SPIRE Fourier Transform Module Description

SPIRE-BSS-DOC-002496

Blue Sky Spectroscopy Inc.

Trevor Fulton

version 1.5
09 May 2007

SPIRE Fourier Transform Module Description

Blue Sky Spectroscopy Inc.
Trevor Fulton

Table of Contents

Chapter 1. Introduction

1.1. Acronyms

Table 1.1. Acronyms

Short Form	Full Name
CQM	Cryogenic Qualification Model
EDP	Engineering Data Process
FFT	Fast Fourier Transform
FT	Fourier Transform
FTS	Fourier Transform Spectrometer
ILS	Instrument Line Shape
iFTS	Imaging Fourier Transform Spectrometer
LVDT	Linear Variable Differential Transformer
MPD	Mechanical Path Difference
NHKT	Nominal Housekeeping Timeline Product
OPD	Optical Path Difference
PCF	Phase Correction Function
PFM	Proto Flight Model
RMS	Root Mean Square
RSRF	Relative Spectral Response Function
SCAL	Spectrometer Calibrator
SMEC	SPIRE Spectrometer Mechanism
SMECOE	Spectrometer Mechanism Optical Encoder
SMECT	Spectrometer Mechanism Timeline Product
SDI	Spectrometer Detector Interferogram Product
SDS	Spectrometer Detector Spectrum Product
SDT	Spectrometer Detector Timeline Product
SPIRE	Spectral and Photometric Imaging REceiver
TBD	To Be Determined
TBW	To Be Written
ZPD	Zero Path Difference

1.2. Scope of Document

This document describes the processing tasks within the SPIRE Fourier Transform package (`herschel.spire.ia.modules.ft`). The individual tasks within the SPIRE Fourier Transform package will be presented in terms of how they relate to one another and how their usage affects the SPIRE spectrometer data products. These processing tasks make up a large portion of the data processing pipeline for the SPIRE Spectrometer, a detailed description of which can be found in [AD02].

1.3. Documents

1.3.1. Applicable Documents

AD01	K. J. King, SPIRE Data Product Definition
AD02	T. L. Lim, SPIRE Pipeline Description, SPIRE-RAL-DOC-002437
AD03	SPIRE Calibration Products Description, http://scott1.bnsc.rl.ac.uk:8080/hcss/test_area/product_descriptions/index.htm

1.4. Document History

Table 1.2. Version and Date

Issue	Date
Version 1.0	25 July 2005
Version 1.1	17 August 2005
Version 1.2	10 August 2006
Version 1.3	05 January 2007
Version 1.4	24 March 2007
Version 1.5	09 May 2007

Chapter 2. Module Description

2.1. Module Description

The purpose of the SPIRE Spectrometer pipeline is to transform the spectrometer dectector samples acquired during a single observation into a set of spectra. The pipeline modules collectively perform four basic operations listed below:

1. **Modify Timelines**

   The processing modules in this group perform time domain operations on the spectrometer dectector samples.

2. **Create Interferograms**

   The processing modules in this group merge the timelines of the spectrometer detectors and spectrometer mechanism into interferograms. In addition, the spectrometer detector samples are split into different sets, with each set defined by a single scan of the spectrometer mechanism.

3. **Modify Interferograms**

   The processing modules in this group perform operations on the spectrometer dectector interferograms. These operations differ from those in the "Modify Timelines" group in that they are designed to act on spatial domain data rather than time domain data.

4. **Transform Interferograms**

   The processing modules in this group transform the interferograms into a set of spectra.

The manner in which the basic operations relate to one another is shown in Figure 2.1.

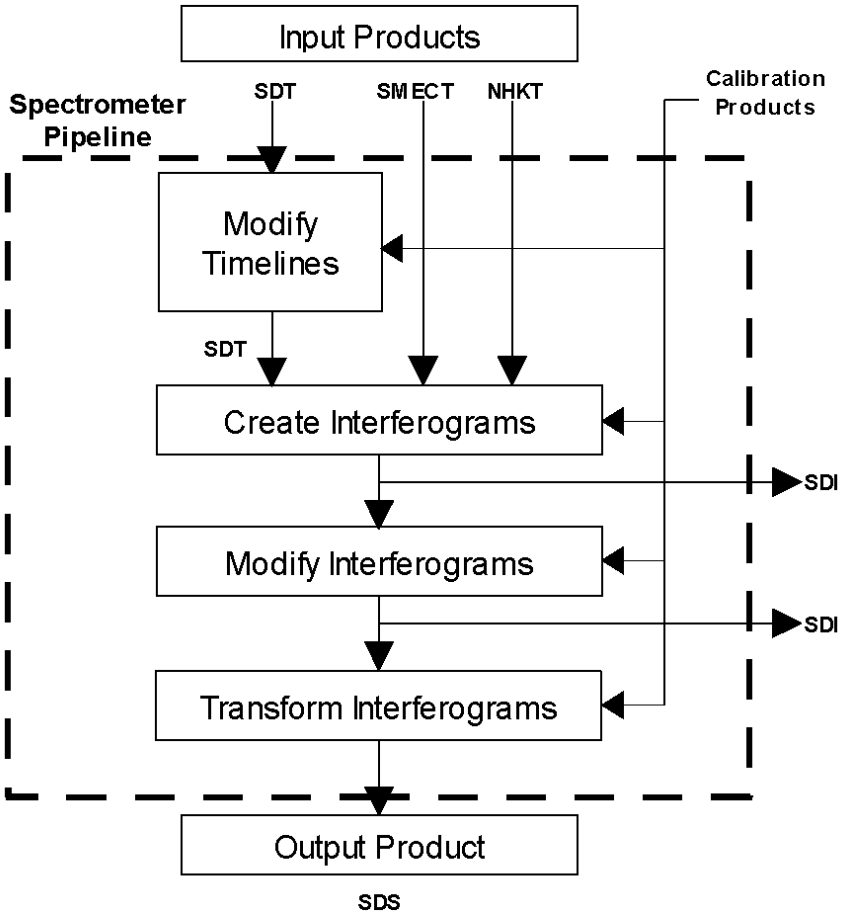
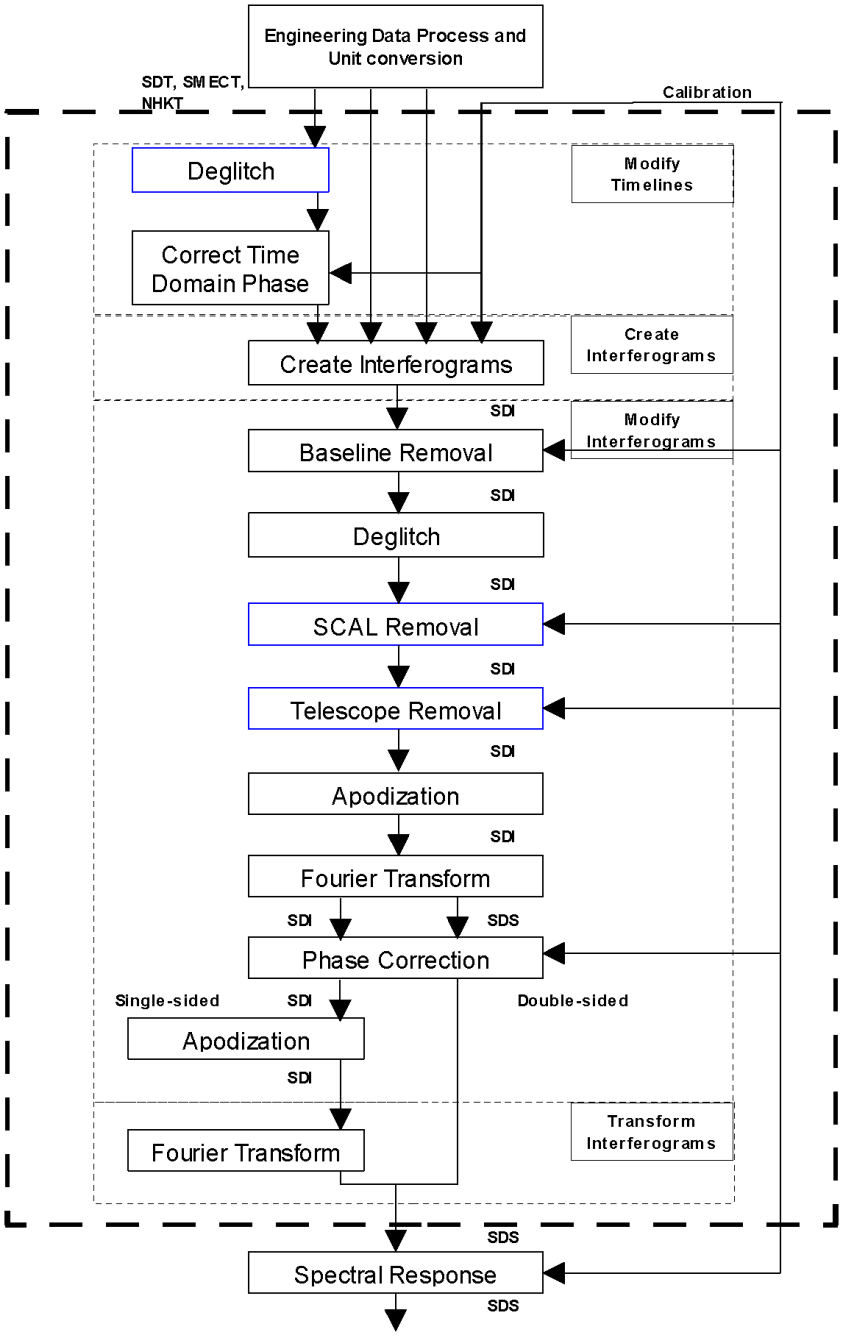


Figure 2.1. Basic functionality of the SPIRE Spectrometer pipeline.

The SPIRE spectrometer data processing pipeline has been divided of a set of atomic tasks, or modules. This choice was made to allow for a degree of flexibility in the manner by which data from spectrometer observations are processed. An overview of the individual modules of the SPIRE spectrometer pipeline and their connection with one another is shown in Figure 2.2 while descriptions of these processing modules are presented in Chapter 3 - Chapter 9.



Individual processing modules and their input and output products are shown. Those modules shown in **blue** are not described in detail in this document.

Figure 2.2. Detailed functionality of the SPIRE Spectrometer pipeline.

Chapter 3. Time Domain Phase Correction

3.1. Purpose

The purpose of this module is to adjust the spectrometer detector samples to account for the time delay induced by the detector read-out electronics and the thermal response of the detectors.

3.2. Inputs

Data Products

SDT	**Spectrometer Detector Timeline Product** This product contains the measured signal sample timelines for each spectrometer detector channel.

Calibration Products

calElectronicsParams	**Read-out Electronics Table** This calibration product contains the values of the resistors and capacitors that comprise the read-out electronics for the SPIRE spectrometer detector channels.
calPixTimeConst	**Pixel Time Contants Calibration Product** This calibration product contains the calibrated thermal time constants for each spectrometer detector channel.

3.3. Description

The time domain phase adjustment is accomplished by convolution of the detector timelines with an inverse delay function. The overall delay function is the combination of the frequency response of the SPIRE spectrometer read-out electronics and the thermal response of the spectrometer detector channels. While the response of the read-out electronics is the same for all spectrometer detector channels, the thermal response is different for each channel. As such, the delay function is computed on a channel-by-channel basis.

3.4. Outputs

SDT	**Spectrometer Detector Timeline Product** The output SDT product contains the corrected signal sample timelines for each spectrometer detector channel.

3.5. Example

```
IA>> from herschel.spire.ia.modules.ft import *
IA>> removeTimeDomainPhase=RemoveTimeDomainPhase()
IA>> sdt=removeTimeDomainPhase(sdt=sdt,
                               calElectronicsParams=calElectronicsParams,
                               calPixTimeConst=calPixTimeConst)
```

Chapter 4. Create Interferograms

4.1. Purpose

A typical SPIRE spectrometer observation consists of a series of scans of the spectrometer mechanism while the instrument is pointed at a given target. In the SPIRE spectrometer, the sampling of spectrometer detectors and the spectrometer mechanism is decoupled; the two subsystems are sampled at different rates and at different times. The purpose of this module is to combine the spectrometer detector timelines and spectrometer mechanism timeline into a set of interferograms whose samples are equidistant for a given SPIRE spectrometer observation.

4.2. Inputs

Data Products

SDT	**Spectrometer Detector Timeline Product** This product contains the measured signal sample timelines for each spectrometer detector channel.
SMECT	**Spectrometer Mechanism Timeline Product** This product contains the timelines for the telemetry parameters related to the spectrometer mechanism (SMEC).
NHKT	**Nominal Houseeeping Timeline Product** This product contains the timelines for the nominal housekeeping telemetry parameters.

Calibration Products

calSmecZpd	**Zero Path Difference Calibration Product** This calibration product contains the values the position of zero path difference (ZPD) for each SPIRE spectrometer detector channel. There are two entries for each channel; one entry gives the SMEC optical encoder value for ZPD, the other entry gives the SMEC LVDT value for ZPD.
calSmecOeOpd	**Optical Encoder to Optical Path Difference Calibration Product** This calibration product contains the conversion factors that relate a step of the SMEC encoder to a step in OPD. Nominally, this conversion factor is equal to 4 for a Mach-Zehder FTS; for the SPIRE FTS this conversion factor will be different for each off-axis detector channel.
calSpecChanMask	**Spectrometer Detector Channel Mask Calibration Product** This calibration product contains entries for each spectrometer detector channel to indicate whether the channel is known to be either noisy or dead. Channels listed as dead will be omitted from the output SDS product.

4.3. Description

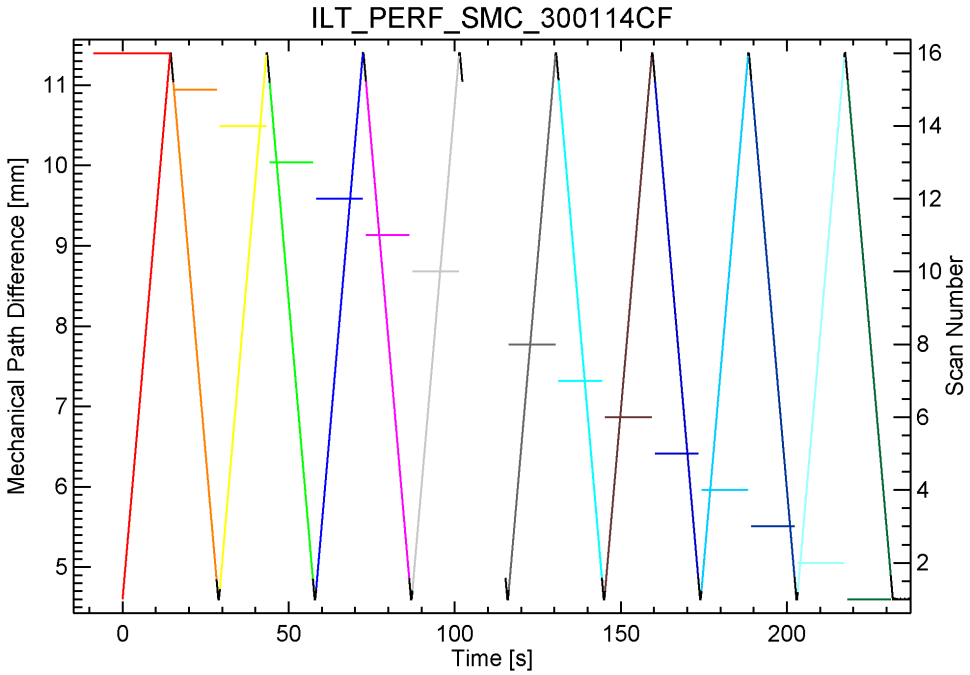
The process by which interferograms are created involves two steps, each of which is described below.

1. **Interpolation of the SMECT timeline.** This step converts the samples in the SMEC timeline from being non-uniform in position to a set of timelines that are uniform in position.

    a. **Create a regular SMEC position timeline.** This step creates a common vector of SMEC positions that will be the basis for the interferograms for all of the spectrometer detector channels and for all of the scans in the observation. This common position vector will

contain samples that are uniformly-spaced in terms of SMEC position. A common set of SMEC positions is desired as it will allow for easier scan-to-scan and channel-to-channel comparisons of the interferograms for a given SPIRE spectrometer observation.

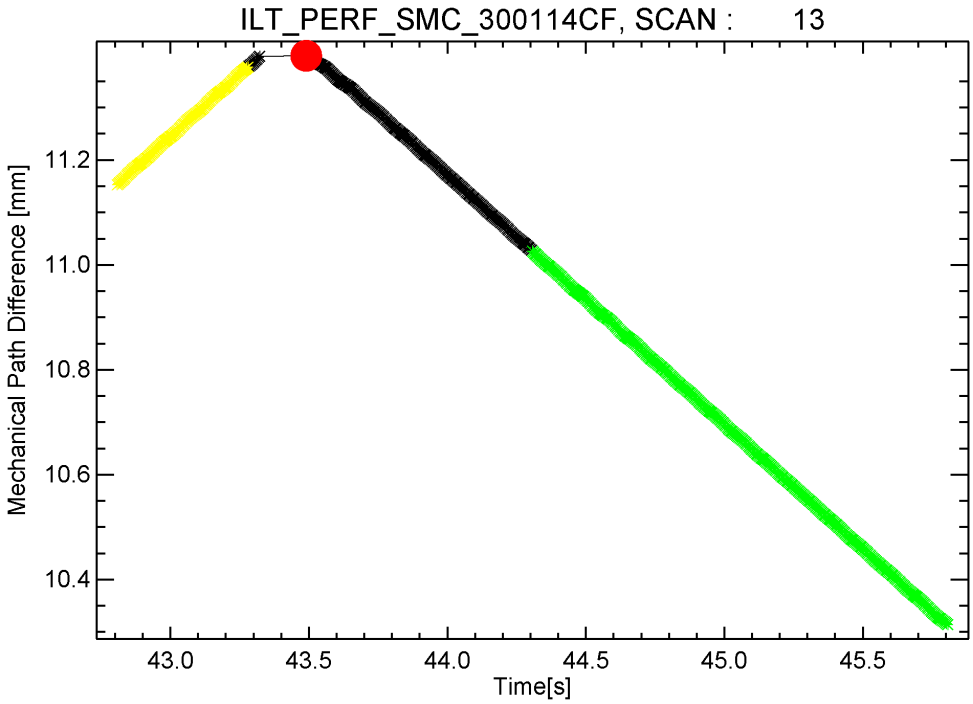
b. **Parse the encoder timeline into discrete scans.** A SPIRE spectrometer observation is composed of discrete building blocks referred to as scans. These scans represent a single movement of the SMEC from one extreme position to another. The scan numbers for an observation are contained in the NHKT product in either the SCANS or SMECSTAT parameter. It should be noted that since the parameters in the NHKT are sampled more sparsely than those in the SMECT (1Hz versus 240Hz), the housekeeping parameters can only provide a rough estimate of the actual delineations between scans. An example of the rough nature of the housekeeping parameters in comparison to the SMECT parameters is shown in Figure 4.1.



The tringluar pattern represents the SMEC poitions as a function of time. The colored portions of the SMEC timeline represent the regions where the SMEC encoder timeline overlaps with the scan ranges given by the SCANS or SMECSTAT housekeeping parameters.

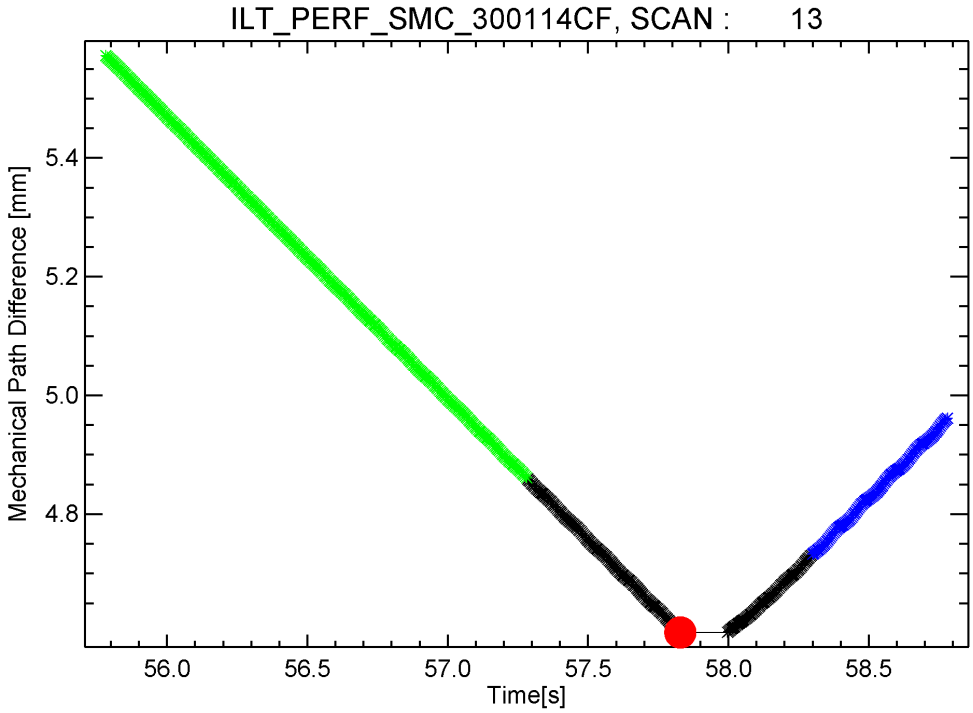
Figure 4.1. Coarse Scan Extrema

A more precise representation of the actual scan limits is found by closer inspection of the SMEC encoder timeline in the scan extrema regions defined by the housekeeping parameters. Figure 4.2 and Figure 4.3 show two close-up views of the encoder timeline from Figure 4.1.



A close up view of the encoder timeline near the start of a given scan. The colored samples represent the scan delineations as defined by the housekeeping parameters. The red circle shows the start of the given scan as defined by the by the point where the encoder timeline changes from a decreasing function to an increasing function.

Figure 4.2. Fine Scan Extrema

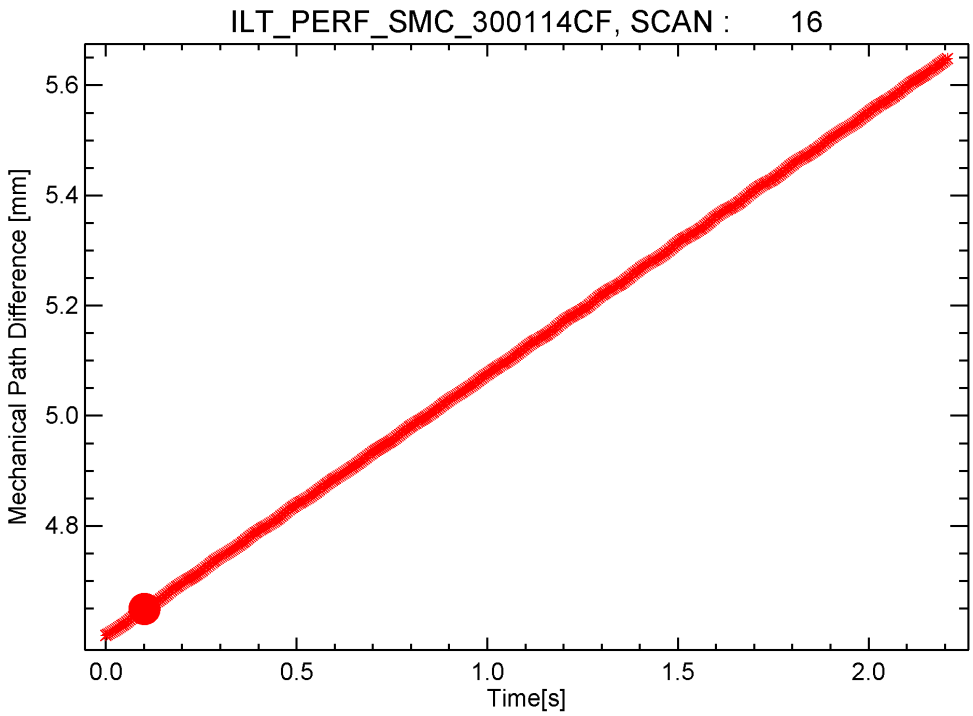


A close up view of the encoder timeline near the end of a given scan. The colored samples represent the scan delineations as defined by the housekeeping parameters. The red circle shows the end of the given scan as defined by the by the point where the encoder timeline

changes from a decreasing function to an increasing function.

Figure 4.3. Fine Scan Extrema

As shown in Figure 4.2 and Figure 4.3, the points where the encoder timeline changes from an increasing to a decreasing function are precise indicators of the actual scan extrema. In some cases, principally for the first scan of an observation, there is no such notable change in the encoder timeline that can be used to delineate the scan extremum. This situation arises when the commanded scan extremum is different that the home position of the SMEC, an example of which is shown in Figure 4.4. For these situations, the scan extremum in the encoder timeline is defined as the sample that is closest to the commanded scan extremum, with the commanded scan extrema given by the SCANSTART and SCANEND parameters in the NHKT.



A close up view of the encoder timeline near the start of the first scan of a given observation. The derived scan start position/time is highlighted by the red circle.

Figure 4.4. Fine Scan Extrema

c. **Interpolate the SMEC timelines.** Once the SMEC timeline has been parsed into discrete scans, the SMEC positions are interpolated from the irregular samples to the regular positions derived above. This step determines, on a scan-by-scan basis, the times when the SMEC reached a series of positions that are regular in MPD. Thse positions will later form the basis for the output interferograms.

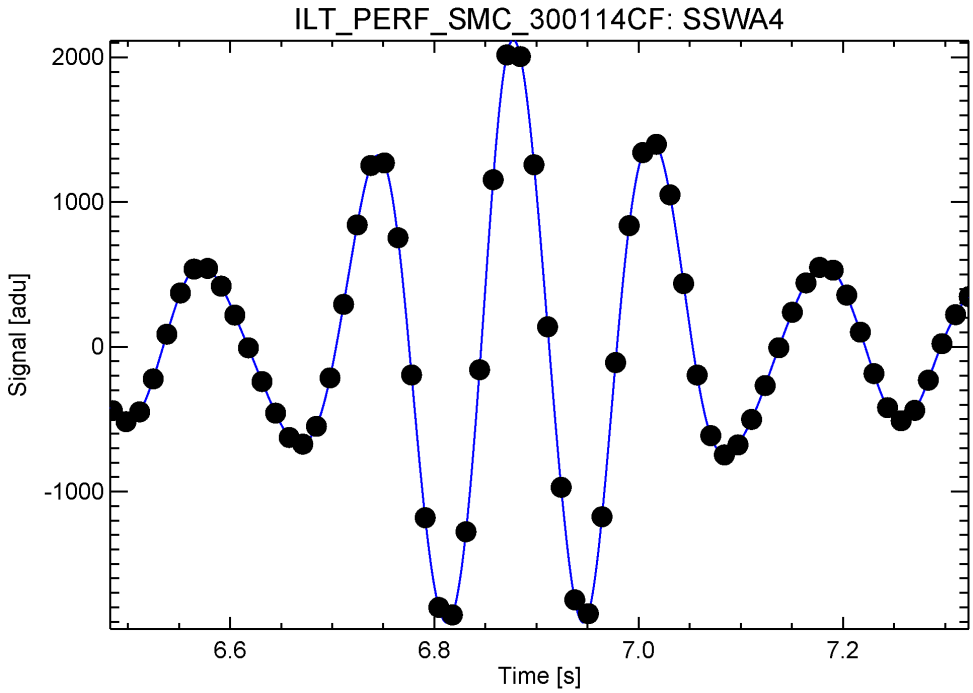
2. **Merge the spectrometer detector and the new SMEC timelines.** This step combines the signal samples in the SDT with the newly-created regularly-spaced SMEC position timelines to create a set of interferograms.

   a. **Check the integrity of the detector channel.** If the input "calSpecChanMask" calibration product indicates that the channel is dead, the channel is omitted from the output SDI product.

b. **Correct for frametime delay.** Adjust the pixel timeline to account for the difference between the time that the pixel was read out and the listed sample time for the detector.

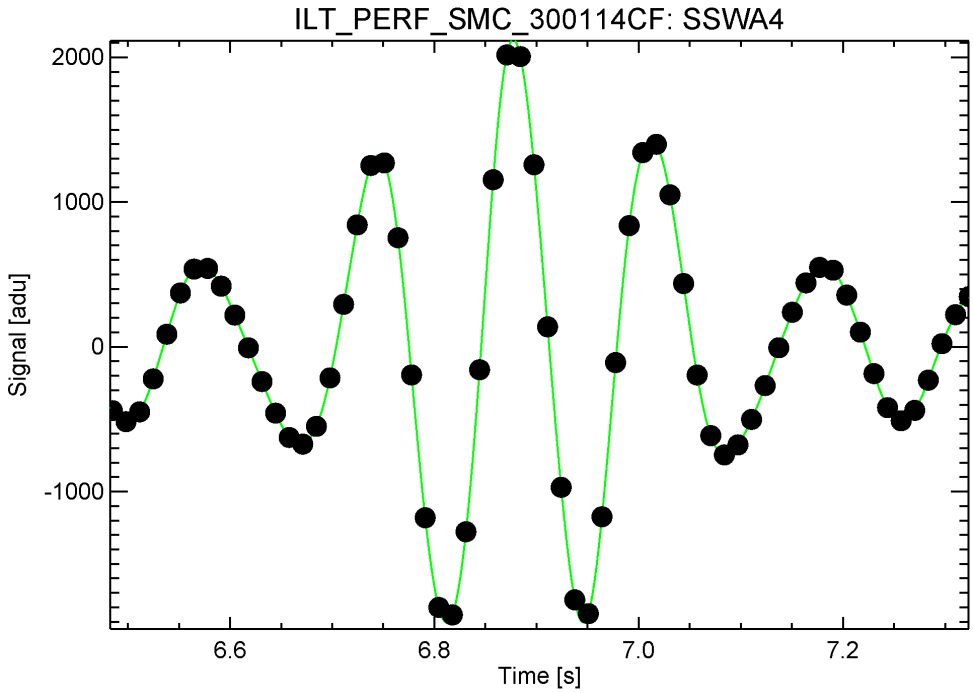
> **Note**
> The EDP may have already taken this into account.

c. **Interpolate the detector timeline.** Interpolation of the SDT timelines is performed by first oversampling (see ???) the signals in time, then interpolating the oversampled timeline to the regularly-spaced SMEC timeline. The signal samples that result from this step are then linked to the regularly-spaced positions from step 1.



An example of a detector timline. The circles are the original samples, the blue line is the oversampled timline.

Figure 4.5. Oversampled Detector Timeline



An example of a detector timline. The circles are the original samples, the green line is the detector timeline interpolated onto the times that correspond to regular SMEC intervals.

Figure 4.6. Interpolated Detector Timeline

d. **Convert from mechanical path difference to optical path difference.** This step shifts and scales the position samples of the newly created spectrometer detector interferograms in such a way as to convert the sample positions from mechanical path difference (MPD) to optical path difference (OPD). The position of ZPD and the conversion of a step in MPD to a step in OPD are derived from the "calSmecZpd" and the "calOeOPD" calibration products, respectively. [1].

> **Note**
>
> The data from the SPIRE PFM test campaigns indicates that for each spectrometer detector channel, there is different position of ZPD for each emitting element (Telescope, SCAL2, SCAL4). As such it may be appropriate to delay the conversion from MPD to OPD until after the effects of SCAL and Telescope effects have been removed from the spectrometer detector interferograms.

4.4. Outputs

SDI **Spectrometer Detector Interferogram Product** The output SDI product contains the interferograms for each spectrometer detector channel for each scan of the observation.

4.5. Example

```
IA>> from herschel.spire.ia.modules.ft import *
```

[1] If "calSmecZpd" or "calOeOPD" calibration products are not specified, default values for ZPD and the MPD to OPD step conversion are used.

```
IA>>
IA>> regSampledIfgmCreation=RegSampledIfgmCreation()
IA>> sdi=regSampledIfgmCreation(sdt=sdt,
                                smect=smect,
                                hkt=nhkt,
                                calSmecZpd=calSmecZpd,
                                calOeOPD=calOeOPD,
                                calSpecChanMask=calSpecChanMask)
IA>>
```

4.6. Quality Control

The quality control parameters for this module focus primarily on the behaviour of the SMEC. A list of the quality control parameters for this module is given in Table 4.1 below.

Table 4.1. Quality Control Parameters

Quality Control Metric	Description	Format	Acceptable Range
Number of missed optical encoder fringes	Compare the SMEC optical encoder and LVDT timelines to determine whether any optical encoder counts were missed.	Integer	0
Average Stage Speed	On a scan-by-scan basis, compute the ratio of the measured SMEC speed to the commanded SMEC speed.	Float	<1%
Stage Speed Deviation	On a scan-by-scan basis, compute the standard deviation of the SMEC speed.	Float	<0.05 mm/s
Scan Extrema	On a scan-by-scan basis, compute the ratio of the measured SMEC scan extrema to the commanded scan extrema.	Float	<1%
SMEC Temperature	The difference between the maximum and minimum SMEC temperature over the coarse of the observation.	Float	<0.05 Kelvin

4.7. Future Development

An alternate method of interferogram creation will be required for the step and integrate mode of the spectrometer. The algorithm for this method is still TBD.

Chapter 5. Baseline Correction

5.1. Purpose

The purpose of this module is to remove the baseline from each of the interferograms of a SPIRE spectrometer observation.

5.2. Inputs

Data Products

SDI **Spectrometer Detector Interferogram Product** This product contains interferograms for each spectrometer detector channel for each scan of the observation.

Calibration Products

calTeleVig **Vignetting Calibration Product** This calibration product contains one curve per spectromter detector channel. These curves describe the amount of vignetting to be expected as a function of MPD.

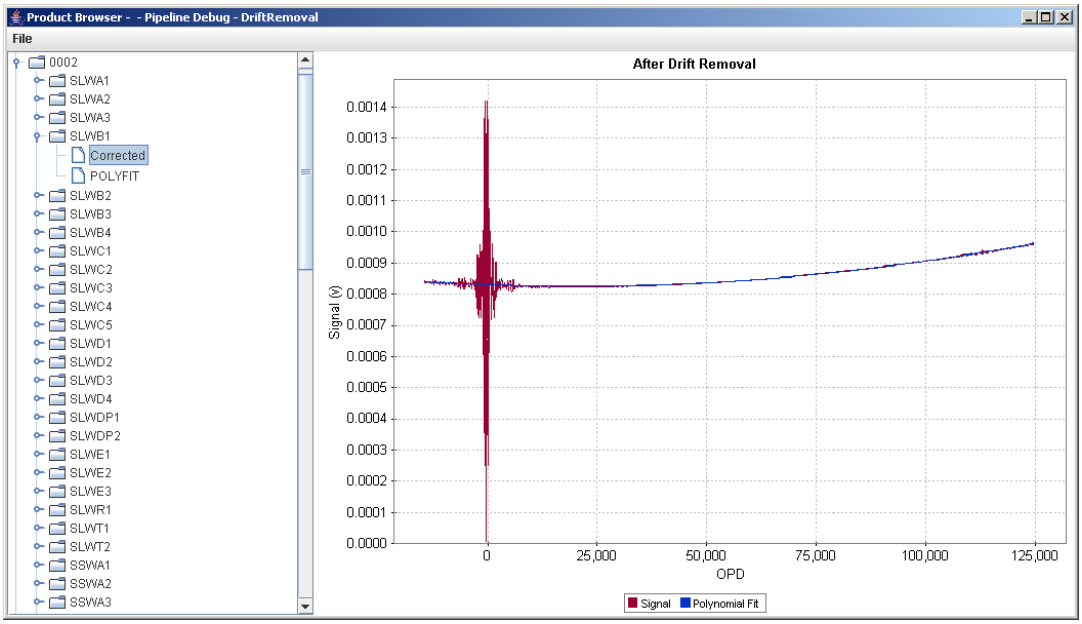
Control Parameters

degree **Degree of Polynomial Fit** This optional parameter defines the degree of the polynomial that is fit to the baseline of the interferograms in the input SDI product.

5.3. Description

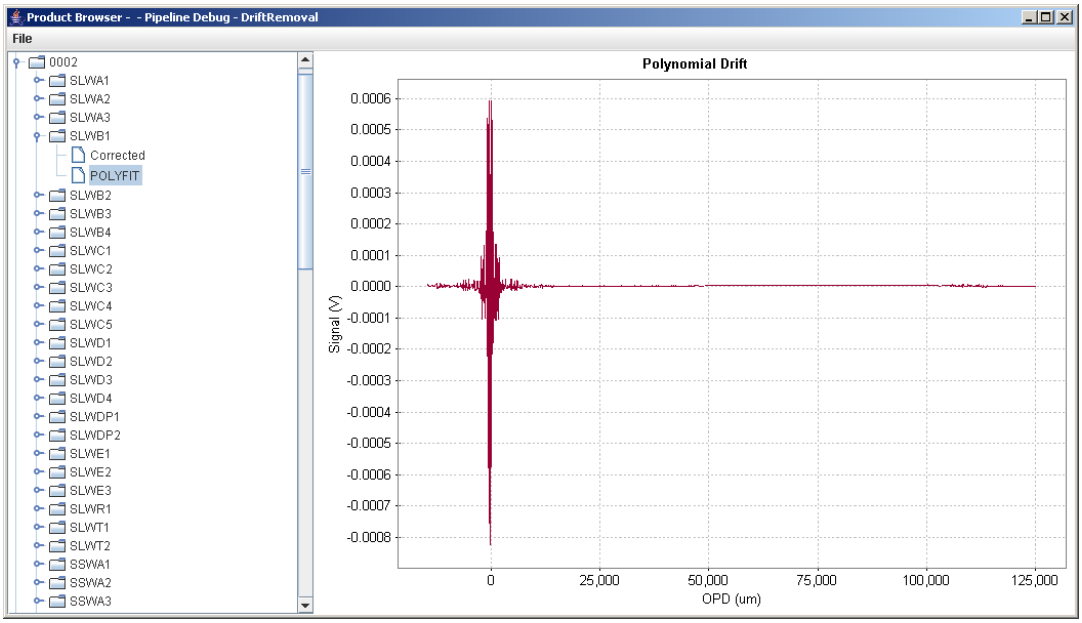
Baseline correction is performed on a scan-by-scan basis for each spectrometer detector channel in the input SDI product. For each interferogram, a polynomial is first fitted to the baseline (see Figure 5.1) then subtracted from the interferogram (see Figure 5.2). The order of the fitted polynomial is the same for all interferograms of the SDI product and is given by "degree" input parameter [1]. The order of the fitted polynomial may be any integer in the range from 1 to 10.

[1] If the "degree" input parameter is not set, the order of the polynomial fit is given by the user property `spire.ia.modules.ft.baselinecorrection.order`.



The red curve is the measured interferogram; the blue curve is a second-order polynomial fitted to the baseline of the interferogram.

Figure 5.1. Interferogram before baseline correction



The interferogram after baseline correction.

Figure 5.2. Corrected Interferogram

5.4. Outputs

SDI **Spectrometer Detector Interferogram Product** The output SDI product contains the corrected interferograms for each spectrometer detector channel for each scan of the observation.

5.5. Example

```
IA>> from herschel.spire.ia.modules.ft import *
IA>>
IA>> baselineCorrection=BaselineCorrection()
IA>> sdi=baselineCorrection(sdi=sdi, degree=2)
IA>>
```

5.6. Future Development

A more appropriate algorithm for baseline correction may be to rely on a set calibrated curves rather than a fitted polynomial. In addition, it may be useful to divide this process into two modules: the first module would remove any time-dependent baseline (e.g. drift due to changes in the temperature of the detectors) from the spectrometer detector timelines; the second module would then remove any position-dependent baseline (e.g. vignetting) from the spectrometer detector interferograms.

Warning

There are some questions as to whether this modules is necessary in the standard SPIRE FTS processing pipeline. It may be that any baseline present in the measured interferograms will only add low, out-of-band frequencies to the final spectra and will therefore not adversely affect the derived spectra. In addition, application of this correction in a less than optimal manner may in fact introduce anomolous features to the derived spectra, in which case this module will do more harm than good.

Chapter 6. Glitch Correction

6.1. Purpose

The purpose of the Glitch Correction task is to remove unwanted localized artifacts from the measured interferograms prior to transformation.

6.2. Inputs

Data Products

SDI **Spectrometer Detector Interferogram Product** This product contains interferograms for each spectrometer detector channel for each scan of the observation.

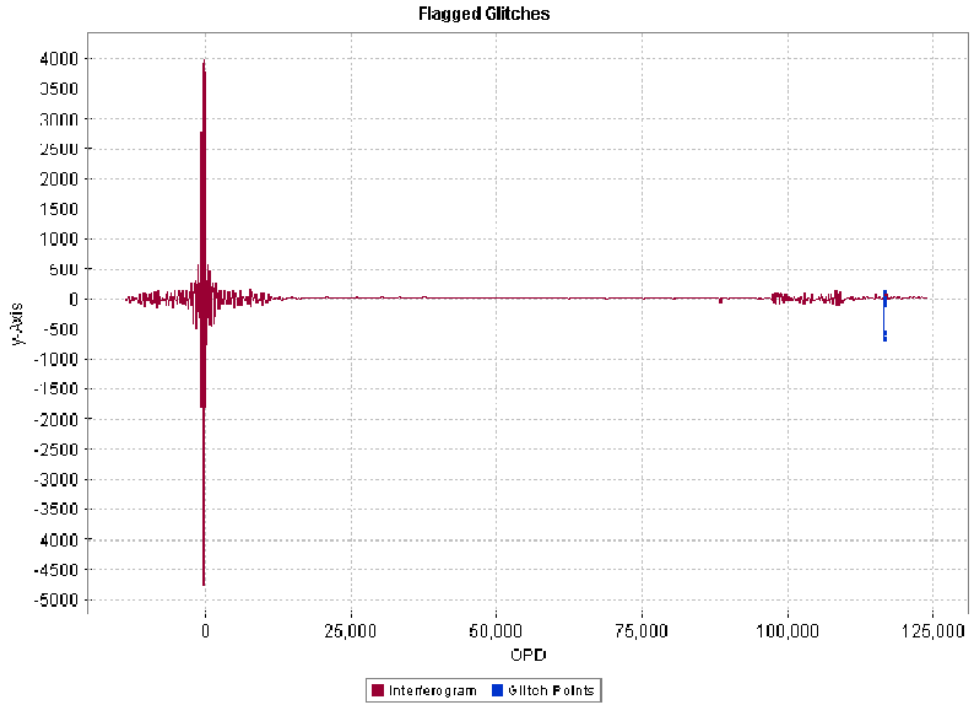
6.3. Description

Localized artifacts in the interferograms, glitches, pose a serious problem for Fourier Transform Spectrometer observations. A glitch that affects as few as one interferogram sample can adversely affect each and every spectral component. Glitches in an interferogram must therefore be identified and removed prior to transformation in order to avoid unwanted spectral artifacts.

The interferogram deglitching module is composed of two steps. The first step identifies the anomolous samples in the spectrometer detector interferograms; the second step removes these samples and repairs the interferograms.

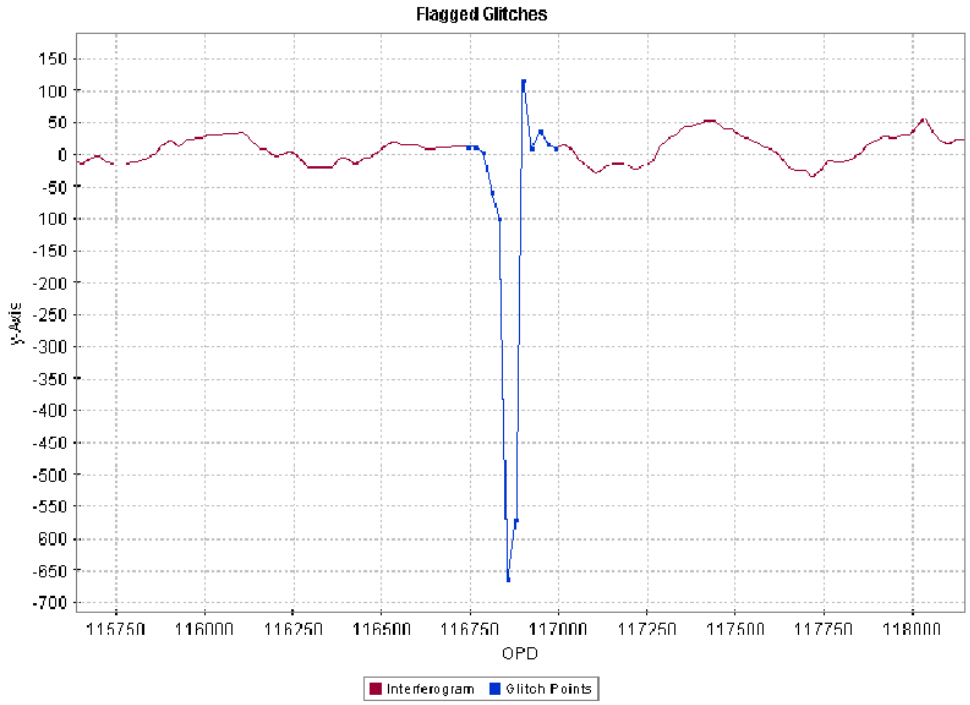
1. **Glitch Identification.** One of the properties of the SDI products created by the SPIRE Fourier Transform package is that the samples for each interferogram for a given detector channel are all on the same, regularly-spaced position grid (see Chapter 4). Since, for a given observation, the interferograms from scan-to-scan for a given channel should be identical within measurement noise, it is possible to detect anomolous samples caused by glitches by comparing the interferograms for a given pixel at each position.

   Glitches are identified by flagging outliers in the scan-to-scan comparison at each sample point using metrics such as standard deviation or skewness (see Figure 6.1 and Figure 6.2). In general, standard deviation is a better metric for observations with a low number of scans (number of scans < 10), while the skewness technique works better for observations with a large number of scans (number of scans >= 10).



The points shown in blue have been flagged as glitches.

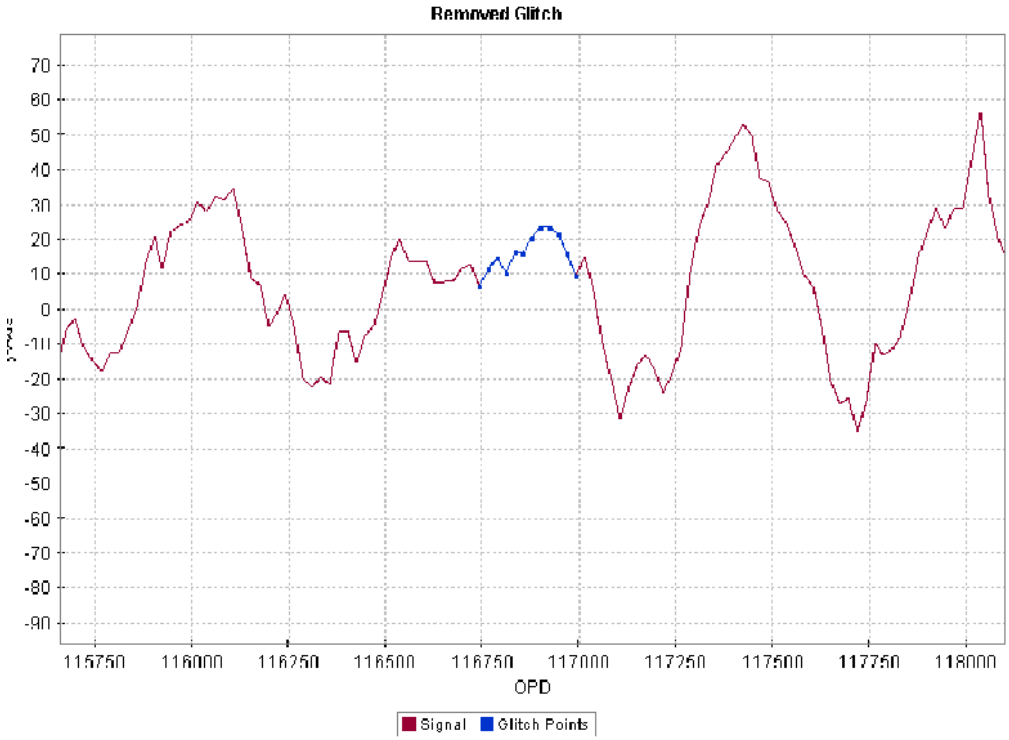
Figure 6.1. Interferogram with Simulated Glitch



Close-up version of the interferogram shown in Figure 6.1. The points shown in blue have been flagged as glitches.

Figure 6.2. Interferogram with Simulated Glitch

2. **Glitch Removal.** After the glitches present in the interferograms of the SDI product have been identified, the points that have been flagged are removed and replaced. The replacement values are taken as the average of the values from the unaffected interferograms at that position (see Figure 6.3).



Interferogram with the glitch points removed and replaced. The points shown in blue have been corrected.

Figure 6.3. Interferogram with Glitch Points Removed and Replaced

Note

The two steps of the interferogram deglitching module rely on a statistical analysis of the measured interferograms. As such, a minimum number of interferograms will be required so that these statistics will be meaningful. If this module is kept as part of the standard processing pipeline for the SPIRE spectrometer, further analysis should be performed to determine the minimum number of required scans per observation so that this number can be added to the proposal handling system.

6.4. Outputs

SDI **Spectrometer Detector Interferogram Product** The output SDI product contains interferograms that have been corrected for glitches.

6.5. Example

```
IA>> from herschel.spire.ia.modules.ft import *
IA>>
IA>> ifgmDeglitcher=IfgmDeglitcher()
IA>> sdi=ifgmDeglitcher(sdi=sdi)
IA>>
```

6.6. Quality Control

A list of the quality control parameters for this module is given in Table 6.1 below.

Table 6.1. Quality Control Parameters

Quality Control Metric	Description	Format	Acceptable Range
Number of glitches per scan	The fraction of points in a scan flagged as glitches as a function of the total number of points in a scan.	Float	< 10%

Chapter 7. Apodization

7.1. Purpose

The purpose of the Apodization task is to correct for and remove effects related to the instrumental line shape of the Fourier Transform spectrometer.

7.2. Inputs

Data Products

SDI **Spectrometer Detector Interferogram Product** This product contains interferograms for each spectrometer detector channel for each scan of the observation.

Control Parameters

apodType **Apodization Type** This parameter defines the type of apodization that is to be performed. If set to "pre", the output SDI will contain samples before and after ZPD. If this parameter is set to "post", the output SDI will contain only those samples where OPD is >= ZPD.

apodFunction **Apodization Function Name** This optional parameter defines the apodization function that is to be applied to interferograms in the input SDI product.

In addition to some long-standing apodization functions, the SPIRE Fourier Transform data processing package provides a number of functions that optimize the tradeoff between minimizing the secondary spectral maxima and reduced spectral resolution. A list of all of the apodization functions available for this task is given in Appendix B

7.3. Description

The natural ILS for a Fourier Transform spectrometer is a cardinal sine or Sinc function. For interferograms that contain features that are at or near the resolution of the instrument (e.g. channel fringes), the natural instrument line shape can introduce secondary maxima in the calculated spectra. The magnitude of these secondary maxima can be reduced by multiplying the interferogram with a tapering or apodizing function prior to transformation. A side effect of apodization is that it leads to a reduction in the resultant spectral resolution.

7.4. Example

The following is an example of pre-apodization; apodization of the interferograms in the SDI prior to phase correction.

```
IA>> from herschel.spire.ia.modules.ft import *
IA>>
IA>> regSampledApodization=RegSampledApodization()
IA>> presdi=regSampledApodization(sdi=sdi,
                                  apodType="pre",
                                  apodFunction="aNB_15")
IA>>
```

The following is an example of post-apodization; apodization of the interferograms in the SDI after

phase correction.

```
IA>> from herschel.spire.ia.modules.ft import *
IA>>
IA>> regSampledApodization=RegSampledApodization()
IA>> sdi=regSampledApodization(sdi=sdi,
                               apodType="post",
                               apodFunction="aNB_15")
IA>>
```

7.5. Outputs

SDI **Spectrometer Detector Interferogram Product** This product contains apodized interferograms for each spectrometer detector channel for each scan of the observation. If pre-apodization is selected the output SDI is a new product that contains double-sided interferograms. If post-apodization is selected the interferograms in the output SDI will have samples for only those OPDs where OPD >= ZPD.

Chapter 8. Phase Correction

8.1. Purpose

The purpose of the Phase Correction task is to modify the interferograms of the input SDI product to correct for any asymmetries in the sampled *signal* about ZPD. These asymmetries, which may arise from mis-sampling of the position of ZPD or the from the presence of dispersive elements in the spectrometer (e.g. optics, electronics), if left uncorrected, lead to phase errors in the resultant spectrum.

8.2. Inputs

Data Products

SDI **Spectrometer Detector Interferogram Product** This product contains interferograms for each spectrometer detector channel for each scan of the observation.

SDS **Spectrometer Detector Spectrum Product** This product contains the double-sided spectra for each spectrometer detector channel for each scan of the observation.

Calibration Products

calBandEdges **Spectrometer Band Edges Calibration Product** This calibration product contains the measured spectral band limits in units of cm^{-1} for each spectrometer detector channel.

Control Parameters

degree **Degree of Polynomial Fit** This optional parameter defines the degree of the polynomial that is fit to the in-band portion of the measured phase for each scan for each spectrometer detector channel.

pcfSize **Phase Correction Function Length** This optional parameter defines the length in points to which the derived phase correction function will be truncated. This parameter is only valid for single-sided interferograms.

pcfApodName **Apodization Function Name** This optional parameter defines the apodization function to be applied to the derived phase correction function. A list of all of the apodization functions available is given in Appendix B. This parameter is only valid for phase correction of single-sided interferograms.

8.3. Description

The even symmetry of a Fourier Transform spectrometer theoretically implies that interferograms recorded by the spectrometer will also exhibit even symmetry. The spectrum of a evenly symmetric interferogram contains only real components, as per Equation 8.0.

$$FT(I_{Symmetric}(x)) = \int I_{Symmetric}(x)e^{i2\pi x(\sigma)}dx = \int I_{Symmetric}(x)e^{i2\pi x(\sigma)}dx = B(\sigma) = B_{Re}(\sigma)$$

The presence of dispersive elements and the possibility that the position of zero path difference not being sampled can result in a measured interferogram whose *signal* samples are not symmetric

about ZPD (see Figure 8.1).

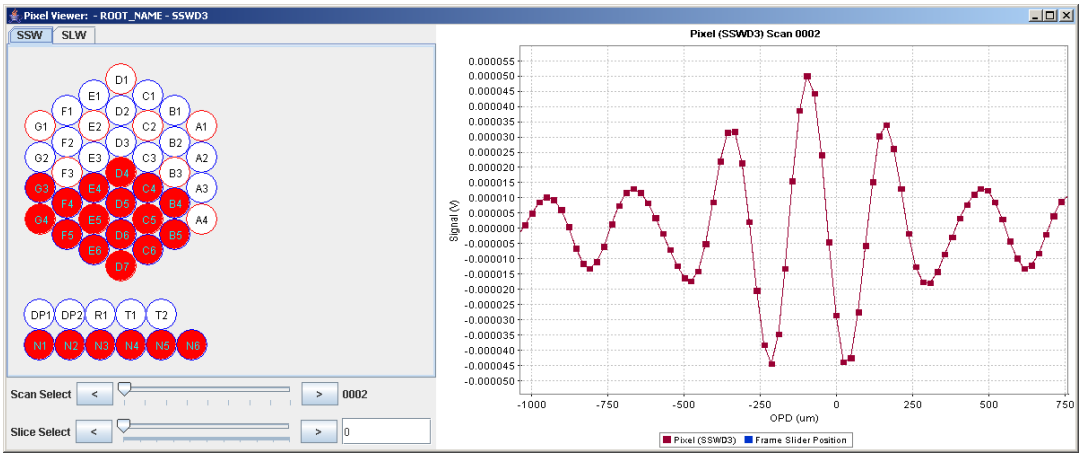


Figure 8.1. Asymmetric Interferogram

Left uncorrected, the spectrum calculated from an asymmetric interferogram will contain both real and imaginary components.

$$FT(I_{Asymmetric}(x)) = \int I(x)e^{i2\pi(x\sigma)}dx = \int I_{Even}(x)\cos(2\pi x\sigma))dx + i\int I_{Odd}(x)\sin(2\pi x\sigma))dx$$

$$FT(I_{Asymmetric}(x)) = B_{Re}(\sigma) + iB_{Im}(\sigma) = B(\sigma)e^{i\Phi(\sigma)}$$

The phase correction process renders the measured interferogram *signal* symmetric about ZPD by moving the components of the spectrum that are located in the imaginary domain to the real domain.

8.4. Double-sided phase correction

If the interferograms in the input SDI product are double-sided (see Section A.1.1), then the resultant spectra will contain phase information for each spectral element. As such, phase correction of these interferograms can take place solely in the spectral domain. Referring to Equation 8.0, the calculated spectrum, $B(\sigma)$, can be corrected by way of multiplication with a phase correction function (PCF) as:

$$\text{Phase Corrected Spectrum} = B(\sigma)e^{i\Phi(\sigma)} \times PCF = B(\sigma)e^{i\Phi(\sigma)} \times e^{-i\Phi(\sigma)} = B(\sigma)$$

It should be noted that the PCF referred to here is not simply the negative of the measured phase. Rather, a fit is made to the measured in-band phase and it is complex exponential of the negative of this fit that is used as the PCF (see Figure 8.3). The basis for using the fitted phase rather than the calculated phase it that, by doing so, the noise associated with the imaginary postion of the spectrum remains in the imaginary domain. For random sources of measurement noise, this can lead to an increase in the signal-to-noise ratio by a factor of $\sqrt{2}$.

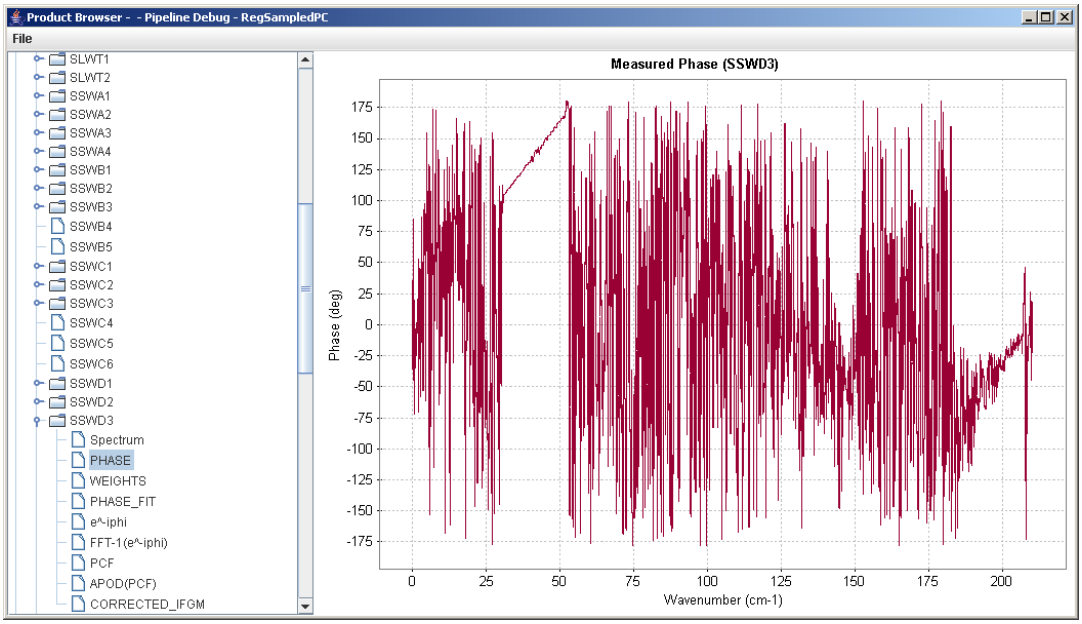
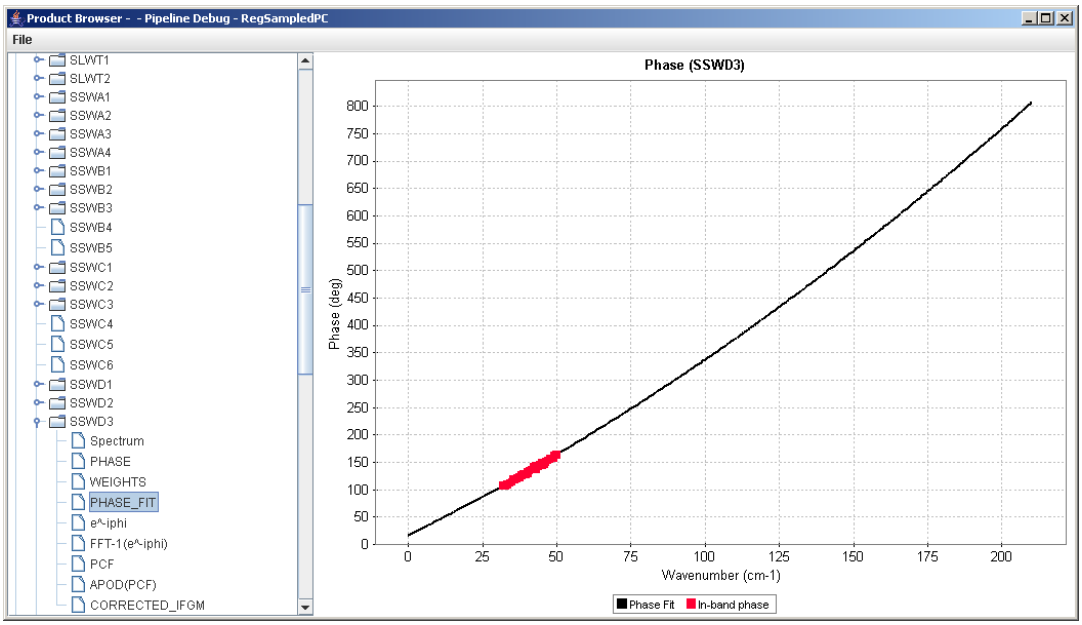


Figure 8.2. Measured phase of a typical asymmetric interferogram



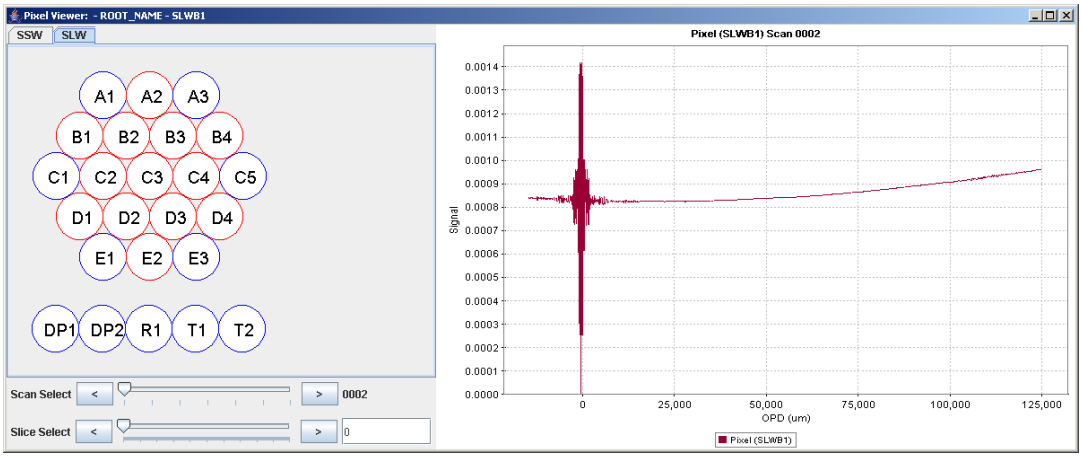
The red curve is the in-band portion of the measured phase; the black curve is a weighted fitted function to the in-band portion of the measured phase.

Figure 8.3. Fit To The Measured Phase

8.5. Single-sided phase correction

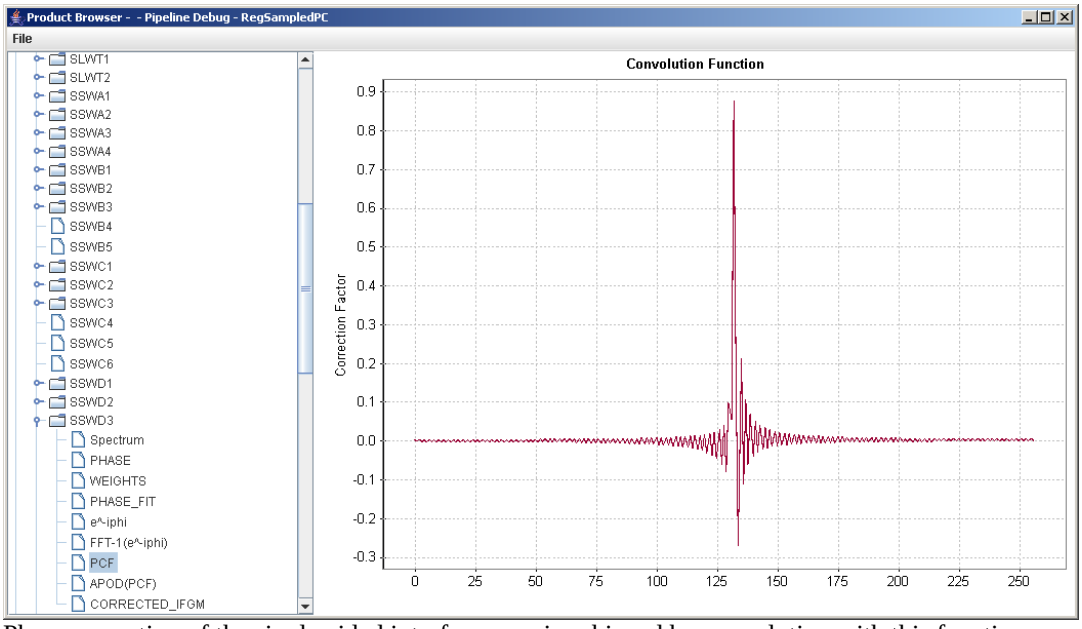
For single-sided interferograms (Section A.1.2), phase correction is performed by first computing the uncorrected spectrum (magnitude and phase) from the double-sided subsets of the single-sided interferograms (see Figure 8.4). The phase correction function is derived from a weighted in-band fit to the phase of the double-sided spectrum. Since this phase correction function is derived from the low-resolution portion of the interferogram, it cannot be directly applied to the uncorrected single-sided spectrum. Single-sided phase correction, by contrast, proceeds in the inteferogram (spatial) domain rather than the spectral domain.

As noted above, the PCF for single-sided interferograms (see Figure 8.4) is computed from the double-sided portion. This PCF is not multiplied with the uncorrected spectrum, instead the inverse FT of the PCF (see Figure 8.5) is convolved with the uncorrected single-sided interferogram, as in Equation 8.0.



The single-sided interferogram contains as a subset, a double-sided interferogram.

Figure 8.4. Single-Sided Interferogram



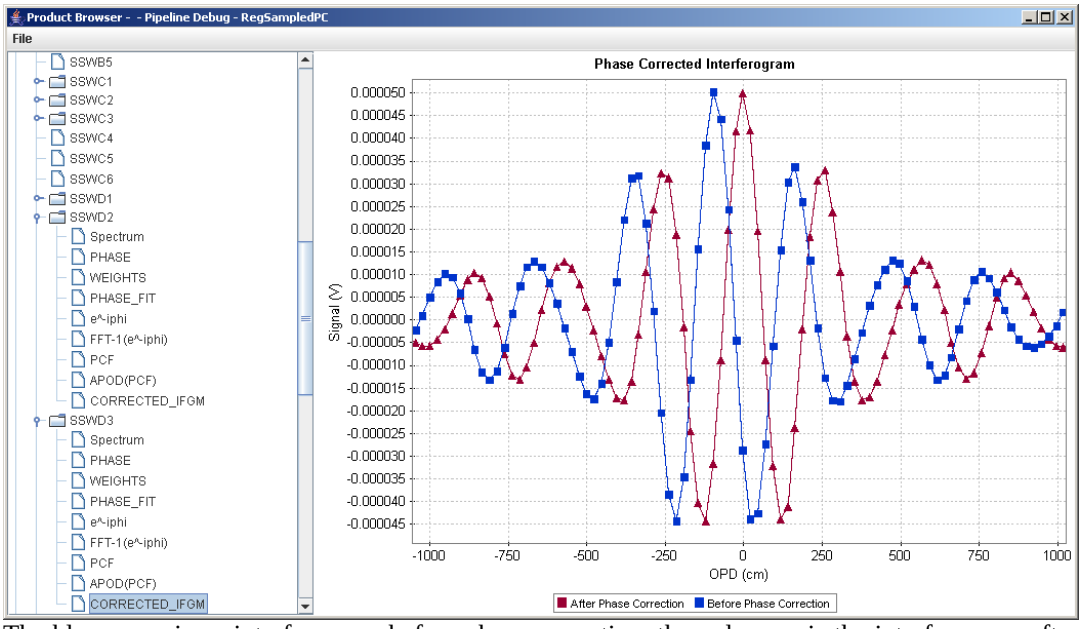
Phase correction of the single-sided interferogram is achieved by convolution with this function.

Figure 8.5. Inverse Transform of Phase Correction Function

$$I_{PC} = I(x) \otimes FT^{-1}(PCF) = I(x) \otimes FT^{-1}\, e^{i\Phi(\sigma)}$$

It should be noted that the operation expressed in Equation 8.0 above is mathematically equivalent to that for double-sided interferograms (Equation 8.0) since, from Fourier theory, multiplication in one domain is equivalent to convolution in the other domain. The sample signals in the interferogram that results from this convolution are rendered symmetric about ZPD (see Figure 8.6). This symmetry means that only that portion of the single-sided interferogram where the optical path dif-

ference is greater than or equal to ZPD is required to compute the resultant spectrum (see Figure 8.7).



The blue curve is an interferogram before phase correction, the red curve is the interferogram after phase-correction.

Figure 8.6. Phase-Corrected Interferogram

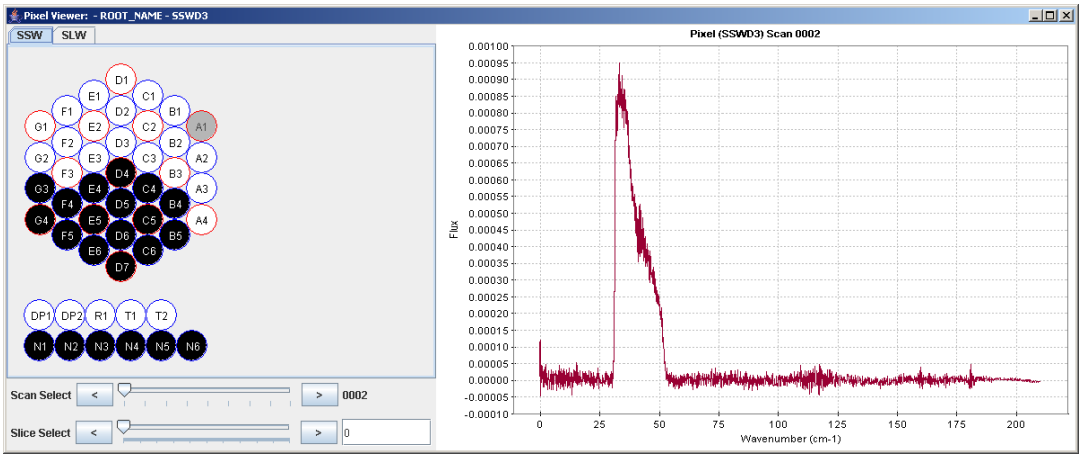


Figure 8.7. Spectrum from Single-Sided Interferogram

8.6. Outputs

SDI **Spectrometer Detector Interferogram Product** This output SDI contains phase-corrected interferograms for each spectrometer detector channel for each scan of the observation.

SDS **Spectrometer Detector Spectrum Product** The double-sided SDS that was provided as an input to this processing module will be modified such that its spectra will be phase-corrected.

8.7. Example

```
IA>> from herschel.spire.ia.modules.ft import *
IA>>
IA>> regSampledFT=RegSampledFT()
IA>> regSampledPhaseCorrection=RegSampledPhaseCorrection()
IA>>
IA>> dsds=regSampledFT(sdi=sdi,
                       side="doublesided")
IA>>
IA>> sdi=regSampledPhaseCorrection(sdi=sdi,
                                   dsSpec=dsds,
                                   calBandEdge=calBandEdge,
                                   pcfSize=128,
                                   pcfApod="aNB_15")
IA>>
```

8.8. Quality Control

A list of the quality control parameters for this module is given in Table 8.1 below.

Table 8.1. Quality Control Parameters

Quality Control Metric	Description	Format	Acceptable Range
Number of in-band phase flips	This parameter counts the number of times that the measured in-band phase changes by $\pm 2\pi$ ($=\pm 360^{o}$). Any such instance may an indication of an improper selection for the ZPD position.	Integer	0

8.9. Future Development

The major drawback to the current implementation of the Phase Correction Task is that the observed data is used to correct itself. This aspect poses a potential problem for the case where the measured data is of poor quality (i.e. the signal-to-noise ratio is low). A phase correction function derived from data of poor quality may not be reliable and its usage could result in further degradation of the data.

Future development of this module will involve using the observations from the ground test campaigns and those from the verification stage to quantify the systematic sources of phase error for the SPIRE iFTS. The phase correction function will then be computed from these known sources of phase error. Performing phase correction in this manner will remove the need to employ a fitting function to compute the phase correction function and remove the dependence of the phase correction function on the quality of the data.

Chapter 9. Fourier Transform

9.1. Purpose

The purpose of this module is to create a set of spectra for each spectrometer detector channel for a given observation.

9.2. Inputs

Data Products

SDI **Spectrometer Detector Interferogram Product** This product contains interferograms for each spectrometer detector channel for each scan of the observation.

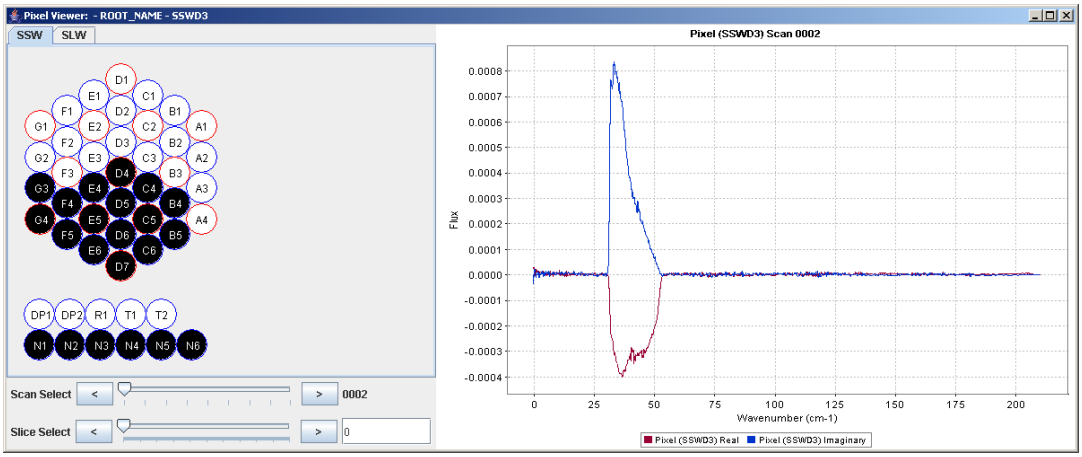
Control Parameters

side **Type of transform to be performed** The value of this parameter directs this task as to whether double-sided (side="doublesided") or single-sided (side="singlesided") FT should be computed.

9.3. Description

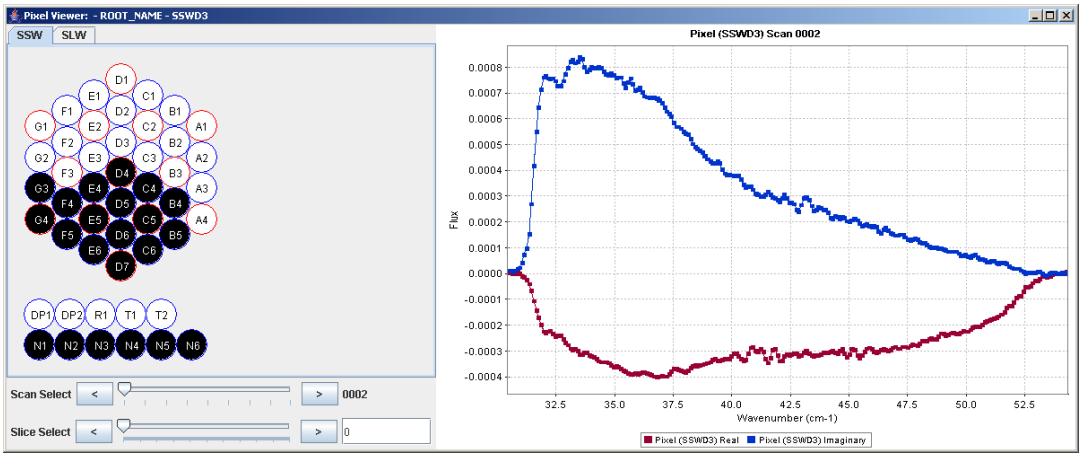
This task can process either single-sided or double-sided interferograms and does so depending on the input parameter "side". If the input interferograms are specified as double-sided then the calculated spectra contain both real and imaginary components. If the input interferograms are single-sided then only the portion of the imput interferograms where the OPD is greater than or equal to ZPD is used the resultant spectra are entirely real.

For the double-sided Fourier Transform, the individual interferograms of the input SDI product are examined and the double-sided portion of each interferogram is used to compute the resultant spectrum. The resultant spectra contain both real and imaginary components (see Figure 9.1, Figure 9.2)).



Spectrum from double-sided interferogram.

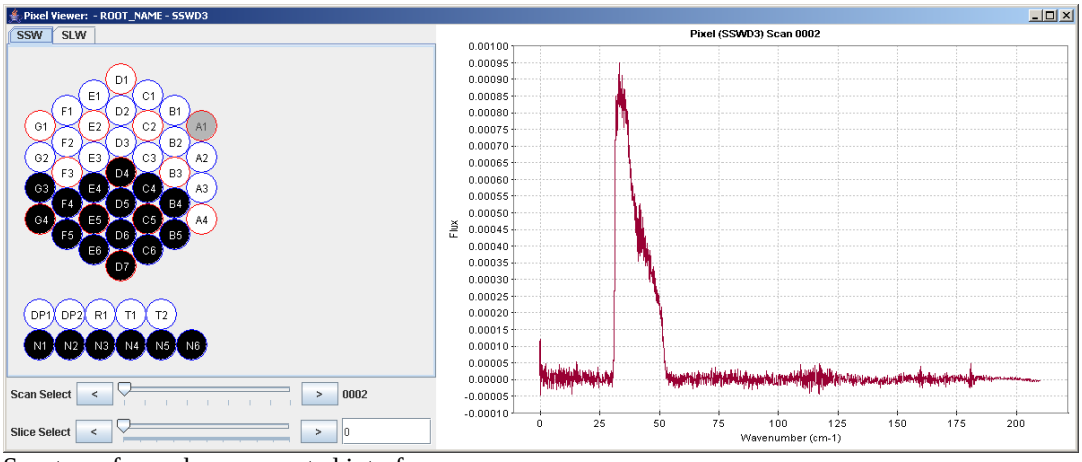
Figure 9.1. Double-sided Spectrum



Close-up version of the spectrum shown in Figure 9.1.

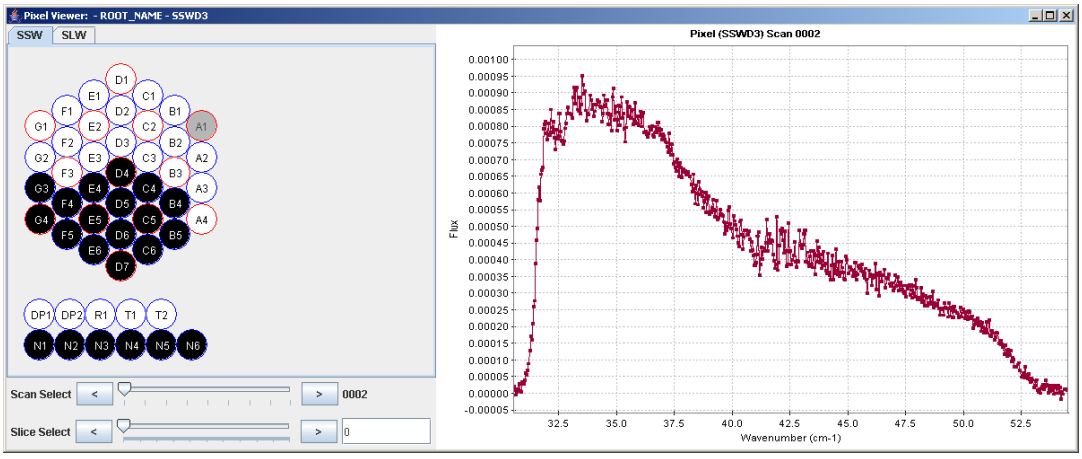
Figure 9.2. Double-sided Spectrum

In the case of the single-sided Fourier Transform, it is assumed that the interferograms in the input SDI product have been phase-corrected (Chapter 8), meaning that only those data on one side of ZPD are independent. As such, only that portion of the interferograms where OPD>=ZPD is used to compute the resultant spectra. As a result, the calculated spectra in the output SDS contain only real components (see Figure 9.3, Figure 9.4)).



Spectrum from phase-corrected interferogram.

Figure 9.3. Single-sided Spectrum



Close-up version of the spectrum shown in Figure 9.3.

Figure 9.4. Single-sided Spectrum

9.4. Outputs

SDS	**Spectrometer Detector Spectrum Product** The output SDS; contains the calculated spectra for each of the interferograms in the input SDI.

9.5. Example

The following are examples of the SPIRE spectrometer Fourier Transform task.

```
IA>> from herschel.spire.ia.modules.ft import *
IA>>
IA>> regSampledDoubleSidedFT=RegSampledFT()
IA>> dsds=regSampledDoubleSidedFT(sdi=sdi, side="doublesided")
IA>>
IA>> regSampledSingleSidedFT=RegSampledFT()
IA>> ssds=regSampledSingleSidedFT(sdi=sdi, side="singlesided")
IA>>
```

Appendix A. Definitions

A.1. Double-sided and Single-sided Interferograms

The terms double-sided and single-sided as used in this document describe the two types of interferograms that can be measured with a Fourier Transform Spectrometer.

A.1.1. Double-sided Interferograms

Double-sided interferograms are defined as those interferograms or that portion of measured interferogram where the sample *positions* are symmetric about the position of zero path difference (ZPD). That is, a double-sided interferogram is one that contains an equal number of samples before and after the ZPD sample [1]. An envelope of a double-sided interferogram is shown in Figure A.1.

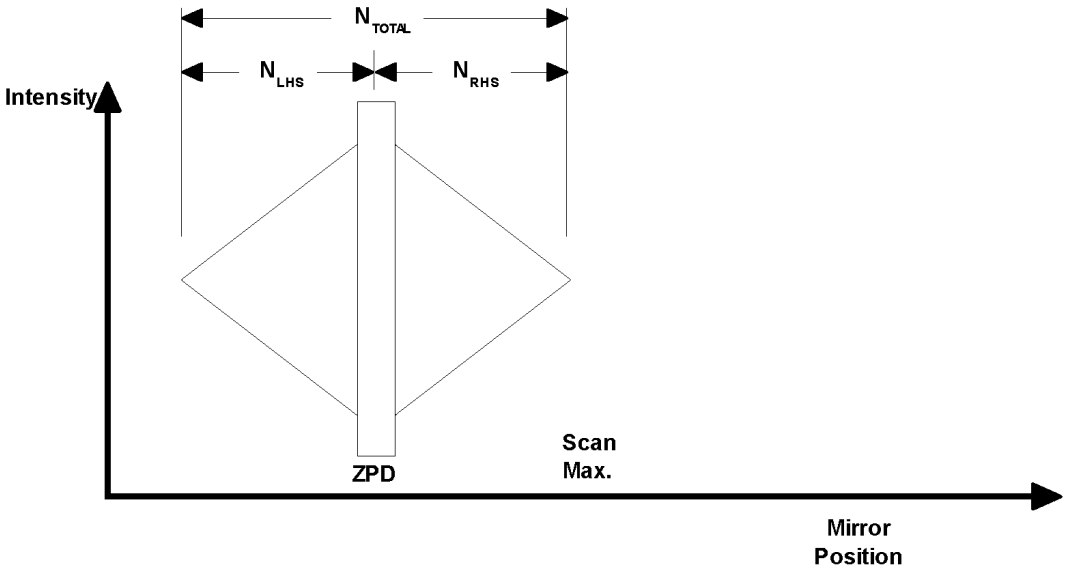


Figure A.1. Envelope of a double-sided interferogram

A.1.2. Single-sided Interferograms

Single-sided interferograms are defined as those interferograms that contain more samples on one side of ZPD than the other. An envelope of a single-sided interferogram is shown in Figure A.2.

[1] Some implementations of the Fourier Transform may require an even number of points (N_{TOTAL} even). In this case, the RHS of the double-sided interferogram will contains an extra point to render the total number of points even.

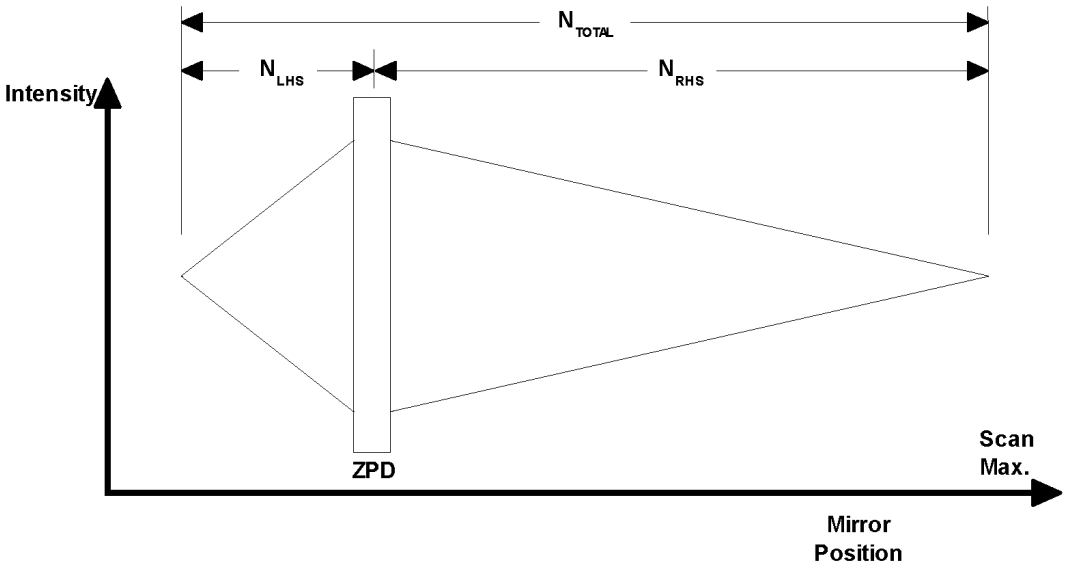


Figure A.2. Envelope of a single-sided interferogram

Appendix B. Apodization Functions

The following is a list and description of all of the apodization functions avalaible in the SPIRE data processing package.

Table B.1. Apodization Functions

Keyword Name	Apodization Function
aNB_W_120	Norton-Beer Weak
aNB_M_140	Norton-Beer Medium
aP_141	P with alpha = -0.0325 and p=0.3
aHM_150	Hamming
aNB_S_160	Norton-Beer strong
aD_174	D with alpha = 0.26
aBH_3_184	Blackman-Harris 3 Terms
aE_195	E with alpha = 0.22
aBH_4_221	Blackman-Harris 4-terms
aBH_M_222	Modified Blackman-Harris 4-terms
aGAUSS	Gaussian
aNB_11	FWHM = 1.1 SINC
aNB_12	FWHM = 1.2 SINC
aNB_13	FWHM = 1.3 SINC
aNB_14	FWHM = 1.4 SINC
aNB_15	FWHM = 1.5 SINC
aNB_16	FWHM = 1.6 SINC
aNB_17	FWHM = 1.7 SINC
aNB_18	FWHM = 1.8 SINC
aNB_19	FWHM = 1.9 SINC
aNB_20	FWHM = 2.0 SINC