (x-x.min)/(x.max-x.min).

You can even sort columns using this mechanism. This is what the x.sortwith property is for. This property is the index of each row in a sorted version of the row. That means that x(x.sortwith) gives the x values sorted, but y(x.sortwith) sorts the y column based on x. This way you can overwrite the x,y columns with new data and then use the sorted version of them in expressions.

This is useful when you want to draw the x/y values using the Plot command, since that command will connect each row together with a line.

Editing table entries

The column list can be thought of as the place where the data is stored. The place where the data is displayed is the table. You can choose which columns are displayed in the table by selecting the “Show” check box in the column settings. This does not affect which columns are visible in the drawing commands. You can also hide an entire column group by using the “Include” check box.

When a column is visible, you can edit the individual values, but only for the number, text, date or binary columns. The other columns can be displayed, but you can only change the rule behind the values and how the values are displayed.

Data Entry

The data table has slightly different input methods than what you might be used to from a spreadsheet. To start editing an entry, double click on it. If you press return (enter) you close this cell and start editing the cell right below it. If you hit tab you go to the next editable cell to the right or to the next line if you are at the end. By holding down the shift key when you hit the tab key you go to to the next editable cell to the left or up to the previous line.

DataGraph will always keep the last row empty, and as you add values to the last row a new and empty row will show up below it. Pasting in data is explained in the next chapter.

Deleting rows

You can also select a set of rows and hit the delete key. The main issue here is that when you select rows you select all of the rows that are displayed in the table. If you just want to delete the rows of a sub-set of columns, what you need to do is to hide (uncheck the Show or Include box) for the columns you don’t want to change and hit the delete key. You will get asked which columns should be affected, but you can decide to only delete the rows from the columns that are visible, the columns in the same group or all columns.

Reorder rows

The simplest way to reorder rows is to drag them around in the table. This is done as follows:

1 - Select the rows
2 - Click and drag the mouse to the left or right.
3 - Drag the selection up/down.
What will happen is that you will start dragging a gray version of the selection. A blue line will show where this block will be moved when you release the mouse. If you hold down the option key, you clone the selection.

If you click and drag the mouse more up/down in step 2, the system interprets this as a change in the selection and does not drag the selection around. If you are not displaying all of the columns, the same window pops up as when you are deleting rows, since it is not clear from context which columns you want to be affected.

Reorder columns
You can reorder columns in the column definition list or in the table. To drag columns around in the table, just select the column and drag them. When you release the mouse, the column will be moved in the column list.

Sorting rows/columns
You can sort data based on a column. The simplest way to do this is to select the column you want to sort, and then go to the action menu below the table, or the Data menu in the main menu bar. There you can select the sorting method. Note that the action menu gives you more options, and the options depend on the particular the column type. For example, the text column can be sorted according to last name and natural sort puts “value 2” before “value 10”.

The sorting options are also available in the context menu for a column. You get this menu by either control clicking (right clicking) on the column header in the table or on the small action menu icon to the right of the column name in the column list.

When you have groups, or not all of the columns are displayed, the action menu uses a sub menu for sorting. Using this sub menu, you can specify exactly which columns you want to include in the sort. This is important because sorting only the visible columns might corrupt the data. For example, say you have a class list of student ID and final grade. You hide the student ID column and then want to sort the table according to name or a grade.

For example, say you have nested groups. The highest level is a group for class section. Inside that, you have a groups for each midterm. Inside that, you have the grades for that midterm. If you select sort for the final grade, you need to specify what you want to sort. Here it makes sense to sort everyone in the section, since sorting the midterm group will certainly corrupt the data.

Editing

Shortcuts are available through the context menu. You get to the context menu by either right clicking or control clicking. If you select rows and control click on the column header you trigger the context menu for that column. This allows you to fill the entries with a fixed value, or use the first two rows to fill out the remainder of the row selection. This only affects the column that is selected.

Find & Filter
DataGraph has a find action, triggered with ⌘F. This command provides the user a search bar above the data table that can be used to filter the display of the current table based on a search string. This does not affect any drawing or change the data, but you can select rows to copy/
cut or delete them. You can also edit individual cells in the selection. Note that this command works on all the data columns that are displayed in the data table.

**Find & Replace**

DataGraph can also be used to find and replace data as needed. For example, if you have extra text in a number or quotes where they shouldn’t be you can easily isolate this data and update the data table.

First, you should select the columns or rows that you want to modify. To open the find and replace window, you can either hit `⌘R` or select *Find and Replace...* from the data menu. In the example below, “Remove non-number” is selected.

Entries that are changed are displayed in red, unchanged entries in black. You can pick several filter options, as well as using a find-replace string match.
Importing/ Pasting Data

The previous chapter dealt with how to change the structure of a table, and how to edit and manipulate values in a table. You can use this to enter in values one by one, but often values are either in a file, or the clipboard. From a high level there are two separate parts, the data source, and the destination. Source is a file or clipboard and destination is rows or columns in the table.

**Data source**: If the data is in the clipboard you can select between two options - *Paste* or *Paste special...*. The paste option uses default rules to figure out what separator is used for the columns and to determine the column type (i.e., number or text). *Paste Special...* brings up a sheet where you can control the separator, transpose the data, and trim the data, as it is pasted in the data table. Note that if you drag in a text region from Text Edit it is handled as a Paste.

If the data is in a file you have three options, as shown below. *Import File...* is similar to the standard paste, and uses the same rules as the paste action to determine what the separator is used (tab or comma). *Import File Portion* is similar, but allows you to select a sub set of the lines in the file. This is helpful when you want to trim off a header or footer before the standard import method. The *Import Special...* option is a very flexible import method, and gives you a tremendous amount of control over how data is imported.

Note that if you drag in a file it is imported by using the *Import File...* option. You can drag the file either onto the data table or into the list of columns. If you drag it into the list of columns it creates a group of columns.

**Data destination**: There are two different import destinations, the data table and the column list. The column list only accepts a dragged file or columns dragged from other DataGraph files. This adds the content as columns or as a group with columns, and is explained at the end of this chapter. When you import a file, the content is either appended or overwrites data that is already there. There are three different ways that data is added:

- If nothing is selected in the table, the data is appended as new columns. The exception is that if there are no rows in the columns that are displayed, the data overwrites the existing columns.

- If you select columns (or no rows are displayed), the selected columns are overwritten with the data, and additional columns are inserted to the right of the last selected column. If the data source has column names this will also overwrite the column names. Note that drawing commands that use the selected columns will still refer to those columns even if the name changed.

- If you select rows, the data adds the data to the columns and overwrites the selected rows. This means that if you want to add data above the first row, you first create an empty row at the top (menu entry in the Data menu) and then overwrite that row with the content. Any additional rows will be added below the last selected row. No columns will be added, and if there is a different number of columns in the incoming data, or the column names don’t match, DataGraph will ask for a clarification. This is explained further in a later section with an example.

**Interpreting data**

When data is written as a text, and here it doesn’t matter if the text is in a file or the clipboard, there are a number of conventions used. DataGraph tries to understand most of those conventions without any additional user input. The easiest is where each entry is separated by tabs. Clipboards from spreadsheets are typically stored this way. But even in that case, sometimes the first row is just numbers, sometimes it is column names. Another common separator are commas, so called CSV (comma separated values). Here the problem is that sometimes the commas are part of a label and in Europe a comma is used as a decimal separator in numbers. Also, sometimes
numbers are separated by spaces, so “4 2 2.3” are three numbers.

DataGraph tries to handle all of these cases automatically when you import a file or use the Paste action.

- **Separator**: If there is a tab character in the text, this is used as the separator. If there is no tab, DataGraph then checks for commas, semi colons or pipes (|) and then finally uses spaces.

- **Title row**: If the first line has no numbers, the row is viewed as a title row. Otherwise, the first line is considered a data row and the columns don’t have any name.

- **Column type**: If DataGraph needs to create a table column to fit the input data, the column type is determined from the first row. If the first row is a number, the column that is created is a standard number column. Otherwise it is a text column. If this is incorrect, you need to convert the column. But if you are overwriting columns or appending rows, the existing types are used.

**Paste Special**
The paste and import functions use the rules explained above along with some automatic substitutions, to convert text blocks from files or the clipboard into a table of numbers or text. But sometimes these rules might not import the data in the way you would like. For example, what does 4,5 mean? DataGraph assumes that you are using a comma separator between two columns and would interpret this as the number 4 and the number 5; however, in many countries the comma is used to separate the decimal, and this is the number four and a half.

If these rules are not what you want, and that will unfortunately happen more often in Europe than the US, the solution is to use the Paste Special... option.

For example, take the following numbers

5.6
5.334.5

If you have these in the clipboard and you select Paste Special..., you will see the following screen, in which the data is interpreted as two comma separated columns.

If the decimal mark here is the comma, there are two settings you need to modify in order to import these numbers correctly. These are illustrated in the following graphic. First, set the separator to “Only Tab” so that the comma (,) will not be used as a separator. Next, click on the number conversion so that 5.334,5 is converted to the standard form for DataGraph. Note that you can still display numbers in the graph using , as a separator. You just need to adjust the separator setting in the style sheet.

**Paste Special...** also has additional features such as being able to paste in a transposed table. So if you have two rows with a lot of numbers, you can convert them into two columns. This is important since drawing commands refer to the columns and use all rows in that column.

**Import Special**
DataGraph is very flexible when it comes to understanding data, and throughout the years the import mechanism has been tweaked to understand files coming from various platforms, tab, comma, space delimited etc. This should hopefully cover most cases, but there are some cases where these rules fail. Rather than depend on other tools to massage the data, you can use the Import Special... method.

You first select the file using the standard file picker, but once you have done that a new window opens up, as shown in the following graphic. In this window, you can specify the structure in the file. In the top right corner, you specify the rules that should be applied to extract the columns. The rules allow you to specify the separator, skip over certain number of lines or characters, search for starting symbols etc.
You can toggle the entries open to specify mappings that should be done, such as what to do when the entry is empty, standardize numbers or texts etc.

The settings for Import Special are saved. The converter is saved in the DataGraph file, but you can drag it into the global list, rename it, drag around etc. When it is in the global list, it is accessible from other files. Just click on the entry in the list in the top left corner to apply the filter to the content of the file. The bottom shows a preview of what happens to the file. For large files, only the first few lines are converted for the preview. When you hit the Import option, the file is converted. The file is not read into memory all at once, so it will be able to handle very large files efficiently.

Examples
The best way to explain the rules that were just described is to show a few examples of basic input.

Overwriting columns: Consider a file with three columns and 5 rows. The file could be comma or tab separated, and the first row contains the titles. Assume you start with an empty table, which just has the default x and y columns, and without selecting anything in the table you pick Import File... from the Data menu.

Select the file and hit Open. The columns are now called A and B instead of x and y and you have a new column C.

The plot command that used to draw x and y is now drawing A and B, since they overwrote the columns that were there.

Next, assume that you have a second file that you want to import. If you don’t have any columns selected and import the file, the columns are now appended and you have a total of six columns.

If instead you select the first three columns (image on left) and import the file, those columns are overwritten (image on right) and the plot command will still draw the A and B columns.

Adding rows: If you need to add rows to columns, the way you do that is to select where you want to insert the rows and then either import or paste in the data. Assume that you have three rows and you want to add 3 rows to the end. The first step is to select the last row in the table, which is always empty. Then paste or import the file. Since the column names are the same, DataGraph will append the content. If the column names have a different name, DataGraph will ask you to clarify what should be done. For example, if the column names are A and B, DataGraph gives you a dialogue box, allowing you to specify how to input the
DataGraph also remembers what you picked, so next time it will start by guessing the same action.

When you want to add rows at the top, the first thing you need to do is to add an empty row. Do that by either using the Data menu or control click (right click) on the first row and add a row to the table.

Then select the empty row and paste the data or import a file. This overwrites the first row, and any additional rows are added, as shown below.

If you select rows and paste in data where the column names are not an exact match, DataGraph will ask you to specify which columns should go where. For example, let's assume that the original table has three columns x, y, and y2. You want to overwrite rows 2, 3, 4. You import content either through the clipboard or a file. This import has different rows, let's say x, x2, and y. DataGraph will ask for a clarification of what you mean to do and to do that it brings up a sheet.

Click on the black dot on the left and drag the arrow to the column on the right. Then when the data is added, the three selected rows are removed and the two rows are inserted instead. This shortens the selected columns but leaves the other columns untouched.

**Dragging a file:** If you drag a file onto a table, the table gets a blue highlight. If you release the file DataGraph responds in the same manner as when you select Import... from the data menu. This is handy when you see the file in the Finder. You can also drag a file into the column list. This action works in a slightly different way. When you drag a file into the column list, you will see a drag line. If you release the mouse, the content of this file is added as a group, with the columns coming from the data file. The name of the group comes from the file name.

If you drag a file onto a group, the content will overwrite the columns inside this group. This is the same as selecting all of the columns in the group and then dragging the file onto the table.
File formats
In addition to text files, DataGraph can import other data types. The file is selected just like the text file. For file formats with binary data, DataGraph creates binary columns so no information is lost. The file formats that DataGraph 3.0 supports are:

- Cricket Graph data files. Cricket Graph was widely used years ago, but the last OS that supported it was OS9. Intel machines cannot run the classic compatibility layer, but DataGraph can import these data files.

- Matlab data files. Matlab is widely used in engineering and sciences, and you can save data in Matlab and import it into DataGraph. The key here is that you need to save the file with the -v4 flag. For example, if you want to save the workspace so that it can be loaded into DataGraph type

  ```
  >> save import.mat -v4
  ```

Don't forget the -v4 flag. See help save in Matlab for more info about how to save only particular variables. All row or column vectors are read in as columns (binary).

- netCDF files. All lists are read in as columns (binary).

- Plot files. One of the first Cocoa applications.

Copying and dragging columns
You can drag columns between DataGraph files. For that you need to drag them from the column list and drop them into a column list in a different DataGraph file. You can also hold down the option key to clone columns in the same DataGraph file.

You can select columns in the column definition list and copy them. This allows you to select entire groups, which you can’t do when you select columns in the table and copy.

One thing to consider is when columns are large. If you select columns in the data table to copy, DataGraph adds a text representation of the data to the clipboard. This means that you can paste the data into a spreadsheet or different application, and the string representation will be used. If you have a few million entries, this can take a few seconds. If you are intending to copy columns within DataGraph itself, it is faster to select the columns in the column definition list and copy from there, since then only the native DataGraph format is added to the clipboard. This means that you cannot paste the data into a different application when copied from the column definition list.

When you overwrite columns, the connections that you have made in drawing commands will not be broken. This means that you don’t have to repeat any of the steps to convert data into a graphical/statistical representation.
Layout

There are three groups of settings that are used to define the overall layout of a graph: the **style**, the **canvas** and the **axis** settings. These are always located above the drawing commands and are shown in the following image.

The **style** settings are used to set overall rules, line thickness, font, color scheme and look of the axes. The **canvas** settings are used to position the drawing area. The **axis** settings determine how the data is mapped to the drawing area defined in the canvas settings. The axis setting also allows you to set tick marks and axis labels. This chapter focuses on these three settings. The content of a graph is created by using the drawing commands, which are described in the next chapter.

**Style**

The style setting is the top entry, and the icon looks like an artist palette.

At the top you select the Main font style, the default pen color, line thickness, fill for markers and marker size. By default, drawing commands will refer to these settings but that can be overwritten.

Once you open the detailed portion by clicking on the disclosure triangle, there are a number of additional settings you can change. These are grouped together by function, with small spaces to separate them visually. The first group defines the built in font styles. Some are used by the axis settings, such as title and axis.

The drawing commands that draw text also refer to these fonts by default. How to use them and overwrite style and color is explained in the next chapter. By default, they all refer back to the Main font defined in the top line, so if you change that font everything in your graph gets updated in a consistent manner.

Below the fonts, you set the color theme. You can use the built in colors or you can overwrite the colors freely. The colors that are selected in the style settings are then accessible to the drawing commands. The drawing commands can pick those colors from a menu rather than having to pick every color separately. The intent is to make it easy to limit the number of colors you use and change the color of several objects at once. By defining them here you can quickly change the color scheme of the entire graph.

Below the color list you specify the drawing style for the axis as shown below. Note that by default the axis line is set to be “pen”. The pen size is set in the top left corner of the style settings. This works since the style defines a variable - **pen** - that is accessible from all command entries in the graph. When you change the pen thickness you change the pen variable at the same time, and this will therefore change the axis line. Since pen is a variable, you can also use it to create expressions. For example, you can set the axis line to be pen*2 to make it twice as wide as the pen. You can also set it to “1.0” when you don’t want it to depend on the pen thickness and always be one pixel. This approach is used in the drawing commands to make the default line thickness be the same as the pen thickness.

In the style settings, you set the grid style and the drawing style for the tick marks. The position of tick marks is controlled by the axis setting or by using the Extra Axis drawing command.

Below the axis style is the language and localization settings. The date tick marks use names such as Jan, Feb, and date formatters allow you to use week day names as well. The default option is to use the settings in the system, but if you want to share a file with someone in a different country or use a different language for other reasons, you can select that here. Here you can also specify whether or not numbers should be formatted with a decimal or a period.
You can save the settings by using the style set and quickly switch between styles. For example, you can set up a grayscale vs color theme or thick presentation theme vs thin scheme for printing.

You can copy a style and paste it into a style setting in a different graph. You do that by selecting the style setting entry (thick blue border) and select Copy. Then go to the graph you want to change and select Paste. You don’t have to select anything.

Canvas
The canvas sets the drawing area for the graph and the title of the graph.

In the canvas, you can specify the figure size by using the “Size” menu. Once you select the size you can set the working magnification. This does not affect the output but makes it easier to work on small graphs or on machines that have very high resolution. The Actual setting in the magnification menu makes the graph appear at the same size as when you print it. Otherwise a graph will look a lot smaller on the screen than when you include it in a text document. It is better to set the size of the graph to the final size in DataGraph rather than scale the figure in another program (e.g., word processor).

Another common setting you need in the canvas detail is the margin settings. If you leave them blank, DataGraph uses automatic rules to determine the space around the data area. This space depends on the title, font, setting in the axis, what drawing commands want to draw outside the data area etc.

The following figure shows how the margins are computed. If you leave the margins as automatic in the canvas, the margin is the sum of the tick mark space and title space. Otherwise the margin space is exactly what you specified, but this will not affect the tick or title space. Note that the margins are set in pixels, but

DataGraph has defined conversion constants so that you can type in 1.1in and it gets converted to pixels.

Note that the tick spacing is adjusted in the axis settings. The title space depends on what is drawn there and is not set directly. If you want to tweak it, you do that by setting the corresponding margin.

Axis
Once the size of the drawing area has been determined by the canvas settings, the axis settings control how the data is mapped onto the graph. This setting is partly also a drawing command, in that it will draw the x and y titles when specified and will draw tick marks when the Draw x/y numbers is checked. This can also be accomplished using the Extra Axis drawing command.

The axis range in x and y is computed in three steps as described below.

Step 1: Compute the union of all the drawing commands. The drawing commands will affect the x and y ranges based on the data that they are representing.

Step 2: Add to that the numbers you specify in the “Include in x/y” field inside the axis settings (see image below). By default this is empty, which means that the range from step 1 is unchanged.

Step 3: Pad this range so that the drawing commands will not go right up against the boundary. The default is “Nice Values” which uses a number of heuristics to pick the padding. You can turn this padding off and pick other methods using a menu.

Step 4: Restrict this range, i.e. crop, by using the range specified. By default this is -∞,∞ which means that the range from step 3 is unchanged.
For example, consider the case where you have a plot where the x range of the plot is \([0.2, 1.5]\). This will be the result from step 1. If you leave the include field blank, step 2 is not changed. If the padding in step 3 is set to “Nice Values”, DataGraph will change the range to \([0.1, 1.6]\) to add space to the left and right. Step 4 is by default set to \(-\infty, \infty\) and will therefore not restrict it at all. Note that this padding is independent of the graph size.

In this case, you might want to include the origin. Do that by typing the number 0 into the “include in x” field. This means that step 2 will make the range \([0, 1.5]\) and then the padding will make it \([0, 1.7]\) to keep things symmetric.

You can also specify the range exactly. Assume for example that you want the range to be exactly \([0, 2]\) independent of what data you have. Then you set the include and restrict fields to 0, 2. Here the padding option doesn’t matter. This means that any data outside that range is not visible. If you are zooming in on data, this is what you want. If you want to see whether or not there is anything outside, but still ensure the range includes at least \([0, 2]\), then you leave the “restrict x” as is and turn off the padding.

**Axis types**
The axis type determines how a coordinate is mapped. Most common is linear, which means that each unit on the axis has the same length in the output. Logarithmic means that first DataGraph computes the logarithm of the data and then draws the result linearly. Logarithmic is better when you draw stock prices and other data where relative growth/reduction is more important than absolute size.

**Crop using the mouse**
When determining the region for an axis, and this also holds for the split axis described below, the last step above (4) is to crop the range using the restrict field. It is certainly possible to enter in your own numerical value here, but if you just want to zoom in, this is not a very fast method. DataGraph has a number of methods to set and modify the cropping range using the mouse and scrolling. The most common method is to click in the background of the axis and drag the mouse. Once you have dragged it a few pixels, you see a blue selection interval. As shown to the right, if you drag the mouse back towards the starting point it will vanish. When you release the mouse, these intervals will overwrite the current restriction range for the x and y axes. By moving the mouse horizontally, you only crop the x axis, and only vertically, and you crop the y axis. If you do this accidentally, the simplest thing is to move the mouse towards the starting point to not crop anything or hit undo after you release the mouse.

Once you have a cropped axes you can see it by the small icons that show up in the top part of the axis settings. This means that the range is restricted, and by clicking on these buttons you reset the range to \(-\infty, \infty\).

If you then move your mouse over the axis, a floating window pops up on top of the axis, as shown in the following image. This window allows you to adjust the range.

You can remove the crop, click on either end of the white bar to adjust one of the ends or click and drag the entire bar to translate the range. You can also use the scroll functionality to translate the range. Note that if you scroll up/down for the x axis and left/right for the y axis, you zoom in or out.

**Split Axis**
By default, the entire drawing region is used by the axis specified in the axis setting. To have more than one graph in the drawing region, you can split the x or y axis. You can split either axis using the Split X or Split Y buttons to the right side of the canvas setting.
When you click the Split X or Split Y button you get additional settings just below the axis settings. A small axis selector is also added to the right hand side of each drawing command that represents the drawing region and highlights the location for the drawing command. You can easily move a drawing command by simply clicking a different axis.

The simplest and a common use for this is to create what is typically called Double Y graph. For example, consider the line plots below that have very different y ranges.

To overlay them, you first need to split the y axis. In the plot command for the red line, click on the small axis selector that was added to the right side. Now the graphs show up one on top of the other, as shown below. The final step is to overlap the two y axes. This is done by selecting “Join Y” on the right hand side of the canvas (see above image).

If you split the x axis once and the y axis once you get a 2x2 matrix of graphs. Each row shares the same y axis and each column shares the same x axis. The ranges are computed independently for each axis so the x axis for the first axis applies steps 1-4 listed before but only for the drawing commands that are in the first x axis.

You can overlap the two x axes by picking Join X.

Joining the Y axis instead gives you

If you want fully independent graphs, you should create them as separate graph (explained in a later chapter) and combine them using the Graphic command.

You can adjust how the split axis is joined. In the graphs above, the Box style in the style setting is set to Axis Box instead of the standard “One Box”. You can change that style, and inside the Split axis entries you can adjust how the break in between the sub-axes is drawn. For Offset box style and break separator you get the result shown below.
**Split axis spacing**
Inside the split axis you can adjust the spacing and layout. The following figure shows what each field controls.

The main axis settings only has controls for the space for x and y tick marks. The overall size is controlled by the canvas setting. The width/height fields are relative to the main axis. The space between axis can be adjusted in several ways. If you set it to Exactly... you can type in the exact space in pixels/mm/inches etc.

**Axis Labels**
If you specify the x or y title, this command will draw them centered with the y title rotated 90 degrees counter clockwise. This is the most common way to draw axis labels, but if you want a more flexible method you can leave the labels here blank and use the Text command to draw the labels. This command allows you to draw it non-centered, rotated left aligned etc etc.

The spacing for the x and y tick marks are controlled in the axis entry. By default the space for the y axis expects the number to about 3-4 digits. If you only have a single digit, such as the integers 1, 2, 3 this is too wide so switch the space to “Narrow”. This scales with the font size. This is used as part of the automatic margin calculation and to find how big the tick mark space is, and this will move the title.
Drawing Command Elements

One of the main differences between DataGraph and traditional drawing environments is how settings for drawing elements are handled. In most other graphing programs, when you see a line or text in a graph, the way to change the line thickness etc. is to click on an element of the drawing, expect it to highlight in some way, and then, either use a dialog box (menu entry or double click) or an inspector window to change the properties behind that element. This same inspector would be used for each line, and you change them one at a time. These settings are stored somewhere behind the scenes for every entry, and just a sub-set is shown at any given time.

In DataGraph, every drawing element is visible in the “drawing command list”. When you click on a drawing element, the drawing element does not highlight on the screen. Instead, the applicable drawing command is selected (highlighted with a blue border). The UI is always visible, even when the drawing element is not selected. You can copy settings between commands, compare them side by side and view all of the settings at the same time. Every change you make is applied immediately, and if you didn’t mean to make a change, you can use the undo mechanism or change it back.

This chapter goes through the common building blocks used in all of the drawing commands. The next chapter explains in more detail a few important drawing commands - Plot, Point, Bar, Pivot, Label, etc. There are a number of other drawing commands available and they are explained in the on-line help.

Level of Detail

There are two levels of detail for each drawing command. The top is always visible and is typically just two lines. These are the settings that you frequently adjust and gives enough information to get a sense of what is drawn. For example in the case of a Plot command, the top two lines show the x and y columns that are used, the line style and point style. When you open the little disclosure triangle you see the rest of the settings, such as if it should be filled, how to connect the markers, if you want a label printed at each data point, set the error bars etc etc. In the lower right corner is a small ? button which will bring up the on-line help for that drawing command.

The UI is fairly compressed, and throughout the UI, as you select menu entries, you can expect additional details to show up to the right of the menu. As an example, consider the line options for the Plot command. The default is to connect the points by straight lines. But there are two other options, Steps and Smooth. If you pick Steps there are three different step styles, so to the right of the Connections menu you get a second menu which allows you to pick which style. The smooth connection is a spline, and there are two different types of splines that you can pick from a menu. With very few exceptions, selecting an entry in a menu will only add or remove user elements to the right or below the menu.

Selecting Data Columns

For drawing commands that use data from the table, the way you refer to the data is by selecting the column from the column menu. This shows the column in the order that they appear in the
column list. Any column groups will show up as a sub-menu. You can also drag a column from the column list and on top of the column menu. This selects that column. Once a column has been selected, you can change the name of the column and move it around. The column selector keeps track of it. Columns that can not be used, such as a Text column if you need a numerical column, are drawn in red. You can still select them, but they won’t be used, and typically nothing will be drawn.

When you select the column, DataGraph pops up an error message that explains why the entry is red. Many drawing commands can not use the # column. This is because # differs from a normal column in that it has no length.

Color

You can tweak the color for every drawing element. For a good graph you should use color sparingly and consistently. Often you want to use the same color for a number of different elements, and rather than having to set the same color over and over, DataGraph by default sets up a color theme and you can pick colors from there. You can still specify colors for a single entry and have hundreds of colors in the same graph.

To explain this mechanism, consider the color picker that controls the line color for a plot. This is a menu, and when you click on that menu you will see 17 preset colors and at the end the “Color Picker”. If you select the color picker you get a small rectangular color tile. When you click on the border around the tile, you bring up the standard color picker from the operating system and you can pick the color. Click on the tile again, or click on a different tile to deselect the color and stop sending color changes to this color selector. You can drag the color tile onto another color tile. The problem with setting each color by using the color picker is that it is tedious to change a color if you decide that that you want a slightly different hue of green. You then need to change several drawing elements. This is why it is better to use one of the 17 predefined colors by selecting them from the menu. The colors are defined in the style settings. The pen and point are defined at the top, and the line, error, fill and background color are defined there as well. These are defined with the standard color pickers. So consider referring to those colors, and then set those using the color picker.

Font Style

Font specification is set up similar to the color picker in that by default, DataGraph sets up font styles that make it easy to use a uniform font, sizes that are consistent and when you change the main font everything changes with it. At the very top, the first line is the main font. In the detail settings are four
more font styles - Title, Axis, Label and Local. Drawing commands will then refer to those fonts and allow you overwrite aspects of them.

When a drawing command uses a font, the first step is to select the font style from a menu. There you can pick from the font styles defined in the style settings or variables. You can then overwrite certain aspects of the font. This is done by the next three menus to the right of the font menu.

There are three things you can overwrite. The first is the font style, as in changing it to italic or bold. The next menu allows you to overwrite the size. This is relative, so you specify how much the font size should be changed. The third menu allows you to overwrite the color. The top entry just uses the color you defined in the parent font, but you can choose a color from the presets defined in the style settings or overwrite it using a color picker.

The style settings also allows you to overwrite aspects of the main font when specifying a font style in the same way. Here “My font” is defined in the variable list. “Main” is the font defined in the first line.

The main font also has a font selector that allows you pick from common fonts that are included with the system. Since the other fonts typically are all based from this font, everything changes consistently. Another option for the main font is the LaTeX font. This is intended for people that use the LaTeX package to typeset papers, and this way the graph will have a consistent look. All font selectors then have a “Specify...” option that allows you specify the font by using the standard font picker from Apple. Just click on the font name to bring up the font selector.

Mask
When you select columns, the drawing command will look at all of the rows in those columns. As you add/remove columns the drawing command is immediately updated. However, in some cases you might want to only use a sub-set of the data. For example you want to only create a scatter plot for rows that have a particular label, take histograms only when a given value lies in a specified range etc. This is what DataGraph calls a mask. You can specify which rows to include, and any other rows are removed for the purposes of that drawing command. For example in the Points command, which is used for scatter graphs, by default the mask entry is “Draw all”, indicating that every row will be used. You can select any column here, so you can mask based on a column that is different than the x or y coordinate. Once you select
the column, new entries will show up on the right of
this menu. The next entry you change is how to
determine if a row in the column should be included
or not. The first two entries use a range, but for <,>
... this range changes into a specific number. Note
that both the number and the range can use global
variables, so you can specify global variables a,b and set
the range to “a,b” so that you can easily change a and
b using a slider.

Using the small + button to the right you can add
additional masking lines. This allows you to limit the
mask further.

Pop up slider
There are a lot of numerical fields in DataGraph. They
control point sizes, line widths, offsets, histogram bin
width etc etc. When you enter in a new number you
have to close the field before the change takes effect.
You can close it by either hitting return, tabbing out of
it or clicking with the mouse somewhere to change
the focus. Changes are made immediately, but to make
this even faster and more immediate DataGraph
allows you to change many of those values by using a
pop up slider. Next to most of the
numerical fields is a
small icon of a slider,
a black slider with a
small blue button.
When you click on that, a small black slider pops up.
The range of this slider is automatically determined
based on the current value and the valid range for that
parameter. For example to change the marker size, just
click and drag the mouse until you get a size that you
like. If the range is not sufficient, move the mouse all
the way to the top/bottom of the slider, release the
mouse and click again. Since the range is determined
by the current value the slider range has changed the
next time you click it.

Text labels
There are a number of aspect to text. There are text
for labeling axis, text labels, you can put labels at
individual data points, to mark connecting lines, inside
legends etc. There are two sources for text, either you
specify it in a text field inside the drawing command
or it comes from a data column. A text field allows
you a lot of flexibility to build up the text, and you can
use global variables and properties from inside
drawing commands. For example, you can put the
result of a fit command inside a text label, axis title etc
using the token mechanism. When you use a column,
either numerical or text, you can adjust the formatting
to set the number of digits for numbers, append and
prepend text such as currency etc.

The text can have formatting statements to set super
and subscripts, insert greek characters, mathematical
symbols, change the font style, color etc. The text
layout is explained in more detail in the on-line
documentation, but here are some of the most
important and frequently used aspects.

Greek/Special Characters: Every text field as
well as the data table supports unicode characters.
That means that in addition to cyrillic, greek, accented
characters etc you have access to a lot of the symbols
such as daggers, copyright signs etc. The standard
method to insert these characters into text is to bring
up the Font panel and click on the Character... palette.
This is certainly possible in DataGraph as well, but
involves a number of clicks. DataGraph speeds this up
by using notation borrowed from TeX (LaTeX). LaTeX
is not what you would call a WYSIWYG editor, and
even though it is used extensively in math and physics
it isn't used that much. But TeX introduces a very
compact notation to insert characters and commands
using the \ character. So the character \alpha is written as
\alpha followed by a space. Similarly \beta, \gamma, \\delta, \nu, \mu etc.

This is used for a lot more special characters, and they
are listed in the on-line help. Or you can look for a
LaTeX reference. Some examples are \euro, \pound,
\permille, \leftarrow, \longleftarrow.
**Super/Subscript:**
Super and subscripts are implemented by using the special characters ^ and _. This means that if you want to type x squared you use x^2, and sub scripts are done with _, so if you want to create a a subscript 0, type a_0. The sub/super script only works for a single entry. To apply it to a longer text, enclose it in curly braces, such as a_{10}.

Since \, ^, _, {, } have special meaning, you have to use a workaround if you want to use them in a label. You do that by putting \ in front, that is \\, \^, \_, \{, \}.

**Math symbols:** In addition to the greek characters and super/sub scripts, DataGraph contains some basic equation structures such as square roots, fractions and sums. If you have very complicated equations you might want to use a small LaTeX to PDF utility such as LaTeXiT where you can use the full power of LaTeX, but in order to use that utility you need to install the entire TeX package. The subset that DataGraph implements is completely independent of any TeX distribution.

The main equations structures that are implemented in DataGraph are \frac for fractions, \sum for sums and \sqrt for square roots. So for example \frac 14 is the 1/4th fraction. If you want multiple characters, enclose them in { }, for example \frac {17}{19}. \sqrt 2 is the square root of 2, but \sqrt{23} is the square root of 23. For sums you can for example use \sum_{n=0}^\infty. Note that the expression parser is also fully unicode compliant.

**Style and color:** You specify the font for the text label by using a font. This means that you can’t use different fonts in the same label. Where people typically need that is if they want to make a word bold or change the color of a part of the text. DataGraph supports this using a different mechanism, and one that works better with the token mechanism. To switch to bold use the \bf or \bold command. Similarly \it (italic) and \rm (plain). This is a switch that is then in effect until something else is specified. You can use {} to limit the range, for example “this \bf box is bold” is equivalent to “this \bf box \plain is bold”.

Color is specified similarly. DataGraph understands \red, \green, \blue, \black, \white

**Multiple Lines** If you paste in multiple lines from a word processor, DataGraph will draw multiple lines for the label. But typing in a multiple line text label is not immediately obvious, since hitting the return key will close the text field. DataGraph does not wrap long lines automatically, so you have to specify where the line break should be. Typing the line in a different application and pasting it in is tedious, so DataGraph has a command to insert a line break. This is the \n command. So “First line\n Second line” will give you a two line label. Note that the space after the n in \n is crucial since there are several commands that start with n. The space after \n is absorbed so if you want a space at the beginning of the next line you need to put two spaces after \n.

**Tokens**
Tokens give you a way to include a dynamic link in a text label. To the right of the text field there is a small + menu where you can create a link. The link points to either a global variable or a property inside a drawing command.

For example, the axis settings entry has fields for the x and y titles. If you click on the + button to the right of the X title, there is a sub menu for the plot command inside the graph. If you select the “X Label” entry from you get a blue token in the text field. This token is the name of the column that is selected. So if you select a different column in the plot command, the x title will change.
You can also use tokens to extract a fit parameter from a Fit function or statistic from a Histogram command. The legend field inside the plot drawing command uses tokens to create the “y vs x” name, as shown below.

Number tokens, either from a variable or a fit parameter etc need to be converted into text. You can adjust the format that is used by clicking on the small pop-up menu in the token. This also previews what the number will look like formatted.

If the parameter is a date, the format options are different.

Displaying numbers
Several drawing commands allow you to display values from the data table or computed results as text labels. For example, the Plot and Point commands allow you to draw a label at each x,y coordinate. The Bar command allows you to draw a label at each bar in a graph. The Pivot command can draw the underlying value on top of each bar.

For each of the above examples, you can set the format that is used to convert numerical values to text. There are two types of formatters, one for numbers and another for dates.

For example, you can adjust the number format and set a prefix/postfix for every label. You can add a currency label such as \euro, \pound, or $ before or after each number. If the column is a date column, the formatting options change so that you can specify the day format from a preset or your own format string.

Fill Style
Several drawing commands allow you to fill in a solid region. The fill settings are similar between drawing commands. The default is typically no fill, and as you select the fill method, the view to the right of the menu contains additional details. For example, the “Solid color” setting shows a single color selector. The pattern allows you to vary the hash pattern, background color, line color and width. This pattern is done by drawing lines and not a bitmap.

The Noise fill option is a fill based on a procedural texture. This is a fractal noise function, called Perlin Noise, and you can adjust the background and foreground color, the weight and noise size. Perlin noise is a spatially coherent noise function between 0 and 1. If the weight (field on the left) is 1, the noise is used to switch between the colors. If the weight is 0.5, the noise is scaled by 0.5 and then used to switch between the colors. The field on the right is the grain size. For interesting effects you can set the colors to be semi-transparent. The noise is computed dynamically and when you print or export to pdf the noise is computed at a higher resolution than the screen resolution so it will print properly. This is implemented as a bitmap, so the pdf/eps output file can be large.

Axis selector
DataGraph allows you to break down a graph into sub-axis.
Each sub-axis is created by clicking on the Split X or Split Y buttons in the canvas settings.

The standard axis settings is for the axis in the lower left corner, and each split axis gets its own settings where you can specify the axis type, ranges, etc., just like the axis settings. To indicate what you are changing, there is a small schematic drawing on the far right side. You can not change anything, it is just to indicate what portion of the graph you are adjusting.

Each drawing command changes a little bit. It gets wider and on the right you get a user interface that looks similar to the graphic in the split axis, but this is a controller you can change. Each axis is drawn as a tile, and this does not change even when you join the X and Y axis in the canvas settings. When you click on the tile, the drawing command is now drawn in this axis, and the axis ranges are adjusted accordingly.

**Exclude/Hide**

Every drawing command has two small check boxes in the top right corner. The difference is that Hide means that even if the graphic is not visible it affects the bounding ranges for the axes. One common use is to use the hide check box to toggle a plot or point on and off. Exclude can for example be used when you don't want to draw something, but might need it later.

Histograms and fits are still computed even if they are excluded, so you can use the results in text fields.
Drawing Commands

The previous chapter explained the building blocks for drawing commands, this chapter goes through some of the most frequently used drawing commands. Not all of the drawing commands are described here, but they are all described in the on-line help. To get to the help from within the application, open up the drawing command and click on the small ? button in the lower right corner. This brings up the page for that command in a small browser window.

The most commonly used drawing commands are Plot, Points and Bars. Plot can draw lines between the data points, while Points only draw the data points. Points is more flexible for drawing points since it can vary the size, color and marker type based on data values. Bars gives you the standard bar graph, side by side and stacked.

The Fit command is computes a fit function to a x,y data set. What is drawn is the fit function, but inside the command you also see additional statistics such as fit quality. You can pick from standard fit functions such as linear, polynomial, exponential and power. You can also specify your analytical function and have DataGraph optimize the parameters that you don’t specify.

The Function command allows you to draw an analytic function, where you specify the function as y = f(x). f(x) can depend on variables, so you can easily explore how functions depend on parameters as well as just overlay them on data.

There are a number of annotation commands. Label and Text allow you to add text descriptions to a data point. Range can highlight certain parts of the domain.

The Extra Axis command allows you to both customize the look of the axis more than the standard settings and also specify unit conversion and coordinate shifts.

Magnify allows you to very quickly include a zoom up of a part of the graph. This is similar to the Loupe tool, except becomes part of the graphic.

The Graphic command allows you to add a graphical element, vector graphic, bitmap and even QuickTime movies. This command is also the way to add in other graphs from the same file. This is explained more in the next chapter.

Histogram and Box are for statistical analysis of the data. This allows you to visualize and quantify distributions.

Lines and Connect can be used to annotate or as low level methods to draw data. You can draw a line connection from values that are typed in as well as coming from columns.

Plot

One of the main methods of visualization. You specify two columns, one for the x coordinates and one for the y coordinates. You can specify how DataGraph connects the coordinates and if it draws a marker at each coordinate. This is all accessible at the top of the Plot command. The line thickness is by default given by the pen variable that is defined in the top entry (style settings). You can change the default or change the line width for each plot.

Multiple plots in the same graph are done by creating multiple Plot drawing commands. There is a shortcut available for drawing a plot. Select two columns in the data table and hit the Plot command icon. This creates a plot with those columns.

The plot command connects points in the graph with lines. The x values don’t have to be increasing, but if there is a gap in the data or a row that can’t be understood, DataGraph breaks the line. You can tell it to connect the points over that gap by clicking in the “Bridge gaps in data” check box.
You can adjust how the coordinates are connected by changing the Connections menu in the command detail.

Below the connections option is the fill settings. This allows you to fill the area below the graph or between two graphs. The default is to fill in the y direction between the x,y graph and the x axis. But the fill style can be changed in several ways. Instead of the X axis you can select a column. If you fill in y, plot x,y columns and select a column y2 as the between column DataGraph uses the y2 column to create a plot (x,y2) and fills between the (x,y) plot and (x,y2) plot.

You can also fill the region between the plot an the y axis, or between a (x,y) and (x2,y).

Below the fill is the mask option. This allows you to mask out parts of the rows. This is a gap, just like if you leave the row blank, so if the “Bridge gaps in data” is not checked you might get a very spotty line between the points.

Below the mask is the label option. This allows you to label each point on the graph. The offset is either a single value or a coordinate. If the offset is a single value, DataGraph tries to figure out how to offset the number so that it won’t overlap the line. If you specify two values, this offset is used at each point. The default offset is point, which is a variable defined in the style settings.

At the bottom is the error bar selector. By default it is set to “No Error”, but there are several entries in that menu. When you select one of the entries, the Plot command gets larger and additional column selectors and options show up. You can select up to four columns. In both x and y if you select one column for the error, the error is the same in the positive and negative direction (symmetric). But if you select “Different” for the negative error you get an additional column selector so that the positive and negative error can be different. If a row is empty, no error bar is drawn. For example, in the graph above, the first row in the “neg y error” column is empty.

**Points**

The Points command only draws the points at x,y locations and does not connect the points like the Plot command does. Where the Point command differs from drawing points using the Plot command is that the Point command has a lot more control over the
The Plot command draws all of the points using the same color, fill, type and size. The Point command allows you to vary all of that based on additional columns.

The simplest way to create a scatter graph is to select the two columns you want to draw and hit the Points command. You can adjust the marker style using the menu in the top right corner. Below the marker menu is the drawing style. There are two size fields. The size of the marker and the size of the line if applicable. Note that the solid markers don’t have a line width, so the line width field is hidden along with the color of the line. The icon on the left is a preview of what the marker will look like. The size is in points, and by default the marker size is “point” and line thickness is “pen” which are values defined in the top entry, the Style settings. The first block in the detail settings allows you to change the marker based on additional columns. You can specify the size of the marker by using a column, and this is typically referred to as a bubble chart. When you specify a column, you get an additional controls which set the scaling from data units to pixels. You can also specify the size to be scaled by the area or diameter of the marker.

The Color and Fill menus allow you to vary the fill color and the line color based on a column. This functionality depends on a color scheme variable to set the color. Once you have selected a column you get a menu for “Scheme”. Here you select the color scheme you want to use. The last entry allows you to create a new color scheme. When you select that, a new entry is added to the variable list. Here you can add matching rules that are very similar to the mask mechanism. You can also use the small gear menu to the right of the name to ask DataGraph to suggest matching rules. You can set up multiple coloring schemes and select between them by using the color scheme menu in the Points command. You can use a color scheme for the line color and fill color.

For the marker style you can use a marker scheme that works similarly, but allows you to put a white circle for negative, half filled between 0 and 3, and solid above three.

Mask, label and error bars work just like they do for the Plot command.

Bars
Bar graphs are the standard way to represent categorical data. Consider the following data table where the Label column is a Text column with the label for each row of data. In each row, there is a pair of numbers.

If you select all three columns and click on the Bars command, the command is already set up to use the first column as the labels column and the next two columns as data columns.

The data columns are drawn next to each other at
each label so you can compare the values.

If no columns are selected when you create a Bars command, what is created is a command without column labels and set up for a single column. The bars are located at the numerical values 1, 2, 3, ... When the axis labels are set to Automatic, the Labels column is used for the axis.

Each bar has an entry at the top of the Bars command. To the right is a +/- button pair that allows you to add additional bars or remove a bar. You can select the entry (click on the small icon for example) and hit the delete key to remove or drag it around to reorder the bars. You can drag a column from the column list (far left) onto a Bars command to add a bar.

Inside the detail of each bar you can specify the error bars for a particular bar and a label column if you want to print the value of each column. The drawing representation is controlled by the first line in the command. First is the representation type and the second is how the bars or area is filled. There are three bar styles, standard (side by side), stacked and stacked area. You switch between the methods by selecting the type from the menu in the top left corner. This also allows you to orient the bars along the y axis.

The graphical representation can be further tweaked by using the bar width to the right of the bar type, and offset and space between bars that is located in the detailed view. These are in grid units.

The fill style is set through the Fill menu. There are four types of fill styles, and for each style you set the details further inside the drawing command. For the solid colors you set the color order, for patterns you set the background color, pattern size and line width etc.

At the bottom you set the drawing style for settings that are set for each bar. For example how error bars are drawn, where the label is drawn in each bar etc.

**Pivot**

Pivot is short for Pivot Table, and is a standard data analysis technique (see wikipedia). Pivot requires one to three input columns. One or two columns are used to specify categories and an optional column can be used to specify values. The categories are used to break the table down into blocks. Those blocks are
virtual tables that are indexed by one or two categories. They are virtual in that you don’t explicitly see them but rather just see digested information from them. If you don’t specify a value column the Pivot table will contain a count of the number of rows in each category. If you specify a value column, the table will contain the selected statistic from the corresponding rows in the values table (i.e., sum).

The values are displayed in a small table inside the drawing command. The drawing command can then present this data as a bar graph, stacked, side by side or stacked area. You can also pick whether or not it is displayed with bars in the x or y direction. You can also extract data in the form of columns and put the data back into the main data table. These columns are dynamic, so when the input data changes, the pivot columns change automatically. You can then use these columns as input for any other drawing commands.

To explain this further, let’s walk through a basic example. In the table on the right there are three columns, number and letter are text columns and value is a numerical column. If you select all three columns and create a Pivot command, those columns are automatically selected in the pivot command as rows/column and value. The default action is to select the “Sum” statistic. In the Pivot table, DataGraph broke the three columns into four groups. In each group, the rows have the same entry in the number and letter column and the four groups correspond to the four different combinations. For example, the combination number=first and letter=B contains three rows. The sum of those three rows is 5. If you don’t select a value column the calculation is just the count so the value will be 3.

Fit

One of the standard analysis methods is to compute a relationship between x and y variables in a data set. The fit command is where you do this in DataGraph. This command computes the fit, draws it and also gives you access to fit parameters and quality measurements. Even if you exclude the command, and therefore don’t draw it in the graph, DataGraph still computes the fit and allows you to use the variables in token fields anywhere in that graph.

The fit command works as follows. At the top of the command you select the x and y columns and what fit function to use. Linear, Quadratic and Cubic are polynomials, and Polynomial allows you to vary the polynomial order quickly. Of course the coefficients of a high order polynomial interpolation are relatively useless, but this is one way to smooth the data.

Exponential and logarithmic fit is not a straight least squares fit, in the sense that it is a least squares fit of the logarithm of the y values (linear function).

Sum means that you can type in a comma separated list of functions you want to use, and DataGraph finds the best linear combination of those functions. For example if you want a quadratic fit that goes through the origin, you can use the functions x and x^2. For a more general fit, use the Arbitrary fit option.

The Arbitrary fit allows you to specify a function with unknown parameters and DataGraph computes the parameters to minimize the difference between the function and the data. Mathematically, this is called generalized least squares and allows you to specify a non-linear function. For example, if you specify the function a/(b+x), and a and b have not been defined as
variables, DataGraph finds the constants a and b that give the best fit. If a is defined as a variable, DataGraph will find the best value for the variable b.

When you have a linear function, there is a unique solution, but for non-linear functions you can have multiple solutions. The iterative process used to find the fit and might find one of the solutions, or not converge at all. To address both of these issues, DataGraph allows you to specify an initial guess for each parameter. The default guess is 1, but you can vary this guess by typing in a new value or by using the pop-up slider.

The LOESS fit is a non-parametrized fit. That means that the fitting function is not an analytical function, but is rather computed by a lower order fit at each data point. Use Wikipedia to get a more detailed description, but the short explanation is that at each point on the interval, DataGraph computes a lower order least squares fit in that neighborhood and uses the value of that fit as the function value. You can adjust the interval width, and select if the local fit is linear or quadratic. The smaller the interval, the noisier the fit function and there might not be enough points to do a local fit.

By default all the rows are used to compute a fit. Rows that are invalid or empty are ignored. You can select a sub-set using two methods or a combination. The first is the “Fit X range” option. The default is Everything, but if you select Specified... you can type in a range for the x values. This will restrict the fit based on a range along the x axis. You can also use the mask mechanism. This allows you to base the selection on other columns.

The fit works by minimizing a term of the form
\[ \sum (y_i - f(x_i))^2 \]
where \( f \) depends on the parameters that you want to optimize. The exception is the exponential and logarithmic fit, where the routine minimizes
\[ \sum (\log(y_i) - \log(f(x_i)))^2 \]
You can specify a weight column, in which case each term is weighted by the corresponding row in the weight column, that is the term that is minimized is:
\[ \sum (w_i (y_i - f(x_i)))^2 \]
The effect this has is that where the weight is larger relative to other parts, the error is smaller.

It is easy to display the results of the fit function in labels, legends etc. In any token field, you can select a result from any of the fit functions. This is updated automatically when the fit changes.

If you want to know the residuals, or which rows were used for the fit, use the “Export Column” menu. There you can copy the values as a column or append the values as a column. If you pick the “Append” entries, DataGraph creates a new column which is then connected to the fit drawing command and updates...
automatically. You can also create this column manually by using the "Other" menu in the column list.

Histogram
The Histogram is used to analyze statistics for a list of values.

In the default setting, it takes a column of numbers (Values menu) determines a bin size and then counts how many values are in each bin. This is then represented graphically based on what histogram type is selected. For example, Bars is the standard view and gives a result similar to the command icon. The default option is Stairs, which does not draw individual bars but instead just the tops, and approximates the probability density function. You can draw the histogram along the x axis or y axis. If you draw it along the x axis, the bins are drawn along the x axis, and you can choose what unit the y values have.

Count means that the y values are how many numbers are in each bin.
Density means the value is count/bin width and probability means count/(width*total count). This is best understood if you consider the integral of the curve. For Density the total integral is the number of values. For Probability the total integral (area) is one.

The Type menu allows you to select how to draw the histogram. A very common method is Bars, but Stairs is useful when you have a large number of bins, and is the default setting for this reason. Left/Right will allow you to compare two distributions, one command will draw the histogram on the left side of each bin, the other on the right side of each bin. The smooth option adds together Gaussians. You can adjust the width of the Gaussians using a slider or by specifying the value exactly.

The histogram command also computes a number of basic statistics for the list such as median, mean, standard deviation etc. These numbers are displayed in a table inside the drawing command, but can also be used in a label.

If you want to compute a statistic for a sub-set of the rows you can use the mask command. For example, consider an example where you have values and genders (M or F). The data consists of two columns “Value” and “Gender” and a number of rows. Select the Gender column as the mask column and include if it matches the string M. This only computes the histogram of the corresponding rows in Values.

It is possible to draw lines at particular points in the distribution. For example if you want to draw lines where the median is, or where you have 10% or 90% of the values to the left, you can use the Lines at field. By default this field is empty, but you can type in a list of numbers between 0 and 1. For convenience the variable % is defined so 50% is the same as 50*0.01 = 0.5. So you can put 0.1, 0.5, 0.9 or 10%, 50%, 90%. This works for all histogram types, even the cumulative density option which is selected to the right of the units menu. The Clip check box determines that the line should be clipped by the distribution shape. This is what the lower graph does in the figure above. The default line style for these lines is the same as the line
for the distribution, but you can specify a different line style.

Box

Box plots, also called Box and Whiskers (see wikipedia), are a way to describe a distribution of numbers, and provide a graphical way to compare two distributions. In the simplest case, you take a column of numbers such as 3, 4, 8, 10, 10, 12, and 17. Select the column and click “Box” to get the graph on the right. This shows five aspects of the distribution: the minima, the maxima, the median, and the first and third quartile.

A more interesting example is when you have two columns that contain multiple populations such as the data set on the left. One column is a text column with labels, the other is a numerical column. Select “Value” for the values column and “Treatment” for the Position column, as shown below.

The plot automatically groups each treatment separately and uses them as the categories.

Outliers are drawn as dots. An outlier is a value that is larger than 1.5 times the Inner Quartile Range (IQR) from the median. IQR is the difference between the first and third quartile. The whiskers are only drawn to the smallest/largest non-outlier.

Inside the command detail, you can adjust the style of the whiskers and pick them as the minima and maxima, instead of IQR, to not draw the outliers as shown in the following graph.

You can also control which parts are binned and how they are presented. You can use the mask to select a sub set (see mask description) of data. You can also specify the bins, and name the bins by using one or two columns.

On the left, there are two columns. The first shows the two treatment labels, the second, the name you want to use for that label. Below the position column there is a string column. Instead of the default (All), you can specify the bins you want to use. To the right of that you can select the display name. By default this is “same” but you can specify a different column. This gives the following graph with only two categories.

Range

Range can be used to highlight a coordinate region in x and/or y. It is also possible to use it as a basic building block. In the simplest case you specify a single interval in either the x or y direction. You can drag the edges of the interval with the mouse. As with all numerical fields, you can use variables when specifying the range. You can specify the range in both x and y directions, so you can use this to draw rectangles, and...
semi- or in-definite rectangles by specifying one or both ends to be infinity.

There are also other ways to specify intervals, and this is selected in the menu on the left side. Alternates for example means that you can color the intervals [0,1], [2,3], [4,5],... without having to use multiple range commands. The Dates option allows you to specify alternating fills based on dates.

For full flexibility use the Columns option. This allows you to specify columns for the start and end of each interval. Each row gives you an interval. A slightly advanced use, is to use the clipping action to fill a portion of a plot. This is done by using two drawing commands. One draws the plot, the other draws the region but clips the result with the output of the plot command. This is done by selecting the command from the “Clip with” menu at the bottom of the Range command.

Label
The Label command allows you to point out a particular point on a graph.

When you create a new Label command, it defaults to an arrow that points to the center of the axis region. You can drag the arrow around the screen, and you can drag the label and keep the same point.

There are three parts to a label. First one is where the anchor point is. This is the point that the arrow is pointing to. There are several different places to base an anchor. The default one is Inside. This is a coordinate in x and y. For X axis, Above and Below you specify the x value as a coordinate, but the y as an offset. You can of course drag the Inside arrow up to the x axis, but the difference is clear when you resize the graph or the data changes, since the inside point has a y coordinate and that coordinate might not work for different data or scaling. For example if you want to point to a particular x location it is better to use the X axis option. Below and X axis are the same if you don’t have multiple y axes. Y axis left and right are similar, but there you specify the y value as a coordinate and the x as an offset in pixels. You can drag these arrows by using the mouse, but arrows along the x or y axis are restricted to only move in that coordinate location, and you need to adjust the offset by entering in a numerical value or using the pop-up slider.

The next step is the arrow going from the anchor point to the label point. The size of that arrow is specified to the right of the anchor point. You can change this by dragging the label. Note that if you use the size (0,0) or select At Start for the Inside label the arrow is not drawn. This part will depend on which position you are using. For example, when you select Below as the position, the
coordinate is now the x coordinate and the y pixel offset and the end offset are given in pixels.

You can also control the style of the arrow. Apart from line thickness and color you can specify the line style. You can for example hide the line, make a straight line or angular/curved arrows. You can also adjust the arrow head. If you don't want an arrow at all, select the top entry for the line style and the last entry for the arrow style.

Text
The Text command is an annotation tool that is, in some ways, similar to the Label command, but has additional functionality when it comes to aligning multiple lines. The Where menu specifies in which region of the graph this label is positioned. The Inside option allows you to specify where the label is relative to a corner of the graph. For example if the Offset is “2cm,1cm” and anchor is upper left, then the label is always that far from the top left corner even as you resize the graph.

The text lines allow you to insert references to variables and values inside drawing commands. This for example allows you to create a table of results, where the values are pulled from fit functions or histograms, even if those commands are excluded from the graph.

One use of the Text command is to create more flexible titles for graphs. The standard axis title will always center the label for the x or y axis and rotate the y title. If you want more flexibility, create a Text command and use that to position the text.

Graphic
This command allows you to add a graphic to the figure. The graphic can either be pasted into a graph window, or can come from another graph. Once the graph is inside the command you can position and scale it. Some examples of what you can use this command for are

- Add a logo in a corner of a graph
- Add an equation that is done in LaTeX or some other equation editor and pasted in. DataGraph does support expressions, but doesn't cover as much as TeX does.
- Stack graphs into a figure. Each graph can be done in a separate graph in DataGraph and remain dynamic, or just pasted.

- Scale bitmaps. Applications like TextEdit don’t have a lot of control over the output size, and DataGraph can be used to scale a bitmap to an exact size.

There are two different ways to specify a graphic. This is selected in a menu inside the command, the Source menu. The Imported option means that the image needs to be copied from a file or come from the clipboard. The Graph option is for specifying a graph in the same DataGraph file.

If the clipboard contains a figure, the paste option is enabled. If you select this option, the clipboard contents overwrites the current figure content. Note that if you have a figure in the clipboard and paste it, DataGraph creates a Graphic command for it. This menu also allows you copy the graphic back into the clipboard. This is useful since DataGraph retains any LinkBack (http://linkbackproject.org) information so for some applications such as LaTeX you can paste it back into the application and continue to edit it.

If you have a graphic file, you can click on the Pick... button and select it, or drag the file to the command list to create a Graphic command for that file. If you drag a QuickTime movie, you also get a field where you can select which time value (in seconds) you want to display.

Creating multiple graphs is explained in the next chapter, but the short version is that a single DataGraph file can have multiple graphs that use the same data set. The graphs show up at the top of the file window, and if you select Graph from the Source menu, you see the same list of miniatures inside the Graphic command. Click on a graph to select it. You can not select the current graph or any graph that directly or indirectly depends on this graph since that would cause an infinite recursion.

The graphic can be positioned relative to different parts of the graph, just like the Label and Text commands. The Graph positions the graphic relative to one of the corners of the current axis. The Coordinate positions the graph at a particular x,y location and this means that the graph is positioned at a particular coordinate in space, and if you crop the axes it might not be visible. You can also position the graph in the title or along the axes, for example if you want a graphical label instead of a text label. The Figure option means that the placement is relative to the overall figure, and therefore not dependent on an axis. Which option you pick really should depend on what should happen as the data changes and you resize the graph. The second line will change depending on what is selected in the Where menu. For example, for the Graph option you pick which corner the graphic should be anchored to and what the offset should be. When you pick the x title, you can specify the x coordinate and the offset in the y direction.

Legend
This is the standard way to give a visual key to label lines, fills etc. The legend name is actually specified in the drawing command, typically at the bottom. This
name could be specified exactly or depend on the column name or result of fit functions etc. The legend command then picks up the names from the command and draws them. You can adjust the drawing style, how many rows to use and where to position the legend. You can for example put the legend above the graph or to the right. If you have a second legend it will draw exactly the same entries unless you use the “From axis” menu. The legend is just like other drawing commands, when you have a split menu, you select which x and y sub-axis it belongs to. This menu allows you specify that instead of using the legend strings from all drawing commands, it only uses the legend strings from the drawing commands that belong to the same axis. This way for example you can have a split x axis and two legends, one for the lines on the left, another for the lines on the right.

The Bars command specifies the legend strings at the top, to the right of the values column. The default is to use the name of the selected column, but you can change that. Note that each bar where the legend is not empty will give you an entry in the legend. If you draw the same data twice and the legend is only drawn once, and the two legend icons are drawn on top of each other.

Magnify
The loupe tool allows you to inspect details of the graph without changing the axis ranges. The Magnify command allows you to include that type of view in the output.

There are two uses for this command. The X/Y range specifies the range for the insert. If this range includes the current cropping area, you are not really magnifying the area in the inset, but displaying a larger area and the current axis is the detailed. To reflect that, DataGraph shows the current cropping area in the inset. You can drag this area, and what that does is to change the current cropping rectangle for the axis (x and y coordinates).

The more common use is where the X/Y range is inside the current axis bounds. In both cases the range of the inset axis is given by the X/Y ranges, but now the box is drawn in the main axis as well. Lines are used to connect these areas visually. When you are magnifying a region (case w) you can change the X/Y ranges using the mouse by dragging the
source rectangle. You change the position and size of the inset by dragging it around using the mouse or by adjusting the numerical fields for size. The size is in pixels, but you can use units such as in, cm, mm. The anchor is one of the four corners of the axis, just like the Graphic and Text command.

You can adjust the drawing style inside the drawing command. For example, if you want to draw the box, connecting line, if the tick marks should be drawn etc.

Most commands have a small magnify selector in the top right corner of the detailed view. This selects how the command interacts with the magnify command. The entry on the far left, where the destination and source are both solid blue means that this command is drawn both in the standard view and magnified view. The next means that it is not drawn in the magnified view, i.e. it is not magnified. The one on the right where only the magnified view is solid blue means that the command is not drawn unless the area is magnified.

**Function**

This allows you to specify an analytic function and a range where it should be drawn. The function needs to be a function of \( x \). If you use additional variable names they have to specified as global variables. But doing that allows you to tie these variables to sliders and change them interactively.

You can also specify a “From \( y(x) \)” function. This allows you to fill between two analytic functions.
Graphs

DataGraph allows you to create multiple graphs from the same data set, and create composite graphs. To create a new graph, click on the “Add Graph” button that is in the toolbar. This adds a new empty graph to the DataGraph file. If you hold down the option key you get a duplicate of the current graph. You can also use the New Graph sub menu in the File menu to create the graph.

At the top of the window you get a horizontal list of the current graphs. Each graph is updated in real time so as you change the data you see the changes in all of the graphs. For graphs that do not have a specified size, the size is given by the size of the current drawing view. As you change this view, either by changing the window or changing the splitters, the other graphs are updated.

When you move the mouse over a graph thumbnail you will notice a few controls. On the top left side you see a button to delete this graph (undo if you accidentally hit this.) Below that is the clone graph button and the button to create an empty graph. At the top is a text field so you can name the graph. This is partly for organization, but also to be able to export multiple graphs at the same time (explained in the next section). If you don’t specify a name the “Untitled” tag will vanish when you move your mouse away. In the top right is a the flag check box. This is also currently only used for exporting graphs.

You can also drag the thumbnails to reorder the graphs.

You can also drag a command onto a thumbnail to add a copy of the command to that graph. Note also that you can select the style, canvas and axis and copy them to the clipboard. Then you can go to a new graph and paste in the settings. This is one way to propagate style changes to other graphs.

Combining graphs

The split axis allows you to create multiple axis which share an x/y range. But in some cases you want to put two graphs next to each other. To do this in DataGraph, create each graph as a separate graph page, and then create an additional graph to combine them.

This takes a little bit of tweaking, since by default a new graph is set up to draw a graph so it has an axis. The “For Composition” option in the New Graph menu does the following three steps.

**Step 1**: Set the axis style to “Only Axis”. This stops you from drawing the box around the graph.

**Step 2**: Turn off numbers for the x and y axis in the axis settings. This removes the remaining two lines.

**Step 3**: Set the margins around the axis to 0. This means that the underlying axis fills the entire graph region.
There is still an axis in the graph, and if you draw any plots it will show up. But for compositing other graphs, all you need are a number of Graphic commands. The Graphic command is then set up so that the source is one of the other Graphs in the file. Use the offset field to position them. It is recommended that you specify the size for all of the graphs, including the composite graphs so that the positioning and size of the graphs is more predictable. This also allows you to align the graphs. You align them along the boundary of the drawing canvas, but if the margin computations are the same the axes are aligned as well.

**Example**

Take as an example a case where you want to draw three graphs in a table. If they need to have the same y axis range, you might want to use the split axis functionality, but if they are truly independent using three graphs is the way to do this.

First step is to create the graphs that should be composed as two separate graphs in the same DataGraph file. By default a graph fills the lower right corner of the DataGraph window and will resize as you change the window size. For compositing graphs this is not helpful, so specify the graph sizes exactly by using the Specified... option in the canvas. This will set the canvas size exactly. If the result is going to be printed the graph size will be surprisingly small on the screen so set the magnification level so that you can view things either in actual magnification (same size as printed) or in 2x,4x view.

In this example the first graph has size 4x4 inches, and the other two are 4x2 inches. This means that the composite graph will be 4 inches high and 8 inches wide.

**Step 1:** Create a composite graph from the File menu.
**Step 2:** Set the canvas size to 8in,4in
**Step 3:** There is one Graphic command already in the list, and select Source as “Graph” and click on the graph that you want to display.
**Step 4:** Set the Anchor to lower left and the offset to 0,0. The graph command will look like this:

![Graph command example](image)

**Step 5:** Now create two more drawing commands. Set the anchor to lower left but in this case set the offset to “4in,0” and “4in,2in”
If the drawing commands have the same font sizes axis padding the axis will align exactly. If you change the style in one of the graphs, you will most likely want to copy that style over to the other graphs. To make this quick, go to to one of the graphs, select the Style settings (top entry), select Copy and then go to the other graphs and select paste. You can toggle between the graphs by using the left and right arrow keys while holding down the command key.
Export

Copy
In the edit menu you can select from three copy options. The top option copies the figure as a pdf and bitmap, allowing the program you paste into to pick the preferred format. Programs such as KeyNote will use the pdf version. You can also copy the figure as bitmap only, with or without transparency. You can also use the context menu (right click/control-click) to copy the figure. You can also copy a particular drawing element from the graph.

Export to a file
In the File menu you can export the graph as a file. You can export the graph using multiple formats. For bitmaps you can also specify the resolution. The default (screen resolution) is 72dpi. Note that the EPS file format does not support semi-transparent objects. What DataGraph does is to automatically replace the transparent color by an opaque one that will show up the same based on a white background. PDF does not however support semi-transparent gradients, and DataGraph automatically adjusts the gradient colors so that they will look the same on a white background. This is done since otherwise the EPS/PDF file that gets generated can not be opened. The SVG format is a vector graphics format that is intended for web pages, and is an xml file. The limitation here is that you can not use an external figure (using the Graphic command) or a textured fill since that is implemented with a bitmap. DTG is a graphic file that is unique to DataGraph and DataTank. This format keeps all of the information, since it is just a file version of the graphic that DataGraph uses internally. The main use of this option is if you want to save a graphic that you intend to include in a different DataGraph file.

The bitmaps don’t have any of these limitations, but for high resolutions they will take a lot more disk space than the vector graphic version. PNG and TIFF can be semi-transparent, which is typically called an alpha channel. Note that even though Mac applications tend to all handle alpha channels properly, some applications do not.

Exporting multiple graphs
The standard Export graph menu only exports the selected graph. If you name and flag the graphs you can then export all of them at the same time by using the “Export Flagged Graphs...” option. This brings up a dialog box to ask you for the folder where the images should be saved.

Movies
You can set up animations in DataGraph. This is done through a special variable. By default the animation variable is called “t” and can be used in any numerical field such as cropping regions, line thicknesses, masking...
rules etc. This is set up as a slider, so you can set the range and if it is restricted to be an integer. Then you set the duration in seconds and can click the small play button to animate the variable on screen. Once you have set up an animation that you like, click the small QuickTime button above the play button or use the File menu. Then select the movie format and frame rate. The movie can be included in a KeyNote presentation or put on a web page.
Command Line Utility

DataGraph is typically used interactively, but when it gets to the point where you feel like you are repeating the action paste, export, paste, export over and over, it might be a good time to look into using either Automator or the command line to simplify that task. DataGraph comes with a command line utility called dgraph. This utility is actually stored inside the DataGraph application. If you are on the command line and type

cd /Applications/DataGraph.app

ls

You see that there is a small executable file called dgraph. This executable is a command line tool. It is crucial that it stays there, so do not copy it or move it. When dgraph runs it checks the location to see where the DataGraph application is located, since DataGraph does not have to be installed in the Applications folder. You can run the command line from there, but typically you set a symbolic link and use that instead. If you want the executable to be in /usr/local/bin, do

ln -s dgraph /usr/local/bin/dgraph

This assumes that you are inside the DataGraph.app wrapper.

Once you have this set up, you can call dgraph from the command line. There are three crucial parts to the argument list. The first is the DataGraph file that you want to use as a template. The second is the data file that you want to import into the table. The third is the output that you want to create.

dgraph action.dgraph file.txt out.pdf

Note that typically one or more of those files will have full path names, so the argument list will be a lot longer. What this does is to open action.dgraph and then overwrite the table with file.txt. This is done non-graphically, but what happens is exactly the same as if you select all of the visible columns in the table and then import file.txt. The output graph is saved in out.pdf. Only the current figure (the one selected when you saved the DataGraph file) is saved.

You can overwrite variables, change sizes, set compression ratios, etc by using command line arguments. To get further explanation, use the -help flag, i.e. dgraph -help