Basic Cube Creation – Test Report – SPIRE-BSS-REP-003246

Issue 0.1
December 29, 2009
Peter Imhof, Blue Sky Spectroscopy

Table of Contents

Scope

This document reports on the tests performed to verify the implementation of the basic cube creation functionality as outlined in the User Requirements for Creating and Processing a Spectral Cube from SPIRE data, issue 0.3, June 2, 2009, SPIRE-BSS-DOC-003175, section G, "Detailed Requirements for Basic Functionality".

Data

The following observations from the in-flight performance verification phase were used:

OD	*OBSID*	*BBID*	*Source*	*#Reps*	*Resolution*	*Spatial Sampling*
123	0x50001988	0xa1060001 - 0xa1060004	Rosette ROF3	4	Low	Point, Intermediate Jiggle
123	0x50001989	0xa1060001 - 0xa1060010	Rosette ROF3	12	Low	Point, Full Jiggle
131	0x50001ADB	0xa1060001	NGC7023	20	Low	Raster, Sparse, 3x3

DP-SPIRE, version 2.0 RC2 was used for the testing.

Test Results

Pre-Processing

CPP-1 The SpireCubePreProcessingTask shall operate on an ArrayList of SpectrometerDetectorSpectrum products from a Level1Context.

OK. The Standard Product Generation (SPG) script uses the following syntax:

```
def makeSpectralList (obs, apodization, logger):
    level1 = obs.level1
    list = []
    mySet =level1.refs.entrySet()
    for entry in mySet:
        item = entry.getKey()
        block = level1.refs[item].product
        list.append(block.refs[apodization].product)
    return list

sdsArray = makeSpectralList(obs, sdsOption, logger)
spc = spirePreprocessCube(sdsList=sdsArray)
```

CPP-2 The SpireCubePreProcessingTask shall verify that the wave column is identical within a small margin of 1e-14 for all detectors within a given scan of a SpectrometerDetectorSpectrum product.

OK. This requirements has been loosened to a wavescale consistency of 1e-8, see SPIRE-1959.

CPP-3 The SpireCubePreProcessingTask shall verify that the wave column is identical within a small margin of 1e-14 for all available scans of a SpectrometerDetectorSpectrum product.

OK with modification. This requirements has been loosened to a wavescale consistency of 1e-8, see SPIRE-1959.

CPP-4 The SpireCubePreProcessingTask shall verify that the wave column is identical within a small margin of 1e-14 for all input SpectrometerDetectorSpectrum products.

OK with modification. This requirements has been loosened to a wavescale consistency of 1e-8, see SPIRE-1959.

CPP-5 The SpireCubePreProcessingTask shall create a data product (SpirePreprocessedCube) which provides the input for tasks implementing the SpectralProjectionAlgorithm interface.

OK. This is how the SPG scripts operate.

CPP-6 The SpirePreprocessedCube shall provide the data from all the columns of the original SpireSpectrum1d datasets (at least wave, flux, error, mask) for the detectors of the SLW array only, from the detectors of the SSW array only, or from all SPIRE detectors.

OK with modification. The SpirePreprocessedCube provides the following accessor methods:

```
public Double1d getWave()

public Double3d getFlux(String type)
public Double3d getFlux()

public Double3d getError(String type)
public Double3d getError()

public Flag getFlag(String type)
public Flag getFlag()
```

The wavescale is identical across all channels and it is therefore not necessary to distinguish between the two detector arrays SLW and SSW.

A flag is provided instead of a mask because flag is used in the spectral simple cube.

CPP-7 The SpirePreprocessedCube shall issue a specific and meaningful log message if not all the requested 3D data structures (flux, error, RA, DEC, flag) have identical lengths.

OK, but not testable. I have not found a way to create an ill-formed SpirePreprocessedCube.

CPP-8 The SpirePreprocessedCube shall issue a specific and meaningful log message if the length of the wavescale does not match the corresponding length of the requested flux cube.

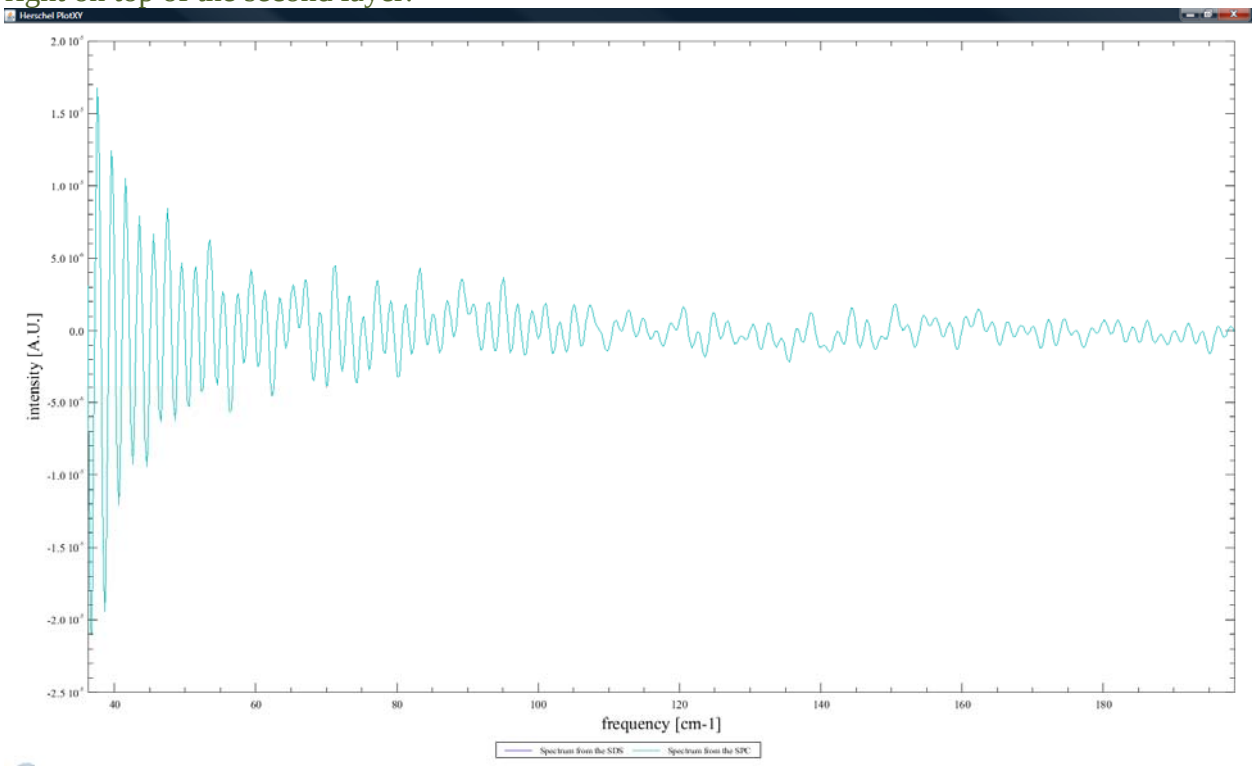
OK, but not testable. I have not found a way to create an ill-formed SpirePreprocessedCube.

CPP-9 The output data shall be sufficient to provide input for a task that implements the SpectralProjectionAlgorithm interface.

OK. This is how the SPG scripts operate.

CPP-10 The SpireCubePreProcessingTask shall only re-order the wave, flux, and error input data from the SpireSpectrum1d datasets to match the required output data format but not change the data values.

OK. Below is a sample overplot for data from SLWA1. Only one layer is visible because it sits right on top of the second layer:



CPP-11 The SpireCubePreProcessingTask shall provide a flag that specifies – at a minimum – which flux samples are not to be used by the spectral projection task.

OK. Note that channels, where the master bit, set by the SPIRE SPG, indicates that samples are unusable, are not set in the SpirePreprocessedCube.

CPP-12 The SpireCubePreProcessingTask shall set a Boolean marker in the flag where the Master bit of a flux sample of the SpireSpectrum1d is set to True.

OK. All flag bits are set to 0 because channels, where the Master bit is set, are not included in the SpirePreprocessedCube.

CPP-13 The SpireCubePreProcessingTask shall propagate those metadata entries that are common to all input products to the output data product.

OK. The propagated metadata entries are contained within each of the originating SDS.

Spectral Projection

SPT-1 The project method of the SpectralProjectionTask shall be able to operate on an unprojected spectral cube as provided by the SpirePreProcessedCube:

Flux Double3d flux per beam
Ra Double3d Right Ascension
Dec Double3d Declination
Error Double3d error of the flux per beam
Flag Flag (3d) data quality flag
WCS WCS (3d) world coordinate system defining the target grid
allowExtrapolation Boolean non-mandatory indicator whether extrapolation be allowed

OK with modification. This is how the SPG scripts operate. An object of type TargetGrid replaced the WCS.

SPT-2 The project method of the SpectralProjectionTask shall be able to operate on a regular spectral cube:

spectralCube SpectralSimpleCube
WCS WCS (3d) world coordinate system defining the target grid
allowExtrapolation Boolean non-mandatory indicator whether extrapolation be allowed

OK with modification. An object of type TargetGrid replaced the WCS.

SPT-3 The project method of the SpectralProjectionTask shall be able to operate on a list of regular spectral cubes:

spectralCube[] SpectralSimpleCube[]
WCS WCS (3d) world coordinate system defining the target grid
allowExtrapolation Boolean non-mandatory indicator whether extrapolation be allowed

OK with modification. An object of type TargetGrid replaced the WCS.

SPT-4 The beamProfile method of the SpectralProjectionTask shall provide the normalized beam efficiency (integrated area/maximum value (TBC) is equal to 1) for a given detector (specified by an index/string (TBC)) as a function of deviation from the center of the detector in the two directions RA and DEC and the rotation angle (not mandatory).

Implementation per channel name and normalized to one at the beam center is OK, but the implemented FWHM is wrong: SPR-SPIRE-2305.

SPT-5 The targetGrid method of the SpectralProjectionTask shall provide a default target grid from RA and DEC information:

Ra Double3d Right Ascension

Dec Double3d Declination

OK. This is how the SPG scripts operate.

SPT-6 The targetGrid method of the SpectralProjectionTask shall provide a default target grid from an ArrayList of World Coordinate Systems.

OK.

SPT-7 The project methods of the SpectralProjectionTask shall create a spectral cube, i.e. a 3D data structure with two spatial and one spectral dimension, as output product.

OK. The output product is a SpectralSimpleCube.

SPT-8 The project methods of the SpectralProjectionTask shall create a spectral cube specifying the flux per pixel (not per beam).

OK. The flux calibration from the SpectrometerDetectorSpectrum is per detector and remains unchanged.

SPT-9 The beamProfile method of the SpectralProjectionTask shall return a number of type double between 0 and 1.

OK.

SPT-10 The targetGrid methods of the SpectralProjectionTask shall return a World Coordinate System with two spatial dimensions.

OK with modification. An object of type TargetGrid replaced the WCS.

SPT-11 The project methods of the SpectralProjectionTask shall set the flux and error at the target grid locations to the values of the closest input data (next neighbour interpolation).

OK by design and verified by test harness.

SPT-12 The project methods of the SpectralProjectionTask shall create a spectral cube that contains all sky positions of the input data if the Boolean input parameter allowExtrapolation is set to True.

OK. The default target grid is indeed larger than the footprint of the detector arrays.

SPT-13 The project methods of the SpectralProjectionTask shall create a spectral cube that includes only those sky positions where data are available below and above the target position in both spatial dimensions if the Boolean input parameter allowExtrapolation is set to False.

Whether extrapolation is required or not, is determined as follows: The extreme RA/DEC values of the input data define a rectangle which must fully contain the target grid in order to process data without extrapolation.

SPT-14 The project methods of the SpectralProjectionTask shall set the Boolean input parameter allowExtrapolation to False as default setting.

N/A. allowExtrapolation is a mandatory parameter.

SPT-15 The beamProfile method of the SpectralProjectionTask shall return the value of a Gaussian beam profile with FWHM of 16 arcsec/35 arcsec for detectors from SSW/SLW regardless of the rotation angle.

The implemented FWHM is wrong by a factor of 1/sqrt(log(Pi)): See SPR-SPIRE-2305.

SPT-16 The targetGrid method of the SpectralProjectionTask shall create a World Coordinate System in two spatial dimensions with the reference pixel at the smallest RA and the smallest DEC values in the input data.

The target grid object is created with the reference pixel in the center of the image.

SPT-17 The targetGrid method of the SpectralProjectionTask shall create a World Coordinate System in two spatial dimensions with square pixel sizes.

OK.

SPT-18 The targetGrid method of the SpectralProjectionTask with an ArrayList of World Coordinate Systems as input shall create a World Coordinate System where the side of one square pixel is equal to the largest interval specified in any of the input World Coordinate Systems.

OK.

SPT-19 The targetGrid method of the SpectralProjectionTask with an RA and a DEC cube as input shall create a World Coordinate System where the side of one square pixel is equal to the difference between the largest RA and the smallest RA divided by the square root of number of individual spectra present in the input data or the difference between the largest DEC and the smallest DEC divided by the square root of number of individual spectra present in the input data – whichever is greater.

OK. The grid interval is computed by dividing by the floor of the square root, reduced by one.

SPT-20 The targetGrid method of the SpectralProjectionTask shall throw a signature exception if the input data straddle the origin of RA or the poles of DEC.

Not implemented.

SPT-21 The SpectralProjectionTask shall ignore any data samples where the flux value is NaN or infty and not throw a Java signature exception.

OK.

SPT-22 The SpectralProjectionTask shall ignore any data samples where the master flag is set to True and not throw a Java signature exception.

Not implemented because the usage of the flag is not yet defined.
If the flag is equal to “UNVALID” everywhere, the task will throw a “java.lang.IllegalArgumentException: No valid spectrum exists. All are flagged as invalid.”

ResampleFrequency

The spectrum toolbox (herschel.ia.toolbox.spectrum.ResampleFrequency) provides the functionality to resample the frequency grid of the spectral cube (satisfies use case D.5.) with two generic interpolation algorithms: an Euler integration scheme in combination with nearest neighbor interpolation and a trapezoidal integration scheme in combination with linear interpolation.