estec

European Space Research
and Technology Centre
Keplerlaan 1
2201 AZ Noordwijk
The Netherlands
Tel. (31) 71 5656565
Fax (31) 71 5656040
www.esa.int

NIRSpec Technical Note
NTN-2011-007

Author(s): S. Birkmann and P. Ferruit

Date of Issue: March 22, 2013
Version: 1.2

Installing the NIRSpec Calibration Software and Database on Mac OS X 10.7 (Lion)

Abstract:

This document describes the necessary steps to install the NIRSpec Calibration Software and Database on a “fresh” Mac OS X 10.7 (Lion) system.

v1.2	22/03/2013	PF	Update of section 3.5.3 with the addition of the cosmolopy and NIPPLS libraries / packages. Update of the contents of the PYTHONPATH variable contents in section 3.5.4.

1 INTRODUCTION

Installing the NIRSpec Calibration Software and Database and the JWST_Python tools on a “fresh” Mac OS X Lion system is not as complicated as you might think. It will take a considerable amount of time (half a day), but should be relatively straight forward if you follow this guide.

2 SYSTEM

The installation described here were carried out using a new Mac Book Pro (2011 Model) with 2.2 GHz Intel Core i7 processor and 8 GB 1333 MHz DDR3 RAM. The Max OS X version was 10.7.2.

3 INSTALLING THE SOFTWARE

This section describes how to install the NIRSpec Calibration Software and Database, as well as the post-processing and extraction tools. This follows the guide written up by Giovanna available on the JWST Wiki and Pierre’s “Installation manual for the JWST_Python tools”, bluntly stealing the content in there and adapting it when necessary.

3.1 Get the software

The easiest way to get the most recent version of the software is to download it from the JWST/NIRSpec CVS server. However, the JWST/NIRSpec CVS repository is also available on the web via this link.
In order to set your system up for this, follow the steps below:

1. Set the `CVSROOT` environmental variable to your `.cshrc` by adding the following line
   ```
   setenv CVSROOT :pserver:${USER}@cvs.rssd.esa.int:/services/repositories/JWST_CVS
   ```

2. Open a new terminal window and move to a place where you want to copy the software. In this example we use `~/Library`
   ```
   /> cd ~/Library
   ```

3. Log in cvs (using your normal Unix password):
   ```
   /> cvs login
   ```

4. Check out the directory tree with all the software (IDL, C and Python):
   ```
   /> cvs co Software
   ```
   This command will create the directory `Software` and will populate it with the current version of the IDL and C codes needed for the preprocessing and the database.

5. Get the `REFERENCE DATA` used by the pre-processing pipeline. The reference files are stored at the ftp site ftp://ftp.rssd.esa.int/pub/jwstlib/refData/. Download these files into the sub-directory `Software/JWST_C/var` created by CVS within the directory you have performed the CVS check-out operation in the step above.

Once you have installed the software for the first time you can keep it up-to-date by doing:

```
/> cd ~/Library/Software
/> cvs update -d
```

every now and then.

3.2 Setup local variables

In `~/Library/Software/JWST_IDL/bin/` edit `start_nirspec` with your favorite editor and set the following system variables as needed:

MYIDL_DIR is the path to your IDL installation and in case of IDL8.1 needs to be changed to `/Applications/itt/idl/idl81/`

NIRSPEC_DIR should point to where you have installed the ”Software” via CVS (in this example `~/Library/Software/JWST_IDL/`

DS_DATA repository to the directories with the DS level test data (use `/JWST` if working on the JWST server or you can mount `/JWST` from the server on your laptop, use any other directory if you keep the data locally)

NRS_DATA repository to the directories with the test data (use `/JWST/Archive/` if working on the JWST server or you can mount `/JWST` from the server on your laptop, use any other directory if you keep the data locally)

The shell script `start_nirspec` will configure your system and start a customized IDL session. Therefore, either create a link to it in a directory that is in your `PATH` system variable

```
/> cd ~
/> sudo ln -s Library/Software/JWST_IDL/bin/start_nirspec /usr/local/bin
```

or include the `JWST_IDL/bin/` directory into your `PATH` variable by adding the following line to your `.cshrc`:

```
setenv PATH "${PATH}:~/Library/Software/JWST_IDL/bin"
```

3.3 Install the needed libraries

SQUAT uses cfitsio, DS9, XPA, and Skim.
If you have these libraries already installed you can skip this step, otherwise keep reading.

3.3.1 Installing cfitsio

1. Get the latest version from http://heasarc.gsfc.nasa.gov/fitsio/ (in this example we used V3.280).

2. Copy the file `cfitsio3280.tar.gz` into a working directory.

3. Untar the file:
   ```
   /> tar -xzvf cfitsio3280.tar.gz
   ```

4. Go to the newly created directory:
   ```
   /> cd cfitsio
   ```

5. Configure and compile the library:
   ```
   /> ./configure --prefix=/usr/ --enable-reentrant
   /> make
   /> make testprog
   /> ./testprog > testprog.lis
   /> diff testprog.lis testprog.out
   ```

6. If compilation was successful, there should be no differences. In this case you can move the library in place via:
   ```
   /> sudo make install
   ```

3.3.2 Installing cmpfit

This library is only needed if you want to create linearity correction reference files using the nrs_linearity C-program.

1. Copy the file cmpfit-1.2-stephan.tgz from /Users/Shared/squat_libraries/ on the JWST server (jwst-macserver) to a working directory.
2. Untar the file:
   ```
   /> tar -xvzf cmpfit-1.2-stephan.tgz
   ```
3. Go into the newly created directory:
   ```
   /> cd cmpfit-1.2
   ```
4. Then compile the library and move it into place:
   ```
   /> make
   /> sudo make install
   ```

This will put the created library into /usr/lib/ and the associated header files into /usr/include/. You have to edit the Makefile if you want them somewhere else (but then again, some of the following installation steps might not work).

3.3.3 Installing DS9

There is no Scisoft package available for Mac OS X 10.7 so far. The 10.6 version does install and some programs like iraf seem to work, but DS9 unfortunately does not (frequent crashes). Therefore we have to install DS9 on its own.
As of the date of this document there is also no "official" DS9 release for Mac OS X Lion. However, there is a beta version for DS9 7.0 available that also runs on Lion (command line only, no application). Installation is as follows:

1. Download the latest beta version binary for Mac OS X 10.7 from the DS9 site. Here we used Version 7.0 Beta 14.
2. Copy the file ds9.darwinlion.7.0b14.tar.gz to a working directory.
3. Untar the file:
   ```
   /> tar -xzvf ds9.darwinlion.7.0b14.tar.gz
   ```
4. Move the resulting files ds9 and ds9.zip into /usr/local/bin/ (this might require sudo mv if you do not have enough privileges):
   ```
   /> sudo mv ds9 /usr/local/bin/
   /> sudo mv ds9.zip /usr/local/bin/
   ```

As long as /usr/local/bin is in your PATH system variable, you should now be able to start up DS9 from the command line. Alternatively you can also put it into /usr/bin/.

3.3.4 Installing the XPA Messaging System

Again, there is no pre-build binary available for Lion, so we have to download the source code and compile it.

1. Download the latest XPA version (here we used XPA 2.1.13 from April 2011) from the XPA messaging system site.
2. Copy the file `xpa-2.1.13.tar.gz` into a working directory.
3. Unzip the file:
   ```
   /> tar -xzvf xpa-2.1.13.tar.gz
   ```
4. Go into the newly created directory:
   ```
   /> cd xpa-2.1.13
   ```
5. Configure, make, and install:
   ```
   /> ./configure --prefix=/usr/local/
   /> make
   /> sudo make install
   ```

3.3.5 Installing Skim

For Skim, the latest version 1.3.18 seems to run fine on Lion. In order to install:

1. Download the installer from the Skim website.
2. Locate the file in Finder and click it.
3. Follow the instructions and drag the Skim icon to the Applications folder icon to install Skim.

3.4 Compile the pre-processing pipeline

In order to compile the NIRSpec pre-processing pipeline open a terminal and type:

```
/> cd ~/Library/Software/JWST_C_src/
/> make pipeline
```

If you want, you can also compile `imcombine` (used for combining data cubes for darks etc.) and `nre_linearity` (used to create linearity correction reference files) by typing:

```
/> make imcombine
/> make nrs_linearity
```

If you want to run `pipeline` from outside IDL, make sure that you add the path to `pipeline` to your `PATH` variable by adding the following line to your `.cshrc`:

```
setenv PATH "${PATH}:~/Library/Software/JWST_C/bin"
```

In this case you have to set the path to the appropriate reference files yourself as well, e.g.:

```
setenv REF_DATA ~/Library/Software/JWST_C/var/FPA104_T3850
```

for FPA 104 at a temperature of $T = 38.50\,\mathrm{K}$.

3.5 Installing Python for the JWST_Python tools

This guide follows the document prepared by Pierre Ferruit, available as `INSTALL.doc` in the `JWST_Python/doc` subdirectory once you installed the NIRSpec software from CVS (see Section 3.1 above). It is partly repeated here, with alterations/omissions due to the use of Mac OS X 10.7.

3.5.1 Installing fink

Here we assume that you have a fresh installation without `fink` installed. You can check this by opening a terminal window and type

```
/> which fink
```

If `fink` is already installed you will get something like

```
/sw/bin/fink
```

and you can directly go to step 5 below. Otherwise, in the terminal you should see

```
fink: Command not found.
```

This means that you need to install `fink` by following these steps:

1. Download the latest version of `fink` from the fink webpage to a working directory. Here we used Version 0.31.4

2. Untar the file and go to the newly created directory:

   ```
   /> tar -xvzf fink-0.31.4.tar.gz
   /> cd fink-0.31.4
   ```

3. If at ESTEC, make sure that you are on the wireless (or know all the proxy settings), then execute the following command:

   ```
   /> ./bootstrap
   ```

   and answer all the questions. If on the wireless it is OK to always select the default. Also note that there is no need to select the 64-bit version anymore, since this is now the default (32-bit version is not available anymore).

4. Once this is done, execute the following commands (assuming that you are on a csh/tcsh shell):

   ```
   /> source sw/bin/init.csh
   /> fink selfupdate
   /> fink configure
   ```

   Leave all options unchanged (there is no unstable tree anymore in fink for Lion).

5. Now execute the following commands:

   ```
   /> fink selfupdate
   /> fink update-all
   ```

6. In order to have fink available in your future terminal windows, add the following line to the .cshrc in your home directory (unless you already did this before):

   ```
   source /sw/bin/init.csh
   ```

3.5.2 Using fink to install Python and its basic libraries

The following instructions assume that you are using a terminal window with fink correctly setup (see end of Section 3.5.1).
We are now using fink to install the necessary software. Note that some of the following installation steps will take quite long (up to hours of compilation). During the installation, fink will sometimes ask you to make some configuration choices. Unless directed otherwise in this manual, it is usually safe to select the default option.

1. Execute the following commands:

   ```
   /> fink install python
   /> fink install ipython-py27
   /> fink install scipy-py27
   /> fink install matplotlib-py27
   ```

   and have a coffee or two while compilation is running.

2. If the .matplotlib directory does not exist in your home directory, create it and copy the matplotlibrc file to that directory:

   ```
   /> cd ~
   /> mkdir .matplotlib
   /> cp /sw/lib/python2.7/site-packages/matplotlib/mpl-data/matplotlibrc ~/.matplotlib/
   ```

3. Edit the matplotlibrc file and change the backend type to GTKAgg.

4. Install the pyfits library:

   ```
   /> fink install pyfits-py27
   ```

5. Make a quick test to see if the libraries are correctly installed:

```
/> ipython
In [1]: import scipy
In [2]: import matplotlib
In [3]: import pylab
In [4]: import pyfits
In [5]: exit()
```

You should not get any error messages.

3.5.3 Installing additional libraries

We also need to install some additional libraries that are used by the JWST Python software. Two of them are available as `fink` packages, but the other two will have to be installed by hand.

1. Open a terminal window and execute the following command:

   ```
   /> fink install pil-py27
   ```

2. Download the latest version of `pywcs` from the Python website to a working directory. Here we used Version 1.10-4.7.

3. Then untar and install the package:

   ```
   /> tar -xvzf pywcs-1.10-4.7.tar.gz
   /> cd pywcs-1.10-4.7
   /> sudo python setup.py install
   ```

4. Download the latest version of `Polygon` from the Polygon website to a working directory. Here we used the latest build from 26 October 2011.

5. Then unzip and install the package:

   ```
   /> unzip jraedler-Polygon2-49a5fa3.zip
   /> cd jraedler-Polygon2-49a5fa3
   /> sudo python setup.py install
   ```

6. Install the `bitstring` package:

   ```
   /> fink install bitstring-py27
   ```

7. Retrieve the cosmolopy library archive as a tar or zip bundle from the cosmolopy website.

8. Unpack the tar or zip bundle and in the newly created folder execute the following command (you can delete the folder and the bundle once this is done):

   ```
   /> sudo python setup.py install
   ```

9. Install the NIPPLS package from Bernhard Dorner. Unpack the tar or zip bundle provided by Bernhard and, in the newly created folder, execute the following command (you can delete the folder and the bundle once this is done):

   ```
   /> sudo python setup.py install
   ```

10. If you want make a quick test to see if the libraries are correctly installed:

    ```
    /> ipython
    In [1]: import PIL.Image
    In [2]: import pywcs
    In [3]: import Polygon
    In [4]: import bitstring
    In [5]: import cosmolopy
    In [6]: import NIPPLS
    In [7]: exit()
    ```

    You should not get any error messages.

3.5.4 Setup the environment for using JWST_Python tools

The JWST_Python tools make use of the files present under the `JWST_Python/lib` directory. Thus it is necessary to add the following line to your `.cshrc`:

```
setenv PYTHONPATH /Users/${USER}/Library/Software/JWST_Python_lib:/Users/${USER}
/Library/Software/JWST_Python_lib/JWSTpylib
```

If you have installed the software somewhere else (i.e. not in `~/Library`), you have to modify the path above accordingly, of course.