Southern African Large Telescope

Title:	**Long-slit reduction with SHyRAF package RELSA**
Author(s):	**Alexei Y. Kniazev**
Doc. number:	**2350AC0099**
Version:	**1.0**
Date:	*December 15, 2006*
Keywords:	Long-slit, Data reduction, Pipeline recipe
Approved:	**David Buckley (Ast Ops Manager)**
	Signature:_____ Date:_____

ABSTRACT

In this document I am studing different recipes for SALT long-slit data reduction.

South African Astronomical Observatory, Cape Town 7925, South Africa

Contents

List of Figures

1 The Basic Ideas

The 2D Long-slit reduction is a standard and very well known procedure for people using IRAF. The *longslit* and *apextract* packages have all the possibilities to make this reduction in the correct way. Unfortunately, I reduced a lot of non-uniform data from different telescopes with different spectral setups and I felt myself not comfortable for a number of reasons:

1. it was hard to follow a uniform system of file naming, where after each step of the standard reduction file names has to be changed in some "standard" way;
2. it was hard to follow a uniform way to save and restore different spectral configurations for the different dates;
3. it was hard to follow a uniform system to save and restore spectrophotometric standards/sensitivity curves for the different spectral configurations for the different dates;
4. it was practically not possible to save and restore specific parameters for the different IRAF tasks to reduce data of different spectral configurations and/or different dates and/or different hardware (telescope+instrument).

Of course, it is very possibly that I am just a very lazy person.

2 The Implementation

The easiest and efficient way to solve most of above problems was to write my own scripts. Personally, I prefer SHyRAF scripting. "SHyRAF scripting" means UNIX Shell that manipulates with data-files and/or environmental variables using Shell-scripting language and possibilities of any of ∼500 standard UNIX programs. Shell calls for any specific tool/programs it needs to make a reduction of data: IRAF, IDL, MIDAS, PyRAF, ftool, perl etc. I like this way, but if you do not you are welcome to use your own.

In this implementation Shell makes temporal IRAF program IRAF_FILE_NAME that finally is executed in the standard way "CL < IRAF_FILE_NAME". Since Shell was designed for file-system operations, it is Shell responsibility to check that all directories, files and configuration files exist and/or properly copied. It is also Shell responsibility to form correct file names, lists and react to any errors.

3 The Package Description

3.1 The General Description

1. The current implementation of the package **RELSA** uses Shell, IRAF, PyRAF, MIDAS and ftool packages. Last two may or may not be used for the work with FITS headers depending on configuration file. IRAF is used by default, but for some interactive operations only PyRAF can be used, because IRAF does not work correct in this case. So, generally, both IRAF and PyRAF have to installed as the first.

2. The package is developed in the way that any new long-slit acquisition system should be easily added. For this reason the package is designed in the way that all acquisition system related steps are belongs to the lowest level scripts – functions. Currently **RELSA** can reduce data from three systems:

RSS – Long-slit mode of Robert Stobie Spectrograph of SALT;

SCORPIO – Long-slit mode of SCORPIO multi-mode instrument installed in the prime focus of the SAO RAS 6 m telescope (BTA);

GSPEC – Grating Spectrograph with SITE CCD at SAAO 1.9m telescope.

3. The package consist of number of Shell programs that could be divided into three basic categories: (a) main tasks, (b) subroutines, (c) functions and (d) configuration files. The current list of these programs is:

`LONG_epar.sh` – configuration file;

`LONG_red.sh` – one of main tasks of Long-slit reduction. Used for total reduction of one spectral setup observed in one date: wavelength calibration and background subtraction for the object(s), wavelength calibration and background subtraction for spectrophotometric standard(s), making of sensitivity curve and flux calibration of object(s). See more details in Section 3.3.1;

`LONG_One_setup.sh` – subroutine of Long-slit reduction. Used for wavelength calibration and background subtraction for the object(s) observed in one spectral setup. See more details in Section 3.4.1;

`LONG_Std_red.sh` – subroutine of Long-slit reduction. Used for making of sensitivity curve of one spectral setup. See more details in Section 3.4.2;

`LONG_flux.sh` – subroutine of Long-slit reduction. Used for flux calibration of object(s) of one spectral setup. See more details in Section 3.4.3;

`LONG_Make_ident.sh` – subroutine. Used for interactive spectral identification of the reference spectrum in one spectral setup. See more details in Section 3.4.4;

`LONG_cor_sens.sh` – function. See more details in Section 3.5;

`LONG_init_wav.sh` – function. See more details in Section 3.5;

`LONG_Make_airm.sh` – function. See more details in Section 3.5;

`LONG_Make_date.sh` – function. See more details in Section 3.5;

`LONG_Make_setup.sh` – function. See more details in Section 3.5;

`LONG_read_des.sh` – function. See more details in Section 3.5.

4. The package could be installed for your own using and be placed into any directory you would like. Hereafter I will call this directory *global* directory. Only one limitation exists: *global* directory with programs has to exist in your ${PATH} environment. I suggested that the best place is your ${HOME}/bin. More details about installation see in the Section 4.

5. Currently **RELSA** has only one configuration file that has name "`LONG_epar.sh`". Any main task or subroutine, when it is executed, looks for this configuration file in the local directory first and in the *global* directory second. All above means that you can have slightly different *local* "LONG_epar.sh" files in each directory you have data. More details about configuration file see in the Section 3.2.

6. Any main task or subroutine can be executed in the format:

```
LONG_<program> param1=value ... paramN=value
                        or
LONG_<program> paramN=value ... param1=value
```

   There is no any limit for a number of parameters.

   To see the list of parameters any main task or subroutine should be executed in the format:

```
LONG_<program> help=yes
        or
LONG_<program>
```

7. All spectral setups are acquisition system dependent. For the acquisition system **RSS** the general format of making of these names is

   LAMPID_GRATING_ARANGLE_GRANGLE_BIN, where:

   LAMPID – reference spectrum identification (content of FITS descriptor **LAMPID**);

   GRATING – grating identification (content of FITS descriptor **GRATING**);

   ARANGLE – articulation angle identification (content of FITS descriptor **CAMANG** in the format of F5.2).
   WARNING: This is NOT the REAL value of articulation angle, but ideal value that was requested.

   GRANGLE – grating angle identification (content of FITS descriptor **GRTILT** in the format of F5.2).
   WARNING: This is NOT the REAL value of grating angle, but ideal value that was requested.

   BIN – binning identification (content of FITS descriptor **CCDSUM**).

   Example: **ThAr_PG0900_27.25_13.63_2x2.fits** ref. spectrum for ThAr
   CuAr_PG0900_27.25_13.63_2x2.fits the same for CuAr

8. Sensitivity curves naming is acquisition system dependent. For the acquisition system **RSS** the general format of making such names is

Std_GRATING_ARANGLE_GRANGLE_BIN_DATE, where:

GRATING – grating identification (content of FITS descriptor **GRATING**);

ARANGLE – articulation angle identification (content of FITS descriptor **CAMANG**);

GRANGLE – grating angle identification (content of FITS descriptor **GRTILT**);

BIN – binning identification (content of FITS descriptor **CCDSUM**);

DATE – date of observation in the format **YYYYMMDD**, where:

YYYY – year;
MM – month;
DD – day.

Example: **Std_PG0900_27.25_13.63_2x2_20061011.fits** sensitivity curve

9. All parameters used by any task/subroutine/function can be *local* and *global*. *Global* parameters are defined in the configuration file first, but can be modified locally. Global parameters that will be modified surely are specified in the configuration file in the format `param=tmp`.

10. One reference spectrum of each spectral setup and all sensitivity curves are saved by **RELSA** in the special directory, the name of which is defined by global parameter `St_red_dir`.

    Example: `St_red_dir="/home/akniazev/ST_red"` my definition for saltastro

11. All tables of identified and used lines for each spectral setup are saved by **RELSA** in the special directory DATABASE, that are located in the directory defined by `St_red_dir` parameter.

12. **RELSA** is constructed in the way to be maximally independent of standard IRAF installation. It means that:
    (1) I would like to have all last versions of spectral distributions for used spectrophotometric standards, most correct reference spectra lines lists etc. to be located locally, since I am improving them and updating them;
    (2) I do not like to care to reinstall it each time IRAF is updated/reinstalled.
    For these reasons the package has the standard directories CALDIR and LINELISTS that are located in the directory also defined by `St_red_dir` parameter.

13. Input data have to be primary reduced and have to have only one-level header.

3.2 Configuration file `LONG_epar.sh`

1. Currently **RELSA** has only one configuration file;
2. Configuration file contains parameters for all tasks/subroutines/functions;
3. Parameters are written in the form `parameter=value` or `parameter="value"`;
4. Parameters and values are case sensitive;
5. Symbol '#' is a comment.
6. All parameter are written in the format "parameter=tmp" are changing inside of the scripts and can not be controled from the configuration file.
7. Some of these parameters are used by many programs, but some are used only by one. Below is the list of such parameters with short comment on each of them:

```
################################################################################
#.COPYRIGHT    (c)  2006  SAAO, CapeTown
#.IDENT        LONG_epar.sh
#.LANGUAGE     SHyRAF  :-)
#.AUTHOR       AKN   (Kniazev A.Y.), SALT
#
#.KEYWORDS
#.PURPOSE      Global definition of all parameteres for the package
#
#.COMMENT
#.VERSION      $Date: 2008/04/28 11:16:21 $
#              $Revision: 1.14 $
################################################################################
#
    #
    #...Database location
    #
#St_red_dir="/home/akniazev/SALT/Data/Reduction/ST_red"
St_red_dir="/home/akniazev/ST_red"
    #
    #...Name of the IRAF temporary file
    #
iraffile="Iraftemp"
################################################################################
    #
    #...For subroutine of reading information form FITS headers:
    #   Type of work with FITS-files: IRAF | ftool | MIDAS
```

```
    #
FITS_work=IRAF
    #
    #...For "FITS_work=IRAF" you need to know IRAF locaion
    #
irafdir=/iraf/iraf/bin.linux
###############################################################################
    #
    #...Observational systems for which these scripts are working:
    #   1. RSS - prime focus spectrograph of SALT
    #   2. SCORPIO - prime focus spectrograph of 6m
    #   3. GSPEC - Grating Spectrograph with SITE CCD at SAAO 1.9m
    #
#
#export System=RSS
#Observatory=saao
#
#export System=SCORPIO
#Observatory=sao
#
export System=GSPEC
Observatory=saao
GRAT=G3
GRANG=7.1
###############################################################################
    #
    #...Global definitions of reduction steps
    #
cosmic=""
subtract=""
wcalib="w"
backgr="s"
calibflux="f"
###############################################################################
    #
    #...Keys for specific programs
    #
#...For "LONG_Std_red.sh"
#
sens_app=no     #...append new star ('yes') to the file with standard stars
#   or create new ('no') file
sens=yes        #...make or not sensitivity curve [yes|no]
copy=yes        #...copy or not the resulted sensitivity curve into
```

```
#   database [yes|no]
################################################################################
#
#    WARNING!!! parameters "parameter=tmp" ARE CHANGING inside of the scripts
#               and CAN NOT be controled from here
#    They are just listed below to show their existence
#
################################################################################
    #
    #...Parameters for IRAF tasks
    #
#...longslit.identify.
#   For "LONG_Make_ident.sh" script
#
    images=tmp
    coordlist_id=tmp
    function_id=chebyshev
    order_id=4
    fwidth=6.
    cradius=6.
#
#...longslit.reidentify.
#   For "LONG_Make_ident.sh" and "LONG_One_setup.sh" scripts.
#
    reference=tmp
    images=tmp
    coordlist=tmp
    interactive=no
    newaps=yes
    override=no
    refit=yes
    nlost=20
    verbose=yes
    trace=yes
    section='middle line'
    shift=INDEF
    search=INDEF
    addfeatures=no
#
#...longslit.fitcoord
#   For "LONG_Make_ident.sh" and "LONG_One_setup.sh" scripts.
#
    images=tmp
```

```
    interactive_fit=no
    combine=no
    functio=legendre
    xorder=5
    yorder=3
#
#...longslit.transform
#   For "LONG_One_setup.sh" script.
#
    input=tmp
    output=tmp
    fitnames=tmp
    interptype=linear
    flux=yes
    blank=INDEF
    x1=INDEF
    x2=INDEF
    dx=INDEF
    y1=INDEF
    y2=INDEF
    dy=INDEF
#
#...longslit.background
#   For "LONG_One_setup.sh" script.
#
    input=tmp
    output=tmp
    axis=2
    interactive_back=no
    naverage=1
    function=chebyshev
    order=5
    low_rej=2.
    high_rej=1.5
    niterate=5
    grow=0.
#
#...apextract.apall
#   For "LONG_Std_red.sh" script.
#
    input=tmp
    output=tmp
    format=onedspec
```

```
    interactive_apall=yes
    nfind=1
    llimit=-25
    ulimit=25
    t_order=6
    ylevel=INDEF
    line=INDEF
#
#...longslit.standard
#   For "LONG_Std_red.sh" script.
#
    input=tmp
    output=tmp
    caldir=${St_red_dir}/caldir/
    interact_st=no
    star_name=tmp
    extinct=${St_red_dir}/caldir/suzextinct2.dat
#    extinct=${St_red_dir}/caldir/ctioextinct.dat
#
#...longslit.sensfunc
#   For "LONG_Std_red.sh" script.
#
    standard=tmp
    sensitiv=tmp
    interactive_sens=yes
#
#...longslit.calibrate
#   For "LONG_Std_red.sh" and "LONG_flux.sh" scripts.
#
    input=tmp
    output=tmp
    sensitivity_cal=tmp
    extinctc=${St_red_dir}/caldir/suzextinct2.dat
```

3.3 Detailed Description of Main Tasks

3.3.1 LONG_red.sh

To be done.

3.4 Detailed Description of Subroutines

3.4.1 `LONG_One_setup.sh`

Execution : `LONG_One_setup.sh obj=obj_list [arc=arc_list]`, where:

obj_list – list of objects of one setup for which reference spectrum listed in the **arc_list** can be used;

arc_list – reference spectrum that have to be used for wavelength calibration of object(s) listed in the **obj_list**.

Algorithm :

1. Script checks that standard identification for this spectral setup exist:
   (a) If reference spectrum was specified, script will check for such spectral setup in the "database" directories, copy it to the current directory and uses for reidentification.
   (b) If there is nothing for such setup, new identification will be started and result will be copied into the "database" directories.
   (c) If there is no reference spectra was specified, the script will check for such spectral setup in the "database" directories, copy it to the current directory and uses it for the 2D wavelength transformation.
   (d) If there is no reference spectra was specified, and there is nothing for such spectral setup in the "database" directories, the script will stop.
2. If the reference spectrum was specified, reidentification and fitcoord will be redone.
3. 2D wavelength transformation will be done on base either new or standard fit for all input objects.
4. Background will be done for all input objects.

IRAF used tasks : *longslit.identify, longslit.reidentify, longslit.fitcoord, longslit.transform, longslit.background*

3.4.2 `LONG_Std_red.sh`

To be done.

3.4.3 `LONG_flux.sh`

To be done.

3.4.4 `LONG_Make_ident.sh`

To be done.

3.5 Detailed Description of Functions

3.5.1 `LONG_cor_sens.sh`

To be done.

3.5.2 `LONG_init_wav.sh`

To be done.

3.5.3 `LONG_Make_airm.sh`

To be done.

3.5.4 `LONG_Make_date.sh`

To be done.

3.5.5 `LONG_Make_setup.sh`

To be done.

3.5.6 `LONG_read_des.sh`

4 Package Installation

- To work with **RELSA** package you need IRAF and PyRAF have been previously installed on your computer;
- Copy from *http : //hren_znaet_otkyda/* files A, B and C into any directory you would like to install the package. Lets suggest you would like to install it into your ${HOME}/bin directory. In this case copy ABC into your ${HOME}/bin directory;
- Make "`tar xzvf file`" for each file: A, B and C;
- Check that ${HOME}/bin in your ${PATH};
- Enjoy your reduction.

5 Inserting of New Acquisition Systems

RELSA is written in the standard way that all acquisition system dependencies are located in lowest level functions. To insert new acquisition system you need:

1. specify the name of new acquisition system in the configuration file (parameter `System`);

2. edit/add specific part into the text of functions: `LONG_cor_sens.sh`, `LONG_init_wav.sh`, `LONG_Make_airm.sh`, `LONG_Make_date.sh`, `LONG_Make_setup.sh`. Each function is self-described.

6 Examples

6.1 Example 1

I have my RSS data observed on 20061026 and primary reduced. I created next lists:

File name	What is that	Name of the List
mrpP200610260003ee.fits	PN in the dwarf elliptical Local Group galaxy	obj.lst
mrpP200610260004ee.fits	another exposure	obj.lst
mrpP200610260005ee.fits	ThAr reference spectrum for this setup	obj_arc.lst
mrpP200610260017ee.fits	Spectrophotometric standard	std.lst
mrpP200610260018ee.fits	ThAr reference spectrum for the standard	std_arc.lst

Reduction:

- I know that all my parameters are fine and I reduced such setup before:

```
LONG_red.sh obj=obj.lst arc=obj_arc.lst std=std.lst std_arc=std_arc.lst
```

- I did not reduced such spectral setup before and I am not sure that I can do it easily. For this reason first, I played only with the objects to see that wavelength calibration is fine and parameters of background procedure produce correct result.

```
LONG_red.sh obj=obj.lst arc=obj_arc.lst
```

  After that I just restart everything. Since setup was identified once, the wavelength calibration after that will be re-done automatically.

```
LONG_red.sh obj=obj.lst arc=obj_arc.lst std=std.lst std_arc=std_arc.lst
```

A Codes of programs

```
#!/bin/sh
#
##############################################################################
#.COPYRIGHT    (c) 2006 SAAO, CapeTown
#.IDENT        LONG_RSS_red.sh
#.LANGUAGE     SHyRAF  :-)
#.AUTHOR       AKN (Kniazev A.Y.), SALT
#
#.KEYWORDS
#.PURPOSE      Script for reduction of LONG-slit RSS data
#
#.COMMENT
#.VERSION      $Date: 2008/04/28 11:18:26 $
#              $Revision: 1.15 $
##############################################################################
#
 #
 # ...Help
 #
help() {
echo "Usage: LONG_red.sh obj=obj_list [arc=obj_arc] [[std=std_list] [std_arc=std_arc_l
echo "where:"
echo "       obj_list - list of objects (without of standards)"
echo "       obj_arc -  list of arcs for object (currently - one)"
echo "       std_list - list of standards"
echo "       std_arc  - list of arcs for standards (currently - one)"
echo ""
}
##############################################################################
 #
 # ...Some initial setups
 #
    #...Global
    #
    if [ -f ./LONG_epar.sh ]; then
. ./LONG_epar.sh
    else
. LONG_epar.sh
    fi
    #
```

```
    #...Local
    #
    eval $*
###############################################################################
#
# ...Check input parameters
#
    #...For help
    #
    if [ "z${help}" = "zyes" -o $# = 0 ]; then
help
exit 1
    fi
    #
    # ...For list of objects
    #
    if [ "z${obj}" = "z" -o ! -s "${obj}" ]; then
echo ""
echo "***** LONG_red.sh: no objects were specified *****"
echo ""
help
exit 1
    fi
###############################################################################
#
# ...Some initial setups for IRAF
#
    if [ ! -f login.cl ]; then
mkiraf
    fi
 #
 # ...Some initial checks
 #
    if [ -d uparm -o -f uparm ]; then
rm -rf uparm
    fi
    cp -rpd ${St_red_dir}/uparm ./
    #
    if [ -d database -o -f database ]; then
rm -rf database
    fi
    cp -rpd ${St_red_dir}/database ./
    #
```

```
    if [ -f loginuser.cl ]; then rm -f loginuser.cl; fi
#############################################################################
#
# ...Make our lists for each setup
#
#
# ...1.Reduce all objects of similar setup
#
    LONG_One_setup.sh obj=${obj} arc=${arc}
    if [ $? != 0 ] ; then
exit 1
    fi
#
# ...2.Reduce standards for this setup
#
    if [ "z${std}" = "z" -o ! -s "${std}" ]; then
echo ""
echo "***** LONG_red.sh: no standards were specified *****"
echo ""
    else
LONG_One_setup.sh obj=${std} arc=${std_arc}
if [ $? != 0 ] ; then
    exit 1
fi
sed 's/.fits/ws.fits/g'  <${std} >${std}a
LONG_Std_red.sh obj=${std}a
if [ $? != 0 ] ; then
    exit 1
fi
    fi
#
# ...3.Make flux calibration for this setup
#
    sed 's/.fits/ws.fits/g'  <${obj} >${obj}a
    LONG_flux.sh obj=${obj}a
#
#...Delete all temporary files
#
    rm -f ${std}a ${obj}a
#############################################################################
```

```
#!/bin/sh
#
###############################################################################
#.COPYRIGHT    (c)  2006  SAAO, CapeTown
#.IDENT        LONG_One_setup.sh
#.LANGUAGE     SHyRAF  :-)
#.AUTHOR       AKN   (Kniazev A.Y.), SALT
#
#.KEYWORDS     Long
#.PURPOSE      Script for reduction of one setup of LONG-slit data
#
#.COMMENT      What script is doing:
#              1. Script checks that calibrations for this spectral setup
#                 exist:
#                 a. If one specified reference spectrum, script will
#                    check for identification and copy it.
#                 b. If no reference spectra was specified, the script
#                    will try to find 2d fit for this setup and copy it.
#              2. If reference spectrum was specified, reidentification
#                 and fitcoord will be re-done.
#              3. 2d transformation will be done on base either new
#                 or standard fit for all input files.
#              4. Background will be done for all input files.
#
#
#.VERSION      $Date: 2006/12/07 14:03:22 $
#              $Revision: 1.22 $
###############################################################################
#
 #
 # ...Help
 #
help() {
echo "Usage: LONG_One_setup.sh obj=obj_list [arc=arc_list]"
echo "where:"
echo "       obj_list - list of objects for this arc"
echo "       arc_list - list of arcs (currently - one)"
echo ""
}
###############################################################################
 #
 # ...Some initial setups
```

```
 #
    #...Global
    #
    if [ -f ./LONG_epar.sh ]; then
. ./LONG_epar.sh
    else
. LONG_epar.sh
    fi
    #
    #...Local
    #
    eval $*
#############################################################################
#
# ...Check input parameters
#
    #...For help
    #
    if [ "z${help}" = "zyes" -o $# = 0 ]; then
help
exit 1
    fi
    #
    # ...For list of objects
    #
    if [ "z${obj}" = "z" -o ! -s "${obj}" ]; then
echo ""
echo "***** LONG_One_setup.sh: no objects were specified *****"
echo ""
help
exit 1
    fi
#############################################################################
#
# ...Some initial setup for IRAF
#
    if [ ! -f login.cl ]; then
mkiraf
    fi
    #
    if [ -f loginuser.cl ]; then rm -f loginuser.cl; fi
#############################################################################
#
```

```
# ...Define our spectral setup and copy data for it
#
    #...In case of only object we need to find calibration
    #   for this setup and copy it to the current directory
    #
    if [ "z${arc}" = "z" -o ! -s "${arc}" ]; then
reference="`cat ${obj}|head -1`"
reference=`LONG_Make_setup.sh ${reference} obj`
reference1=`echo ${reference}|sed 's/NONE//g'`
reference=`ls ${St_red_dir}/*${reference1}.fits|head -1`
reference=`basename ${reference}|sed 's/.fits//g'`
if [ -f ${St_red_dir}/${reference}.fits -a -f ${St_red_dir}/database/fc${reference} ];
    cp ${St_red_dir}/${reference}.fits ./
    cp ${St_red_dir}/database/fc${reference} ./database
    echo ""
    echo "***** I was need to copy ${reference} setup from the database *****"
    echo ""
else
    echo "***** $0: I need to exit, because there is no identification *****"
    echo "***** $0: for requested ${reference1} setup in our database  *****"
    exit 1
fi
echo "${reference}.fits" >$$
arc=$$
    else
    #...In case of reference we need to find identification
    #   for this setup and copy it to the current directory
    #
reference="`cat ${arc}|head -1`"
reference=`LONG_Make_setup.sh ${reference} obj`
if [ -f "${St_red_dir}/${reference}.fits" -a -f "${St_red_dir}/database/id${reference}
    cp ${St_red_dir}/${reference}.fits ./
    cp ${St_red_dir}/database/id${reference} ./database
else
    echo "There is no identification for ${reference} setup in our database"
    LONG_Make_ident.sh arc=${arc}
    if [ $? != 0 ] ; then
echo "***** $0: Something wrong *****"
exit 1
    fi
fi
    fi
#
```

```
# ...Define our reference spectrum and prepare
#    this table
#
    coordlist=‘echo ${reference}|tr "_" " "|awk ’{print $1}’|tr ’[:upper:]’ ’[:lower:]
    if [ -f ${St_red_dir}/linelists/${coordlist}.dat ]; then
cp ${St_red_dir}/linelists/${coordlist}.dat ./
    else
echo "***** $0: There is no table with coordinate list in our directory *****"
exit 1
    fi
#
# ...Make our lists and delete existed files
#
    sed ’s/.fits/w.fits/g’   <${obj} >${obj}${wcalib}
    sed ’s/.fits/ws.fits/g’  <${obj} >${obj}${wcalib}${backgr}
    sed ’s/.fits//g’ <${arc} >${arc}1
    rm -rf ‘cat ${obj}${wcalib} ${obj}${wcalib}${backgr}‘
    ref_spec=‘cat ${arc}1|tr -d ’\r’‘
#
# ...make iraf input file
#
    echo print "’***** Vsemy prishla jopa *****’"   >  $iraffile
    echo ’set imtype          = "fits"’             >> $iraffile
    echo reset stdimage        = imt6               >> $iraffile
    echo images                                     >> $iraffile
    echo plot                                       >> $iraffile
    echo dataio                                     >> $iraffile
    echo lists                                      >> $iraffile
    echo tv                                         >> $iraffile
    echo utilities                                  >> $iraffile
    echo noao                                       >> $iraffile
    echo twodspec                                   >> $iraffile
    echo longslit                                   >> $iraffile
    echo print "’***** Reducing of ARCs *****’"     >> $iraffile
    #
    #...Check that this spectrum reidentification exist
    #   It can be like that in case some spectra are using
    #   the same reference spectrum
    #
    if [ ! -s database/id${ref_spec} ]; then
echo reidentify reference=${reference} images=${ref_spec} interactive=${interactive} n
echo reidentify reference=${ref_spec} images=${ref_spec} interactive=${interactive} ne
    fi
```

```
    #
    #...Check that this two-dimensional solution exists for this
    #   reference spectrum.
    #
    if [ ! -s database/fc${ref_spec} ]; then
echo fitcoord   images=\@${arc}1 interactive=${interactive_fit} combine=${combine} fun
    fi
    echo print "'***** Reducing of objects ***** '" >> $iraffile
    echo hedit images=\@${obj} fields=DISPAXIS value=1 add=yes addonly=yes ver-  >> $i
    echo transform input=\@${obj} output=\@${obj}${wcalib} fitnames=\@${arc}1 interpty
    echo background input=\@${obj}${wcalib} output=\@${obj}${wcalib}${backgr} axis=${a
    echo logout                                          >> $iraffile
#
#...Execution of IRAF depending on request for interactive work
#   with "fitcoord" task or/and "background" task
#
    if [ ${interactive_fit} = "no" -a ${interactive_back} = "no" ]; then
cl < ${iraffile}
    else
echo '! kill -9 `ps aux|fgrep ${USER}|fgrep python|fgrep -v fgrep|awk "{print \$2}"`'
mv ${iraffile} loginuser.cl
iraffile="loginuser.cl"
#        xgterm -e "cl"
pyraf -n
    fi
###############################################################################
#
#...Delete all temporary files
#
    rm -f ${obj}${wcalib} ${obj}${wcalib}${backgr} ${reference}.fits plotfile ${arc}1
    rm -f ${iraffile} ${coordlist}.dat
###############################################################################
```

```
#!/bin/sh
#
###############################################################################
#.COPYRIGHT    (c)  2006  SAAO, CapeTown
#.IDENT        LONG_Std_red.sh
#.LANGUAGE     SHyRAF  :-)
#.AUTHOR       AKN   (Kniazev A.Y.), SALT
#
#.KEYWORDS     Long
#.PURPOSE      Script for reduction of standards and making sensitivity function
#
#.COMMENT      What script is doing:
#
#
#.VERSION      $Date: 2008/04/28 11:18:26 $
#              $Revision: 1.20 $
###############################################################################
#
 #
 # ...Help
 #
help() {
echo 'Usage: LONG_Std_red.sh obj=std_list [sens_app=yes|no] [sens=yes|no [copy=yes|no]
echo "where:"
echo "       std_list - list of standards"
echo '       sens_app=yes|no - append new star to the file with standard stars or crea
echo '       sens=yes|no     - make or not sensitivity curve finally'
echo '       copy=yes|no     - copy or not sensitivity curve into the database directo
echo "      "
echo "Program suggests: "
echo "1. All objects were 2D-linearized and background was subtracted."
echo '2. Each file in the input list has .fits extension'
echo "      "
}
###############################################################################
 #
 # ...Some initial setups
 #
    #...Global
    #
    if [ -f ./LONG_epar.sh ]; then
. ./LONG_epar.sh
```

```
    else
. LONG_epar.sh
    fi
    #
    #...Local
    #
    eval $*
###############################################################################
#
# ...Check input parameters
#
    #...For help
    #
    if [ "z${help}" = "zyes" -o $# = 0 ]; then
help
exit 1
    fi
    #
    # ...For list of objects
    #
    if [ "z${obj}" = "z" -o ! -s "${obj}" ]; then
echo ""
echo "***** LONG_Std_red.sh: no standards were specified *****"
echo ""
help
exit 1
    fi
###############################################################################
#
# ...Some initial setup for IRAF
#
    if [ ! -f login.cl ]; then
mkiraf
    fi
    #
    if [ -f loginuser.cl ]; then rm -f loginuser.cl; fi
###############################################################################
#
# ...Create output names
#
    #...For the standard
    #
    file=‘cat ${obj}|head -1‘
```

```
    std_out=‘LONG_Make_setup.sh ${file} std‘
    ddate=‘LONG_Make_date.sh ${file}‘
    std_out=${std_out}_${ddate}
    if [ ${sens_app} = "no" ]; then
rm -f ${std_out}
    fi
#
# ...make iraf input file
#
    iraffile_st=${iraffile}
    iraffile="loginuser.cl"
    echo print "’***** Vsemy prishla jopa *****’"   >  $iraffile
    echo ’set imtype          = "fits"’            >> $iraffile
    echo reset stdimage        = imt6              >> $iraffile
    echo images                                    >> $iraffile
    echo plot                                      >> $iraffile
    echo dataio                                    >> $iraffile
    echo lists                                     >> $iraffile
    echo tv                                        >> $iraffile
    echo utilities                                 >> $iraffile
    echo noao                                      >> $iraffile
    echo twodspec                                  >> $iraffile
    echo apextract dispaxis=1                      >> $iraffile
#
# ...Main loop
#
    for i in ‘cat ${obj}‘
    do
aa=‘echo ${i}|sed ’s/.fits/1d/g’‘
bb=‘echo ${i}|sed ’s/.fits/.0001.fits/g’‘
cc=‘echo ${i}|sed ’s/.fits//g’‘
    #
    #...Create name for input file with data for standard star
    #
star_name=‘LONG_read_des.sh ${i} OBJECT|sed ’s/ //g’|tr ’-’ ’_’|tr ’[:upper:]’ ’[:lowe
#
    if [ ! -f ${caldir}/${star_name}.dat ]; then
echo ""
echo "***** LONG_Std_red.sh: no file ${star_name}.dat for this star *****"
echo "***** LONG_Std_red.sh: in the directory ${caldir} *****"
echo ""
rm -rf ${iraffile}
exit 1
```

```
    fi
#
    #
    #...Continue to make iraf file
    #
echo imdel ${bb},${aa}.fits                          >> $iraffile
echo apall input=${i} format=${format} interactive=${interactive_apall} nfind=${nfind}
#
#...You have to delete next string in case of blyadskogo pyraf
#
echo ${cc}                                           >> $iraffile
#
echo rename ${bb} ${aa}.fits                         >> $iraffile
echo longslit                                        >> $iraffile
LONG_Make_airm.sh ${i} ${aa}.fits ${Observatory} $iraffile
if [ $? != 0 ] ; then
    exit 1
fi
echo standard input=${aa}.fits output=${std_out} caldir=${caldir} interact=${interact_
    done
#
    if [ ${sens} = "yes" ]; then
echo imdel ${std_out}.fits >> $iraffile
echo '! LONG_cor_sens.sh' ${aa}.fits ${std_out} >> $iraffile
echo sensfunc standard=${std_out} sensitiv=${std_out}.fits interactive=${interactive_s
    fi
    echo logout                                              >> $iraffile
#
#    echo '! kill -9 `ps aux|fgrep ${USER}|fgrep python|fgrep -v fgrep|awk "{print \$2
#    pyraf -n
    xgterm -e "cl"
###############################################################################
#
#...Next iraf to calibrate standards itself
#
    if [ ${sens} = "yes" ]; then
rm -rf ${iraffile}
iraffile=${iraffile_st}
echo print "'***** Vsemy prishla jopa *****'"   >  $iraffile
echo 'set imtype          = "fits"'              >> $iraffile
echo reset stdimage        = imt6                >> $iraffile
echo images                                      >> $iraffile
echo plot                                        >> $iraffile
```

```
echo dataio                                             >> $iraffile
echo lists                                              >> $iraffile
echo tv                                                 >> $iraffile
echo utilities                                          >> $iraffile
echo noao                                               >> $iraffile
echo twodspec                                           >> $iraffile
echo longslit                                           >> $iraffile
for i in `cat ${obj}`
do
    in_cal=`echo ${i}|sed 's/.fits/1d.fits/g'`
    out_cal=`echo ${in_cal}|sed 's/1d/1d'${calibflux}'/g'`
    echo imdel ${out_cal} >> $iraffile
    echo calibrate input=${in_cal} output=${out_cal} sensitivity=${std_out}.fits extin
done
echo logout                                               >> $iraffile
    #
cl < ${iraffile}
    #
    #...Copy all files into Database directory
    #
if [ "${copy}" = "yes" ]; then
    cp -v ${std_out} ${std_out}.fits ${St_red_dir}
fi
    fi
###############################################################################
#
#...Delete all temporary files
#
    rm -rf ${iraffile}
###############################################################################
```

```
#!/bin/sh
#
###############################################################################
#.COPYRIGHT    (c)  2006  SAAO, CapeTown
#.IDENT        LONG_flux.sh
#.LANGUAGE     SHyRAF  :-)
#.AUTHOR       AKN   (Kniazev A.Y.), SALT
#
#.KEYWORDS     Long
#.PURPOSE      Script for correction for sensitivity curve
#
#.COMMENT      What script is doing:
#
#
#.VERSION      $Date: 2008/04/28 11:18:26 $
#              $Revision: 1.7 $
###############################################################################
#
 #
 # ...Help
 #
help() {
echo 'Usage: LONG_flux.sh obj=obj_list [sens_file=sensitivity]'
echo "where:"
echo "       obj_list    - list of objects"
echo '       sensitivity - use this sensitivity file instead of standard one'
echo "      "
echo "Program suggests: "
echo "1. All objects were 2D-linearized and background was subtracted."
echo '2. Each file in the input list has .fits extension'
echo "      "
}
###############################################################################
 #
 # ...Some initial setups
 #
    #...Global
    #
    if [ -f ./LONG_epar.sh ]; then
. ./LONG_epar.sh
    else
. LONG_epar.sh
```

```
    fi
    #
    #...Local
    #
    eval $*
###############################################################################
#
# ...Check of input parameters
#
    #...For help
    #
    if [ "z${help}" = "zyes" -o $# = 0 ]; then
help
exit 1
    fi
    #
    # ...For list of objects
    #
    if [ "z${obj}" = "z" -o ! -s "${obj}" ]; then
echo ""
echo "***** LONG_flux.sh: no objects were specified *****"
echo ""
help
exit 1
    fi
###############################################################################
#
# ...Some initial setup for IRAF
#
    if [ ! -f login.cl ]; then
mkiraf
    fi
    #
    if [ -f loginuser.cl ]; then rm -f loginuser.cl; fi
###############################################################################
#
# ...Create input name for sensitivity curve
#
    if [ "z${sens_file}" = "z" ]; then
file=‘cat ${obj}|head -1‘
std_out=‘LONG_Make_setup.sh ${file} std‘
ddate=‘LONG_Make_date.sh ${file}‘
std_out="${std_out}_${ddate}.fits"
```

```
if [ ! -f ${std_out} ]; then
    if [ -f ${St_red_dir}/${std_out} ]; then
cp -v ${St_red_dir}/${std_out} ./
    else
echo "***** LONG_flux: There is no sensitivity curve for this setup *****"
echo "***** LONG_flux: Your setup is ${std_out} *****"
echo
exit 1
    fi
else
    echo "***** The sensitivity curve exists for this setup *****"
    echo "***** Your setup is ${std_out} *****"
    echo
fi
    else
std_out="${sens_file}"
if [ ! -f ./${std_out} ]; then
    if [ -f ${St_red_dir}/${std_out} ]; then
cp -v ${St_red_dir}/${std_out} ./
    else
echo "***** The specified sensitivity curve does not exists *****"
echo
exit 1
    fi
fi
    fi
#
# ...make iraf input file
#
    echo print "'***** Vsemy prishla jopa *****'"   >  $iraffile
    echo 'set imtype          = "fits"'             >> $iraffile
    echo reset stdimage        = imt6               >> $iraffile
    echo images                                     >> $iraffile
    echo plot                                       >> $iraffile
    echo dataio                                     >> $iraffile
    echo lists                                      >> $iraffile
    echo tv                                         >> $iraffile
    echo utilities                                  >> $iraffile
    echo noao                                       >> $iraffile
    echo twodspec                                   >> $iraffile
    echo longslit                                   >> $iraffile
#
# ...Main loop
```

```
#
    for i in `cat ${obj}`
    do
aa=`echo ${i}|sed 's/.fits/'${calibflux}'.fits/g'`
echo imdel ${aa}                              >> $iraffile
LONG_Make_airm.sh ${i} ${i} ${Observatory} $iraffile
if [ $? != 0 ] ; then
    exit 1
fi
echo calibrate input=${i} output=${aa} sensitivity=${std_out} extinction=${extinct} >>
    done
#
    echo logout                                       >> $iraffile
    cl < ${iraffile}
#
#...Delete all temporary files
#
    rm -rf ${iraffile}
##############################################################################
```

```
#!/bin/sh
#
###############################################################################
#.COPYRIGHT    (c)  2006  SAAO, CapeTown
#.IDENT        LONG_init_wav.sh
#.LANGUAGE     SHyRAF  :-)
#.AUTHOR       AKN (Kniazev A.Y.), SALT
#
#.KEYWORDS
#.PURPOSE      Calculate of initial wavelength and step for current setup
#
#.COMMENT
#.VERSION      $Date: 2006/12/07 14:04:42 $
#              $Revision: 1.6 $
###############################################################################
#
if [ -f ./LONG_epar.sh ]; then
    . ./LONG_epar.sh
else
    . LONG_epar.sh
fi
file=$1
param=$2
###############################################################################
#
# ...Some initial setups for IRAF
#
    if [ ! -f login.cl ]; then
mkiraf
    fi
    #
    if [ -f loginuser.cl ]; then rm loginuser.cl; fi
###############################################################################
#
#...Main body
#
if [ ${System} = "RSS" ]; then
    if [ -f ${file} ]; then
grat=`LONG_read_des.sh ${file} GRATING|sed 's/ //g'|sed 's/PG//g'`
grangle=`LONG_read_des.sh ${file} GR-ANGLE|sed 's/ //g'|awk '{printf "%7.3f",$1}'`
arangle=`LONG_read_des.sh ${file} AR-ANGLE|sed 's/ //g'|awk '{printf "%10.6f",$1}'`
start=`echo "1"|awk "{printf \"%8.3f\", (1e7/${grat})*(sin(0.017453292*${grangle})+sin
```

```
#       blue_end=‘echo "1"|awk "{printf \"%8.3f\", (1e7/${grat})*(sin(0.017453292*${g
blue_end=‘echo "1"|awk "{printf \"%8.3f\", (1e7/${grat})*(sin(0.017453292*${grangle})+
cent_beg=‘echo "1"|awk "{printf \"%8.3f\", (1e7/${grat})*(sin(0.017453292*${grangle})+
cent_end=‘echo "1"|awk "{printf \"%8.3f\", (1e7/${grat})*(sin(0.017453292*${grangle})+
red_beg=‘echo "1"|awk "{printf \"%8.3f\", (1e7/${grat})*(sin(0.017453292*${grangle})+s
end=‘echo "1"|awk "{printf \"%8.3f\", (1e7/${grat})*(sin(0.017453292*${grangle})+sin(0
naxis=‘LONG_read_des.sh ${file} NAXIS1|awk ’{print $1}’‘
step=‘echo "1"|awk "{printf \"%8.5f\", (${end}-${start})/${naxis}}"‘
echo "${start} ${step} ${blue_end} ${cent_beg} ${cent_end} ${red_beg}"
    #
if [ "${param}" = "yes" ]; then
    echo print "’***** Vsemy prishla jopa *****’"   >  $iraffile
    echo ’set imtype          = "fits"’            >> $iraffile
    echo reset stdimage        = imt6              >> $iraffile
    echo images                                    >> $iraffile
    echo plot                                      >> $iraffile
    echo dataio                                    >> $iraffile
    echo lists                                     >> $iraffile
    echo tv                                        >> $iraffile
    echo utilities                                 >> $iraffile
    echo noao                                      >> $iraffile
    echo twodspec                                  >> $iraffile
    echo hedit images=${file} fields=CRVAL1 value=${start} add=yes addonly=yes ver-  >
    echo hedit images=${file} fields=CRPIX1 value=1 add=yes addonly=yes ver-  >> $iraf
    echo hedit images=${file} fields=CDELT1 value=${step} add=yes addonly=yes ver-  >>
    echo hedit images=${file} fields=CD1_1 value=${step} add=yes addonly=yes ver-  >>
    echo logout                                    >> $iraffile
    cl < ${iraffile}
    rm -f ${iraffile}
fi
    #
    else
echo "***** LONG_init_wav.sh: No file ${file} *****"
exit 1
    fi
else
    echo -n "OK"
fi
################################################################################
```

```
#!/bin/sh
#
###############################################################################
#.COPYRIGHT    (c)  2006  SAAO, CapeTown
#.IDENT        LONG_Make_airm.sh
#.LANGUAGE     SHyRAF  :-)
#.AUTHOR       AKN   (Kniazev A.Y.), SALT
#
#.KEYWORDS
#.PURPOSE      Calculation of airmass
#
#.COMMENT
#.VERSION      $Date: 2008/04/28 11:18:26 $
#              $Revision: 1.6 $
###############################################################################
#
file=$1
file1=$2
Observatory=$3
iraffile=$4
    #
    #...First, calculate airmass
    #   Second, generate IRAF command
    #
if [ -f ${file} ]; then
    if [ "${System}" = "RSS" ]; then
    #
    #...First
    #
airmass=‘LONG_read_des.sh ${file} TELALT|sed ’s/ //g’|awk ’{printf "%6.3f",$1}’‘
airmass=‘echo "1"|awk "{print 1/cos(0.017453292*(90-${airmass}))}"|tr -d ’\r’‘
    #
    #...Second
    #
echo hedit images=${file1} fields=AIRMASS value=${airmass} add=yes addonly=yes ver-
echo hedit images=${file1} fields=OBSERVAT value=${Observatory} add=yes addonly=yes ve
#
    elif [ "${System}" = "SCORPIO" ]; then
    #
    #...First
    #
airmass=‘LONG_read_des.sh ${file} Z|sed ’s/ //g’|awk ’{printf "%6.3f",$1}’‘
```

```
airmass=‘echo "1"|awk "{print 1/cos(0.017453292*(${airmass}))}"|tr -d ’\r’‘
    #
    #...Second
    #
echo hedit images=${file1} fields=AIRMASS value=${airmass} add=yes addonly=yes ver-
#
#...If System is unknown
#
#
    elif [ "${System}" = "GSPEC" ]; then
    #
    #...First
    #
echo hedit images=${file1} fields=OBSERVAT value=${Observatory} add=yes addonly=yes ve
echo setairmass images=${file1} update+ override+       >> $iraffile
#
#...If System is unknown
#
    else
echo hedit images=${file1} fields=AIRMASS value=1.0 add=yes addonly=yes ver-       >>
    fi
else
    echo ""
    echo "***** LONG_Make_airm.sh: Input file {file} does not exist *****"
    echo ""
    exit 1
fi
```

```
#!/bin/sh
#
################################################################################
#.COPYRIGHT    (c)  2006  SAAO, CapeTown
#.IDENT        LONG_Make_date.sh
#.LANGUAGE     SHyRAF  :-)
#.AUTHOR       AKN (Kniazev A.Y.), SALT
#
#.KEYWORDS
#.PURPOSE      Creation of date in the format 20060606
#
#.COMMENT
#.VERSION      $Date: 2008/04/28 11:18:26 $
#              $Revision: 1.3 $
################################################################################
#
if [ ${System} = "RSS" ]; then
    ddate=`echo $1 | tr -d "[a-z]"|tr -d "[A-Z]"|tr -d '.'|sed 's/....$//'`
    echo ${ddate}
elif [ ${System} = "SCORPIO" ]; then
    name=`pwd`
    ddate=`basename ${name}`
    echo ${ddate}
elif [ ${System} = "GSPEC" ]; then
    ddate=`LONG_read_des.sh ${1} DATE-OBS|sed 's/-//g'`
    echo ${ddate}
fi
```

```
#!/bin/sh
#
###############################################################################
#.COPYRIGHT    (c)  2006  SAAO, CapeTown
#.IDENT        LONG_Make_ident.sh
#.LANGUAGE     SHyRAF  :-)
#.AUTHOR       AKN (Kniazev A.Y.), SALT
#
#.KEYWORDS     Long
#.PURPOSE      Script making idendification of one setup of LONG-slit data
#
#.COMMENT      What script is doing:
#
#.VERSION      $Date: 2006/12/07 14:02:13 $
#              $Revision: 1.5 $
###############################################################################
#
 #
 # ...Help
 #
help() {
echo "Usage: LONG_Make_ident.sh arc=arc_list"
echo "where:"
echo "       arc_list - List with reference spectrum for this setup"
echo "      "
echo 'Program suggests that input file has .fits extension'
echo "      "
}
###############################################################################
 #
 # ...Some initial setups
 #
    #...Global
    #
    if [ -f ./LONG_epar.sh ]; then
. ./LONG_epar.sh
    else
. LONG_epar.sh
    fi
    #
    #...Local
    #
```

```
    eval $*
################################################################################
#
# ...Check input parameters
#
    #...For help
    #
    if [ "z${help}" = "zyes" -o $# = 0 ]; then
help
exit 1
    fi
    #
    # ...For list of objects
    #
    if [ "z${arc}" = "z" -o ! -s "${arc}" ]; then
echo ""
echo "***** LONG_Make_ident.sh: List with data was not specified *****"
echo ""
help
exit 1
    fi
################################################################################
#
# ...Some initial setup for IRAF
#
    if [ ! -f login.cl ]; then
mkiraf
    fi
    #
    if [ -f loginuser.cl ]; then rm -f loginuser.cl; fi
################################################################################
#
# ...Define our spectral setup
#
    spec=‘cat ${arc}‘
    reference=‘LONG_Make_setup.sh ${spec} arc‘
    if [ ${reference} = "NONE_NONE_NONE_NONE_NONE" ]; then
echo "***** LONG_Make_ident.sh: ${reference} *****"
echo "***** LONG_Make_ident.sh: Can not get spectral setup for this system *****"
exit 1
    else
cp ${spec} ${reference}.fits
    fi
```

```
#
# ...And calculate initial wavelength and step for it
#    (if we can do it, of course)
#
    LONG_init_wav.sh ${reference}.fits yes
    if [ $? != 0 ] ; then
 echo "***** LONG_Make_ident.sh: Something wrong *****"
 exit 1
    fi
#
# ...Define our reference spectrum and prepare this table
#
    coordlist=‘echo ${reference}|tr "_" " "|awk ’{print $1}’|tr ’[:upper:]’ ’[:lower:]
    if [ -f ${St_red_dir}/linelists/${coordlist}.dat ]; then
cp ${St_red_dir}/linelists/${coordlist}.dat ./
    else
echo "***** LONG_Make_ident.sh: There is no table with coordinate list in our director
exit 1
    fi
#
# ...make iraf input file
#
    echo print "’***** Vsemy prishla jopa *****’"   >  $iraffile
    echo ’set imtype          = "fits"’            >> $iraffile
    echo reset stdimage        = imt6              >> $iraffile
    echo images                                    >> $iraffile
    echo plot                                      >> $iraffile
    echo dataio                                    >> $iraffile
    echo lists                                     >> $iraffile
    echo tv                                        >> $iraffile
    echo utilities                                 >> $iraffile
    echo noao                                      >> $iraffile
    echo twodspec                                  >> $iraffile
    echo longslit                                  >> $iraffile
    echo identify images=${reference} coordlist=${coordlist}.dat function=${function_i
    echo reidentify reference=${reference} images=${reference} interactive=${interacti
    echo fitcoord images=${reference} interactive=yes combine=${combine} functio=${fun
    echo logout                                    >> $iraffile
    echo ’! kill -9 ‘ps aux|fgrep ${USER}|fgrep python|fgrep -v fgrep|awk "{print \$2}
#
#...Execution of IRAF depending on request for interactive work
#   with "fitcoord" task or/and "background" task
#
```

```
    mv ${iraffile} loginuser.cl
    iraffile="loginuser.cl"
    pyraf -n
#
#...Copy all output files into Database directory
#
    cp -v ${reference}.fits ${St_red_dir}
    cp -v database/id${reference} ${St_red_dir}/database
    cp -v database/fc${reference} ${St_red_dir}/database
#
#...Delete all temporary files
#
    rm -f ${iraffile} ${coordlist}.dat
###############################################################################
```

```
#!/bin/sh
#
###############################################################################
#.COPYRIGHT    (c)  2006  SAAO, CapeTown
#.IDENT        LONG_Make_setup.sh
#.LANGUAGE     SHyRAF  :-)
#.AUTHOR       AKN   (Kniazev A.Y.), SALT
#
#.KEYWORDS
#.PURPOSE      Creation of setup name
#
#.COMMENT
#.VERSION      $Date: 2008/04/28 11:18:26 $
#              $Revision: 1.10 $
###############################################################################
#
. ./LONG_epar.sh
file=$1
std=$2
    #
    #...For RSS
    #
if [ "${System}" = "RSS" ]; then
#
#...Read of LAMP ID
#
    LAMPID=`LONG_read_des.sh ${file} LAMPID|sed 's/ //g'`
#
#...Read of grating name
#
    GRATING=`LONG_read_des.sh ${file} GRATING|sed 's/ //g'`
#
#...Read of grating angle
#
    GRANGLE=`LONG_read_des.sh ${file} GRTILT|sed 's/ //g'|awk '{printf "%5.2f",$1}'`
#
#...Read of Articulation Station
#
    ARANGLE=`LONG_read_des.sh ${file} CAMANG|sed 's/ //g'|awk '{printf "%5.2f",$1}'`
#
#...Read of Binning factor
#
```

```
    BIN=‘LONG_read_des.sh ${file} CCDSUM|sed "s/’//g"|sed ’s/ /x/g’‘
#
#...Forming of output
#
    if [ "${std}" = "std" ]; then
echo -n "Std_${GRATING}_${ARANGLE}_${GRANGLE}_${BIN}"
    else
echo -n "${LAMPID}_${GRATING}_${ARANGLE}_${GRANGLE}_${BIN}"
    fi
    #
    #...For SCORPIO
    #
elif [ "${System}" = "SCORPIO" ]; then
#
#...Read of Binning factor
#
    BIN=‘LONG_read_des.sh "${file}" BINNING|sed ’s/ //g’|sed "s/’//g"‘
#
#...Forming of output
#
    if [ "${std}" = "std" ]; then
echo -n "Std_VPHG550G_${BIN}"
    else
echo -n "HeNeAr_VPHG550G_${BIN}"
    fi
    #
    #...For GSPEC
    #
elif [ "${System}" = "GSPEC" ]; then
#
#...Read of Binning factor
#
    BIN="1x1"
#
#...Read of grating name
#
    GRATING=${GRAT}
#
#...Read of grating angle
#
    GRANGLE=${GRANG}
#
#...Forming of output
```

```
#
    if [ "${std}" = "std" ]; then
echo -n "Std_${GRATING}_${GRANGLE}_${BIN}"
    else
#
#...Read of LAMP ID
#
LAMPID=‘LONG_read_des.sh ${file} ARC-LAMP|sed ’s/ //g’‘
    #
echo -n "${LAMPID}_${GRATING}_${GRANGLE}_${BIN}"
    fi
else
#
#...Forming of output for common case
#
    echo -n "NONE_NONE_NONE_NONE_NONE"
fi
```

```
#!/bin/sh
#
###############################################################################
#.COPYRIGHT    (c)  2006  SAAO, CapeTown
#.IDENT        LONG_cor_sens.sh
#.LANGUAGE     SHyRAF  :-)
#.AUTHOR       AKN (Kniazev A.Y.), SALT
#
#.KEYWORDS
#.PURPOSE      Try to exclude data for gaps from the sensitive curve file
#
#.COMMENT
#.VERSION      $Date: 2006/11/15 22:04:21 $
#              $Revision: 1.2 $
###############################################################################
#
if [ -f ./LONG_epar.sh ]; then
    . ./LONG_epar.sh
else
    . LONG_epar.sh
fi
fits_file=$1
sens_file=$2
#
#...Main body
#
if [ ${System} = "RSS" ]; then
    if [ -f ${file} ]; then
numbers=‘LONG_init_wav.sh ${fits_file} no‘
blue_end=‘echo ${numbers}|awk ’{print $3}’‘
cent_beg=‘echo ${numbers}|awk ’{print $4}’‘
cent_end=‘echo ${numbers}|awk ’{print $5}’‘
red_beg=‘ echo ${numbers}|awk ’{print $6}’‘
awk "{if ( \$1 < ${blue_end} || \$1 > ${cent_beg} ) print \$0}" ${sens_file} >${sens_f
awk "{if ( \$1 < ${cent_end} || \$1 > ${red_beg} ) print \$0}" ${sens_file}$$ >${sens_
rm ${sens_file}$$
    #
    else
echo "***** $0: No file ${file} *****"
exit 1
    fi
else
```

```
    cp ${sens_file} /dev/null
fi
```

```
#!/bin/sh
#
###############################################################################
#.COPYRIGHT    (c)  2006  SAAO, CapeTown
#.IDENT        LONG_read_des.sh
#.LANGUAGE     SHyRAF  :-)
#.AUTHOR       AKN (Kniazev A.Y.), SALT
#
#.KEYWORDS
#.PURPOSE      Read_of_descriptor from FITS file
#
#.COMMENT
#.VERSION      $Date: 2006/12/06 17:17:55 $
#              $Revision: 1.8 $
###############################################################################
    #
    # ...Some initial setups
    #
       #...Global
       #
    if [ -f ./LONG_epar.sh ]; then
. ./LONG_epar.sh
    else
. LONG_epar.sh
    fi
#
    file=$1
    des=$2
    if [ -f "${file}" ]; then
    #
    #...For MIDAS
    #
if [ "${FITS_work}" = "MIDAS" ]; then
    if [ "${des}" = "OBJECT" ]; then des="IDENT" ; fi
    if [ "${des}" = "NAXIS1" ]; then des='NPIX' ; fi
if [ -f ${HOME}/midwork/login.prg ]; then
    mv ${HOME}/midwork/login.prg ${HOME}/midwork/login.prg$$
    echo -n `inmidas xh -p -j "read/des ${file} ${des} H; bye"|tr -d '\r'`
    mv ${HOME}/midwork/login.prg$$ ${HOME}/midwork/login.prg
else
    echo -n `inmidas xh -p -j "read/des ${file} ${des} H; bye"|tr -d '\r'`
fi
```

```
elif [ "${FITS_work}" = "ftool" ]; then
    #
    #...For ftool
    #
    export HEADAS='/usr/local/bin'
    ftool='/usr/local/bin/headas-init.sh'
    . $ftool
    fkeypar $file+0 ${des}
#            echo -n `pget fkeypar value | sed 's/ //g' | sed "s/'//g" `
    echo -n `pget fkeypar value | sed "s/'//g" `
    #
    #...For IRAF
    #
elif [ "${FITS_work}" = "IRAF" ]; then
    echo -n `${irafdir}/x_images.e hedit images=${file} fields=${des} value="." add="n
else
    echo -n "-100"
fi
    else
echo -n "-100"
    fi
```