Introduction

The Gumtree scripting interface has a numpy-like Python library that makes data treatment easier. It also has a GUI library that automatically generates control panels for scripts. Same scripts can be loaded in both the desktop application and the web service application for online data treatment. The scripting interface benefits both the users and the developers. Users can easily make scripts to run experiments or treat the data, with a graphical interface automatically created for these scripts. Developers save a lot of time in compiling and deploying products when using this feature.

Features

- Scientific Computing Library: More than numpy
  - multi-dimensional array object
  - mathematical operations
  - error propagation
  - carrying axes information
- Library to Read Data from Different Sources
- Plotting Library
  - 1-D and 2-D plotting
  - real-time data plotting
  - fitting support
- Instrument Control Module
- Auto-GUI Support
  - generic GUI interface
  - desktop implementation
  - web service implementation

Scripting Can Be Used to

- Treat Experiment Data
  - preview data and send feedback to experiment
  - treat data on a local computer or on the server
- Align Instrument
- Create Quick Experiment Interfaces
- Make Highly Customisable Features

Sample Code - Arrays

```python
def setup_script():
    # create a 2D array, with dimensions: 3x4
    x = arange(0, 90.0, 90.0/12, [3, 4])
    # multiply a random array
    y = rand([3, 4])
    # print the array, with squares of errors and axes
    print x
```

Sample Code - Auto GUI

```python
def auto_gui():
    # set UI label of the control panel
    __script__.title = 'Kowari Reduction'
    # set the number of columns of the control panel
    __script__.numColumns = 2
    # create a new group: a collection of parameters
    g_jump = Group('Jump to Scan Index')
    g_region = Group('Region Selection')
    g_efficiency = Group('Efficiency Correction')
    # set the height of the group
    g_region.rowspan = 3
    # set the number of columns in the group
    g_efficiency.numColumns = 2
    # create a new parameter, giving type, initial value, and command to run when changing the value
    ind_jump = Par('int', -1, options = [], command = 'jump_to_index()')
    var_jump = Par('float', float('NAN'), [], command = 'jump_to_var()')
    # set the UI label of the parameter
    ind_jump.title = 'select data index'
    var_jump.title = 'select scan variable'
    # add parameters to a group
    g_jump.add(ind_jump, var_jump)
```

Server Side Scripting with Browser Interface

```python
def server_side():
    # set the title of the parameters
    ind_jump.title = 'select data index'
    var_jump.title = 'select scan variable'
    # add parameters to a group
    g_jump.add(ind_jump, var_jump)
```