Chapter 1. PACS Launch Pads

1.1. Introduction

Sep 2014. The pipeline chapters have been updated. This has been written for track 13 of HIPE, but much will also be valid for track 14.

*Welcome to the PACS data reduction guide (*PDRG*) #. We hope you have gotten some good data from PACS and want to get stuck in to working with them. This guide begins with a series of "launch pads" that from Chap. 1; essentially quick-start guides to working with PACS data. These will show you the fastest ways to get your data into HIPE, to inspect the HSA-pipeline reduced cubes (spectroscopy) or images (photometry), and will outline what you need to consider before you start to reduce the data yourself through the pipeline.* For PACS observations we recommend you always re-reduce your data: to include the latest calibrations, so that you can check the intermediate results (deglitching, map-making and masking, flat-fielding, etc), and because some pipeline tasks are not carried out by the automatic pipeline that the HSA-retrieved data were processed through.

Contents: Chap. 2 takes you through what you need to do and know before you start pipeline processing your data, Chap. 3 is dealing with the different PACS photometry pipelines and Chaps 4 and 5 contain more detailed information about photometry data processing (e.g. deglitching and MADmap). In the appendix are issues that are common to photometry and spectroscopy such as explained the organisation and contents of PACS data products.

Additional reading can be found on the HIPE help page, which you can access from the HIPE "*Help>Help Contents*" menu. This covers the topics of: HIPE itself, I/O, scripting in HIPE, and using the various data inspection and analysis tools provided in HIPE. We will link you to the most useful bits of this documentation—we do not repeat the information given there, only material that is PACS-specific is in this *PDRG*. You can also consult the PACS public wiki for the Observer's Manual and calibration information and documentation (herschel.esac.esa.int/twiki/bin/view/Public/PacsCalibrationWed?template=viewprint). This is also linked from the PACS section of the HIPE help page. Information on the calibration of PACS data is **not** covered in this *PDRG*, you have to look at the wiki for these.

1.2. PACS Data Launch Pad

1.2.1. Terminology

The following Help documentation acronyms are used here (the names are links): **DAG**: the Data Analysis Guide; **SG**, the Scripting Guide.

Level 0 products are raw and come straight from the satellite. **Level 0.5** products have been partially reduced, corrected for instrument effects generally by tasks for which no interaction is required by the user. **Level 1** products have been more fully reduced, some pipeline tasks requiring inspection and maybe interaction on the part of the user. **Level 2** products are fully reduced, including tasks that require the highest level of inspection and interaction on the part of the user. **Level 2.5** products, which are found for some of the pipelines, are generally those where observations have been combined or where simple manipulations have been done on the data.

Text written *like this* usually means we are referring to the class of a product (or referring to any product of that class). Different classes have different (java) methods that can be applied to them and different tasks will run (or not) on them, which it why it is useful to know the class of a product. See the SG to learn more about classes. Text written `like this` is used to refer to the parameter of a task.

1.2.2. Getting your data from the archive into HIPE

There are a number of ways to get Herschel data into HIPE, and those data can come in more than one format. The PACS pipelines use one method, which is documented in the pipeline script, but here

we give an overview of the simplest ways to get Herschel data into HIPE. This topic is covered in detail chap. 1 of the DAG.

Herschel data are stored in ESA's Herschel Science Archive (HSA),

- They are identified with a unique number known as the Observation ID (obsid)
- HIPE expects the data to be in the form of a **pool**, so HSA data must be *imported* into HIPE
- A **pool** is like a database, with observations organised as an **Observation Context**, containing links to all the data, calibration, and supplementary files, and including the raw, partially-, and fully-reduced data products

There are several ways to get observations imported from the HSA, or disk, into HIPE:

- Directly from the HIPE command line, using

```
obsid = 134..... # enter your own obsid
# to load into HIPE:
myobs = getObservation(obsid, useHsa=True)
# to load into HIPE and at the same time to save to disk:
myobs = getObservation(obsid, useHsa=True, save=True)

# You must be logged on to the HSA for this to work:
# See the DAG sec. 1.4.5.
```

  See the *DAG* sec. 1.4.5 for more information on getObservation (for example, how to log-on to the HSA before you can get the data). If you use the parameter save=True in getObservation then the data are at the same time saved to disk to your MyHSA: note that saving can take a while. Information and links about accessing MyHSA are also provided in sec. 1.4.5.

  This method is useful for single observations and brings the data directly into HIPE in the format that the PACS pipeline requires.

- Download a tar file (which is not a pool) from the HSA. See the *DAG* sec. 1.4.7 for more information on this. This is the method to use if you have several observations, or a very large one.

  If you are getting a tarfile, then you will have gone into the HSA, identified your observations, and asked for them to be send via ftp. From the tar file you get, you will need to import the data into HIPE. This is explained in the *DAG* sec. 1.5; to summarise, after having untarred the file you do the following in HIPE:

```
#   Get the data from the HSA as a tarball
#   On disk, untar the tarball, e.g.
cd /Users/me/fromHSA
tar xvf meme1342.tar

# in HIPE, then
myobsid=1342..... # enter your obsid
mypath="/Users/me/fromHSA/me1342
myobs=getObservation(obsid=myobsid,path=mypath)

# obsid is necessary only if more than one observation
# is in that directory, i.e. if your tarfile has several
# obsids in it
```

- Import directly into HIPE from the HSA via its GUI, which is useful for single observations and brings the data directly into HIPE in the format that the PACS pipeline requires. This is covered in the *DAG* sec. 1.4.5 ("using the GUI: HUI"). Briefly, you need to find your observation (e.g. by obsid), then in the query results tab of the HUI, click the download icon to access a "Retrieve Products" menu, from which select to download "All".

Tip

To get to the HSA from HIPE use the viewer from the HIPE menu *Window>Show View>Data Access>Herschel Science Archive*. Anything you download directly into HIPE from there will appear in the *Variables* pane of HIPE, and often the command will also be echoed to the *Console*.

After you have done this you can:

- If you loaded your data into HIPE using the first or third method above, then you must save the observation to disk.

  If using getObservation you can chose two ways to save the data to disk: either to your MyHSA, using the parameter save=True in getObservation as explained above; or to a pool on disk, using the command saveObservation.

  To use saveObservation:

```
# "myobs" is the name of the ObservationContext
pooln = "NGC3333"
saveObservation(myobs, poolName=pooln)
```

  where the observation goes to your **local store** (HOME/.hcss/lstore) to a directory (pool) with the name that is either the obsid or as it given in `poolName`. The *DAG* does not explain saveObservation but it does explain the same parameters of getObservation: see sec. 1.7.

  Note: if using saveObservation, the *ObservationContext* is not saved by default with the calibration tree (calibration files) it was reduced with. If you wish to do that, see Sec. 2.6: you need to set the parameter saveCalTree=True, which will save whatever calibration tree is in your *ObservationContext*.

  f you used the second method you do not need to save your observation as a pool, you could keep them as they are. If you do want to convert your observation data on disk from the "HSA-format" to the pool format, you can simply use saveObservation as in the example above.

- Once the data are saved to this pool with saveObservation, on your disk, then any other time you want to access them you should use getObservation,

```
obsid = 134...... # enter your obsid here
pooln = "NGC3333"
myobs=getObservation(obsid, poolName=pooln)
```

  The *DAG* explains these parameters of getObservation: see sec. 1.7 to know how to access the data if they were saved elsewhere than your local store. The command will get the data as located in the local store (HOME/.hcss/lstore) and with a directory (pool) name that is either the obsid (e.g. "342221212") or the value set by `poolName`. You can also use a GUI to get data, e.g. from your MyHSA: see sec. 1.7 of the *DAG*.

So, to summarise the command-line methods:

```
obsid = 134...... # enter your obsid here

# Direct I
#   Get the data from the HSA and then save to
#   /Users/me/.hcss/lstore/MyPoolName
myobs=getObservation(obsid, useHsa=True)
saveObservation(myobs, poolName="MyPoolName")
#   Then, to later get those data
myobs=getObservation(obsid, poolName="MyPoolName")
#
#   Get the data from the HSA and then save to
#   /Users/me/.hcss/lstore/[obsid as a string]
myobs=getObservation(obsid, useHsa=True)
saveObservation(myobs)
#   Then, to later get those data
```

```
myobs=getObservation(obsid)

# You must be logged on to the HSA for this to work:
# See the DAG sec. 1.4.5.
# See Sec. 2.6 to learn about saving and then
# restoring the caltree



# Direct II
#   Get the data from the HSA and immediately save
#   to /Users/me/.hcss/lstore/MyPoolName
myobs=getObservation(obsid, useHsa=True, save=True, poolName="MyPoolName")
#   Then to later get those data
myobs=getObservation(obsid, poolName="MyPoolName")



# DIRECT III
#   Get the data from the HSA as a tarball
#   On disk the data are in directories off of /Users/me/fromHSA
#   In HIPE, then
myobsid=1342..... # enter your obsid
mypath="/Users/me/fromHSA/me1342
myobs=getObservation(obsid=myobsid,path=mypath)
obsid = 134...... # enter your obsid here
mypath="/Users/me/fromHSA/me1342
myobs=getObservation(obsid=myobsid,path=mypath)
```

For the GUI-based methods, read chap. 1 of the DAG. For full parameters of getObservation, see its URM entry: here. (Note: there are two "getObservation"s in the URM. The one I link you to is the correct one, it is also the first in the URM list.)

1.2.3. Looking at your fully-reduced data

Once the data are in HIPE, the *ObservationContext* will appear in the HIPE Variables panel. To look at the fully-reduced, final Level 2 product (images for the photometer) do the following,

- Double-click on your observation (or right-click and select the **Observation Viewer**)
- In the directory-like listing on the left of the Observation viewer (titled "Data"), click on the + next to the "level2"
- Go to HPPMAPB to get the blue map or the HPPMAPR to get the red Naive map. The map will open to the right of the directory-like listing, but if you want to view it in a new window then instead double-click on the "HPPMAPB" (or right-click to select the **Standard Image Viewer**

To learn more about the layers of the *ObservationContext* and what the products therein are, see Sec. ???.

1.3. PACS Photometry Launch Pad

The following Help documentation acronyms are used here: **DAG**: the Data Analysis Guide; **PDRG**: PACS Data Reduction Guide.

1.3.1. Does the observation data need re-processing?

The PACS ICC recommend that you always reprocess your data,

- The pipeline tasks and the calibrations are still undergoing improvement and the pipeline that the reduced data you got from the HSA may not have been the most recent
- There are some pipeline stages that for all but the most simple data you ought to inspect the results of the pipeline task yourself, to decide if you wish to change the default parameters

Information about calibration files held in the **calibration tree**:

- When you start HIPE, HIPE will begin by looking for a calibration file update: Sec. 2.5.1.
- To check what version of calibration files and the pipeline your HSA-gotten data were reduced with, and to compare that to the current version and to see what has changed, see Sec. 2.5.4.
- You can also look at the Meta data called calTreeVersion, see Sec. 2.5.4.
- To load the calibration tree into HIPE when you pipeline process, see Sec. 2.5.3.

1.3.2. Re-processing with the pipeline scripts

The subsequent chapter of the *PDRG*, linked to below, cover different pipelines each.

The pipeline script you will run will depend on the observing mode and the science target,

- *Chopped point source data*: see Sec. 3.3 for observations taken in chop-nod photometry mode (an old mode).
- scan-map and mini scan-map for point sources*: see Sec. 3.2.1 for observations containing mainly point sources and small extended sources*
- Extended sources using MADMap*: see Chap. 3.2.2 for observations of extended sources (only use when scan and cross scan data are taken).*
- Extended source using JScanam *see Sec. 3.2.3 for observations of extended sources (only use when scan and cross scan data are taken).*
- *Extended source using Unimap* see Sec. 3.2.4.2 for observations of extended sources.
- The pipeline scripts contain all the pipeline tasks and simple descriptions of what the task are doing. But if you want to know all the details you need to consult the pipeline chapters (links above). Individual pipeline tasks are also described in the PACS *User's Reference Manual* (PACS *URM*).
- The pipelines take you from Level 1 ((calibrated data cubes in Jy/detector pixel)) to Level 2 (fully-processed). If a Level 2.5 is done, that means maps have been combined.

To access the scripts, go to the HIPE menu *Pipelines>PACS>Photometer*. The scripts assume:

- The data are already on disk or you can get them from the HSA using getObservation (so you must know the Observation ID)
- You have the calibration files on disk; normally you will use the latest update, but you can run with any calibration tree version: see Sec. 2.5.3 to know how to change the version of the calibration tree you are using.
- You chose to do the red or the blue camera separately

To run the scripts,

- Read the instructions at the top, and at least skim-read the entire script before running it
- Although you can run most all in one go, it is *highly* recommended you run line by line at least for the first time
- If you are going to comment within the script or change parameters, then first copy the script to a new, personalised location and work on that one (HIPE menu *File>Save As*): otherwise you are changing the script that comes with your HIPE installation

As you run the scripts,

- Plotting and printing tasks are included with which you can inspect the images and masks themselves. The plots will open as separate windows
- The scripts will save the data into FITS files after each Level (this is a difference with the spectroscopy pipeline)

1.3.3. Considerations when running the pipeline

Considerations concerning the technicalities of running the pipeline are:

- If you chose to run the pipeline remotely or as part of bulk processing you might want to disable the plotting tasks by commenting out the lines starting with "Display(...)"
- **Memory vs speed**: the amount memory you assign to HIPE to run the pipeline depends on how much data you have, but >=4Gb for sure is recommended.

  If you wish to fiddle with your data (other than using the plotting tasks provided in the pipeline) it would be a good idea to do that in a separate running of HIPE.
- Save your data at least at the end of each Level, because if HIPE crashes you will lose everything that was held only in memory (the scripts, by default save your data after each Level so DO NOT modify that part)

Things to look out for in your data as you run the pipeline are:

- Saturated and Glitched data
- Non-smooth coverage map (the coverage map is not uniform but the transitions should be fairly smooth towards the edges)
- Up and down scan offsets (distorted Point Spread Function)
- Dark spots around bright point sources (sign of inappropriate high-pass filtering)

1.3.4. Further processing

There are a number of tasks that can be used to inspect and analyse your PACS Level 2 images. For a first quick-look inspection (and even for some image manipulation) we recommend the tasks' GUIs. The tasks are listed in the *Tasks* panel under Applicable if the image is highlighted in the Variables panel. Double-click on the task will call up its GUI, except for the Standard Image Viewer which is invoked by a right-click on the image in the Variables panel and selecting *Open with>Standard Image Viewer*

- If you just want to look at the images you can use the **Standard Image Viewer**: see *Sec 4.4 of the DAG*:
- The **annularAperturePhotometry task**: (see *Sec 4.21 of the DAG*) Performs aperture photometry using simple circular aperture and a sky annulus. There are other aperture photometry tasks: fixedSky, pacsAnnularSky, rectangular.
- The **sourceExtractorDaophot and sourceExtractorSussextractor**: (see *Sec 4.19 of the DAG*) Extracts sources from a simple image using different algorithms.
- The **sourceFitter**: (see *Sec 4.20 of the DAG*) Fits a 2D Gaussian to a source in a specified rectangular region on an image.

See the image analysis chapter of the ***Data Analysis Guide*** chap. 4 for more information on image processing in HIPE.

Chapter 2. Setting up the pipeline

2.1. Terminology

Level 0 products are raw and come straight from the satellite. **Level 0.5** products have been partially reduced and corrected for instrument effects generally by tasks for which no interaction is required by the user. **Level 1** products have been more fully reduced, some pipeline tasks requiring inspection and maybe interaction on the part of the user. **Level 2** products are fully reduced, including tasks that require the highest level of inspection and interaction on the part of the user. **Level 2.5** products, which are found for some of the pipelines, are generally those where observations have been combined or where simple manipulations have been done on the data.

The *ObservationContext* is the product class of the entity that contains your entire observation: raw data, HSC-reduced products (levels), calibration products the HSC reduced with, auxiliary products such as telescope pointing, and etc. You can think of it as a basket of data, and you can inspect it with the Observation Viewer. This viewer is explained in the *Herschel Owners Guide* chap. 15, and what you are looking at when you inspect a PACS *ObservationContext* is explained in Sec. ???.

The Level 2 (and also 2.5) photometry product is a *SimpleImage* that contains a standard two-dimensional image, in particular the following arrays: "image" as an array 2D (e.g. double, integer); "error" as an array 2D (e.g. double, integer); "exposure" as an array 2D (e.g. double, integer); "flag" as a short integer array 2D. It also contains Meta data that provide unit and World Coordinate System information. The definition of *Frames* give above is valid also for photometry. The photometry pipeline does not push the products into *ListContexts* as it does not use slicing.

To learn more about what is contained in the *ObservationContext and Frames*, see Sec. ???.

The following (Help) documentation acronyms are used here: *DAG*: the Data Analysis Guide; *PDRG*: this PACS Data Reduction Guide; HOG*:* HIPE Owner's Guide.

2.2. Getting and saving your observation data

2.2.1. Getting

The fastest ways to get the *ObservationContext* into HIPE were explained in Sec. 1.2. We expand on that here, but do first read Sec. 1.2. If you get your data via the HSA-GUI as a "send to external application" then it should be an *ObservationContext* already.

If you have the data already on disk but as gotten from the HSA as a tarball:

```
# on disk, untar the tarball, e.g.
cd /Users/me/fromHSA
tar xvf meme1342.tar
# look at it: ls meme1342

# in HIPE,
myobsid=1342..... # enter your obsid
mypath="/Users/me/fromHSA/me1342
myobs=getObservation(obsid=myobsid,path=mypath)

# obsid is necessary only if more than one observation
# is in that directory, i.e. if your tarfile has several
# obsids in it
```

Get the data from the HSA directly on the command line:

```
obsid = 134..... # enter your obsid here

# Direct I
#   Get the data from the HSA and then save to
```

```
#   /Users/me/.hcss/lstore/MyPoolName
myobs=getObservation(obsid, useHsa=True)
saveObservation(myobs, poolName="MyPoolName")
#   Then, to later get those data
myobs=getObservation(obsid, poolName="MyPoolName")
#
#   Get the data from the HSA and then save to
#   /Users/me/.hcss/lstore/[obsid as a string]
myobs=getObservation(obsid, useHsa=True)
saveObservation(myobs)
#   Then, to later get those data
myobs=getObservation(obsid)

# You must be logged on to the HSA for this to work:
# See the DAG sec. 1.4.5.
# See later to learn about saving and then
# restoring the caltree

# Direct II
#   Get the data from the HSA and immediately save
#   to /Users/me/.hcss/lstore/MyPoolName
myobs=getObservation(obsid, useHsa=True, save=True, poolName="MyPoolName")
#   Then to later get those data
myobs=getObservation(obsid, poolName="MyPoolName")
```

Or if the data are on a pool on disk (not ex-tarfile format, but HIPE-format), you use:

```
# for data in [HOME].hcss/lstore/me1234
obsid=1342..... # enter your obsid
myobs=getObservation(obsid,path=mypath)
```

The full set of parameters for getObservation can be found in its URM entry: here. (Note: there are two "getObservation"s in the URM. The one I link you to is the correct one, it is also the first in the URM list.)

2.2.2. Saving

You use the task saveObservation for this, and to run this task with all the parameters set:

```
# To save in /Users/me/bigDisk/NGC1 where "bigDisk" is a replacement for
# the "local store" default location (see below)
pooln="NGC1"
pool="/Users/me/bigDisk"
saveObservation(obs, poolName=pooln, poolLocation=pooll,
saveCalTree=True|False, verbose=True|False)
```

Where the only parameter you *need* to set is the "obs"—by default the data is saved to HOME/.hcss/lstore/[obsid as a string]. All other parameters are optional. The data will be saved to a pool (directory) located in the local store, whether that local store is the default HOME/.hcss/lstore or /Users/me/bigDisk as in the example above.

Or, as already mentioned above, you can save as you get the data:

```
# Direct II
#   Get the data from the HSA and immediately save
#   to /Users/me/.hcss/lstore/MyPoolName
myobs=getObservation(obsid, useHsa=True, save=True, poolName="MyPoolName")
#   Then to later get those data
myobs=getObservation(obsid, poolName="MyPoolName")
```

You can save to anywhere on disk, though by default the data go to [HOME]/.hcss/lstore with a `poolName` that is the obsid (observation number) as a string. If the directory does not exist, it will be created. If it does, then new data are added to it. Note that if you add the same obsid to the same pool a second time, then using getObservation later to get the *ObservationContext* will get you only the latest saved data. There is a parameter, `saveCalTree`, which is a switch to ask to save the calibration tree that is contained in the *ObservationContext* (myobs): True will save it, and the default False will not. Saving with the caltree takes up more space on disk and more time to work, but if you want to be able

to access the calibration tree that the data were reduced with by the pipeline (either that which the HSA ran or that which you run), you should first attach the calibration tree to the *ObservationContext* and then set this parameter to True. If you have gotten the data just now from the HSA then the calibration tree will be attached.

Alternatively, the task getObservation also has a parameter that will save the data to disk, to your MyHSA, and including the calibration tree. See the URM entry to learn more, and see also the *DAG* sec. 1.4 to learn more about getObservation, used on data from the HSA or from disk.

2.3. What and where are the pipeline scripts?

In the following chapters we describe how to run the photometry pipelines that are offered via the HIPE *Pipeline* menu. In this chapter we explain the setting up of the pipelines. You will then skip to the chapter that is of the pipeline appropriate for your AOT.

All the photometry pipelines are standalone and provide a full processing of your data, with all the necessary steps required to produce a FITS image of your science target. Here we give a short summary of the purpose of each pipeline, although their names are quite self explanatory.

- *Chopped point source data*: see Sec. 3.3 for observations taken in chop-nod photometry mode (an old mode).
- *scan-map and mini scan-map for point sources*: see Sec. 3.2.1 for observations containing mainly point sources and small extended sources
- *Extended sources using MADMap*: see Chap. 3.2.2 for observations of extended sources (only use when scan and cross scan data are taken).
- *Extended source using JScanam* see Sec. 3.2.3 for observations of extended sources (only use when scan and cross scan data are taken).
- *Extended source using Unimap* see Sec. 3.2.4.2 for observations of extended sources.

The pipeline scripts can be found in the HIPE *Pipelines>PACS>Photometer* menu. Load and copy (*File>Save As*) to a unique name/location the pipeline script you want to run, because otherwise if you make changes and save the file, you will be overwriting the HIPE default version of that pipeline script. Henceforth, to load your saved script you will use the HIPE *File>Open File* menu. Read the instructions at the beginning of the script and at least skim read the entire script before running it. They are designed such that they can be run all in one go, after you have set up some initial parameters, but it is recommended that you run them line by line, so you have better control over them.

We remind you here that you should consult the AOT release notes and associated documentation before reducing your data. These inform you of the current state of the instrument *and the calibration*. Information about the calibration of the instrument will be important for your pipeline reductions—any corrections you may need to apply to your data after pipeline processing will be written here. Information about spectral leakages, sensitivity, saturation limits, and PSFs can also be found here. These various documents can be found on the HSC website, in the PACS public wiki: here.

Note

Spacing/tabbing is very important in jython scripts, both present and missing spaces. Indentation is necessary in loops, and avoid having any spaces at the end of lines in loops, especially after the start of the loop (the if or for statement). You can put comments in the script using # at the start of the line.

2.4. How much can you improve on the automatic pipeline?

Before you being pipeline reducing the data yourself, it is a valid question to ask: how much can I improve on what I have already seen in the HSA-obtained Level 2 product (better known as the

"SPG"—Standard Product Generation—data)? The answer to this depends on when the data you have were reduced by the PACS Standard Product Generation pipeline that is run by the Herschel Science Centre to populate the Herschel Science Archive, and on the type of observation you have. The data products contained in the Herschel Science Archive might be produced by a previous pipeline version, and therefore some of the algorithms and calibration files it used may be older than those in your version of HIPE(how to check is shown in Sec. 2.5.4). Note that you can always use the on-demand processing option provided by the Herschel Science Archive to run the latest version of the PACS Standard Product Generation pipeline. This option is especially interesting for those who do not have a machine with tens of Gigabytes of RAM that is needed to perform PACS data reduction. The pipeline is continually being updated. In any way it is always advisable to inspect your level2/level2.5 data to see whether the parameters with which the SPG pipeline was run are appropriate for your observations. To check which version of HIPE the SPG data were reduced with, type, in the Console of HIPE, the following: HIPE> print obs.getMeta()["creator"].string where "obs" is your ObservationContext; you can also look at the version of the calibration tree with: HIPE> print myobs.["calTreeVersion"].long. If you are reading this PDRG via HIPE then you will be working in Track 10|11 of HIPE. To figure out what calibration tree is the latest, simply load it and look:

```
calTree=getCalTre(obs-myobs)
print calTree
# version number is printed near top of listing
```

2.5. Calibration files and the calibration tree

2.5.1. Installing and updating the calibration files

First, you should consult the AOT release notes and associated documentation (e.g. Observer's Manual and Performance and Calibration documents), these being important for informing you of the current state of the instrument *and the calibration*. Information about spectral leakages, sensitivity, saturation limits, ghosts and PSFs can also be found there. These various documents can be found on the HSC website, in the PACS public wiki: here.

The calibration files are not provided with the HIPE build, rather you are offered to chance to update them only when they need to be updated. If you open HIPE and you get a pop-up telling you to install the calibration products, it means that the calibration file set has been updated by the PACS team and you are being offered the chance to get that update. Click on "Install" and the new calibration products will be downloaded and installed. They are placed in [HOME]/.hcss/data/pcal-community (or pcal-icc, but only for the PACS team).

If this is the very first time you are using HIPE and hence you have never installed any calibration files before, then you should select "Install", otherwise you will have no calibration files at all. If you have done this before, and hence you do have a calibration file set, then you can chose whether to update or not. Why would you not? Well, if you are in the middle of processing data you may want to continue with the calibration files you are already using, rather than downloading new files and possibly having to start again (for consistency's sake), although just because you update does not mean you need to use the updated calibration tree: see Sec. 2.5.3 for information about how to set the calibration tree version you use in the pipeline.

2.5.2. Checking what has been updated

The updater GUI tells you which calibration files have been changed. To see the relevant information about the release, in the calibration updater pop-up click on "Show details...". In the new panel that appears, look at the "Release Notes" tab for a summary of the new set version. In there will be listed the calibration files (the FITS files) that have been included in the update and information about the changes made.

You can also look at the individual "Files" tab to see what (if anything) has changed in the individual files that are being updated. Some files will have no information in them, most of the information is

in the Release Notes tab, and in the Files tab in the files called PCalBase_TimeDependency_FM_v#, which also contain a summary of the release. If more than one version number of calibration files are listed, you will be more interested in the highest version number.

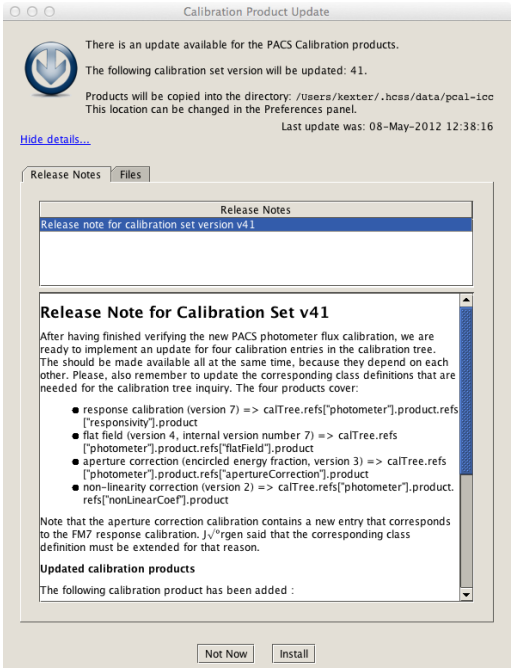

Figure 2.1. Updating the calibration files

To check on which pipeline tasks this will affect, check the pipeline scripts, as they comment on which calibration files are used by the tasks that use calibration files (e.g. you are told "# used cal files: observedResponse, calSourceFlux" for the task specDiffCs).

The calibration files take up about half a gigabyte, so you may need to install them in a directory other than the default [HOME]/.hcss/data/pcal-community. If you want to install them elsewhere then: in the updater pop-up click "Not Now"; go to the HIPE Preferences panel (from the Edit menu); click on *Data Access>Pacs Calibration*; in the "Updater" tab that is now in the main panel change the name of the directory in the space provided. Do not click on anything else—you do want to use the "Community Server" as these are the products that have been tested, the "ICC" ones are still in the process of being validated. Click to Apply and Close. Then go to the Tools menu of HIPE, and select *pacs-cal>run Updater*. Voilà.

You can also inspect the calibration sets and products with a **Calibration Sets View**. This allows you to inspect the calibration sets that have been installed on your system. You get to this view via the HIPE menu *Window>Show View>Workbench>Calibration sets*. The view will show the release notes for the selected set (numbered boxes at the top), or the calibration file list for the selected set (viewing the notes or the file list are chosen via the central drop-down menu). The calibration file list is just a list of what calibration files, and their version numbers, are included in the selected set, and the release note you will see is the general one for that set. A new release of a calibration set will include some updated calibration files and also all the rest that have not changed.

2.5.3. The calibration tree

Before beginning the pipeline you will need to define the calibration tree to use with your reductions. The calibration tree contains the information needed to calibrate your data, e.g. to translate grating position into wavelength, to correct for the spectral response of the pixels, to determine the limits above which flags for instrument movements are set. The calibration tree is simply a set of pointers to the calibration files in your installation, it is not the calibration files themselves. Tasks that use

calibration files will have the parameter `calTree`, which you set to the name you have given to the calibration tree (see below).

To use the latest calibration tree you have in your installation is done with,

```
calTree=getCalTree(obs=myobs)
```

Where "obs=myobs" is setting the parameter `obs` to the *ObservationContext* you are going to be working on, here called "myobs". This is done so that those few calibrations that are *time-specific* will take, as their time, the time of your observation.

If you want to reduce your data with an older calibration tree, you can do this simply by typing

```
calTree=getCalTree(version=13) # to use version 13
```

If you want to use the calibration tree that is with the *ObservationContext* (assuming it has been saved there), you type,

```
calTree=myobs.calibration
```

This will always be present if you have just gotten the data from the HSA, and will be present if whoever saved the *ObservationContext* remembered to save it with the calTree (see Sec. 2.6).

2.5.4. Comparing calibration file versions

To compare the version of the calibration files you will use by default when you begin pipeline processing your data, to those used by the HSC when the automatic pipeline was run, you do the following: where "myobs" is the name of the *ObservationContext*, type,

```
# The caltree that comes with you data
print myobs.calibration
print myobs.calibration.spectrometer
# The caltree you have on disk, this is the command that loads
# the calibration tree
#   that you will later use when you run the pipeline
calTree=getCalTree(obs=myobs)
# And to then inspect it
print caltree
print caltree.spectrometer
# Now you can compare all the version numbers that are printed
# to Console
```

The parameter `obs` (set to myobs here) simply specifies that the calibration tree will take the versions of the calibration files that are from the time that your observation took place, for those few calibration files which are time-sensitive.

Note that to print out the information on the calibration tree from "myobs" (the first command in the script above) it is necessary that the calibration tree is there in "myobs". This will be the case for SPG reduced data if you have only just gotten it from the HSA and loaded it into HIPE. But if you used saveObservation to save it first to disk, or if you are looking at an *ObservationContext* someone gave you, then to get hold of the calibration tree of that *ObservationContext* it must be that the calTree was attached to and saved with the ObservationContext when running saveObservation. This is done by using the `saveCalTree`=True option, as explained in the next section. For this reason it may also be worth saving the calibration tree you will use when *you* reduce your data.

You can also check the calibration version your HSA-data were reduced with by looking at the Meta data "calTreeVersion" in the *ObservationContext*. This gives you the "v"ersion number of the calibration tree used to reduce those data,

```
print obs.meta["calTreeVersion"].long
```

To find out what version of HIPE your data were reduced with, check the Meta data called "creator", it will tell you something like SPG V7.3.0, which means Standard Product Generator in HIPE v 7.3.0.

2.6. Saving your *ObservationContext* and its calibration tree to pool

As stated previously, and repeated here, if you wish to save the calibration tree with your ObservationContext, then you should follow these instructions for the command-line methods:

```
obsid = 134...... # enter your obsid here

# example I: save with saveObservation to $HOME/.hcss/lstore/MyFirst
myobs=getObservation(obsid, useHsa=True)
saveObservation(myobs,poolName="MyFirst",saveCalTree=True)
# followed later by
myobs=getObservation(obsid, poolName="MyFirst")
calTree=obs.calibration

# example II: save when you get from the HSA
myobs=getObservation(obsid,useHsa=True,save=True,poolName="MyFirst")
# then later:
myobs=getObservation(obsid,poolName="MyFirst")
calTree=myobs.calibration

# via tarfile
file = "/Users/me/me10445555"
myobs = getObservation(obsid,file)
calTree=myobs.calibration
```

Why you would want to save the calibration tree? Whether you are saving data you got directly from the HSA, or data you have pipeline reduced yourself with the latest calibration tree, it is worth saving the fully-reduced *ObservationContext* with the caltree so that if you later wish to compare the reductions to later ones you do, you can at least check that the calibration trees are the same; and so that when you write up your results, you can find out which calibration tree you used. But otherwise you do not need to: the calibration files themselves are held on disc, all you need to know is the calibration tree version that was used to reduce the data.