Chapter 1. PACS Launch Pads

1.1. Introduction

May 2012. The PDRG of Track 8 has been split into two: a Spectroscopy and a Photometry document. You are reading the spectroscopy data reduction guide. As well as splitting from photometry, the spectrosopy pipeline chapters have been updated for the new pipeline scripts, including descriptions of the improved flatfielding and point source loss correction tasks. The wavelength switching pipeline chapter has been added. This has been written for track 9 of HIPE, but much will also be valid for track 10.

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:

- Section 1.2 the Launch Pad for Data, which is a summary of getting PACS data into HIPE and how to have a quick look at them. This is followed by the Spectroscopy Launch Pad (Section 1.3), which is a summary of how to run the pipelines and what to consider before and while you run them.
- Chapter 2 takes you through what you need to know and do before you start pipeline processing, such as the calibration files, getting and saving data, where the pipeline scripts are and more.
- Chapter 3, Chapter 4, Chapter 5 are the pipeline chapters for the chop-nod, unchopped, and wavelength switching AOTs.
- Chapter 6 includes scripts and general information about plotting PACS data for diagnostic reasons as you pipeline process; shows you the various viewers you can use to inspect your PACS data; shows you how to work with masks; and how to combine the spectra of different spaxels.
- Chapter 7 contains the guide to "drizzle", which is an advanced spectral projection task that combines the cubes of dithered observations for single wavelength regions and creates a larger, and spatially better sampled cube. This task is an alternative to the final pipeline task "specProject", which is much faster and more general but not as good for AORs that requested spatial dithering. This chapter also explains how to run specProject on fitted cubes (i.e. you fit the spectra and then run the spectral projection task on them, rather than first spectrally projecting and then fitting), and finally how to combine PACS and SPIRE SED observations of point sources.

  > **Tip**
  >
  > *spectral projection* simply means taking one cube, with its spatial layout, and for each wavelength point and each spaxel, projecting the data onto a new spatial grid (e.g. smaller spaxels and a more regular spatial grid), to make a new cube.

- Chapter 8 the appendix includes information on the layout of PACS data—what are in them, how to inspect them. We then give more detail (in the form of information tables) on the "information" arrays (Status, BlockTable, Meta data) held in PACS products and which are used by certain pipeline tasks.

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 necessary 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. A quick on terminology

The following Help documentation acronyms are used here (the names are links): **DAG**: the Data Analysis Guide; **SaDM**, the Scripting and Data Mining 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 SaDM 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 you used getObservation you could 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 full parameters of this task (including: saving the calTree, savin to elsewhere than the local store) can be found in Sec. 2.2.

  Note: if using saveObservation, the *ObservationContext* is not saved by default with the calibration tree it was reduced with. If you wish to do that, see Sec. 2.6.5: you need to set the parameter `saveCalTree`=True, which will save whatever calibration tree has been added to your *ObservationContext*.

If 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. "134......") 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. 2.6.5 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)
```

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* pane. To look at the fully-reduced, final Level 2 cubes, 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"
- There will be two types of Level 2 cubes:

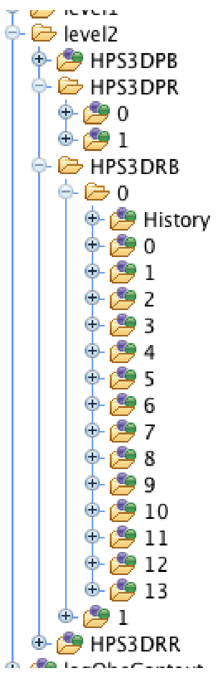

Figure 1.1. The cubes of Level 2: the Projected with one cube per wavelength range (0 and 1) and per pointing (0 to 13), and the Rebinned wiht one cube per wavelength range (-0 and 1)

- The "rebinned cubes", of class *PacsRebinnedCube* and held in a *ListContext* (basically a list container for products). There will be one cube per wavelength range and per pointing specified in your observation. These cubes have 5x5 spaxels of size 9.4 arcsec on a side, and note that they have an irregular WCS (which you will not notice when you look at them in a viewer: see Sec. 3.7.4). **These are the final science-grade cubes of the pipeline for staring observations**, and also for mapping observations that have large offsets (i.e. which were not executed in a spatial dither pattern).
- The "projected cubes", which are of class *SpectralSimpleCube* and are held in a *ListContext*, and which are the result of the pipeline task specProject. There will be one cube per wavelength range specified in your observation. These cubes have spaxel sizes of 3 arcsec. These are the mosaicked and spatially-spectrally regridded versions of the rebinned cubes (hence on cube per wavelength: the "per pointing" have been combined). **These are the final science-grade cubes for mapping observations**, and mainly so if you used a dithering mapping pattern. For such observations you may want to try the task **drizzle** (Chapter 7), rather than the default pipeline task, to perform an alternative spectral-spatial projection. For this you need to re-reduce the data (as you ought to in any case) and near the end of the pipeline, step out and run drizze (read Chapter 7).

  See e.g. Sec. 3.7.4 to learn more about the differences between these two cubes.

- Go to the + next to the HPS3DPR (red *projected* cube) or HPS3DPB (blue *projected* cube) or to the + next to the HPS3DRR (red *rebinned* cube) or HPS3DRB (blue *rebinned* cube). The list of numbers

(+0, +1...) are the individual cubes, one per wavelength range specified in your AOR (observation request) and also one per pointing for the rebinned cubes (see the figure above). The tooltip that appears when you hover over a cube in the listing will help you work out what the contents of that cube is. Click on the number (the "0"s and "1"s in the figure above, not the "0 to 13"s) to see the cube open to the right of the directory-like listing, or to see it in a new window, right-click and chose the **Spectrum Explorer** (SE: see the *DAG* chap. 6). From the SE you can look at the spectra of your spaxels, perform mathematical operations, extract spectra or sub-cubes, make velocity and flux maps and fit your spectra.

To learn more about the layers of the *ObservationContext* and what the products therein are, see Sec. 8.2.

1.3. PACS Spectroscopy Launch Pad

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

A *spaxel* is a spatial pixel. It is the spatial unit of the integral field spectrograph that PACS is. Each spaxel contains the spectrum from one unique "pixel" on the sky. The native spaxel sizes of PACS are 9.4x9.4 arcsec. (Note: the HIFI term for spaxel is pixel.)

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 automatic pipeline that the HSC runs may not have been the most recent. Currently there are still some tasks or refinements that the HSA data were not subjected to, which you can do if you reprocess yourself. This included the spectral flatfielding.
- There are some pipeline stages that for all but the simplest targets and observations you ought to inspect the results of the pipeline task yourself, to decide if you wish to change the default parameters; inspecting these intermediate products is much easier if you re-run the pipeline. This includes the flatfielding, and subtracting the off-source data ("telescope" background) from the on-source data (your astronomical object)

The calibration files used by the pipeline are held in a **calibration tree**:

- When you start HIPE, HIPE will begin by looking for a calibration file update: Sec. 2.6.1.
- 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. 2.6.4.
- You can also look at the Meta data called "calTreeVersion" (print myobs.["calTreeVersion"].long) to see the "v"ersion number of the calibration tree used to reduce your *ObservationContext* by (called "myobs") by the automatic pipeline: see Sec. 2.6.4.
- To load the calibration tree into HIPE when you pipeline process (and which is not necessary until then), see Sec. 2.6.3. (Quick guide:calTree=getCalTree(obs=myobs), and then to see what version it is: print myobs.calTree)

The *ObservationContext* you get from the HSA will come with a calibration tree, the one the HSA used. If you are reprocessing the data, you should use the latest calibration tree instead.

1.3.2. Re-processing with the pipeline scripts

Subsequent chapter of the *PDRG*, linked to below, cover the different pipelines.

The pipeline script you will run will depend on the observing mode.

- *Chop-Nod line scan and short (wavelength) range scan*: Sec. 3.3 for the standard pipeline; Sec. 3.5 for the background normalisation pipeline script, which offers an alternative background subtraction and flux calibration to the standard pipeline; Sec. 3.4 for a "testing your data" pipeline, where on-source and off-source cubes are created so you can compare the background that will be subtracted from the source spectra (after running this script you have to go back to run one of the other two pipelines).

  If your data are from the range scan AOT but the range is short, the read the advice at the top of Chap. ??? to decide whether to run the line scan+short range scan pipeline or the large range and SED pipeline.

- *Chop-Nod large (wavelength) range and SED scan*: Sec. 3.3 for the standard pipeline; Sec. 3.5 for the background normalisation pipeline script, which offers an alternative background subtraction and flux calibration to the standard pipeline; Sec. 3.6 for a script that will show you how to combine the results into (longer) spectra; Sec. 3.4 for a "testing your data" pipeline, where on-source and off-source cubes are created so you can compare the background that will be subtracted from the source spectra (after running this script you have to go back to run one of the other two pipelines).

- *Unchopped line and short (wavelength) range scan*: Sec. 4.3.

  If your data are from the range scan AOT but the range is short, the read the advice at the top of Chap. ??? to decide whether to run the line scan+short range scan pipeline or the large range and SED pipeline.

- *Unchopped range scan (including SED)*: Sec. 4.3 for a single (on- or off-source) observation, and Sec. 4.4 for an on-source and an off-source observation to be subtracted from each other.

- *Wavelength switching* (an old mode): Chap. 5.

- There is no difference in any of the pipeline scripts in handling point sources and extended sources. The only difference (other than the AOT) is whether you have pointed observations (in which case the final Level 2 cubes that you perform science measurments on should be the *PacsRebinnedCube*s) or mapping (if dithered, then you can perform your final science measuments on the projected cubes created by specProject or drizzle; if large-step rastered you ought to perform science measurments on the *PacsRebinnedCube*s as with pointed). We also offer tasks that will extract and point-source loss correct the spectra of a point source, from a *PacsRebinnedCube*. The differences between the pipelines within each AOT (chopped/unchopped/line scan/range scan) are currently not large.

- The pipeline scripts contain all the pipeline tasks, pipeline helper plotting tasks, and simple descriptions of what the tasks are doing. But if you want to know all the details you need to consult the pipeline chapters (links above). The individual pipeline tasks are more fully described in the PACS *User's Reference Manual* (PACS *URM*), along with all the other (PACS) tasks.

- The pipelines take you from Level 0 (raw) to Level 2 (fully-processed) or, where applicable, Level 2.5 (fully-processed and combined/extracted).

To access the scripts, go to the HIPE menu *Pipelines>PACS>Spectrometer*. 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. 2.6.3 to know how to change the version of the calibration tree you are using.

- You do the red and the blue camera separately.

- Different wavelength ranges and different pointings within a single observation are dealt with in the pipelines. For unchopped SED AOTs the off-source observations are a separate obsid to the on-source observations, and so must be dealt with separately and then subtracted.

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 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 data layout or the spectra themselves. The plots will open as separate windows.
- You will be offered various ways to save the data: within the *ObservationContext* (and from there to pool) or as so-called **SlicedFrames**, **SlicedPacsRebinnedCubes** or **SlicedProjectedCubes** to a pool. Saving as FITS cubes or spectra is only possible towards the end of the pipeline, when single FITSable products are created.

1.3.3. Considerations when running the pipeline

Considerations concerning the technicalities of running the pipeline are:

- If you chose to run the pipeline helper plotting tasks (by setting *verbose=1* in the pipeline script) then the plots will open in new windows. This is something to be aware of if you are running the pipeline remotely or as part of bulk processing.
- **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. (Consider the size of your observation in pool on disk compared to the memory you have assigned to HIPE.) You can improve the memory use of the pipeline, at the cost of speed, by using a **temporary pool** (Sec. 2.5). If you have multiple wavelength ranges in your data you can improve the speed and memory use by selecting to process only one range at a time (this is explained within each pipeline script when you get to the place where you can separate out ranges).

  If you wish to fiddle with your data (other than using the plotting tasks provided in the pipeline) you could consider doing 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.

Stages to check as you run the pipeline, and which we discuss as we take you through the pipeline are:

- saturated and glitched data, and "transients" (mainly for unchopped data)
- wavelength grid to chose
- spatial grid for the Level 2 cubes (for mapping observations only)
- off-source spectra vs on-source spectra (for the chopped and unchopped pipeline)
- flatfielding
- rebinned vs projected cubes (aka pointed vs mapping observations)

1.3.4. Extra and post-pipeline

After the pipeline chapters there is additional information about working with PACS spectroscopy, either within or after the pipeline:

- Chap. 6 to learn about plotting and inspecting your data as you pipeline process them.
- Chap. 7 explains some additional pipeline tasks (such as "drizzle" to spectrally project cubes) and post-pipeline tasks (such as making line maps and combining PACS and SPIRE spectra).
- App. 8 is a compilation of information about PACS products, and how to understand their layout.

1.3.5. Further processing

There are a number of GUIs that can be used to inspect your PACS Level 2 cubes (the *SpectralSimpleCube* and the *PacsRebinnedCube*, i.e. both of the Level 2 cubes, projected and rebinned). The GUIs are called up with a right-click on the cube name, either where it lies in the *Variables* pane of HIPE or where it is within the Observation viewer in the *Editor* pane. Note that all of these GUIs work on the individual cubes, not on the *ListContext* they are contained within (see Sec. 8.2.3). (Tip: you can drag and drop a cube from the Observation viewer to the *Variables* pane and so extract it out of the *ObservationContext*.) So you need to click, in the Observation viewer, and either select to view or drag and drop, the cubes themselves—the 0,1,2,3,4... in Fig 1.1. (Another tip: when clicking on a cube in the Observation viewer, it can sometimes take a long time to receive a response: it is much faster using the command-line syntax to extract a cube out of the *ObservationContext*. The first time you drag and drop, copy the command echoed to the *Console* and in the future you can use that.)

- To scroll through 2D wavelength slices of your cubes you can use the **Standard Cube Viewer**.
- # The **SpectrumExplorer** (see the *DAG* chap. 6). This is a spectral visualisation tool for spectra and cubes. It allows for an inspection and comparison of spectra from individual spaxels or from separate cubes, and it gives access to various mathematical tasks via its **Spectral Toolbox** menu (see chap. 6.4).
- # The **Cube ToolBox**, which you also access via the SpectrumExplorer: see the *DAG* (chap 6.4, chap 6.5). Together with the Spectrum Toolbox, this allows you to inspect your cube spatially and spectrally at the same time. It also has analyses tasks#you can make line flux maps, position—velocity diagrams and maps, and extract out spectral or spatial regions.
- # The **Spectrum Fitter GUI**, which you also access via the SpectrumExplorer: see the *DAG* chap. 7. This is to will fit the spectra of your cubes. (For cubes it should be accessed via the Toolbox menu of the Spectrum Explorer, **not** directly from the HIPE *Tasks* panel.)