# Data display specification

## Principles

This is a specification for data display code that works with QCoDeS. These are the general principles I think it should follow:

1. It should be able to read from a QCoDeS database being used in an experiment.
2. It needs to be at least as easy to learn as QCoDeS.
3. It needs to be possible to make it run automatically every time we run doNd()
4. It should build as much as possible on an existing solution, so that we
   1. Save work
   2. Can piggy-back off existing documentation and examples
   3. Benefits from future development.

This solution could be a Python library, or a data analysis program that we have access to such as Matlab or Origin.

1. Our focus should be on display, with basic analysis as a bonus.

## Commands we need

### Basic display and organisation

**Show\_sweeps[[1]](#footnote-1):**

(Similar to showwaves() in Igor)

Plots all sweeps**[[2]](#footnote-2)** from a list argument, each in its own window and live-updating, from a QCoDeS database. The plotted sweeps should look something like this:



The program should automatically recognise appropriate axis labels and ranges from the database; in future it should be able to make more complicated plots when required (such as X vs Y, or plots in polar coordinates or containing error bars).

Suggestion: allow for wildcards in the list argument.

**Show\_num:**

(Similar to shownum())

Plot all sweeps from a given run number (or a list of numbers)

**Append\_sweeps:**

(Similar to appendwaves())

Append sweeps from a list argument to the topmost graph.

**Append\_num:**

(Similar to appendnum())

Append sweep with a given number (or list of numbers) to the topmost graph. The command should be selective about what sweeps it appends; for example it might do the following:

1. Sweeps of the same number of dimensions, independent parameter(s), and dependent parameter;  
   and if none, then:
2. Sweeps of the same number of dimensions and independent parameter(s)  
   and if none, then:
3. Sweeps of the same number of dimensions  
   and if none, then:
4. Any sweep with the given number.

### Basic plot adjustments

**Set\_range:**

Sets range on X, Y, and Z axes. Includes an option to autoscale.

**Trace\_color:**

Set colour on a given sweep (or list of sweeps, or all sweeps) on the top graph.

Suggestion: include an option to colour sweeps individually in a way that makes them distinguishable.

**Suggestion: Other commands to set line style and thickness**

**Map\_color:**

Set colour on a given heatmap (or list of maps, or all maps, on the top graph).

### Basic interaction with a graph

**Drag and zoom**

Obviously highly desirable

**Color scale adjustment**

It would be good to be able to do this using sliders, ideally with gamma control as SpyView does.

**Cursors**

Read off individual data points (and if possible, differences between data points) using some combination of mouse and arrow keys.

### Export

Export to vector PDF and PNG format. (+svg)

### Other useful things that Igor does that might be nice

These verge into being cosmetic improvements or analysis, and might be better done in a dedicated environment such as Matlab or Origin. However, it’s also nice to do them on the fly:

**1 D cuts through 2D data[[3]](#footnote-3)**

**Collapse a 2D sweep down to its average along X or Y axis**

**Differentiate, smooth, subtract linear fit (based on a rectangular region), subtract the average of each column/row, or normalise each column row, in a plot**

**Calculate the average (and standard error?) of a region of plot**

1. Feel free to use different names for the commands. [↑](#footnote-ref-1)
2. I suggest we call each dataset, i.e. a measurement of variable V vs x,y,z. etc., a sweep. (Igor calls them waves, but this might be confusing.) [↑](#footnote-ref-2)
3. I can show you how this works in Igor. [↑](#footnote-ref-3)