CCD Characterization Software

A development based on Python

Nicolás Haddad

European Organization for Astronomical Research in the Southern Hemisphere

Paranal Observatory

Summary:

*A CCD characterization software written in Python and using some well supported modules, like **PyFITS**, **pylab** and **NumPy** has been developed to allow our Engineers perform fast and interactive analysis of CCD images. The software can be installed in various platforms, like Windows, Linux and MacOS and has the big advantage over other software tools of being free.*

This is an ongoing project, and up to now gain, linearity, CTE, row and column average and plotting, image statistic and plotting, image median stacking and image addition, subtraction and division have been implemented.

Introduction

Software packages like Matlab and IDL with a well established reputation in the scientific community have been used for a long time and provide many routines to perform data reduction on CCDs. The main disadvantage of these packages is their high cost which discourages its installation in multi-CPU environments.

Since some time we are looking for an open source alternative that can be multi-platform, easy to learn, with Object Oriented Programming capabilities that allows our Instrumentation engineers to analyze in situ the CCD images either for normal system testing or whenever there is a suspicion that something can be wrong on our detectors.

Python is a general-purpose high-level programming language which emphasizes code readability and clear syntax. It allows both object oriented or structured programming. A very important strength of this language is an active and strong community of developers that is producing a large amount of modules (software extensions).This new application, which has been baptized CTK (CCD Tool Kit), make use of the PyFITS, NumPy and matlibplot modules.

Ipython is used for the interactive shell terminal. It offers some additional shell syntax, code highlighting and tab completion when compared with the normal python shell terminal.

Python Modules used in CTK

PyFITS

PyFITS (http://www.stsci.edu/resources/software_hardware/pyfits) provides an interface to FITS formatted files. It allows full access to image data and keywords in both read and write mode.

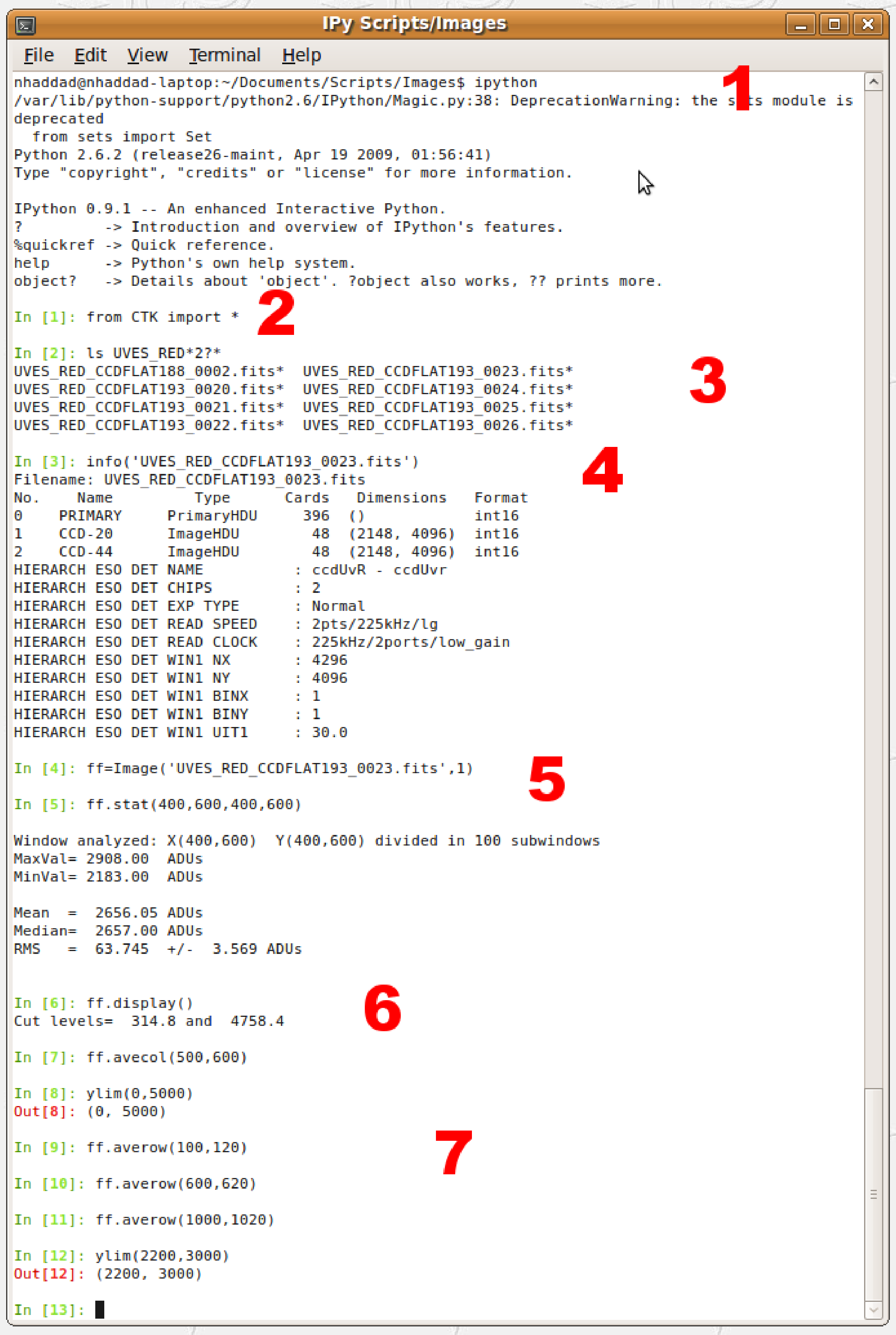
NumPy

NumPy (http://numpy.scipy.org/) is an extension to python, adding support for large, multi-dimensional arrays and matrices.

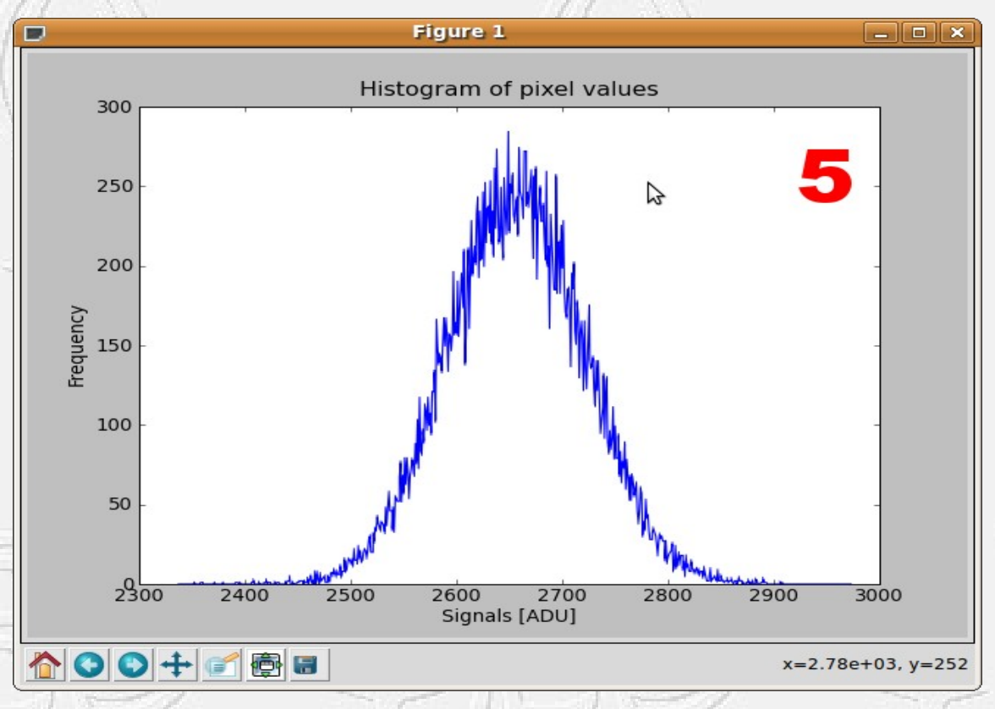
matplotlib

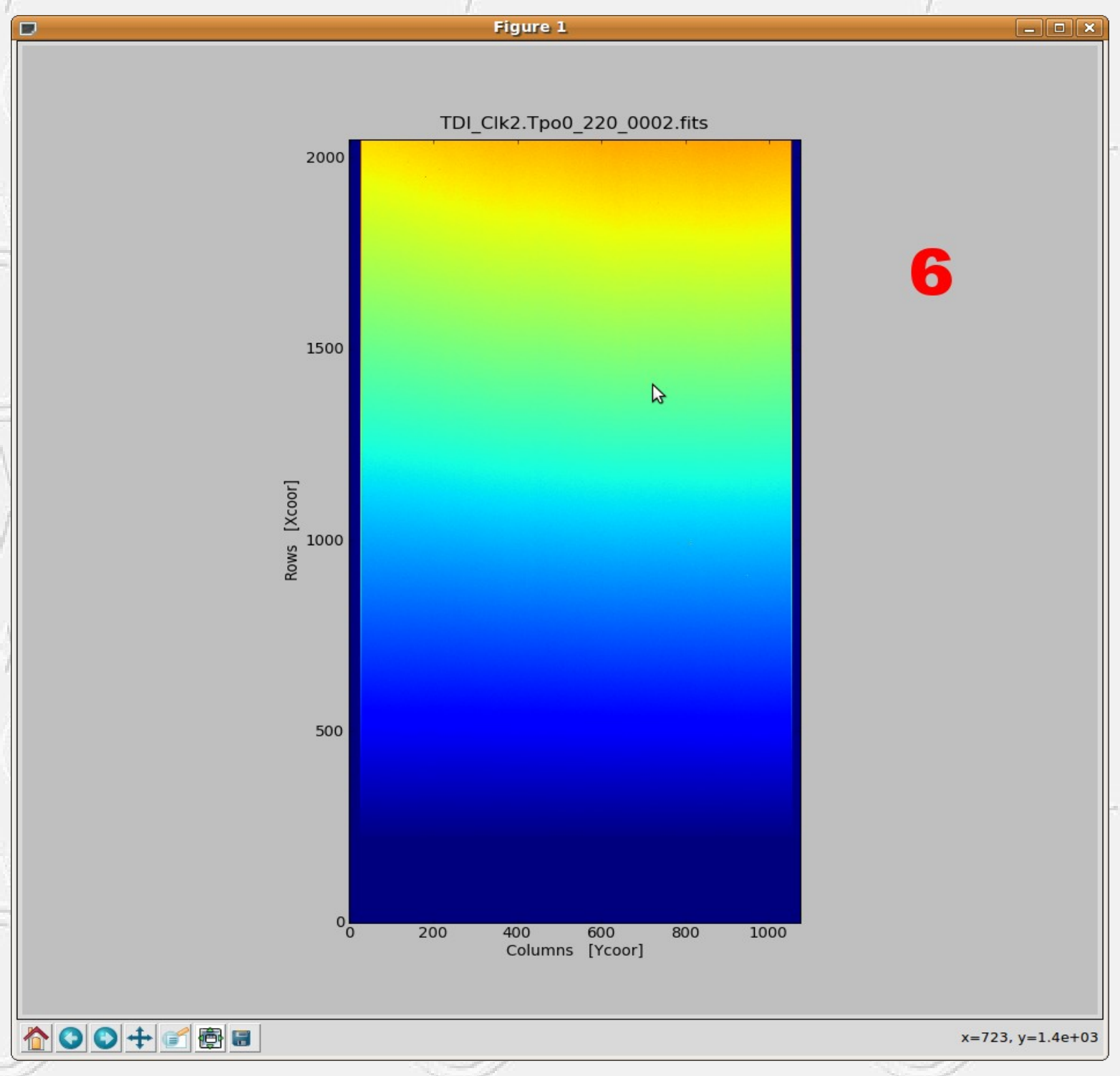
matplotlib (http://matplotlib.sourceforge.net/) is a Python 2D plotting library to produce publication quality figures. It can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc with just a few lines of codes.

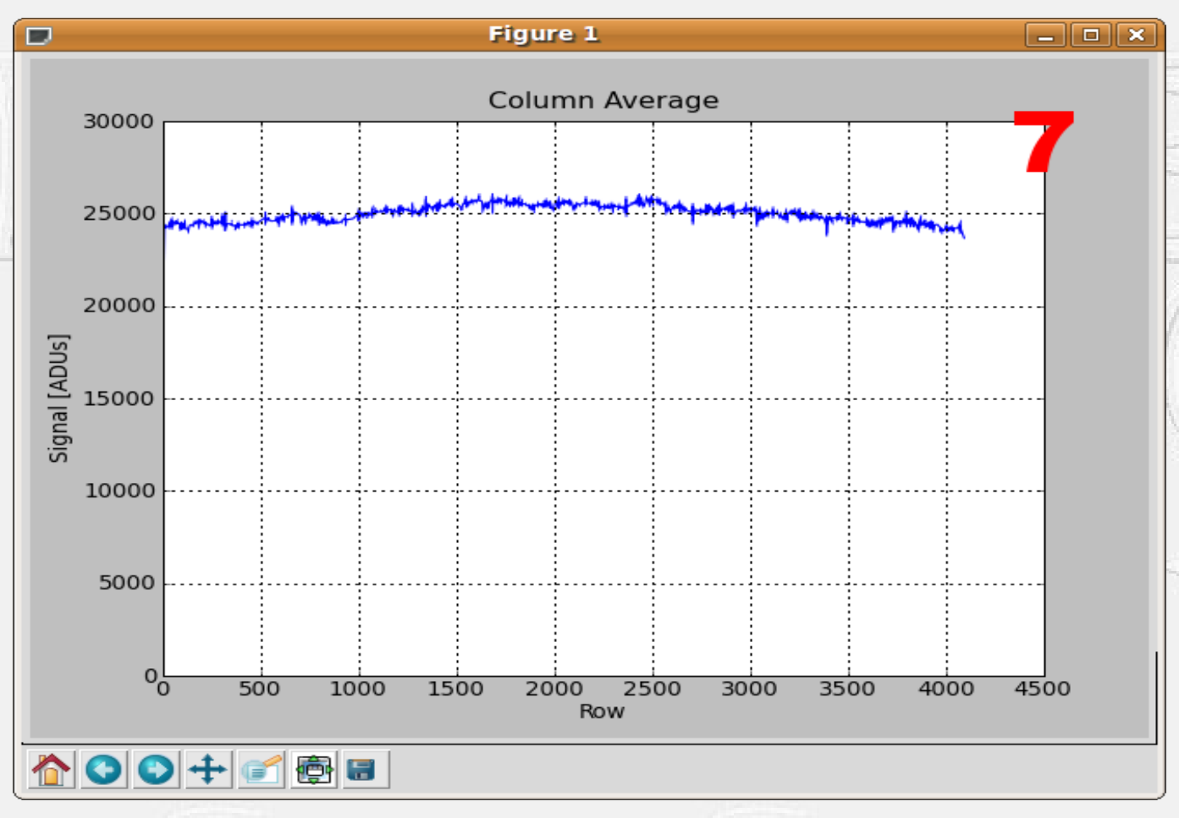
The CTK module can be used inside an interactive Python shell (python or ipython), or the data can be analyzed in batch mode, calling the appropriate reduction routine from the normal shell console. For future enhancements, the development of a Graphic User Interface (GUI) is foreseen to simplify the usage of this toolkit which will also be written in python.

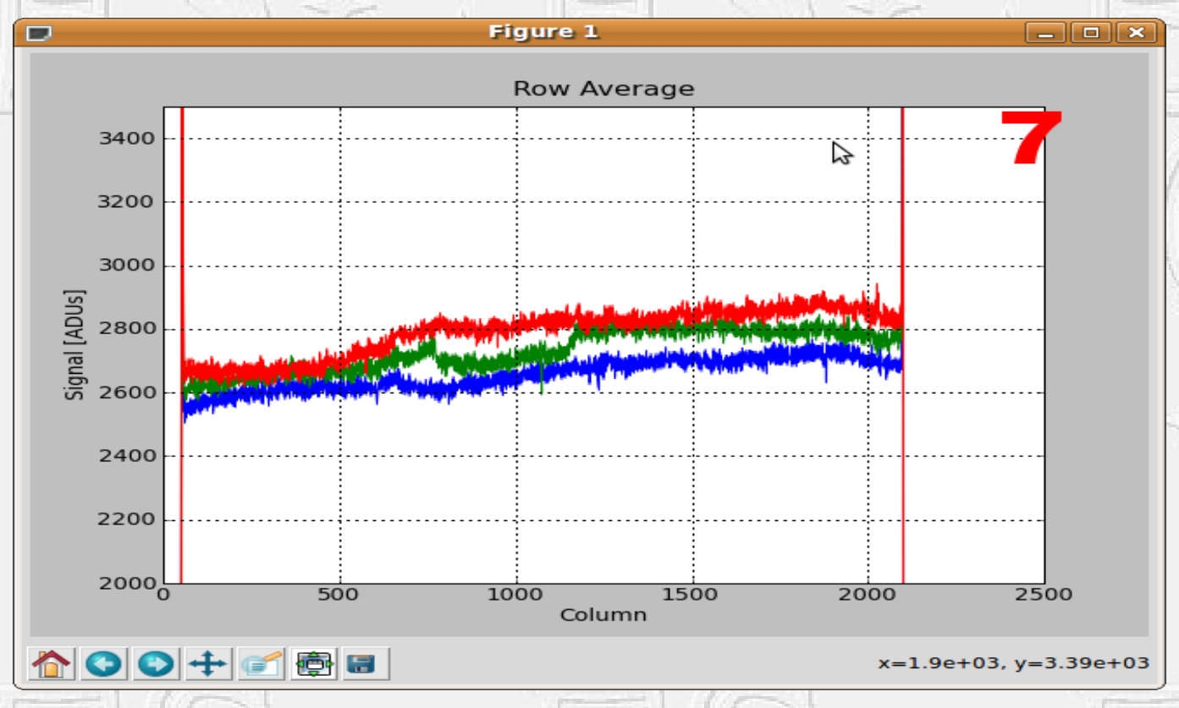


```
nhaddad@nhaddad-laptop:~/Documents/Scripts/Images$ ipython
/var/lib/python-support/python2.6/IPython/Magic.py:38: DeprecationWarning: the sets module is
deprecated
  from sets import Set
Python 2.6.2 (release26-maint, Apr 19 2009, 01:56:41)
Type "copyright", "credits" or "license" for more information.

IPython 0.9.1 -- An enhanced Interactive Python.
?         -> Introduction and overview of IPython's features.
%quickref -> Quick reference.
help      -> Python's own help system.
object?   -> Details about 'object'. ?object also works, ?? prints more.

In [1]: from CTK import *

In [2]: ls UVES_RED*2?*
UVES_RED_CCDFLAT188_0002.fits*  UVES_RED_CCDFLAT193_0023.fits*
UVES_RED_CCDFLAT193_0020.fits*  UVES_RED_CCDFLAT193_0024.fits*
UVES_RED_CCDFLAT193_0021.fits*  UVES_RED_CCDFLAT193_0025.fits*
UVES_RED_CCDFLAT193_0022.fits*  UVES_RED_CCDFLAT193_0026.fits*

In [3]: info('UVES_RED_CCDFLAT193_0023.fits')
Filename: UVES_RED_CCDFLAT193_0023.fits
No.    Name         Type      Cards   Dimensions   Format
0    PRIMARY     PrimaryHDU     396  ()            int16
1    CCD-20      ImageHDU        48  (2148, 4096)  int16
2    CCD-44      ImageHDU        48  (2148, 4096)  int16
HIERARCH ESO DET NAME         : ccdUvR - ccdUvr
HIERARCH ESO DET CHIPS        : 2
HIERARCH ESO DET EXP TYPE     : Normal
HIERARCH ESO DET READ SPEED   : 2pts/225kHz/lg
HIERARCH ESO DET READ CLOCK   : 225kHz/2ports/low_gain
HIERARCH ESO DET WIN1 NX      : 4296
HIERARCH ESO DET WIN1 NY      : 4096
HIERARCH ESO DET WIN1 BINX    : 1
HIERARCH ESO DET WIN1 BINY    : 1
HIERARCH ESO DET WIN1 UIT1    : 30.0

In [4]: ff=Image('UVES_RED_CCDFLAT193_0023.fits',1)

In [5]: ff.stat(400,600,400,600)

Window analyzed: X(400,600)  Y(400,600) divided in 100 subwindows
MaxVal= 2908.00  ADUs
MinVal= 2183.00  ADUs

Mean   =  2656.05 ADUs
Median=  2657.00 ADUs
RMS    =  63.745  +/-  3.569 ADUs


In [6]: ff.display()
Cut levels=  314.8 and  4758.4

In [7]: ff.avecol(500,600)

In [8]: ylim(0,5000)
Out[8]: (0, 5000)

In [9]: ff.averow(100,120)

In [10]: ff.averow(600,620)

In [11]: ff.averow(1000,1020)

In [12]: ylim(2200,3000)
Out[12]: (2200, 3000)

In [13]:
```







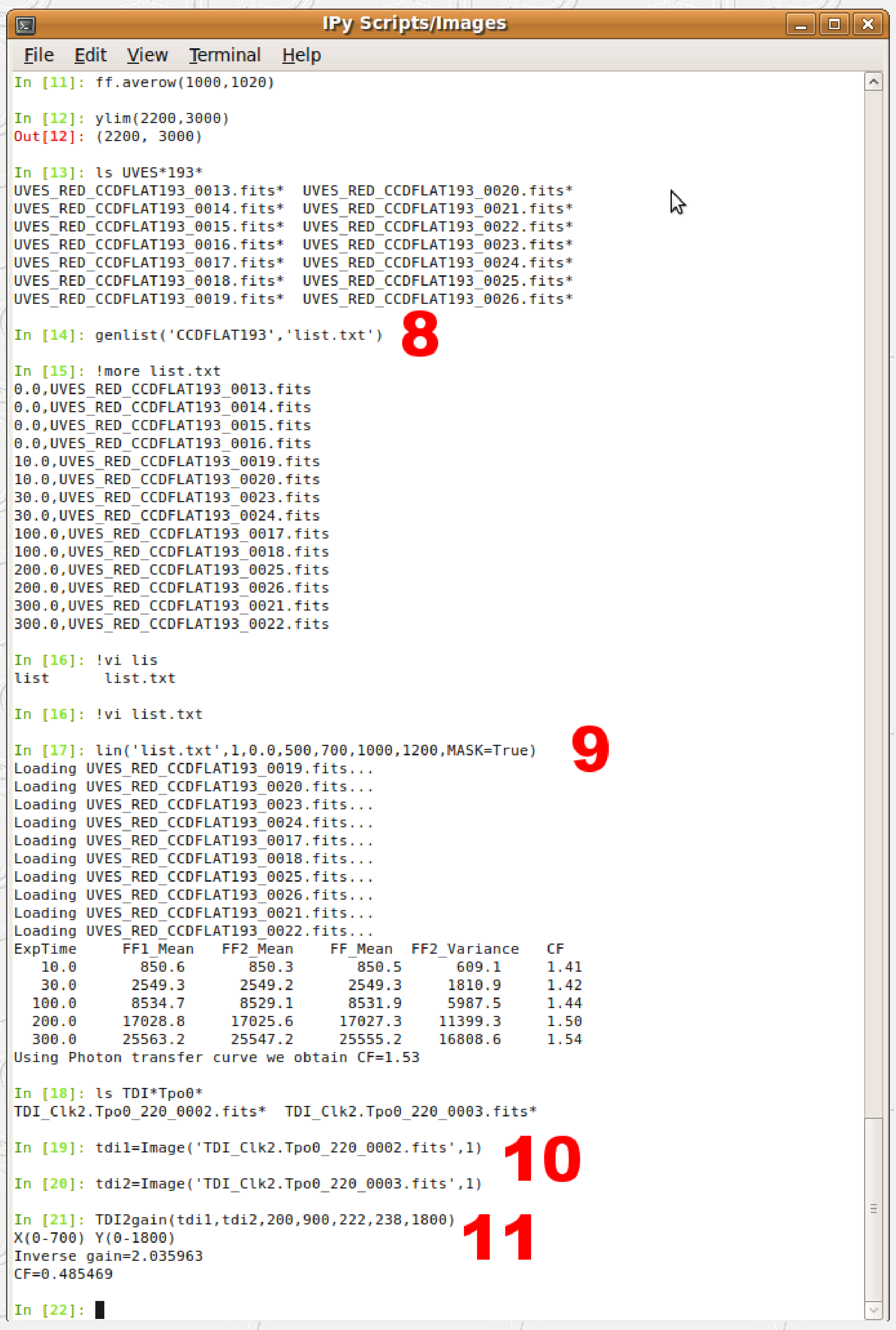


A mini tutorial on CTK

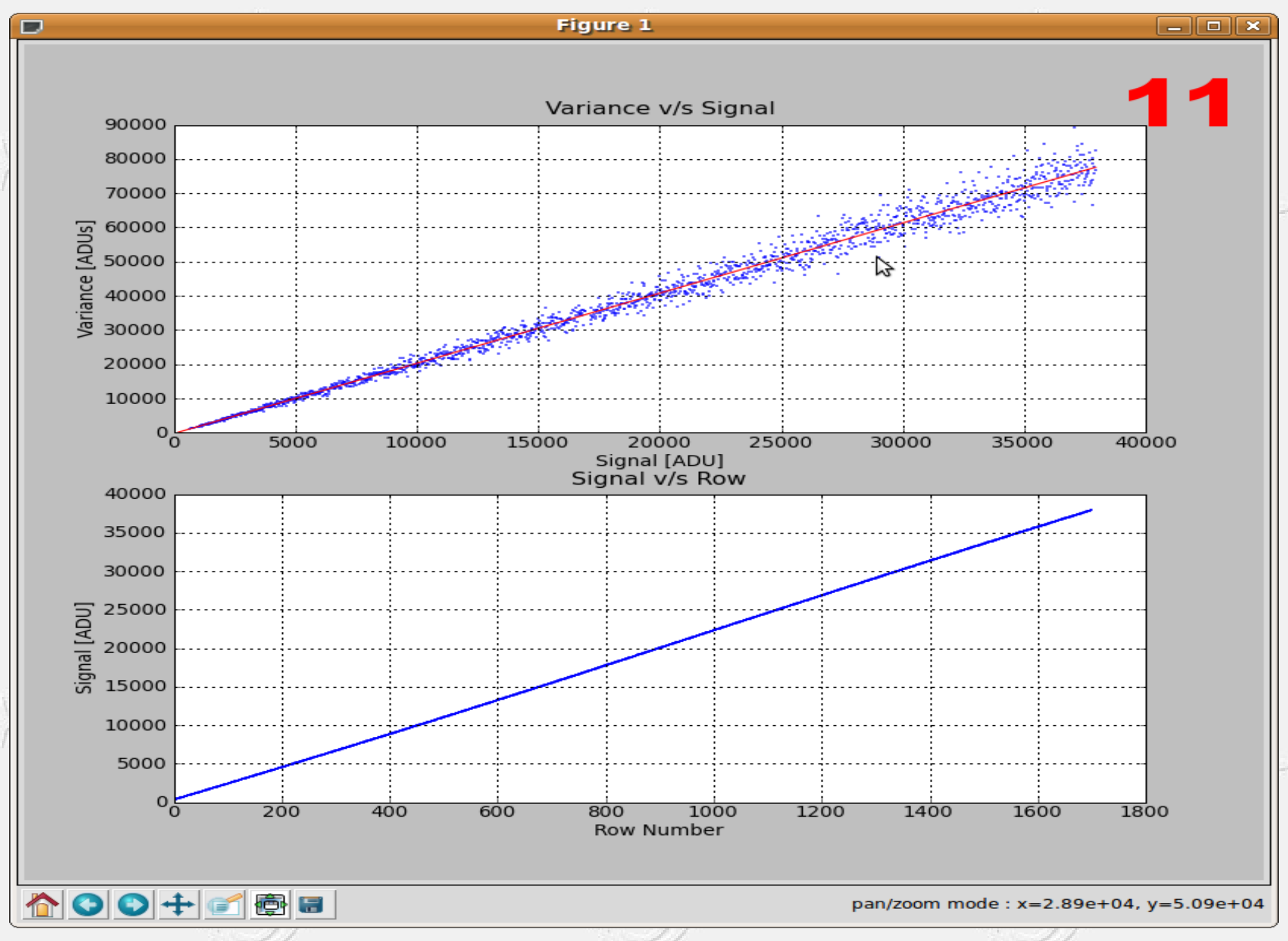
As an example of an interactive session with CTK, the following screen-shots shows how we can use Python + CTK to perform some typical operations during data analysis.

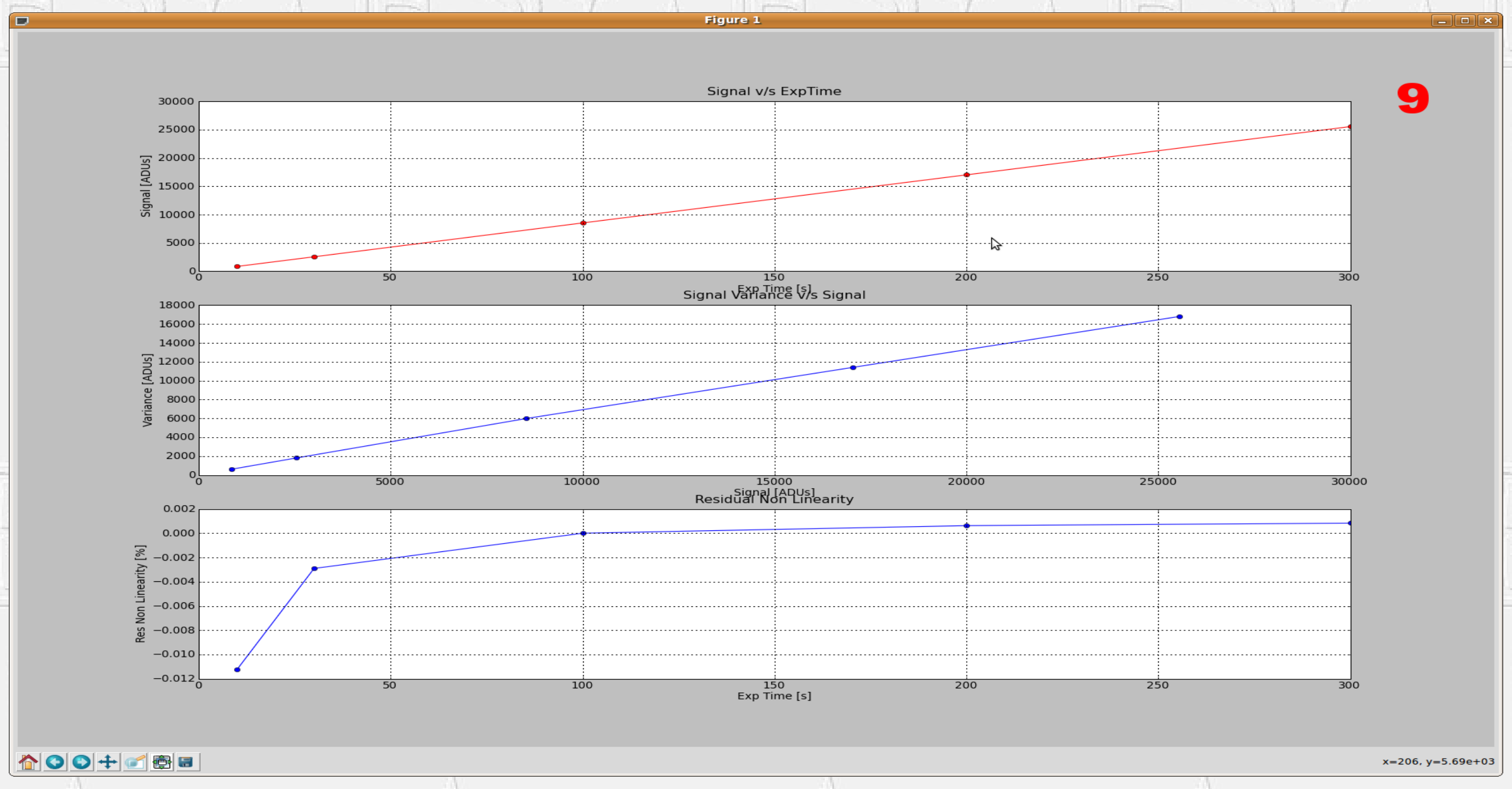
The sequence of operations is:

1) start ipython
2) load the CTK module
3) list the fits file in the current directory
4) check the structure on one of the files
5) load an image, perform statistic on a given area
6) display the image
7) plot the average value for some columns and later for some rows
8) generate a text file listing the images to be used for the linearity test.
9) run the linearity test on the images, plotting the results and saving the data to a text file.
10) load a couple of TDI images
11) perform linearity and gain analysis using the TDI images



```
In [11]: ff.averow(1000,1020)

In [12]: ylim(2200,3000)
Out[12]: (2200, 3000)

In [13]: ls UVES*193*
UVES_RED_CCDFLAT193_0013.fits*  UVES_RED_CCDFLAT193_0020.fits*
UVES_RED_CCDFLAT193_0014.fits*  UVES_RED_CCDFLAT193_0021.fits*
UVES_RED_CCDFLAT193_0015.fits*  UVES_RED_CCDFLAT193_0022.fits*
UVES_RED_CCDFLAT193_0016.fits*  UVES_RED_CCDFLAT193_0023.fits*
UVES_RED_CCDFLAT193_0017.fits*  UVES_RED_CCDFLAT193_0024.fits*
UVES_RED_CCDFLAT193_0018.fits*  UVES_RED_CCDFLAT193_0025.fits*
UVES_RED_CCDFLAT193_0019.fits*  UVES_RED_CCDFLAT193_0026.fits*

In [14]: genlist('CCDFLAT193','list.txt')

In [15]: !more list.txt
0.0,UVES_RED_CCDFLAT193_0013.fits
0.0,UVES_RED_CCDFLAT193_0014.fits
0.0,UVES_RED_CCDFLAT193_0015.fits
0.0,UVES_RED_CCDFLAT193_0016.fits
10.0,UVES_RED_CCDFLAT193_0019.fits
10.0,UVES_RED_CCDFLAT193_0020.fits
30.0,UVES_RED_CCDFLAT193_0023.fits
30.0,UVES_RED_CCDFLAT193_0024.fits
100.0,UVES_RED_CCDFLAT193_0017.fits
100.0,UVES_RED_CCDFLAT193_0018.fits
200.0,UVES_RED_CCDFLAT193_0025.fits
200.0,UVES_RED_CCDFLAT193_0026.fits
300.0,UVES_RED_CCDFLAT193_0021.fits
300.0,UVES_RED_CCDFLAT193_0022.fits

In [16]: !vi lis
list      list.txt

In [16]: !vi list.txt

In [17]: lin('list.txt',1,0.0,500,700,1000,1200,MASK=True)
Loading UVES_RED_CCDFLAT193_0019.fits...
Loading UVES_RED_CCDFLAT193_0020.fits...
Loading UVES_RED_CCDFLAT193_0023.fits...
Loading UVES_RED_CCDFLAT193_0024.fits...
Loading UVES_RED_CCDFLAT193_0017.fits...
Loading UVES_RED_CCDFLAT193_0018.fits...
Loading UVES_RED_CCDFLAT193_0025.fits...
Loading UVES_RED_CCDFLAT193_0026.fits...
Loading UVES_RED_CCDFLAT193_0021.fits...
Loading UVES_RED_CCDFLAT193_0022.fits...
ExpTime     FF1_Mean   FF2_Mean    FF_Mean  FF2_Variance   CF
   10.0       850.6      850.3      850.5      609.1     1.41
   30.0      2549.3     2549.2     2549.3     1810.9     1.42
  100.0      8534.7     8529.1     8531.9     5987.5     1.44
  200.0     17028.8    17025.6    17027.3    11399.3     1.50
  300.0     25563.2    25547.2    25555.2    16808.6     1.54
Using Photon transfer curve we obtain CF=1.53

In [18]: ls TDI*Tpo0*
TDI_Clk2.Tpo0_220_0002.fits*  TDI_Clk2.Tpo0_220_0003.fits*

In [19]: tdi1=Image('TDI_Clk2.Tpo0_220_0002.fits',1)

In [20]: tdi2=Image('TDI_Clk2.Tpo0_220_0003.fits',1)

In [21]: TDI2gain(tdi1,tdi2,200,900,222,238,1800)
X(0-700) Y(0-1800)
Inverse gain=2.035963
CF=0.485469

In [22]:
```





Detectors for Astronomy, 12 - 16 October 2009, ESO Garching