Astronomical Data Analysis Software and Systems VI
ASP Conference Series, Vol. 125, 1997
Gareth Hunt and H. E. Payne, eds.

Dynamic Dynamic Queries (DDQ)

Peter Teuben

Astronomy Department, University of Maryland, College Park

Abstract. We describe an implementation of Dynamic Queries (DQ), a recently developed method as an alternative to the perceived complicated SQL-type operations on relational databases by a very intuitive graphical interface.

This prototype implementation, written within the NEMO package, uses ASCII sliders and the PGPLOT graphics interface and uses a novel interactive analysis back-end tool that is dynamically associated with the query. We call this technique Dynamic Dynamic Queries (DDQ); it can be easily implemented in other existing table manipulation applications.

1. Introduction

Dynamic Queries (DQ) is a recently developed querying technique (Shneiderman 1993) which proved to be a good alternative to the often difficult to learn but very general and flexible SQL-type interface to a relational database. The query in DQ is formulated using a number of graphical widgets that represent a column from a table (columns can be numeric as well as textual). Using two sliders a range search can be set for any column, and the resulting view of this multiply **and**-ed selection is dynamically displayed in a canvas. The canvas can display the data as simple points, but additional information can optionally be added to these points (color, size, etc.). Once sufficient selections have been made, some implementations of DQ then allow the displayed data to be queried in more detail by, e.g., clicking on the point. A number of specific examples of DQ have been discussed in recent years, in particular paying attention to user interface issues.

This paper describes a generic (ASCII-widget) implementation where the input dataset can be an arbitrary table-like dataset. A selection of the viewing coordinates X and Y is made (as a virtual column computed from the existing columns) as well as the columns that are allowed to be be queried. In addition to the above described features, we also implemented a generic (user definable) interactive analysis tool that is dynamically associated with the query. We call this technique Dynamic Dynamic Queries (DDQ).

2. Data Structures

The input dataset has M parameters (columns) for given N observations (rows). Sticking to the two-dimensional aspect of a display screen (see Swayne et al. 1991

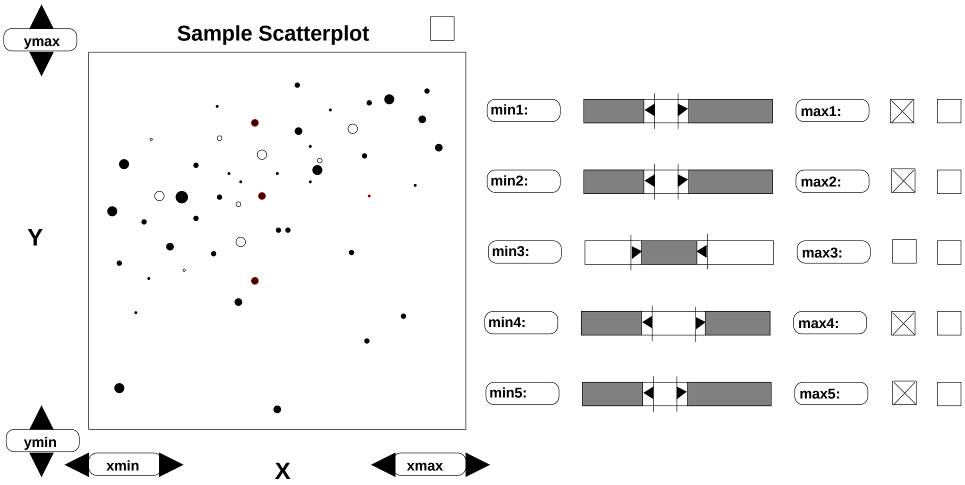

Figure 1. Overall layout of a DDQ GUI tool; sliders on the right (see also Fig. 2), the dynamic view of the selected X and Y coordinates in a scatter plot on the left. The X-Y viewport can be controlled by the xmin/xmax and ymin/ymax buttons.

for an alternative view[1] of these multi-variate data using the XGobi program), we choose two virtual columns (mathematical expressions derived from any of the M parameters) for display and any of the M columns as sliders for DQ interaction.

3. Implementation

The current toy version is implemented in the NEMO[2] package under the name `tabview`. The graphics interface used was that of PGPLOT, although being flexible, a little slow for this kind of application. In addition, rapid querying of large databases can be optimized with more elaborate database structures, such as the multilist, the grid file, k-d tree and the quad tree.

3.1. Startup

At startup a number of sliders (columns) are selected for DQ interaction, and the program overviews simple statistics and all their possible correlations are computed. A sample output with 6 sliders would be:

```
    % tabview in=p100.list xcol=%1 ycol=%2

Reading table p100.list with 6 columns and 100 rows

### Overview statistics:
Slider    : Npts      min      max      mean    sigma skewness kurtosis
1         :  100  -7.1859   5.7947  1.43e-07   1.2811  -1.1083   12.732
```

[1] http://lib.stat.cmu.edu/general/XGobi/

[2] http://www.astro.umd.edu/nemo/

```
2        :  100  -2.3444   7.3121 -5.83e-08   1.1374   3.5281   19.521
3        :  100  -3.4209   5.2131 -1.97e-07   1.1072   1.0584   5.8222
4        :  100 -0.78246   1.1218 -4.76e-08   0.3856   0.3441  -0.1828
5        :  100 -0.95056   1.0854  7.13e-08   0.3906  -0.0894  -0.1902
6        :  100  -1.1176   1.0182  4.87e-08   0.3763  -0.0878   0.3303

### Pearson correlation matrix:
   :      1     2     3     4     5     6
 1 :   1.00
 2 :   0.22  1.00
 3 :   0.54  0.14  1.00
 4 :  -0.07 -0.03 -0.10  1.00
 5 :   0.07  0.02 -0.06  0.06  1.00
 6 :  -0.12 -0.07 -0.08  0.02 -0.06  1.00
Xvar %1: displayed min= -7.51038 max= 6.11921
Yvar %2: displayed min= -2.58581 max= 7.55352
```

The prompt in "command-line mode" is that of the currently activated slider:

```
Slider 1 [-7.18587 5.7947 0 5.7947]:
```

with the following commands implemented:

```
l <num>     modify 'lo' slider
h <num>     modify 'hi' slider
l s <step>  set step in 'lo' slider
h s <step>  set step in 'hi' slider
b <step>    step both 'lo' and 'hi' slider
+/-         change stepping sign/direction
<digit>     change slider to interact with
r           reset lo/hi to min/max for this slider
s           show min/lo/hi/max for all sliders
u           update screen new
f           interactive cursor based flagging of points
o <file>    output table, w/ optional override out= filename
x           swap lo and hi (invert logic)
q           quit
!cmd        execute a shell command 'cmd'
|cmd        pipe visible data as ASCII table to 'cmd'
?           this help plus status
```

3.2. Post-Processing

At each step of the interaction the visible data (or its complement) can be processed. `tabview` implements this by manually piping the resulting data through a tool of the user's choice, but one can imagine this tool to be dynamically associated with the drawing engine, in which case it brings a whole new dimension to the exploration of this kind of data. Here is a sample session:

```
Slider 2 *[1 5.5 7 7]: |tabhist - 2 xmin=0 xmax=24 yapp=1/xs
[1090 data piped]
1090 values read
min and max value in column 2: [74 : 509]
Number of points     : 1090
Mean and dispersion  : 305.88 122.864
Skewness and kurtosis: -0.537889 -0.920467
```

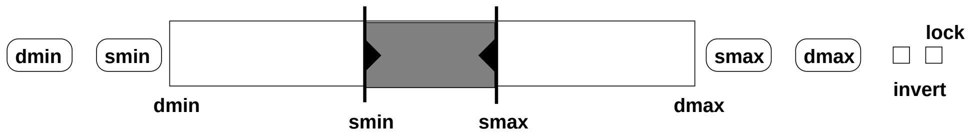

Figure 2. An expanded view of the slider widget that would be needed. The four buttons "dmin," "dmax," "smin," "smax" on either side of the slider can also be edited. The lock and invert check buttons on the right determine the way the slider moves (lock will fix the width (smax−smin) of the slider) and if the inside or outside of the slider is selected (greyed).

4. Conclusions

A definite advantage of DQ is its fast exploration capability of a multi-dimensional database (see also XGobi). Users can quickly discover which regions of this $M \times N$-dimensional database are densely and sparsely populated, find out which variables correlate, and how they depend on other variables in the dataset etc. With added analysis tools DQ will be a very powerful visualization and analysis tool.

An often quoted disadvantage of DQ has been its poor match of current hardware and software, although new implementations by IVEE (see also their excellent Spotfire Java demo[3]) prove that this may be a thing of the past. DQ needs a rapid search algorithm and display strategy (screens must be updated within 100ms in order to satisfy users). To the software side we can add the work required to integrate DQ into existing database systems, although this paper attempts to remedy this somewhat.

A second disadvantage is the limitation of range queries on numeric values. The HCI Lab. at the University of Maryland has experimented with a filter/flow metaphor which then provides full boolean functionality, albeit at a great cost of users to comprehend the system. This is a field in development.

Acknowledgments. We would like to acknowledge the HCIL at the University of Maryland for their inspiring work, and also Erik Wistrand (IVEE Development AB) for the use of their sample graphics during the poster session.

References

Shneiderman, B. 1993, IEEE Software, 11, 70 (DQ)

Swayne, D., Cook, D., & Buja, A. 1991, User's Manual for XGobi, Bellcore Technical Memorandum (XGobi)

Teuben, P. J. 1995, in ASP Conf. Ser., Vol. 77, Astronomical Data Analysis Software and Systems IV, ed. R. A. Shaw, H. E. Payne, & J. J. E. Hayes (San Francisco: ASP), 398 (NEMO)

[3]http://www.ivee.com