Leibniz-Institut für
Astrophysik Potsdam

# Python bindings for the Common Pipeline Library

**Ole Streicher[1], Peter M. Weilbacher[1]**

[1] Leibniz-Institut für Astrophysik Potsdam (AIP), Germany

## Introduction

The Common Pipeline Library (CPL) is a set of routines written by ESO to provide a standard interface for the VLT instrument data reduction pipelines.

Traditionally, VLT data reduction modules („recipes“) are executed via the command line tool EsoRex or a graphical user interface like Reflex.

Python-cpl allows to seamlessly embed VLT recipes into a standard Python environment, enabling comfortable interactive use (via IPython) as well as complex scripts.

CPL recipes are modelled as callable python objects, which allow access to their metadata and online documentation.

## Interactive use

Here, a simple IPython (PyLab) session is shown that generates a master bias using 5 raw bias files and a bad pixel table, and displays it in a plot:

```
import pyfits, cpl

badpixtbl = pyfits.open('/d/muse_badpix.fits')
# Use only the first 100 entries here:
badpixtbl[1].data = badpixtbl[1].data[:100]

bias = cpl.Recipe('muse_bias')
bias.param.nifu = 1
bias.calib.BADPIX_TABLE = badpixtbl

res = bias(('BIAS%2i.fits' % i) for i in range(5))
# Show the resulting master BIAS in a plot
matplotlib.pyplot.figimage(res.MASTER_BIAS[1].data)
```

The main properties for interactive use are:

- **Seamless integration of pyfits**

  Input and calibration frames may be given as file names or as `pyfits.HDUList` objects. As shown in the example above, `pyfits.HDUList` objects may be changed before they are used in the recipe. By default, `pyfits.HDUList` object are returned as results of the recipe.
- **Recipe parameters and frame tags**

  Parameters and calibration frame tags are modelled as normal Python attributes and can be accessed via IPython's tab completition feature.
- **Recipe specific online help**

  The recipe documentation is available as `__doc__` attributes and can be accessed with the '?' feature of IPython.
- **EsoRex support**

  The initialization files of EsoRex can be used to initialize python-cpl. Also, SOF files can be read or written
- **Easy access to recipe parameters of result frames**

  The recipe parameters, input frames and calibration frames can be retrieved from existing frames and re-used for new computations.

- **Simple parallel processing**

  Recipes can be executed in background, allowing a simple parallelization of similar tasks. Different processes will usually run in different working directories and do not interfere with each other. Access to a result of a background process automatically waits for the finalization of the recipe.
- **Detailed recipe stacktrace**

  Recipe crashes because of segmentation faults or similar are converted into a Python exception without compromising the stability of the Python environment. The stack trace contains the recipe C source code, if available:

```
cpl.result.RecipeCrash: Recipe Traceback
 File "[...]/CPL_recipe.c", line 953, in CPL_exec()
 File "[...]/sky.c", line 579, in sky_compute()
  muse_img *whitelight = to_whitelight(pixtable);
 File "[...]/resmpl.c", line 2846, in to_image()
  img = cpl_imagelist_collapse_create(cube);
SIGSEV: Segmentation Fault
```

## Automated processing

Python-cpl implements a number of features specific for batch processing:

- **Standard Python logging**

  The CPL messages are inserted into the Python log system, enabling a flexible log processing (output targets and formats, filters etc.).
- **Access to recipe metadata**

  All available metadata (available parameters, required input and calibration frames etc.) are available in Python, allowing the implementation of a generic processing tool.
- **Access to specific pipeline versions**

  If different pipeline versions are available, a specific version may be selected for the recipe creation.

## Applications

Python-cpl is currently used in the MUSE consortium for the implementation of the MuseWISE data management system (see poster **P117**). Other uses include laboratory measurement processing and pipeline development.

A simple command line tool to run CPL recipes is provided in the package.

## Acknowledgement

We acknowledge support by the German Verbundforschung through the MUSE/D3Dnet project (grant 05A11BA2).

## Further information and contact

**`http://www.aip.de/~oles/python-cpl`**

**`Ole Streicher <ole@aip.de>`**