

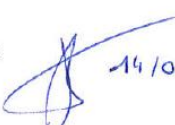
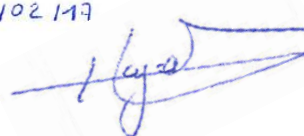


Sentinel-1 Level 1 Detailed Algorithm Definition

Prepared By: Riccardo Piantanida  14/02/2017

Checked By: Guillaume Hajduch  14/02/17

Quality Assurance: Julie Poullaouec  14/02/17

Project Manager: Guillaume Hajduch  14/02/17

Document Number: SEN-TN-52-7445

S-1 MPC Nomenclature: DI-MPC-IPFDPM

S-1 MPC Reference: MPC-0307

Sentinel-1 IPF CDRL Number: PAL1-1

CHANGE RECORD

From issue 1.0 to 1.6, the Sentinel-1 Level 1 Detailed Algorithm Definition was maintained by a consortium led by MDA under the reference S1-RS-MDA-52-7445.

The S-1 IPF and associated documentation is then maintained by the S-1 Mission Performance Centre. From the issue 2.0 the Sentinel-1 Level 1 Detailed Algorithm Definition is maintained by The S-1 Mission Performance Center which is a consortium led by CLS.

ISSUE	DATE	PAGE(S)	DESCRIPTION
1/0	June 24, 2009	All	First Issue
1/1	July 30, 2010	All	First Issue, First Revision
			Updates for major PDR L1 RIDs:
		Section 3.1	<ul style="list-style-type: none"> Added TOPSAR scientific overview (PDRL1-ALG-2)
		All	<ul style="list-style-type: none"> Added normalisation strategy (PDRL1-ALG-1, PDRL1-ALG-2)
		Section 4.2	<ul style="list-style-type: none"> Updated internal calibration strategy (PDRL1-ALG-1, PDRL1-ALG-4)
		Section 9.2.1	<ul style="list-style-type: none"> Added look extraction calculation (PDRL1-ALG-2)
		Section 6.2.4 Section 6.4	<ul style="list-style-type: none"> Updated TOPSAR sections (PDRL1-ALG-2, PDRL1-ALG-15 to PDRL1-ALG-20)
		Section 4.2.3.2	<ul style="list-style-type: none"> Added Replica Validation (PDR-ALG-8)
		Section 7.3.2	<ul style="list-style-type: none"> Enhanced description of optimal sub-swath cut ((PDRL1-ALG-25)
		Appendix D	<ul style="list-style-type: none"> Added Appendix ‘Summary of Auxiliary and Internal Parameters Referenced in Document’ (PDRL1-ALG-28)
			Updates for medium and minor PDR L1 RIDs:
		As marked	<ul style="list-style-type: none"> PDRL1-ALG-6, PDRL1-ALG-7, PDRL1-ALG-11, PDRL1-ALG-18, PDRL1-ALG-29, PDRL1-ALG-30, PDRL1-ALG-32, PDRL1-ALG-33.
			Updates for Change Request #2:
		Section 8	<ul style="list-style-type: none"> Added Slicing Support Section
		Section 9.1	Updated the Raw Data Decoding section

ISSUE	DATE	PAGE(S)	DESCRIPTION
1/2	August 20, 2010	All	First Issue, Second Revision
		Section 4.2	Internal calibration updates
		Section 6.1.1	Added support for the SWST bias
		Appendix D	Updated table D-1 as per updates to section 4.2
1/3	May 18, 2011	All	First Issue, Third Revision
		All	Removed Browse products.
		Sections 6 and 7	Moved some algorithm steps from SLC processing to post-processing.
		Section 8	Updated slicing algorithms.
		Figure 3-3	Removed input ISP product.
			Removed output product metadata.
			Replaced intermediate processed data with internal SLC product.
			Updates for major Delta PDR L1 RIDs:
		Section 6.1.2	S1IPFPDR-233: Section 6.1.2.1 was added to discuss SWST change handling.
		As marked	Updates for medium and minor Delta PDR L1 RIDs:
			S1IPFPDR-230, S1IPFPDR-232, S1IPFPDR-234, S1IPFPDR-235, S1IPFPDR-238, S1IPFPDR-239, S1IPFPDR-241.
			The following RIDs were raised but no updates to this document were required:
			S1IPFPDR-215, S1IPFPDR-231, S1IPFPDR-236, S1IPFPDR-237, S1IPFPDR-240, S1IPFPDR-242, S1IPFPDR-243, S1IPFPDR-244
1/4	Sept. 27, 2012		First Issue, Fourth Revision
		Section 4.2	Updates to internal calibration.
		Section 4.3	Discussion of pre-processing missing line handling.
		Section 5	Updated Doppler Centroid estimation sections to reflect changes to IPF, including replacing SNR with RMS error.
		Eq. 6-14, 6-40	Corrections to TOPS sweep bandwidth equation.

ISSUE	DATE	PAGE(S)	DESCRIPTION
		Section 9.17	Updated noise vector formula.
		Sections 7.3.4, 9.18, 9.19	S1IPF-58: Updated Application Scaling and Absolute Calibration Vectors sections to clarify the descriptions. Updates for PDR RIDs/Actions:
		Section 8.1	S1IPFPDRL2-12: Added calculation for number of slices
		Section 9.19	S1IPFPDRL2-53 / DPDRL2-A3: Updates to calibration vectors to use common SLC/GRD formula.
		Section 9.23	S1IPFPDR-179: Added bi-static correction. Updates for CDR L1&L2 minor RIDs:
		All	IPFCDR-21, IPFCDR-24, IPFCDR-85, IPFCDR-87, IPFCDR-88, IPFCDR-89, IPFCDR-90, IPFCDR-91, IPFCDR-94, IPFCDR-106
1/5	Mar. 28, 2014		First Issue, Fifth Revision
		4.2	Updated Internal Calibration
		5.4	Updated absolute DC estimation steps
		6.1.1	Added note about chirp duration being limited for spurious signal correction.
		6.1.4	Corrected equation for the application of the SWST bias.
		9.2	Added Spurious Signal Correction and Receiver Gain Compensation
		9.3	Moved the definition of the Nominal Replica to section 4.2
		Table D-1	Removed replica whitening flag. Removed electronic losses constants. Added $TXPL_{nom}$.
1/6	Nov. 28, 2014		First Issue, Sixth Revision
		2.2	Added reference to EOFCFI technote.
		4.2.1.2	Corrected description of PG calibration pulses.
		4.2.1.4	Corrected section format and added descriptions of the parameters in the time delay equation.
		6.1.1	Updated normalisation of the RRF.

ISSUE	DATE	PAGE(S)	DESCRIPTION
		6.1.2/6.2.2	Moved the range compression section under the azimuth pre-processing section.
		6.2.4	Added f_{ref} parameter. Updated the calculation of the number of spectral replicas. Updated the range of the azimuth frequency.
		6.3.4.1	Updated azimuth matched filter normalisation.
		6.4.1	Updated the calculation of the number of time replicas. Updated the time support range.
		6.4.1.2	Updated the de-ramping function.
		6.4.1.4	Updated the re-ramping function.
		7.1	Updated range processing normalisation.
		7.2	Updated azimuth processing normalisation.
		9.4	Corrected section reference for PG. Corrected application of PG to the signal data.
		9.5.1	Updated for complex EAP correction. Added reference to EOCCI procedure for calculation of the elevation boresight angle.
		9.17.1	Updated for the application of complex EAP correction.
		9.19	Corrected equation for application of absolute calibration vectors.
		Table C-1	Updated summary of normalisation steps.
		Table D-1	Corrected source data for nominal roll steering input parameters. Added nominal chirp coefficients. Added Δt_{guard1} and Δt_{suppr} .
2/0	Feb. 29, 2016		Second Issue
		9.5.1	IPF-151: Updated EAP correction formula
		9.15	IPF-178: Updated AAP correction formula for TOPSAR data
		7.3.2.1	IPF-179: Removed section on optimal sub-swath cut to perform range direction merging of TOPSAR GRD data



Sentinel-1

Ref:
MPC Nom: DI-MPC-IPFDPM
MPC Ref: MPC-0307
Issue/Revision: 2/1
Date: 31/01/2017

ISSUE	DATE	PAGE(S)	DESCRIPTION
2/1	Jan. 31, 2017		Second Issue, First Revision
		6.2.4.3/6.4.1.3	IPF-256: Added description of the new method of low-pass filtering and resampling

TABLE OF CONTENTS

1	INTRODUCTION.....	1-1
1.1	Purpose.....	1-1
1.2	Scope.....	1-1
1.3	Key Terminology	1-2
1.4	Document Structure	1-2
2	DOCUMENTS.....	2-1
2.1	Applicable Documents	2-1
2.2	Reference Documents	2-1
3	OVERVIEW.....	3-1
3.1	TOPSAR Specific Processing.....	3-5
3.1.1	Mode Overview	3-5
3.1.2	Scientific Overview	3-6
3.2	Dual-Polarisation Processing	3-8
4	PRE-PROCESSING ALGORITHMS	4-1
4.1	Raw Data Analysis.....	4-1
4.1.1	IQ Quadrature departure.....	4-3
4.2	Internal Calibration	4-4
4.2.1	Algorithm Overview.....	4-5
4.2.2	Algorithm Implementation	4-17
4.3	Downlink Header Validation	4-19
4.3.1	Missing Line Detection	4-25
4.4	Terrain Height Function.....	4-26
5	DOPPLER CENTROID ESTIMATION ALGORITHMS.....	5-1
5.1	Absolute DC Calculation (from Orbit & Attitude)	5-3
5.2	Fine DC Estimation.....	5-6
5.2.1	DCE Pre-Conditioning (TOPSAR only)	5-7
5.2.2	Correlation DC Estimator (CDCE)	5-7
5.3	Fine DC Estimates Unwrapping	5-8
5.3.1	Algorithm Overview.....	5-9
5.3.2	Algorithm Implementation	5-10
5.4	Absolute DC Estimation	5-10
5.5	Polynomial Fitting.....	5-11
5.5.1	DC Quality Measurement.....	5-12
5.6	Processing Blocks Dimensions	5-12
6	SLC PROCESSING ALGORITHMS.....	6-1
6.1	Range Processing	6-2
6.1.1	Range Reference Function (RRF)	6-3
6.1.2	Range Dependent Gain Correction.....	6-4
6.1.3	SWST Bias Correction	6-4
6.2	Azimuth Pre-Processing.....	6-5

6.2.1	Azimuth Zero-Padding	6-5
6.2.2	Range Compression.....	6-5
6.2.3	Azimuth Forward FFT.....	6-9
6.2.4	Azimuth Frequency UFR (TOPSAR only)	6-9
6.3	Azimuth Processing	6-17
6.3.1	Secondary Range Compression (SRC).....	6-19
6.3.2	Range Cell Migration Correction (RCMC).....	6-20
6.3.3	Range Resampling (TOPSAR only).....	6-21
6.3.4	Azimuth Compression.....	6-21
6.4	Azimuth Post-processing (TOPSAR only)	6-23
6.4.1	Azimuth Time UFR.....	6-24
7	POST-PROCESSING ALGORITHMS.....	7-1
7.1	Post-Processing Range Processing.....	7-4
7.2	Post-Processing Azimuth Processing.....	7-5
7.2.1	De-scalloping.....	7-6
7.3	Post-Processing Output Processing.....	7-9
7.3.1	Thermal Noise Removal (GRD Only).....	7-9
7.3.2	Burst Merging (TOPSAR GRD Only)	7-10
7.3.3	Quick-Look (QL) Image Generation	7-11
7.3.4	Output Processing.....	7-11
8	SLICING SUPPORT	8-1
8.1	L0 Input Slice Definition	8-1
8.2	Internal Signal Data Slice Definition	8-2
8.2.1	Stripmap Case	8-3
8.2.2	TOPSAR Case.....	8-4
8.3	Output Slice Definition	8-5
8.3.1	Stripmap SLC Case	8-5
8.3.2	TOPSAR SLC Case.....	8-7
8.3.3	Stripmap GRD Case	8-7
8.3.4	TOPSAR GRD Case	8-7
9	COMMON AND SUPPORTING ALGORITHMS	9-1
9.1	Raw Data Decoding	9-2
9.1.1	Raw Data Decoding Overview	9-2
9.1.2	Raw Data Decoding Algorithms	9-2
9.2	Raw Data Correction.....	9-7
9.2.1	I Q Bias Correction.....	9-7
9.2.2	Spurious Signal Correction	9-8
9.2.3	Receiver Gain Compensation.....	9-9
9.3	Nominal Replica	9-11
9.4	Drift Compensation.....	9-11
9.5	Bank of Elevation Antenna Pattern (EAP) Correction Vectors	9-11
9.5.1	EAP Correction	9-12
9.6	Range Spreading Loss Correction.....	9-13
9.7	Weighting Window	9-14
9.8	Azimuth Frequency Vector	9-14

9.9	Generic Unfolding and Resampling (UFR) (TOPSAR only)	9-14
9.10	Effective Radar Velocity.....	9-15
9.11	Azimuth FM Rate	9-16
9.12	Focusing Azimuth Block Overlap (Stripmap only)	9-17
9.13	Focusing Azimuth Block Length (Stripmap only).....	9-18
9.14	Antenna Steering Rate and DC Rate (TOPSAR only).....	9-18
9.15	Azimuth Antenna Pattern.....	9-19
9.16	Bank of GR/SR LUTs	9-20
9.16.1	GR/SR Polynomial Coefficients	9-21
9.16.2	GR/SR LUT.....	9-22
9.17	Bank of Thermal Noise Estimation Vectors	9-22
9.17.1	Algorithm Overview.....	9-23
9.17.2	Algorithm Implementation	9-25
9.18	Application Scaling.....	9-26
9.19	Absolute Calibration Vectors.....	9-27
9.20	Fourier Transform and Energy of a Signal.....	9-29
9.21	Look Extraction	9-29
9.22	Resampling Interpolator.....	9-31
9.23	Bi-static Delay Correction.....	9-31
9.23.2	Section 6.2.2 Range Compression.....	B-2

LIST OF FIGURES

Figure 3-1	Sentinel-1 Acquisition Modes	3-1
Figure 3-2	Sentinel-1 Level 1 Product Family Tree	3-3
Figure 3-3	Sentinel-1 IPF Architecture	3-4
Figure 3-4	Schematic Representation of the TOPSAR Mode Acquisition.....	3-5
Figure 3-5	TOPSAR versus SCANSAR Azimuth Antenna Pattern Weighting	3-8
Figure 4-1	Timing of Calibration Signal Sampling Window	4-8
Figure 4-2	Flow Chart for Extracted Replica Coefficients Derivation	4-14
Figure 5-1	High Level DCE Algorithm	5-1
Figure 5-2	Calculation of Individual Absolute DC Estimates	5-2
Figure 5-3	SAR Geometry	5-4
Figure 6-1	SLC Processing Algorithm.....	6-2
Figure 6-2	Range Compressed Data Buffer Containing Blackfill to Account for SWST Change	6-7
Figure 6-3	Range Lines with SWST Changes and No SWL Changes.....	6-8
Figure 6-4	Range Lines with No SWST Changes but with SWL Changes	6-8
Figure 6-5	Range Lines with SWST Changes and SWL Changes	6-9
Figure 6-6	Time-Frequency Diagram of an Input TOPSAR Azimuth Line	6-10
Figure 6-7	Time-Frequency Diagram of a TOPSAR Azimuth Line after Mosaicking.....	6-13
Figure 6-8	Low-pass Filter Sample Mask for Azimuth Frequency UFR.....	6-15
Figure 6-9	Comparison of Low-pass Filter responses for Azimuth Frequency UFR	6-16
Figure 6-10	Time-frequency Diagram of a TOPSAR Azimuth Line after Frequency Unfolding (without re-sampling)	6-17
Figure 6-11	Time-frequency Diagram of a TOPSAR Azimuth Compressed Line	6-24
Figure 6-12	Time-frequency Diagram of a TOPSAR Azimuth Line after Time-domain Mosaicking.....	6-26
Figure 6-13	Low-pass Filter Mask for Azimuth Time UFR	6-28
Figure 6-14	Comparison of Low-pass Filter responses for Azimuth Time UFR	6-29
Figure 6-15	Time-frequency Diagram of a TOPSAR Azimuth Line after Time-domain Unfolding	6-30
Figure 7-1	L1 Post-Processing Algorithm (High Level View)	7-2
Figure 7-2	L1 Post-Processing Algorithm (Detailed View).....	7-3
Figure 7-3	Scalloping Gain versus Focused Burst Azimuth Time for IW Mode	7-7
Figure 9-1	Decoding Process	9-3
Figure 9-2	Timing Schematic for Receiver Gain Compensation.....	9-9
Figure 9-3	Generic Unfolding and Re-sampling.....	9-15
Figure 6-2	Range Compressed Data Buffer Containing Blackfill to Account for SWST Change	Erreur ! Signet non défini.
Figure 6-3	Range Lines with SWST Changes and No SWL Changes.....	Erreur ! Signet non défini.
Figure 6-4	Range Lines with No SWST Changes but with SWL Changes.....	Erreur ! Signet non défini.
Figure 6-5	Range Lines with SWST Changes and SWL Changes	Erreur ! Signet non défini.

LIST OF TABLES

Table 4-1	Sentinel-1 IPF Downlink Header Validation.....	4-21
Table 9-1	Raw Data Encodings	9-3
Table 9-2	Raw Data Sample Types.....	9-4
Table 9-3	Encoding Terms.....	9-4
Table B-1	2D Data Domains.....	B-1
Table A-1	Normalisation Steps Summary	A-1
Table B-1	Summary of Auxiliary and Internal Parameters Referenced in the Document.....	B-1

ACRONYMS AND ABBREVIATIONS

ACCC	Average Cross Correlation Coefficient
AAP	(2-way) Azimuth Antenna Pattern
APDN	Azimuth Plane Distribution Network
BAQ	Block Adaptive Quantization
DC	Doppler Centroid
CDCE	Correlation Doppler Centroid Estimator
CDR	Critical Design Review
CLS	Collecte Localisation Satellites
CWL	Calibration Window Length
DCE	Doppler Centroid Estimation (or Estimate)
DEM	Digital Elevation Model
DFF	Doppler Frequency Function
DTAR	Distributed Target Ambiguity Ratio
ECR	Earth Centered Rotational (Coordinate System)
EPDN	Elevation Plane Distribution Network
FDBAQ	Flexible Dynamic BAQ
FFT	Fast Fourier Transform
FM	Frequency Modulation
GRD	Ground-Range Detected (image or product)
ISLR	Integrated Side Lobe Ratio
SR/GR	Slant Range to Ground Range
G/S	Ground Segment
GSI	Generalized Stolt Interpolation
HR	High Resolution
I and Q	In-phase and Quadrature
IEEE	Institute of Electrical and Electronics Engineers
IFFT	Inverse Fast Fourier Transform
IPF	Instrument Processing Facility
ISP	Instrument Source Packet
L0	Level 0
L1	Level 1

L2	Level 2
LR	Low Resolution
LSI	Linear Stolt Interpolation
MDA	MDA Systems Ltd.
MR	Medium Resolution
NEBZ	Noise-Equivalent Beta-Zero
NESZ	Noise-Equivalent Sigma-Zero
NEG	Noise-Equivalent Gamma
NORM	Normalisation (algorithmic steps tagged by this abbreviation are part of the IPF normalisation strategy)
PCC	Pulse Coded Calibration
PG	Transmit Power - Receive Gain (product)
PDR	Preliminary Design Review
PRF	Pulse Repetition Frequency
PSLR	Peak-Side Lobe Ratio
PTAR	Point-Target Ambiguity Ratio
QL	Quick-Look (image)
RCM	Range Cell Migration
RCMC	Range Cell Migration Correction
RDA	Range-Doppler Algorithm
RR	Reduced Resolution
RRF	Range Reference Function
RX	Receive
SAR	Synthetic Aperture Radar
SNR	Signal-to-Noise Ratio
SOA	Straight Orbit Assumption
SRC	Secondary Range Compression
S/S	Space Segment
SWST	Sampling Window Start Time
TOPSAR	Terrain Observations for Progressive Scans SAR
TPL	Tx Pulse Length
TSR	Target Slant Range coordinate system
TX	Transmit
TX/RX	Transmit/Receive



Sentinel-1

Ref:
MPC Nom: DI-MPC-IPFDPM
MPC Ref: MPC-0307
Issue/Revision: 2/1
Date: 31/01/2017

UFR

Un-Folding and Re-sampling

1 INTRODUCTION

1.1 Purpose

This document describes the processing algorithms employed by the Sentinel-1 Image Processing Facility (IPF) for the generation of Sentinel-1 Level 1 (L1) products. The algorithms apply to the processing of Sentinel-1 acquisition modes: Stripmap, Interferometric Wide-swath (IW), Extra-wide-swath (EW) and Wave. In IW and EW the data is acquired with the Terrain Observation with Progressive Scans SAR (TOPSAR) technique. For the purpose of this document, IW and EW will be referred to collectively as the TOPSAR mode.

1.2 Scope

This document satisfies the Sentinel-1 IPF deliverables PAL1-1 described in the Sentinel-1 Product Definitions & Instrument Processing Facility Development Statement of Work (SOW) [A-1] and the MDA Proposal to ESA for Sentinel-1 IPF Contract Change Notice No. 2 [A-5]. This document defines the algorithms that fulfil the requirements in the Sentinel-1 IPF System Requirements Document (SRD) [A-2].

This document presents the Level 1 processing algorithms and equations as they will be implemented in the Sentinel-1 IPF software. Document [R-1] describes the Sentinel-1 IPF software design for the implementation of the Level 1 processing algorithms and equations.

This document does not attempt to provide detailed derivations of the equations describing the algorithms, which are discussed in other documents. Many of the Sentinel-1 IPF algorithms discussed in this document are also described in further detail in [R-5]. For the TOPSAR-specific algorithms, additional details, justifications, and descriptions of the mode design and algorithms can be found in [R-2] and [R-4].

As the IPF software will also support the processing of ENVISAT ASAR Stripmap and Wave mode data, this document also presents the ASAR specific processing algorithms. Note that, apart from pre-processing, the IPF will employ the same algorithms for both Sentinel-1 and ENVISAT ASAR, with relatively minor, local adjustments to be reflected in the software design document [R-1].

The algorithm for the Internal Calibration is based on the current version of Document [A-3], modified by information contained in Document [R-13].

1.3 Key Terminology

Azimuth line – a sequence of samples for which the azimuth (or slow) time varies from sample to sample.

Range line – a sequence of samples for which the range (or fast) time varies from sample to sample. Echoes of RADAR pulses are the original range lines input to the IPF.

Raw (or Signal) Data – the SAR echo data. For reasons of compatibility with existing conventions, sometimes raw data (as in Raw Data Decoding or Raw Data Correction) and sometimes signal data (as in ‘Signal Data file’, the output of the pre-processing stage) will be used. The raw data is usually BAQ encoded. However, in some cases ‘raw data’ is understood as BAQ decoded data (as in Raw Data Correction which is performed on BAQ-decoded data). The BAQ state of the signal (raw) data will be made clear in each context.

Azimuth Block – A group of consecutive range lines. Unless otherwise specified, ‘azimuth block’ will be used in a generic sense to also designate bursts (TOPSAR) or vignettes (Wave Mode) besides Stripmap azimuth blocks.

1.4 Document Structure

This document is structured as follows:

Section 1 introduces the purpose, scope key terminology and structure of the document.

Section 2 lists the applicable and reference documents.

Section 3 provides an overview of the Level 1 product types and processing modules supported by the Sentinel-1 IPF.

Section 4 describes the pre-processing algorithms used on Level 0 data.

Section 5 presents the DCE processing algorithms.

Section 6 presents the SLC image processing algorithms.

Section 7 presents the L1 post-processing algorithms.

Section 8 presents supporting algorithms for processing sliced data.

Section 9 describes a number of either lower level algorithms or algorithms common to multiple processing modules.

Appendix A provides a list of major symbols used in the document.

Appendix B provides summarizes the domains of the 2D data that makes the object of various stages of the processing.

Appendix A presents a summary of the Sentinel-1 processor normalisation scheme. Note that each normalisation step is also presented in the context of the particular algorithm to which it applies in one of Sections 1 to 8. For easy identification, the normalisation steps are tagged with the abbreviation NORM.

Appendix B presents a summary of the auxiliary and internal parameters referenced in the document.

2 DOCUMENTS

2.1 Applicable Documents

The following documents of the date/revision indicated form part of this document to the extent referenced herein. Any conflict between this document and any of the applicable documents should be brought to the attention of MacDonald Dettwiler for resolution.

- | | | |
|-----|----------------------------|--|
| A-1 | GMES-DFPR-EOPG-SW-07-00006 | Sentinel-1 Product Definitions & Instrument Processing Facility Development Statement of Work, Issue/Revision 4/1, 23-05-2008. |
| A-2 | S1-RS-MDA-52-7452 | Sentinel-1 IPF System Requirements Document. Issue/Revision 2/2. Apr. 20, 2012. MacDonald Dettwiler. |
| A-3 | S1-PL-ASD-PL-0001 | Sentinel-1 SAR Instrument Calibration and Characterisation Plan, Issue 7.1, June 7, 2013, EADS Astrium. |
| A-4 | S1-RS-MDA-52-7440 | Sentinel-1 Product Definition, Issue/Revision 2/5, Mar. 28, 2014, MacDonald Dettwiler. |
| A-5 | CCN No.2 | Contract Change Notice No. 2, Changes in ESRIN Contract No. 21722/08/I-LG, June 21, 2010. |
| A-6 | S1-IF-ASD-PL-0007 | Sentinel-1 SAR Space Packet Protocol Data Unit, Issue 10, Jan 3, 2013, EADS Astrium. |
| A-7 | 01-7833A | MDA Proposal to ESA for Sentinel-1 IPF Contract Change Request #05. June 5, 2013. MDA. |

2.2 Reference Documents

The following documents provide useful reference information associated with this document. These documents are to be used for information only. Changes to the date/revision number (if provided) do not make this document out of date.

- | | | |
|-----|-------------------|---|
| R-1 | S1-DD-MDA-52-7453 | Sentinel-1 IPF Design Document, Issue/Revision 1/3, September, 2012, MacDonald Dettwiler. |
|-----|-------------------|---|

- R-2 S1-TN-ARE-GS-0001 Technical Note on Sentinel-1 SAR Processing Algorithms, Issue 4.0, February 26, 2010, Aresys.
- R-3 IEEE Research Paper Estimating the Doppler Centroid of SAR data, IEEE Trans. on Aerospace and Electronic Systems, Vol. 25, No. 2, pp. 134-140, March 1989, S. N. Madsen.
- R-4 IEEE Research Paper TOPSAR: Terrain Observation by Progressive Scans, IEEE Trans. Geoscience and Remote Sensing, Vol. 44, No. 9, September 2006, Francesco de Zan and Andrea Monti Guarnieri.
- R-5 Digital Processing of Synthetic Aperture Radar Data, 2005 Artech House, Inc., Ian G. Cumming and Frank H. Wong.
- R-6 PO-RS-ESA-GS-00245 ENVISAT Ground Segment: ASAR Processing and Product Quality Requirements, Issue 3.3, 1 July 1997, ESA
- R-7 Synthetic Aperture Radar Systems and Signal Processing, 1991 John Wiley & Sons Inc, John C. Curlander and Robert N. McDonough.
- R-8 PX-TN-51-0756 Technical Note: ENVISAT ASAR Data Decoding, Issue/Revision 1/0, January 7, 2000, MacDonald Dettwiler.
- R-9 Aerospace Avionics Systems, 1993 Academic Press Inc. George M. Siouris.
- R-10 PO-TN-MDA-GZ-2069 ENVISAT PF-ASAR Doppler Centroid Frequency Estimator, Issue 2, Revision A, 11/09/1997, MacDonald Dettwiler.
- R-11 IEEE Research Paper Doppler Centroid Estimation for ScanSAR Data, IEEE Trans. Geoscience and Remote Sensing, Vol. 42, Issue 1, pp. 14-23, January 2004, Ciro Cafforio, Pietro Guccione and Andrea Monti Guarnieri.
- R-12 S1-TN-ASD-PL-0021 Orbit Dependent Instrument Parameters, Issue 1, 30 July 2008, EADS Astrium.
- R-13 S1-DD-DLR-SY-002 GMES Sentinel-1 SAR Calibration Algorithms, Issue/Revision 2.0, April 26 2010, DLR.
- R-14 S1-IC-MDA-52-7454 Sentinel-1 Interface Control Document, Issue/Revision 1/8, Mar. 28, 2014. MacDonald Dettwiler.



Sentinel-1

Ref:
MPC Nom: DI-MPC-IPFDPM
MPC Ref: MPC-0307
Issue/Revision: 2/1
Date: 31/01/2017

R-15 EOCFI-NOTE-052

Sentinel-1 IPF Attitude Quaternions Usage, Issue 2.1,
Oct. 14, 2014. ESA

3 OVERVIEW

The Sentinel-1 SAR can be operated in one of four nominal acquisition modes (see Figure 3-1):

- Stripmap Mode (SM)
- Interferometric Wide-swath Mode (IW)
- Extra-Wide-swath Mode (EW)
- Wave Mode (WV)

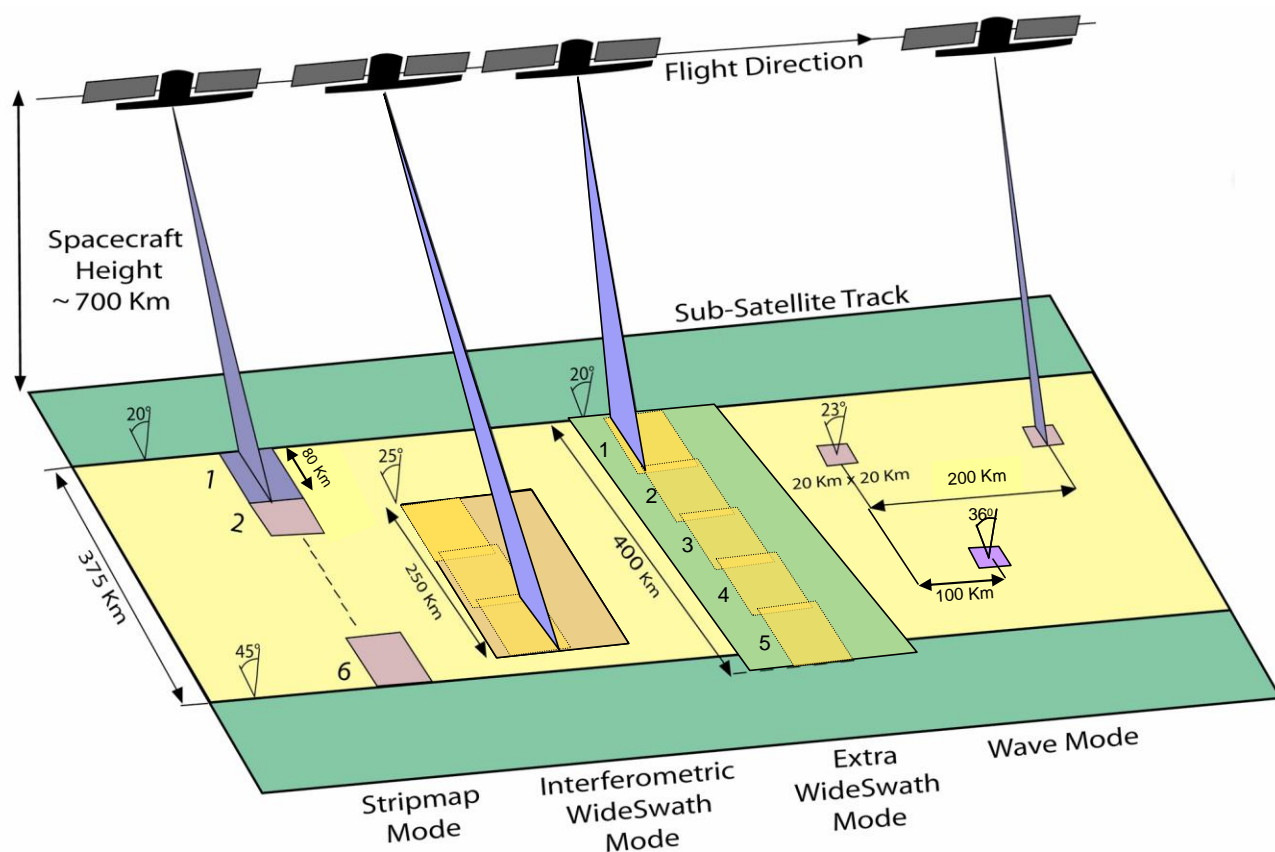


Figure 3-1 Sentinel-1 Acquisition Modes

A brief description of the Sentinel-1 acquisition modes can be found in [A-4]. Although these four modes all have different properties, they generally fall into two categories: single beam, single-swath Stripmap modes (SM and WV), and multi-swath TOPSAR modes (IW and EW). Note that the WV mode is a sampled Stripmap mode in which ‘vignettes’ of data are acquired alternatively at two different incidence angles (i.e. in ‘leap frog’ mode). However, this particular type of acquisition has no impact on the processing algorithms employed: each vignette is treated as Stripmap data while processed individually.

The Sentinel-1 Instrument Processing Facility (IPF) can generate the following Level 1 products from these four acquisition modes:

- Slant Range, Single-Look Complex (**SLC**)
- Ground Range, Multi-Look, Detected (**GRD**)

GRD products are further classified according to their resolution into:

- Full Resolution (**FR**) (SM mode)
- High Resolution (**HR**) (SM, IW and EW mode)
- Medium Resolution (**MR**) (SM, IW, EW and WV modes)

The definition for each product type is given in [A-4]. The Sentinel-1 L1 product family is shown in Figure 3-2.

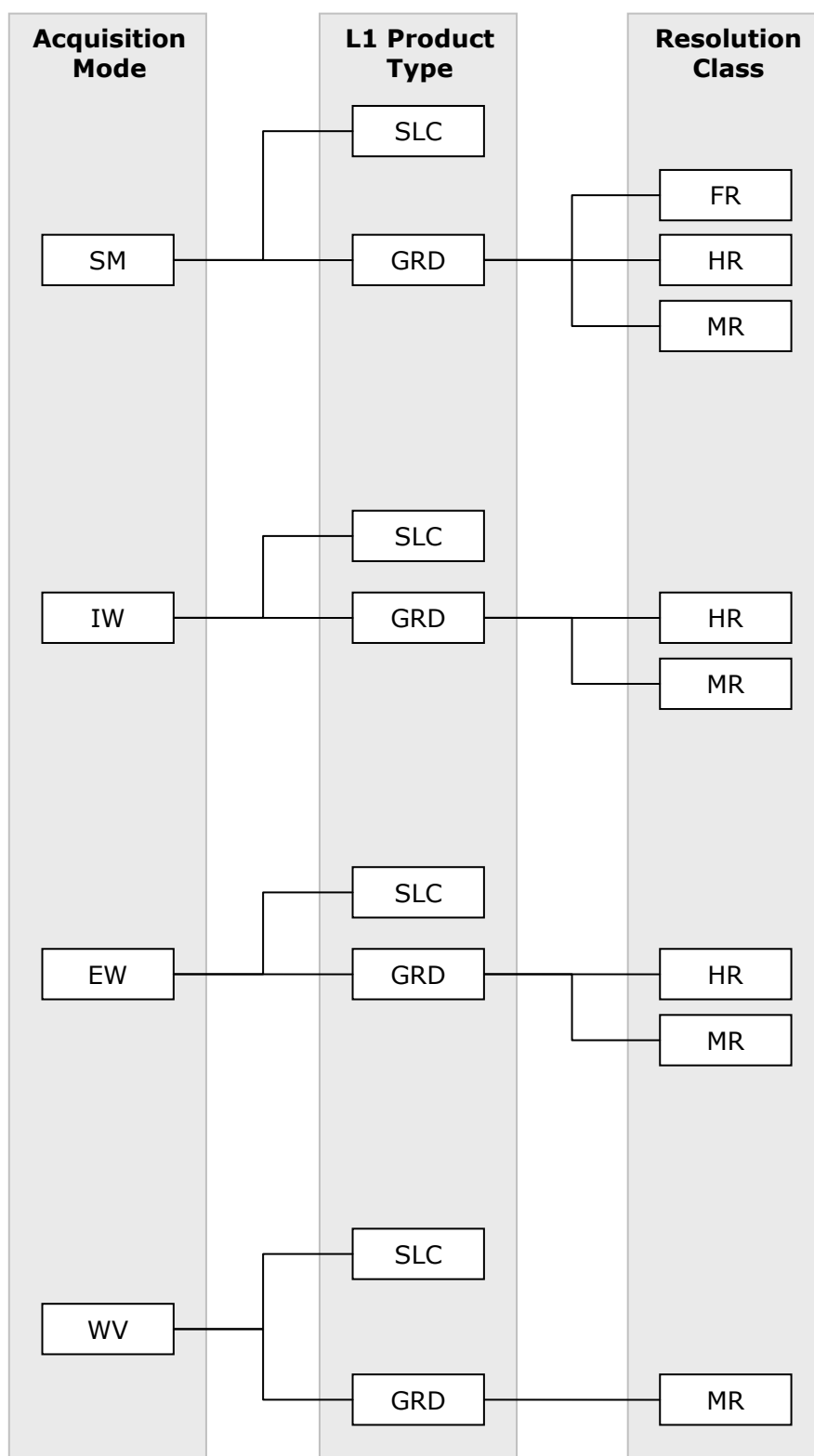


Figure 3-2 Sentinel-1 Level 1 Product Family Tree

In order to generate these products, the Sentinel-1 IPF is composed of a number of software components, as are shown in Figure 3-3. The complete Sentinel-1 IPF software design is described in [R-1].

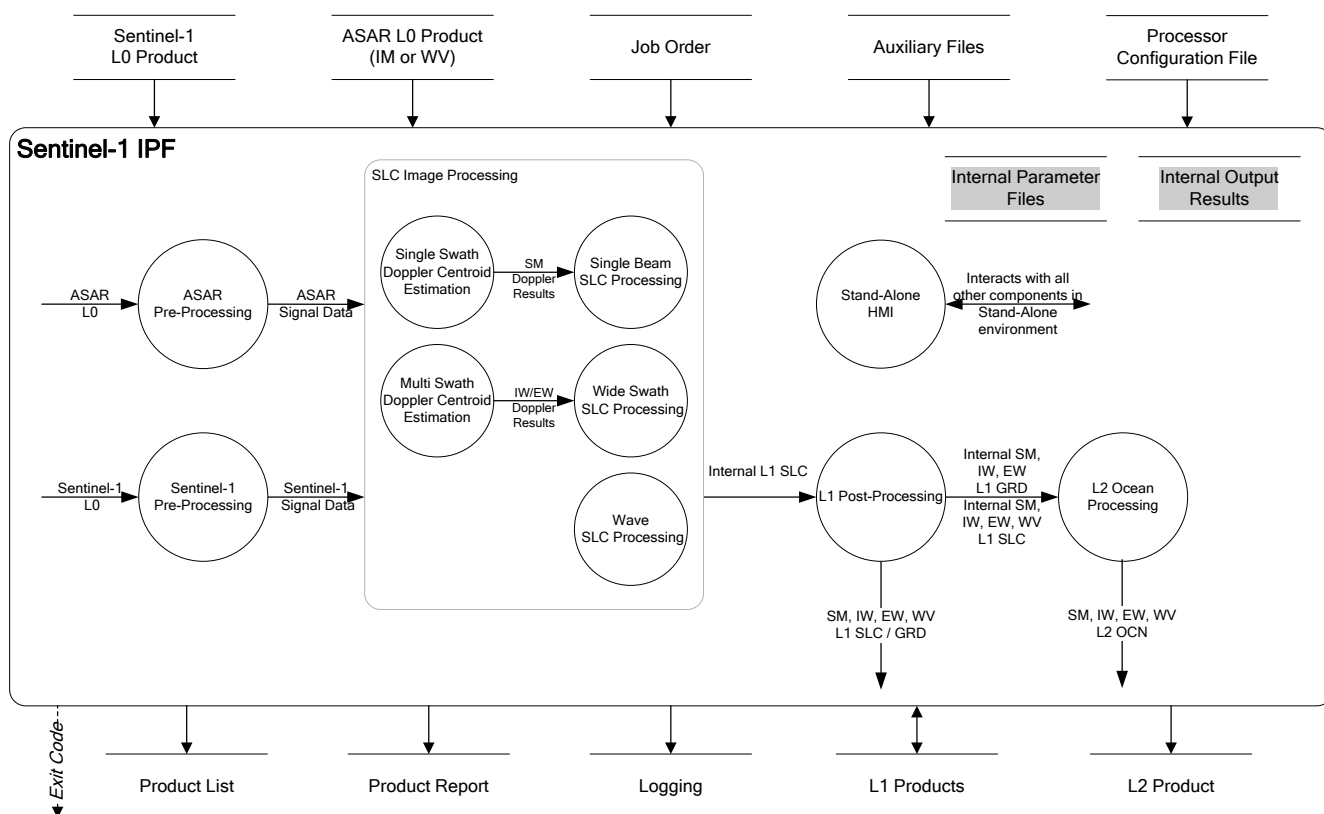


Figure 3-3 Sentinel-1 IPF Architecture

The IPF supports the processing of dual-polarisation data (Stripmap or TOPSAR) but since the processing of the data from the two polarisations is identical (with possible minor exceptions), typically, the dual-polarisation case will not be explicitly mentioned. However, algorithmic aspects specific to dual-polarisation processing are presented in Section 3.2.

The algorithms implemented in the Sentinel-1 IPF are described in this document as follows:

- **The Pre-Processing algorithms** are described in Section 4.
- **The Doppler Centroid Estimation algorithms** are described in Section 5.
- **The SLC Processing algorithms** are described in Section 6.
- **The L1 Post-Processing algorithms** are described in Section 7.
- **The slicing supporting algorithms** are described in Section 8.
- **The common and supporting algorithms** are described in Section 9.

Note that the Doppler centroid estimation algorithm for Wave mode is encapsulated within the L1 Wave SLC Processing component, and not as a separate process as in the case of the other modes.

Algorithms in each of these groups may rely on common or supporting algorithms, which are described in Section 9.

In addition to Sentinel-1 processing, the IPF also supports the processing of ENVISAT ASAR Image Mode (IM) and Wave Mode (WV) data. The Sentinel-1 algorithms described in this document are also applicable to ASAR processing, where processing ASAR IM is equivalent to Sentinel-1 SM, and processing ASAR WV is equivalent to Sentinel-1 WV. The key differences between Sentinel-1 and ASAR processing algorithms are related to the decoding and calibration of input raw data, due to the differences between the Sentinel-1 and ASAR Instrument Source Packet (ISP) data format.

3.1 TOPSAR Specific Processing

The Sentinel-1 IW and EW modes acquire data of wide swaths (composed of 3 and 5 sub-swaths respectively) using the TOPSAR imaging technique.

3.1.1 Mode Overview

The TOPSAR imaging is a form of ScanSAR imaging in the sense that the data is acquired in bursts, by cyclically switching the antenna beam among multiple adjacent sub-swaths. However, in TOPSAR, for each burst, the beam is electronically steered from backward to forward in the azimuth direction. In this way, the TOPSAR bursts are much longer than the ScanSAR bursts. Figure 3-4 schematically represents the acquisition of three bursts on as many sub-swaths with the TOPSAR mode, emphasizing the rotation of the antenna.

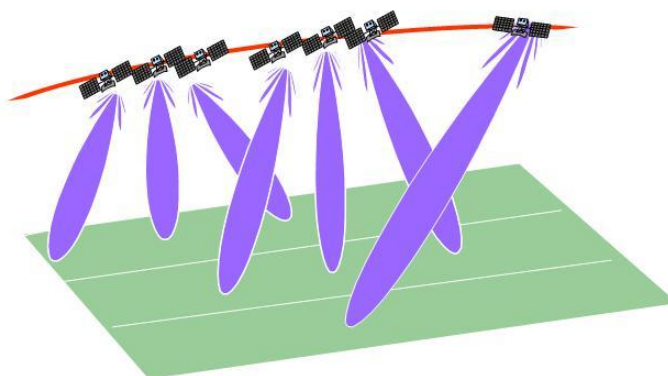


Figure 3-4 Schematic Representation of the TOPSAR Mode Acquisition

As a consequence, each target is illuminated for a shorter period of time but by the entire antenna footprint; therefore in TOPSAR, the resolution loss is achieved by shrinking the footprint rather than by slicing it as in the case of ScanSAR. This has several beneficial implications on the image quality, like for example the achievement of uniform Noise-Equivalent Sigma Zero (NESZ) and ambiguity levels within the bursts.

TOPSAR-specific concepts and algorithms are described in the document as follows:

- The antenna steering rate and the DC rate due to the steering are described in Section 9.14
- The required azimuth pre and post-processing of the data is described in Sections 6.2.4 and 6.4
- The de-ramping of the data prior to performing the following algorithms:
 - Fine DC estimation, as described in Section 5.2.1
 - GRD Azimuth processing as described in 7.2

The following subsection presents a mathematical overview of the TOPSAR principles.

3.1.2 Scientific Overview

A convenient and intuitive way to better understand the principles of the TOPSAR mode is by introducing the concepts of the antenna steering rate and the “shrinking factor”.

Consider, first, a burst acquisition with a fixed antenna like in conventional ScanSAR: the weight of the azimuth antenna pattern on the acquired SAR echoes, as a function of the azimuth time η , can be expressed as:

$$G_{aap,SCAN}(\eta) = \text{sinc}^2\left(\frac{L}{\lambda}\psi(\eta)\right) \quad (3-1)$$

where

L = Antenna length

λ = Wave length

V_r = Effective sensor velocity (see Section 9.10)

R_0 = Zero-Doppler sensor-target distance

$\psi(\eta)$ = Azimuth angle

The azimuth angle is defined by:

$$\psi(\eta) \approx \frac{V_r \eta}{R_0} \quad (3-2)$$

Considering now the acquisition of a TOPSAR burst, the $G_{aap}(\eta)$ function becomes:

$$G_{aap, TOPS}(\eta) = \text{sinc}^2\left(\frac{L}{\lambda}(\psi(\eta) + k_\psi \eta)\right), \quad -\frac{T_b}{2} \leq \eta \leq \frac{T_b}{2} \quad (3-3)$$

where

k_ψ = Antenna steering velocity

T_b = Burst length

Note that for the purposes of this section, the azimuth antenna pattern can be modeled as a *sinc* function, neglecting the grating lobes that arise from its steering.

Combining equations 3-2 and 3-3, the following expression is obtained:

$$G_{aap, TOPS}(\eta) = \text{sinc}^2\left(\frac{L}{\lambda} \frac{V_r \eta}{R_0} \left(1 + \frac{R_0 k_\psi}{V_r}\right)\right), \quad -\frac{T_b}{2} \leq \eta \leq \frac{T_b}{2} \quad (3-4)$$

Comparing equation (3-4) with equation (3-1) it can be noticed that the target is illuminated by the steered antenna for a ground footprint that is equivalent to that of a fixed antenna, but shrunk by the “shrinking factor”:

$$k = 1 + \frac{R_0 k_\psi}{V_r} \quad (3-5)$$

Therefore, from the azimuth resolution point of view, the TOPSAR mode can be thought as a strip-map with an equivalent antenna length k times larger than the real length, hence with a resolution k times lower.

Figure 3-5 presents a comparison between $G_{aap, TOPS}(\eta)$ obtained for TOPSAR- IW1 ($k \cong 4$) and $G_{aap, SCAN}(\eta)$ for a Scan-SAR acquisition with the same burst-length and obviously $k = 0$. It can be noticed the “shrinking” effect on the antenna pattern: the side-lobes of the azimuth pattern can be seen as effect of rotation in the TOPSAR case, while only the main-lobe is seen for Scan-SAR.

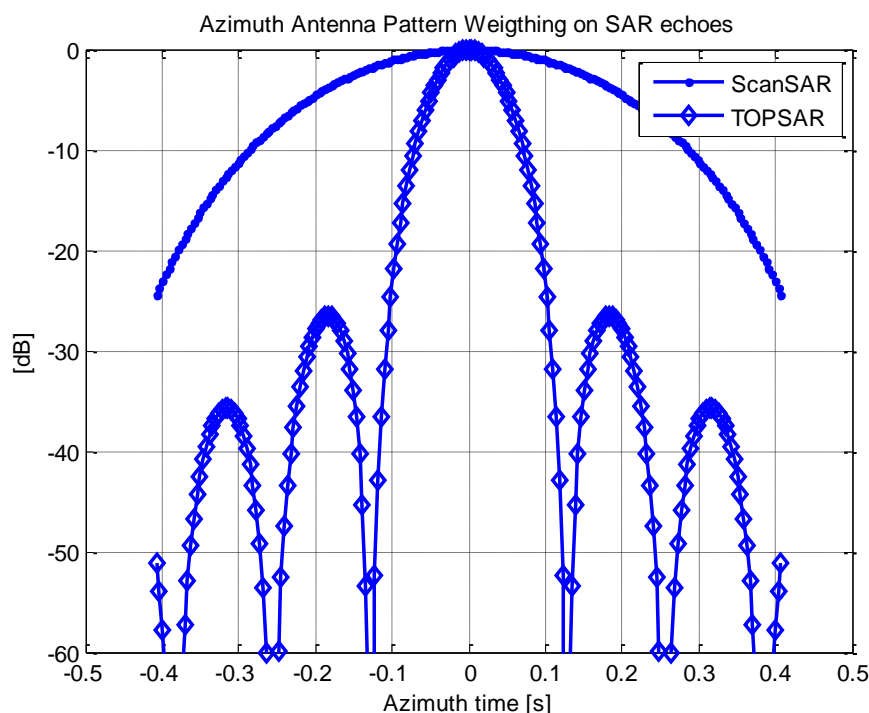


Figure 3-5 TOPSAR versus SCANSAR Azimuth Antenna Pattern Weighting

3.2 Dual-Polarisation Processing

The processing steps required for the dual-polarisation data processing are essentially the same as those required for the processing of single-polarisation data, and generally speaking each polarisation is processed in the same manner. However, there are a number of additional considerations when processing dual-polarisation data:

1. A separate image is generated for each polarisation, and polarisation-specific correction and calibration factors will be applied during the processing of each polarisation (for example, calibration pulses, antenna patterns, etc.).
2. The same estimated Doppler centroid must be used for processing both polarisations, so that the two resulting images are accurately co-registered. Therefore the DC estimated from only one of the two polarisation images will be used. In particular, the co-polarisation channel will be used, since the cross-polarisation channel has the potential to be noisier and therefore less reliable.
3. The internal calibration (replica reconstruction and PG calculation) of the cross-polarisation channel depends on values from the co-polarisation channel.

Note that the inter-channel corrections are implicitly applied by using the correct reference replica for the range compression and the correct drift compensation, as described in Sections 4.2.1.4 and 4.2.1.5.

4 PRE-PROCESSING ALGORITHMS

The Pre-processing module of the Sentinel-1 IPF has four main algorithmic components:

1. **Raw data analysis**, as described in Section 4.1
2. **Internal calibration**, as described in Section 4.2
3. **Downlink header validation**, as described in Section 4.3
4. **Terrain height function**, as described in Section 4.4.

Note that the algorithms described here are directed specifically at the Sentinel-1 Pre-Processing module, rather than the ASAR Pre-Processing module. The ASAR Pre-Processing module performs similar operations, but implemented specifically for handling ASAR data. For instance, ASAR raw data must also be decoded, but it is encoded using a different algorithm.

The ASAR calibration algorithms are described in [R-6], and the raw data decoding algorithms are described in [R-8].

4.1 Raw Data Analysis

Raw data analysis is required in order to perform corrections of the I and Q channels of the raw signal data. The classical raw data correction (applied for instance in the case of ENVISAT-ASAR and RADARSAT-2) involves (see also Section 9.2):

- I/Q bias removal
- I/Q gain imbalance correction
- I/Q non-orthogonality correction

For Sentinel-1 however, the instrument's receive module performs the demodulation in the digital domain, therefore the I/Q gain imbalance and I/Q non-orthogonality corrections are no longer necessary.

The raw data analysis necessary for the raw data correction of ASAR data is defined in [R-6]. Since the IPF also supports the processing of ASAR data, for completeness, the ASAR raw data analysis scheme is reproduced in this section.

Even though for Sentinel-1 the I/Q gain imbalance and the I/Q non-orthogonality corrections are not necessary, they will be made available optionally, using configuration input parameters. Irrespective to the correction flag though, the Raw Data Analysis described in this section will be performed and the results reported for both ASAR and Sentinel-1 data.

Note that the raw data analysis is performed on a sub-set of the BAQ-decoded Level 0 data. For this purpose, only the sub-set to be analysed is BAQ decoded at this stage. See Section 9.1 for a description of the BAQ decoding as well as the stages where the BAQ decoding of the entire Level 0 data set is performed.

The raw data analysis consists of the following:

1. Calculate the mean of the raw data.

The means of the raw data I and Q channels, μ_I and μ_Q respectively, are calculated as follows:

$$\mu_I = \frac{1}{NM} \sum_k \sum_j I_{kj} \quad (4-1)$$

$$\mu_Q = \frac{1}{NM} \sum_k \sum_j Q_{kj} \quad (4-2)$$

where N and M are the dimensions of the raw data sub-set to be analyzed, and $\mathbf{I} = \text{Re}\mathbf{E}$ (4-3)

$$\mathbf{Q} = \text{Im}\mathbf{E} \quad (4-4)$$

2. Calculate the standard deviations of the raw data.

The standard deviations of the raw data I and Q channels, σ_I and σ_Q respectively, are calculated as follows:

$$\sigma_I = \sqrt{\frac{1}{NM} \sum_k \sum_j (I_{kj} - \mu_I)^2} \quad (4-5)$$

$$\sigma_Q = \sqrt{\frac{1}{NM} \sum_i \sum_j (Q_{ij} - \mu_Q)^2} \quad (4-6)$$

3. Calculate the IQ gain imbalance, ρ , and its lower and upper bounds, ρ_1 and ρ_2 .

$$\rho = \frac{\sigma_I}{\sigma_Q} \quad (4-7)$$

$$\rho_1 = 1 - \frac{3}{\sqrt{NM}} \quad (4-8)$$

$$\rho_2 = 1 + \frac{3}{\sqrt{NM}} \quad (4-9)$$

4. Calculate the IQ quadrature departure, θ , and its lower and upper bounds, θ_1 and θ_2 .

The detailed description of this algorithm is presented in Section 4.1.1 below.

5. Set the statistics significance flags

For each of the μ_I and μ_Q , ρ and θ , assess if the value obtained is significant, i.e. if the value is outside computed lower/upper bounds. If this is the case, a predefined value (read from the input auxiliary file) will be used instead. More precisely:

- I-bias Significance flag is set to:
FALSE, if $\frac{-3\sigma_I}{\sqrt{NM}} \leq \mu_I \leq \frac{3\sigma_I}{\sqrt{NM}}$, TRUE otherwise.
- Q-Bias Significance flag is set to:
FALSE, if $\frac{-3\sigma_Q}{\sqrt{NM}} \leq \mu_Q \leq \frac{3\sigma_Q}{\sqrt{NM}}$, TRUE otherwise.
- IQ Gain Significance flag is set to:
FALSE, if $\rho_1 \leq \rho \leq \rho_2$, TRUE otherwise.
- IQ Quadrature Departure Significant flag is set to:
FALSE, if $3\sigma^- \leq \sin(\theta) \leq 3\sigma^+$, TRUE otherwise.

4.1.1 IQ Quadrature departure

1. For each range line k , calculate the correlation coefficient between I and Q channels:

$$c_k = \frac{S_{IQ}}{\sqrt{S_{II} \cdot S_{QQ}}} \quad (4-10)$$

where:

$$S_{IQ} = \sum_{j=1}^M (I_j \cdot Q_j) - \left(\frac{1}{M} S_I \cdot S_Q \right) \quad (4-11)$$

$$S_{II} = \sum_{j=1}^M (I_j^2) - \left(\frac{1}{M} S_I^2 \right) \quad (4-12)$$

$$S_{QQ} = \sum_{j=1}^M (Q_j^2) - \left(\frac{1}{M} S_Q^2 \right) \quad (4-13)$$

$$S_I = \sum_{j=1}^M I_j, \quad S_Q = \sum_{j=1}^M Q_j \quad (4-14)$$

2. For each range line k , calculate Z_k as follows:

$$Z_k = 0.5 \ln \left(\frac{1 + C_k}{1 - C_k} \right) \quad (4-15)$$

3. Calculate the mean and the standard deviation of the vector $\mathbf{Z} = (Z_k)_{k=1,2,\dots,N}$:

$$\mu_z = \frac{1}{N} \sum_{k=1}^N Z_k \quad (4-16)$$

$$\sigma_z = \sqrt{\frac{1}{N} \sum_{k=1}^N (Z_k - \mu_z)^2} \quad (4-17)$$

4. Calculate the IQ quadrature departure θ and its lower and upper bounds θ_1 and θ_2 :

$$\theta = \arcsin(C) \quad (4-18)$$

$$\theta_1 = \arcsin(C - \sigma^-) \quad (4-19)$$

$$\theta_2 = \arcsin(C + \sigma^+) \quad (4-20)$$

where:

$$C = \tanh(\mu_z) \quad (4-21)$$

$$\sigma^+ = \tanh(\mu_z + \sigma_z) - C \quad (4-22)$$

$$\sigma^- = C - \tanh(\mu_z - \sigma_z) \quad (4-23)$$

4.2 Internal Calibration

The internal calibration of the Sentinel-1 images is based on information extracted from the calibration measurements and the noise measurements associated with every data take, as part of the initial, interleaved and final internal calibration sequence performed by the instrument.

There are two types of calibration sequences:

1. Image calibration sequences composed of calibration pulses at the nominal imaging bandwidth, using the unique chirp parameters for each sub-swath in the instrument mode; and,
2. PG calibration sequences composed of calibration pulses at full bandwidth, using the chirp parameters for the first sub-swath in the instrument mode.

Image calibration sequences are present in the pre-amble and the post-amble of each datatake. The image calibration data may be used to create reference replica that is used to generate the range reference function as described in section 6.1.1.

PG calibration sequences are present in the pre-amble and post-amble and are interleaved within the image sequences of each datatake. PG calibration data are used to perform the instrument drift as described in section 9.4.

Generally the information in this section applies to both the image calibration data and the PG calibration data. An explicit distinction is made in all cases where there are differences between the two types of calibration data.

This section presents:

1. **The algorithm overview**, as described in Section 4.2.1
2. **The algorithm implementation**, as described in Section 4.2.2

4.2.1 Algorithm Overview

This section presents the following aspects of the internal calibration:

1. **Nominal replica generation**, as described in Section 4.2.1.1
2. **Calibration pulse extraction**, as described in Section 4.2.1.2
3. **Replica reconstruction**, as described in Section 4.2.1.3
4. **Replica extraction and time delay estimation**, as described in Section 4.2.1.4
5. **Instrument drift derivation**, as described in Section 4.2.1.5
6. **Replica coefficients generation**, as described in Section 4.2.1.7
7. **PG product validation**, as described in Section 4.2.1.8
8. **Noise measurements processing**, as described in Section 4.2.1.9

4.2.1.1 Nominal Replica Generation

This section describes the generation of the nominal replica vectors for the image and PG calibration operations. The following symbols are used in this section:

- *TXPL* Transmit pulse length [s]
- *TXPSF* Transmit pulse start frequency [Hz/s]
- *TXPRR* Transmit pulse ramp rate [Hz/s²]

4.2.1.1.1 Nominal Image Replica Generation

The nominal image replica is generated from the nominal image phase coefficients array which is a configurable input parameter. The nominal image replica at sample time t_n is generated as:

$$NomChirp_{image}(t_n) = \frac{1}{n_{samples}} \cdot \exp \left[2\pi j \cdot \left(\varphi(1) \cdot t_n + \varphi(2) \cdot t_n^2 \right) \right],$$

$$n = \frac{-TXPL}{2}, \dots, \frac{TXPL}{2} - 1 \quad (4-24)$$

where φ is an array of four configurable chirp phase coefficients nominally defined as:

- $\varphi(0) = 0$
- $\varphi(1) = TXPSF - TXPRR \cdot \frac{-TXPL}{2}$
- $\varphi(2) = \frac{TXPRR}{2}$
- $\varphi(3) = 0$

4.2.1.1.2 Nominal PG Replica Generation

The nominal PG replica is generated from the chirp pulse parameters extracted from the downlink headers. The nominal PG replica at sample time t_n is generated as:

$$NomChirp_{PG}(t_n) = \frac{1}{n_{samples}} \cdot \exp \left[2\pi j \cdot \left(TXPSF \cdot t_n + \frac{TXPRR}{2} \cdot t_n^2 \right) \right],$$

$$n = 0, \dots, TXPL - 1 \quad (4-25)$$

4.2.1.2 Calibration Pulse Extraction

The Sentinel-1 IPF supports six types of internal calibration measurements (or signals):

1. Transmit Calibration Signals (*TXcal*)
2. Transmit H Isolation Calibration Signals (*TXHcaliso*)
3. Receive Calibration Signals (*RXcal*)
4. EPDN Calibration Signals (*EPDNcal*)
5. Tile Amplifier Calibration Signals (*TAcad*)
6. APDN Calibration Signals (*APDNcal*)

The path covered by a calibration pulse depends both on its type and on the signal polarisation.

In principle, the replicas could be obtained directly from these 5 calibration measurements. Nevertheless, in order to cancel the effects of the leakage signals, the Pulse Coded Calibration (PCC) approach has been introduced. This approach raises to 9 the number of the required pulse measurements for a single replica reconstruction: two PCC pulses for each *TXcal*, *RXcal*, *TAcad* and *EPDNcal* plus one calibration pulse, which is not coded with the PCC technique, for *APDN*. Depending on the different acquisition modes, these 9 pulses will be placed (see the calibration time-line in Section 4.10 of [A-3]):

- at the beginning and at the end of the data take for SM, and of each vignette, for WV
- periodically within the data take (SM)
- at the end of each burst (IW/EW)

The calibration signals for both image and PG calibration are contained in dedicated packets. The image calibration signals are sampled at the sampling frequency of the measurement data (which is the sampling frequency after the decimation filtering) while the PG calibration signals are sampled at 100 MHz. The calibration signals are not BAQ or FDBAQ encoded, so that they can be used directly (see Section 9.1.2). The image calibration signals are sub-swath and polarisation dependent while the PG calibration signals for each sub-swath are all acquired with the receive parameters of the first sub-swath within the mode. They are therefore only polarisation dependent and for the PG calibration signals the difference in the receive gain power between the sub-swaths is compensated for by applying a gain offset for each sub-swath that is calculated from the image calibration data as described in Section 4.2.1.5.

In the following, a generic calibration packet received at time η_n will be indicated as a vector of M complex samples (note that n represents the rank 1 or 2 as required for equation 4-27):

$$XXcal(m, n) = a(m, n) \cdot e^{j\phi(m, n)}, \quad m = 0, \dots, M - 1 \quad (4-26)$$

The calibration signal acquisition timing is described in Section 5.3 of [A-6] and the timing diagram is repeated in Figure 4-1 below. The M complex samples of the calibration signal are equivalent to $Ncal$ samples in Figure 4-1.

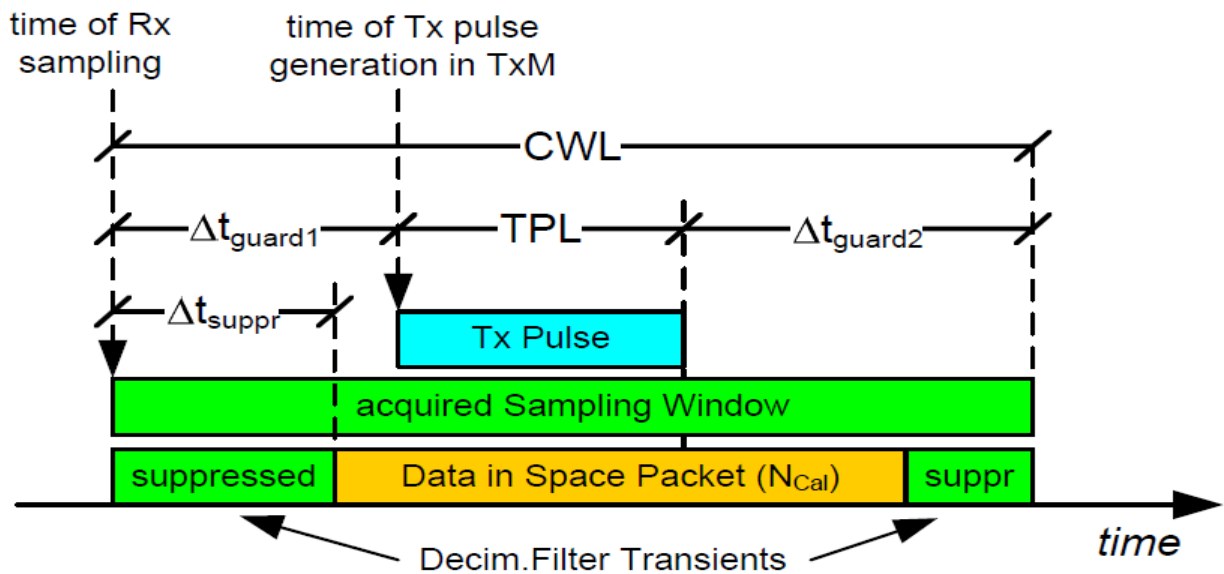


Figure 4-1 Timing of Calibration Signal Sampling Window

The PCC-2 coding technique used for four of the pulse types is a particular case of the more general PCC-N technique. The PCC-N technique is implemented by sending a series of N coherent calibration pulses in parallel through the desired signal path. The individual successive signals are multiplied by factors “+1” or “-1” (the latter adding a phase shift of 180 degrees). Each path is identified by a unique sequence.

The coding operation can be written as:

$$Y = H_N \cdot S \quad (4-27)$$

where Y and S are respectively the vector of measured and of input signals, while H_N is a Hadamard matrix of dimension $N = 2^k$. Thanks to the properties of H_N this equation can be easily inverted to obtain the decoded vector S :

$$S = \frac{1}{N} \cdot H_N^T \cdot Y \quad (4-28)$$

In the PCC-2 case this inversion is not needed in its general form; this can be avoided by simply taking into account that the two pulses are shifted by 180 degrees. The selection of the pulses included in the calibration signal reconstruction is controlled by configurable input parameters. In the equations below the variables x and y are used to represent the calibration pulses selected for the signal reconstruction.

As a consequence, from the 8 PCC-coded calibration measurements, one can derive the following 4 calibration signals that will be used for the replica reconstruction:

$$\begin{aligned} \mathbf{s}_{TXVcal}(m) &= \frac{TXVcal(m, x) - TXVcal(m, y)}{2} \\ \mathbf{s}_{RXcal}(m) &= \frac{RXcal(m, x) - RXcal(m, y)}{2} \\ \mathbf{s}_{EPDNcal}(m) &= \frac{EPDNcal(m, x) - EPDNcal(m, y)}{2} \\ \mathbf{s}_{TAc al}(m) &= \frac{TAc al(m, x) - TAc al(m, y)}{2}, \quad m = 0, \dots, M - 1 \end{aligned} \quad (4-29)$$

As already mentioned, the *APDNcal* calibration measurement is not PCC-coded because it contains only passive elements and cannot be phase shifted. The *APDNcal* has a low signal level and so a number of N pulses are used to improve the signal to noise ratio (where N is a configurable input parameter):

$$\mathbf{s}_{APDNcal}(m) = \frac{1}{N} \cdot \sum_{n=1}^N APDNcal(m, n), \quad m = 0, \dots, M - 1 \quad (4-30)$$

The calibration sequence for the transmit H polarisation signal, *TXHcal*, is not PCC2 coded. Instead a unique calibration pulse, *TXHcaliso* (transmit H calibration isolation) pulse is subtracted from the *TXHcal* pulse to form the signal:

$$\mathbf{s}_{TXHcal}(m) = TXHcal(m, x) - TXHcaliso(m, y), \quad m = 0, \dots, M - 1 \quad (4-31)$$

4.2.1.3 Replica Reconstruction

The replica is the instrument response function describing the chirp and the impact of the instrument chain on the actually transmitted radar signal. It is derived by combining in an appropriate way the calibration pulses $\mathbf{s}_{TXcal}(m)$, $\mathbf{s}_{RXcal}(m)$, $\mathbf{s}_{EPDNcal}(m)$, $\mathbf{s}_{TAc al}(m)$ and $\mathbf{s}_{APDNcal}(m)$ derived as described in Section 4.2.1.1.

For compactness, the formulae in this and the subsequent sections are provided only for the HH/HV dual polarisation case, the VV/VH case being analogous.

A replica is reconstructed for each calibration packet, m as described in [R-13], namely:

$$rep_{HH,m}(t) = F^{-1} \left\{ \frac{F\{s_{TXHcal,m}(t)\} \cdot F\{s_{RXHcal(H),m}(t)\} \cdot F\{s_{TAHcal,m}(t)\}}{F\{s_{EPDNHcal(H),m}(t)\} \cdot F\{s_{APDNcal(H),m}(t)\}} \right\} \quad (4-32)$$

$$rep_{HV,m}(t) = F^{-1} \left\{ \frac{F\{s_{TXHcal,m}(t)\} \cdot F\{s_{RXHcal(V),m}(t)\} \cdot F\{s_{TAHcal,m}(t)\}}{F\{s_{EPDNHcal(H),m}(t)\} \cdot F\{s_{APDNcal(H),m}(t)\}} \right\} \quad (4-33)$$

where the letter H or V within the name of the pulse denotes the polarisation of the transmit path and the letter in parenthesis represents the channel where the pulse is received.

Note also that the calculation of $rep_{HV,m}$, the replica for the cross-polarisation channel is dependent on the pulse from the co-polarisation channel, except $s_{RXHcal(V),m}(t)$. If for some reason the co-polarisation pulses are not available, the cross-polarisation replica cannot be calculated, and the nominal replica could be used instead. This also leads to a co-polarisation dependency for the PG calculated for the cross-polarisation channel, as the PG is derived from the reconstructed replica.

4.2.1.4 Replica Extraction and Internal Time Delay Estimation

The internal time delay is the deviation of the replica location from the location of “Tx Pulse” in Figure 4-1 above. It can be computed **from the PG calibration data** as follows:

$$\Delta t_{int} = \Delta t_{peak}^{PG} - (\Delta t_{guard} - \Delta t_{suppr}) \quad (4-34)$$

where:

- Δt_{peak}^{PG} is the correlation peak location computed in step 2 below
- Δt_{guard} is a configurable input parameter for the time of the transmit pulse within the calibration data packet described in Figure 4-1
- Δt_{suppr} is a configurable input parameter for the time of the suppressed decimation filter transients within the calibration data packet described in Figure 4-1

The **average internal time delay over all PG calibration sequences** will be used to time shift the reconstructed image replica (described below) and the range matched filter as described in Section 6.1.1.

As shown in Figure 4-1 above the chirp replica is in the middle of the acquired sampling window. The location of the replica first sample shall be determined by correlating the output from Section 4.2.1.3 above with the nominal chirp. The complete procedure to extract the replica is as follows:

1. **Cross correlate the reconstructed replica** (from Section 4.2.1.3) **with the nominal chirp** (from Section 4.2.1.1)
2. **Search for the correlation peak** using FFT zoom and parabolic interpolation to refine the peak location. The location of the replica first sample is at the correlation peak. **For PG calibration sequences**, convert this peak location to time and denote it as Δt_{peak}^{PG} . If this cross-correlation step fails to find a valid peak location, the current calibration sequence shall be discarded.
3. If this is an **image replica**:
 1. **Validate the replica** by comparing the 3dB width, PSLR and ISLR against the predefined thresholds, which are configurable input parameters. If any of the checks fails, set *replicasValid* = *FALSE* for the current replica and search for the next calibration pulse. Otherwise continue with the subsequent steps.
 2. **Time shift the reconstructed image replica** such that the correlation peak location is on a sample boundary. This time shift shall be the average Δt_{int} over all **PG calibration sequences** and shall be done by using phase ramp in the frequency domain.
 3. **Extract TXPL complex samples** starting from the first sample location determined by the correlation peak.

4.2.1.5 Rx Gain Offset

The PG calibration pulses are generated using the chirp parameters and gain settings from the first sub-swath of the mode. The gain offset between the first sub-swath and each subsequent sub-swath must be considered when applying the instrument drift correction to the data. To this end, the mean receive gain power for each sub-swath n is **calculated from the image calibration data** as:

$$meanRxCalPow(n) = mean(abs(s_{Rxc}^{cal}(n))^2) \quad (4-35)$$

4.2.1.6 Instrument Drift Derivation

The IPF shall perform instrument phase and gain drift correction. There are two options for implementing this correction:

- Drift correction from the PG product
- Drift correction from a PG product model

4.2.1.6.1 Drift Compensation from PG Product

One of the goals of the internal calibration is to measure the product of the transmit power and the receive gain, also known as the PG product. A detailed description of the PG Product calculation can be found in [R-13].

The PG product can be derived from the replicas, which in turn can be derived from the calibration packets, as described in Section 4.2.1.3, therefore a given number of PG products will be available within a single data take.

The available PG products will then be interpolated in order to obtain a complex correction factor for each received signal echo. The compensation of the PG gain is then performed on an echo by echo basis, after the raw data correction and prior to any other processing step, as described in Section 9.4.

The first step of the algorithm is to compress the replicas reconstructed as described in Section 4.2.1.3, using the first extracted valid replica.

$$rep_cp_{HH,m}(t) = F^{-1} \{ F \{ rep_{HH,m}(t) \} F^* \{ rep_{HH,1}(t) \} \} \quad (4-36)$$

$$rep_cp_{HV,m}(t) = F^{-1} \{ F \{ rep_{HV,m}(t) \} F^* \{ rep_{HV,1}(t) \} \} \quad (4-37)$$

The PG correction factor for sub-swath n corresponding to the calibration packet m can then be calculated by dividing the reference PG with the compressed replica peak and applying the receive gain power ratio for the current sub-swath:

$$PG_{HH,m} = \frac{PG_{ref}}{\max(rep_cp_{HH,m}(t))} \cdot \frac{meanRxCalPow_{HH}(1)}{meanRxCalPow_{HH}(n)} \quad (4-38)$$

where:

- PG_{ref} is a reference absolute PG that will be defined by offline analysis of the acquired data. Note: PG_{ref} for the co-polarization will be different from PG_{ref} for the cross polarization. Both PG_{ref} will be stored in the auxiliary file.
- $\max(rep_cp_{HH,m}(t))$ is the maximum (peak) of the compressed replica

Note that in order to accurately determine the maximum of the compressed replica, $\max(rep_cp_{HH,m}(t))$, the result is first resampled at a finer sampling spacing (via an FFT). This operation is the same as the one performed for the replica extraction as described in Steps 2 of Section 4.2.1.4.

The drift coefficients of the other polarisations are calculated in a similar way.

The timestamp of each drift coefficient is defined to be in the centre of the corresponding calibration sequence m . This set of drift coefficients represents the set of time-stamped PG values that will later be linearly interpolated and applied to each echo source packet as described in Section 9.4.

4.2.1.6.2 Drift Compensation from PG Product Model

The phase/gain drift of the instrument shall be constantly monitored and long term stability analysis shall be available during the lifetime of Sentinel-1. If the drift presents a slow and predictable variation in time, as expected, a proper drift model could be defined and provided as a PG product model, a configurable input LUT. The usage of the PG product model is controlled by a configurable input parameter, *getDriftFromPGmodel*.

4.2.1.7 Image Replica Coefficients Generation

The extracted image replica is usually noisy and noise removal should be done before using the extracted image replica for range compression. Noise removal can be done by fitting polynomial coefficients to the amplitude and phase of the extracted replica. In this section we propose an algorithm to compute the polynomial coefficients of the extracted replica.

The main steps involved in the proposed algorithm are represented in Figure 4-2.

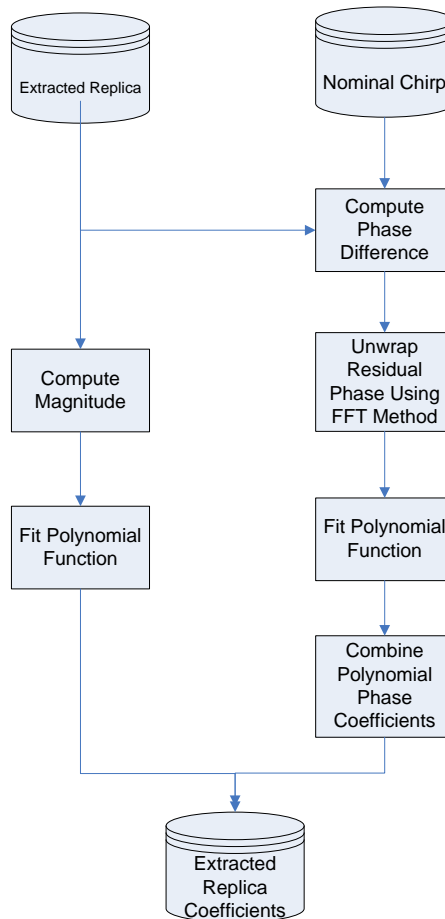


Figure 4-2 Flow Chart for Extracted Replica Coefficients Derivation

Note that the algorithm is performed independently on each polarisation channel, therefore in the following sections, the polarisation notation will be omitted from the pulses notations.

4.2.1.7.1 Amplitude Coefficients

The vector of replica amplitudes is defined as:

$$a_{Ch}(n) = |Ch_{IntCal}(n)|, \quad n = 0, \dots, N-1 \quad (4-39)$$

The following operations are performed as shown in the left part of the flow chart in

Figure 4-2 and involve the following steps:

1. **Compute the replica amplitude.**

2. **The amplitudes vector is fitted with a polynomial function** $a_{Ch}^{FIT}(\tau)$ **sampled on the same grid.** The order of the polynomial is a configurable input parameter.

In the Range Reference Function (RRF) generation, the amplitude is inverted so the range compressed output will have a flat spectrum. The inverted amplitude function can be expressed as:

$$b_{Ch}(n) = \left(\frac{a_{Ch}^{FIT}(n)}{\bar{a}_{Ch}^{FIT}(n)} \right)^{-1} = \left(\frac{a_{Ch}^{FIT}(n)}{\frac{1}{N} \sum_{n=0}^{N-1} a_{Ch}^{FIT}(n)} \right)^{-1} = \frac{\frac{1}{N} \sum_{n=0}^{N-1} a_{Ch}^{FIT}(n)}{a_{Ch}^{FIT}(n)} \quad (4-40)$$

4.2.1.7.2 Phase Coefficients

The steps to compute the replica phase coefficients are shown in the right part of the flow chart in

Figure 4-2:

1. **Compute the phase difference between the nominal chirp and the extracted replica** by multiplying the replica with the conjugate of the nominal chirp:

$$Ch(n) = ref_{IntCal}(n) \cdot ref_{nom}^*(n), \quad n = 0, \dots, N-1 \quad (4-41)$$

Note: the extracted replica has been time aligned with the nominal chirp as described in Section 4.2.1.4.

2. Unwrap the phase difference using FFT method as described in Section 5.3
3. **Fit polynomial to the unwrapped phase.** The order of the polynomial is a configurable input parameter.
4. Combine the unwrapped phase polynomial coefficients with the nominal coefficients to obtain the extracted replica phase coefficients.

4.2.1.8 PG Product Validation

Two types of validation are performed on the PG product (the validation is performed independently on each channel; therefore the polarisation indexing will be omitted):

- **Relative validation.** This validation consists in determining if a PG product value is within a given threshold from the mean of all the PG product values. To this end, the mean and standard deviation of the PG products' amplitude and phase, $\mu_{P,amp}$, $\mu_{P,phase}$, $\sigma_{P,amp}$ and $\sigma_{P,phase}$, are computed.

The PG product value P_m is qualified as invalid if:

$$|P_{m,amp} - \mu_{P,amp}| > pgAmpStdFraction \cdot \sigma_{P,amp} \quad (4-42)$$

OR

$$|P_{m,phase} - \mu_{P,phase}| > pgPhaseStdFraction \cdot \sigma_{P,phase} \quad (4-43)$$

where *pgAmpStdFraction* and *pgPhaseStdFraction* are configurable input parameters.

- **Absolute validation.** This validation consists in determining if a PG product value within a given threshold from the corresponding value of the PG product model. Then, the PG product value P_m is qualified as invalid if:

$$|P_{m,amp} - P_{mod,m,amp}| \geq maxPgAmpError \quad (4-44)$$

OR

$$|P_{m,phase} - P_{mod,m,phase}| \geq maxPgPhaseError \quad (4-45)$$

where $P_{mod,m}$ is the PG product model value and *maxPgAmpError*, *maxPgPhaseError* are configurable input parameters.

If the percentage of the invalid PG products does not exceed a certain threshold, given by the configurable input parameter *invalidPgMaxFraction*, the invalid PG values will be discarded (and only the valid values will be further used in the linear interpolation and application to the data). Otherwise, all the calculated PG product values will be discarded and replaced with the corresponding PG product model values.

Note that the receive gain power ratio for the current sub-swath is not considered during PG validation.

Note that the relative validation requires that all IPF instances perform the PG product calculation for all the packets.

4.2.1.9 Noise Measurements Processing

Noise measurements are acquired during each data take as part of the initial and final internal calibration packet. These measurements are recorded by switching off the TX signal for a sufficient number of TX pulses before the noise measurement. The noise measurements are recorded separately for each sub-swath.

The noise level is computed by calculating the mean power of the available noise packets for each sub-swath. The noise level will be used to determine the noise profile of the output image, as describe in Section 9.17.

The generic noise packet will be indicated as an array of complex samples:

$$Noise(m, n) = a(m, n) \cdot e^{j\phi(m, n)}, \quad m = 1, \dots, M$$

where:

M = number of noise packets
N = number of samples

The noise data are processed as follows:

1. The I/Q Bias Correction is applied as described in Section 9.2.1.
2. The Spurious Signal Correction is applied as described in Section 9.2.2.
3. The Receiver Gain Compensation is applied as described in Section 9.2.3.
4. The mean power is computed as:

$$\sigma_n^2 = \frac{\sum_{m=1}^M |Noise(m, n)|^2}{MN} = \frac{\sum_{m=1}^M a(m, n)^2}{MN} \quad (4-46)$$

For noise data, different encoding options (see Section 9.1.2) may be available.

4.2.2 Algorithm Implementation

This section presents the algorithmic steps of the internal calibration. Note that for brevity, only the HH channel case will be specified, if the other channels are to be treated in the same way:

1. **Process the PG calibration data:**
 - **Generate the nominal PG chirp from the downlink chirp parameters** as described in Section 4.2.1.1.2. This nominal chirp will be used for replica extraction, time delay estimation and PG product calculation.
 - **For each PG calibration sequence m**, perform the following:
 - **Extract the calibration pulses required for replica reconstruction**, as described in Section 4.2.1.2.
 - **Reconstruct the PG replica**, as described in Section 4.2.1.3.
 - **Extract the replica and estimate the internal time delay** as described in Section 4.2.1.4 and report result.
 - **Calculate the PG product** as described in Section. 4.2.1.6.1.
 - **Average the estimated internal time delays** and pass it to the SLC processor to time shift the chirp during RRF generation.
 - Calculate the mean and the standard deviation of the PG product values as described in Section 4.2.1.8.

- **For each PG calibration sequence m , validate the PG product and report the result**, as described in Section 4.2.1.8. This means, set $isPgProductValid(m) = FALSE$ if:
 - Relative validation fails OR
 - Absolute validation fails
 - **Calculate the percentage of the invalid PG product values** (out of the total number of calibration sequences).
 - **Determine the PG product source to be used for drift compensation:**
 - If the percentage of invalid PG product values exceeds $invalidPgMaxFraction$ OR $getDriftFromPGmodel = TRUE$
 - Use the PG product model $P_{mod,HH,m}$, $m = 0,1,...,M-1$ for the drift compensation.
 - Else
 - Set the drift compensation from the PG product dynamically derived from the replicas (at step 5)
 $P_{HH,m} = d_{HH,m}$, $m = 0,1,...,M-1$. Note: only the values for which $isPgProductValid(m) = TRUE$ will be linearly interpolated and used for drift compensation.
 - End
2. **Process the image calibration data:**
- **Generate the nominal image chirp from the provided coefficients** as described in Section 4.2.1.1.1. This nominal chirp will be used for replica extraction and replica coefficients generation.
 - **For each calibration packet m , perform the following:**
 - Extract the calibration pulses required for replica reconstruction, as described in Section 4.2.1.1.
 - Reconstruct the image replica, as described in Section 4.2.1.3.
 - Extract the replica and shift the replica as described in Section 4.2.1.4.
 - Calculate the receive gain power offset as described in Section 4.2.1.5
 - Set the replica coefficients:
 - If the RRF will be generated using the extracted replica then estimate the replica coefficients from the first extracted replica as described in Section 4.2.1.7 and pass the replica coefficients to the SLC processor; otherwise,
 - If the RRF will be generated using the nominal chirp then pass the nominal chirp coefficients to the SLC processor.
 - Extract the complex gain:

- If the RRF will be generated using the nominal chirp then extract the inter-channel complex gain for the nominal chirp from the configuration file; otherwise,
 - If the RRF will be generated using the extracted chirp then extract the inter-channel complex gain for the extracted replica from the configuration file, and pass it to the SLC processor.
3. **Perform the noise level estimation** as described in Section 4.2.1.9. The result will later be used to derive the noise vectors, as described in Section 9.17.

Note that in the slicing case, each IPF instance will perform the same calculation in order to determine the unique reference replica and the PG product values for the entire segment.

4.3 Downlink Header Validation

The Sentinel-1 pre-processing component of the IPF will validate fields within the downlink. The purposes of this validation are:

- To establish the values of fields in the downlink that are needed by later stages of processing;
- To validate that individual field values are correct and have not been effected by bit errors;
- To detect missing lines and gaps; and
- To check that the transmission pattern of the downlink follows the pattern expected by the Sentinel-1 IPF, for example the order of sub-swaths, the length of bursts, and the sequence of echo versus calibration versus noise packets.

In general, two validation strategies will be employed:

1. Majority polling, where a field that is expected to have a constant value across a data take, sub-swath or pulse type is monitored across a group of packets and the value that is used the majority of the time is considered the correct value. This majority value is then used to check for errors in this field across the rest of the data take.

In some cases (e.g. the SWST field), this value will be constant for a period of time, but then switch to a new value at some point. In this case, majority polling is used to distinguish between a bit error and a genuine change in the value.

2. Timeline consistency checks, where a basic understanding of the expected packet sequence and the way different fields change relative to each other within the timeline is exploited. For instance, the counter fields and packet type fields can be cross checked against each other to determine whether they fit the expected timeline.

A preliminary description of the validation that will be performed per field in the Sentinel-1 IPF is listed in Table 4-1. These Sentinel-1 downlink fields are described in [A-6]. This table contains the following information:

- The name and hierarchy of each field;
- Whether the field value is constant around orbit (within a particular mode/swath);
- Whether the field value is constant within a data take;
- Whether the field value is constant within a swath (for a particular data take/mode);
- Whether the field value is constant for a particular packet type (either echo, noise or calibration);
- Whether additional information from the Radar Database is required in order to interpret the field (e.g. in order to use the Elevation Beam Address, the elevation beam patterns from the Radar Database must also be provided to the IPF);
- Whether a look-up table or conversion law is required in order to interpret the value of the field (e.g. the BAQ Mode code in the downlink has a meaning that must be translated using a LUT);
- The validation strategy to apply to each field.

Table 4-1 Sentinel-1 IPF Downlink Header Validation

Field	Constant around Orbit	Constant during Data Take	Constant within Swath	Constant for Packet Type	Translation LUT or Law Required	Validation Strategy
Packet Primary Header						
Packet Version Number	YES	YES	YES	YES	NO	Majority poll across data take
Packet Identification => Packet Type	YES	YES	YES	YES	NO	Majority poll across data take
Packet Identification => Secondary Header Flag	YES	YES	YES	YES	NO	Majority poll across data take
Packet Identification => Application Process Identifier	YES	YES	YES	YES	NO	Majority poll across data take
Packet Sequence Control => Sequence Flags	YES	YES	YES	YES	NO	Majority poll across data take
Packet Sequence Control => Packet Sequence Count	NO	NO	NO	NO	NO	Cross-check against expected value from the reconstructed instrument timeline, the Space Packet Count and the Mode PRI Count to detect and correct bit errors. See also Section 4.3.1 for a description of the missing line detection strategy.
Packet Data Length	NO	NO	NO	NO	NO	Check against valid range (61 to 65533). If field is invalid, pre-processing will fail, as offset to next packet cannot be determined.
Packet Secondary Header						
Datation Service => Coarse Time	NO	NO	NO	NO	NO	Check sequence across multiple packets
Datation Service => Fine Time	NO	NO	NO	NO	NO	Check sequence across multiple packets
Fixed Ancillary Data Field => Sync Marker	YES	YES	YES	YES	NO	Majority poll across data take
Fixed Ancillary Data Field => Data Take ID	YES	YES	YES	YES	NO	Majority poll across data take
Fixed Ancillary Data Field => ECC Number	YES	YES	YES	YES	YES	Majority poll across data take
Fixed Ancillary Data Field => Spare	NA	NA	NA	NA	NO	Ignore
Fixed Ancillary Data Field => Test Mode	YES	YES	YES	YES	YES	Majority poll across data take
Fixed Ancillary Data Field => Rx Channel ID	YES	YES	YES	YES	YES	Majority poll across data take
Fixed Ancillary Data Field => Instrument Configuration ID	YES	YES	YES	YES	NO	Majority poll across data take

Sentinel-1

Ref:
MPC Nom: DI-MPC-IPFDPM
MPC Ref: MPC-0307
Issue/Revision: 2/1
Date: 31/01/2017

Field	Constant around Orbit	Constant during Data Take	Constant within Swath	Constant for Packet Type	Translation LUT or Law Required	Validation Strategy
Sub-commutated Ancillary Data Service => Sub-commutated Ancillary Data Word Index	NO	NO	NO	NO	NO	Cross-check against Space Packet Count
Sub-commutated Ancillary Data Service => Sub-commutated Ancillary Data Word	NO	NO	NO	NO	YES*	No validation * A LUT/Conversion law is required for translation of the Data Time Stamps and Pointing Status Fields. Other fields do not require a LUT/Conversion law.
Counters Service => Space Packet Count	NO	NO	NO	NO	NO	Cross-check against expected value from the reconstructed instrument timeline, the Packet Sequence Count and the Mode PRI Count to detect and correct bit errors. See also Section 4.3.1 for a description of the missing line detection strategy.
Counters Service => PRI Count	NO	NO	NO	NO	NO	Cross-check against expected value from the reconstructed instrument timeline, the Packet Sequence Count and the Space Packet Count to detect and correct bit errors. See also Section 4.3.1 for a description of the missing line detection strategy.
Radar Configuration Support Service => Spare	NA	NA	NA	NA	NA	Ignore
Radar Configuration Support Service => BAQ Mode	NO	NO	NO	YES	YES	Majority poll across data take for echo packets
Radar Configuration Support Service => BAQ Block Length	YES	YES	YES	YES	YES	Majority poll across data take for echo packets
Radar Configuration Support Service => Spare	NA	NA	NA	NA	NA	Ignore
Radar Configuration Support Service => Range Decimation	NO	NO	YES	NO	YES	Majority poll within swath
Radar Configuration Support Service => Rx Gain	NO	NO	YES	NO	YES	Majority poll within swath
Radar Configuration Support Service => Tx Ramp Rate	NO	NO	YES	NO	YES	Majority poll within swath
Radar Configuration Support Service => Tx Pulse Start Frequency	NO	NO	YES	NO	YES	Majority poll within swath within swath
Radar Configuration Support Service => Tx Pulse Length	NO	NO	YES	NO	YES	Majority poll within swath
Radar Configuration Support Service => Spare	NA	NA	NA	NA	NA	Ignore
Radar Configuration Support Service => Rank	NO	NO	YES	NO	NO	Majority poll within swath
Radar Configuration Support Service => PRI	YES (except S5)	NO	YES	NO	YES	Majority poll within swath

Sentinel-1

Ref:
MPC Nom: DI-MPC-IPFDPM
MPC Ref: MPC-0307
Issue/Revision: 2/1
Date: 31/01/2017

Field	Constant around Orbit	Constant during Data Take	Constant within Swath	Constant for Packet Type	Translation LUT or Law Required	Validation Strategy
Radar Configuration Support Service => SWST	NO	NO	NO	NO	YES	Majority poll across multiple packets, check for change
Radar Configuration Support Service => SWL	NO	NO	NO	NO	YES	Majority poll across multiple packets, check for change
Radar Configuration Support Service => SAS SSB Message (Imaging/Noise) => SSB Flag	NO	NO	NO	YES	YES	Check against expected value from imaging/noise/calibration sequence, determined from timeline and counters
Radar Configuration Support Service => SAS SSB Message (Imaging/Noise) => Polarisation	YES	YES	YES	YES	YES	Majority poll within data take
Radar Configuration Support Service => SAS SSB Message (Imaging/Noise) => Temperature Compensation	NO	NO	NO	NO	YES	Majority poll across multiple packets, check for change
Radar Configuration Support Service => SAS SSB Message (Imaging/Noise) => Spare	NA	NA	NA	NA	NA	Ignore
Radar Configuration Support Service => SAS SSB Message (Imaging/Noise) => Elevation Beam Address	NO	NO	YES	YES	YES	Majority poll within swath
Radar Configuration Support Service => SAS SSB Message (Imaging/Noise) => Azimuth Beam Address	NO	NO	YES	YES	YES	Majority poll within swath
Radar Configuration Support Service => SAS SSB Message (Calibration) => SSB Flag	NO	NO	NO	YES	YES	Check against expected value from imaging/noise/calibration sequence, determined from timeline and counters
Radar Configuration Support Service => SAS SSB Message (Calibration) => Polarisation	YES	YES	YES	YES	YES	Majority poll within data take
Radar Configuration Support Service => SAS SSB Message (Calibration) => Temperature Compensation	NO	NO	NO	NO	YES	Majority poll across multiple packets, check for change
Radar Configuration Support Service => SAS SSB Message (Calibration) => Spare	NA	NA	NA	NA	NA	Ignore
Radar Configuration Support Service => SAS SSB Message (Calibration) => SAS Test	YES	YES	YES	YES	YES	Ignore
Radar Configuration Support Service => SAS SSB Message (Calibration) => Cal Type	NO	NO	NO	NO	YES	Check against expected value from imaging/noise/calibration sequence, determined from timeline and counters. Check against Signal Type field.
Radar Configuration Support Service => SAS SSB Message (Calibration) => Calibration Beam Address	NO	NO	NO	NO	NO	Ignore

Sentinel-1

Ref:
MPC Nom: DI-MPC-IPFDPM
MPC Ref: MPC-0307
Issue/Revision: 2/1
Date: 31/01/2017

Field	Constant around Orbit	Constant during Data Take	Constant within Swath	Constant for Packet Type	Translation LUT or Law Required	Validation Strategy
Radar Configuration Support Service => SES SSB Message => Calibration Mode	YES	YES	YES	YES	YES	Majority poll within data take
Radar Configuration Support Service => SES SSB Message => Tx Pulse Number	NO	NO	YES	NO	NO	Majority poll within swath
Radar Configuration Support Service => SES SSB Message => Spare	NA	NA	NA	NA	NA	Ignore
Radar Configuration Support Service => SES SSB Message => Signal Type	NO	NO	NO	NO	YES	Check against expected value from calibration sequence, determined from timeline and counters
Radar Configuration Support Service => SES SSB Message => Swap Flag	NO	NO	NO	NO	NO	Check against expected value from imaging/noise/calibration sequence, determined from timeline and counters
Radar Configuration Support Service => SES SSB Message => Swath Number	NO	NO	YES	NO	NO	Determine swath sequence from timeline and counters. Majority poll within swath

4.3.1 Missing Line Detection

After any possible bit errors have been detected and corrected (as described in Table 4-1), missing line detection is performed. There are two cases in which echo packets can be missing from the input data and both of these cases are detected by monitoring the current value of each packet counter from the source packet headers against the previous value of that packet counter.

Let:

$$PSCT_{increment} = PSCT_{current} - PSCT_{previous} \quad (4-47)$$

$$SPCT_{increment} = SPCT_{current} - SPCT_{previous} \quad (4-48)$$

$$PRICT_{increment} = PRICT_{current} - PRICT_{previous} \quad (4-49)$$

Where:

PSCT = Packet Sequence Count from primary ISP header

SPCT = Space Packet Count from secondary ISP header

PRICT = Mode PRI Count from secondary ISP header

The first case of missing lines is echo packets physically missing from the input data due to some reason external to the SAR instrument; for example, packets missing due to a problem with the downlink which causes echo packets to be lost during transmission. Packets lost due to their absence from the input data are referred to as **downlink missing lines**. Downlink missing lines are detected by comparing the current value of all three space packet counters against the previous value of the respective counter. If the result is one (1) for all counters then the packet sequence is intact. If the result is greater than one (1) and the same for all counters then the number of missing lines is calculated using the SPCT from the secondary packet header as follows:

$$numDownlinkMissingLines = SPCT_{increment} - 1 \quad (4-50)$$

The second case of missing lines is echo packets missing from the input data due to an anomaly in the SAR instrument that prevents it from generating a SAR space packet during the image acquisition. Packets lost for this reason are referred to as **instrument missing lines**. Instrument missing lines are present in the input data when the PSCT increments by one (1), while the PRICT and SPCT increment by an amount equal to each other and that amount is greater than one (1). The number of instrument missing lines is then calculated using the SPCT as follows:

$$numInstrumentMissingLines = SPCT_{increment} - 1 \quad (4-51)$$

4.4 Terrain Height Function

During pre-processing, the IPF will build a vector of (average) terrain height values along the azimuth direction over the scene. Note that for TOPSAR, the same vector will apply for all sub-swaths. In the dual-polarisation case, the IPF will calculate only one terrain height function that will be applied to both polarisations. The azimuth spacing of the vector elements is a configurable, input parameter, except in the case of Wave mode where it is calculate once per cell.

Thus, the terrain height function is defined by a set of pairs:

$$(\eta_i, h_i), \quad i = 0, 1, 2, \dots, n-1$$

where (η_i) is a set of azimuth time values that covers the entire azimuth extent of the scene. Each terrain height value h_i is obtained by averaging over the subset of a DEM corresponding to an azimuth block centered on η_i and covering the entire range dimension of the scene. The size of the azimuth block used for averaging is a configurable input parameter.

Intermediate values of the terrain height, if later required by the SLC processor or the L1 Post-Processor, will be obtained by linear interpolation.

5 DOPPLER CENTROID ESTIMATION ALGORITHMS

This section presents a Doppler Centroid Estimation (DCE) approach based on algorithms and experience from the ENVISAT ASAR operational single beam and ScanSAR processors.

At a very high level, the DCE processing can be represented as in Figure 5-1:

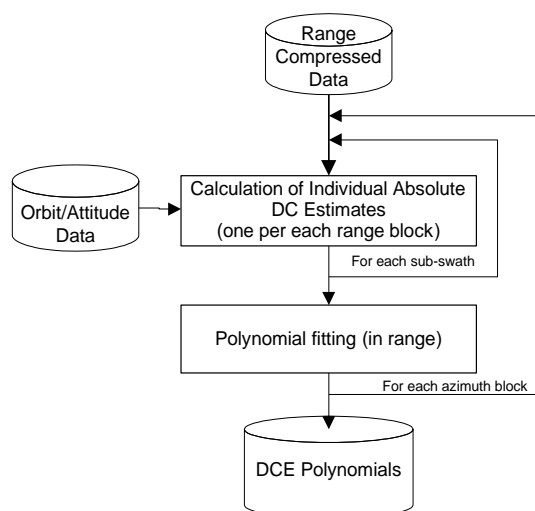


Figure 5-1 High Level DCE Algorithm

A discussion on the DCE processing blocks dimensions is presented in Section 5.6.

The two high level algorithmic blocks illustrated in Figure 5-1 are presented in the next sections as follows:

Calculation of Individual Absolute DC Estimates

The calculation of the individual DC estimates (for each range and azimuth block) is the same for both single-swath (Stripmap and Wave) modes and for the multi-swath (TOPSAR) modes (with the exception of a TOPSAR pre-conditioning step described in Section 5.2.1). This stage, which is illustrated in more detail in Figure 5-2, consists of the following steps:

1. **Absolute DC calculation (from Orbit & Attitude)**, as described in Section 5.1
2. **Fine DC estimation**, as described in Section 5.2
3. **DC estimate quality measurement**, as described in Section 5.5.1
4. **Fine DC estimates unwrapping**, as described in Section 5.3

5. **Polynomial Fitting**, as described in Section 5.5
6. **Absolute DC estimation**, as described in Section 5.4

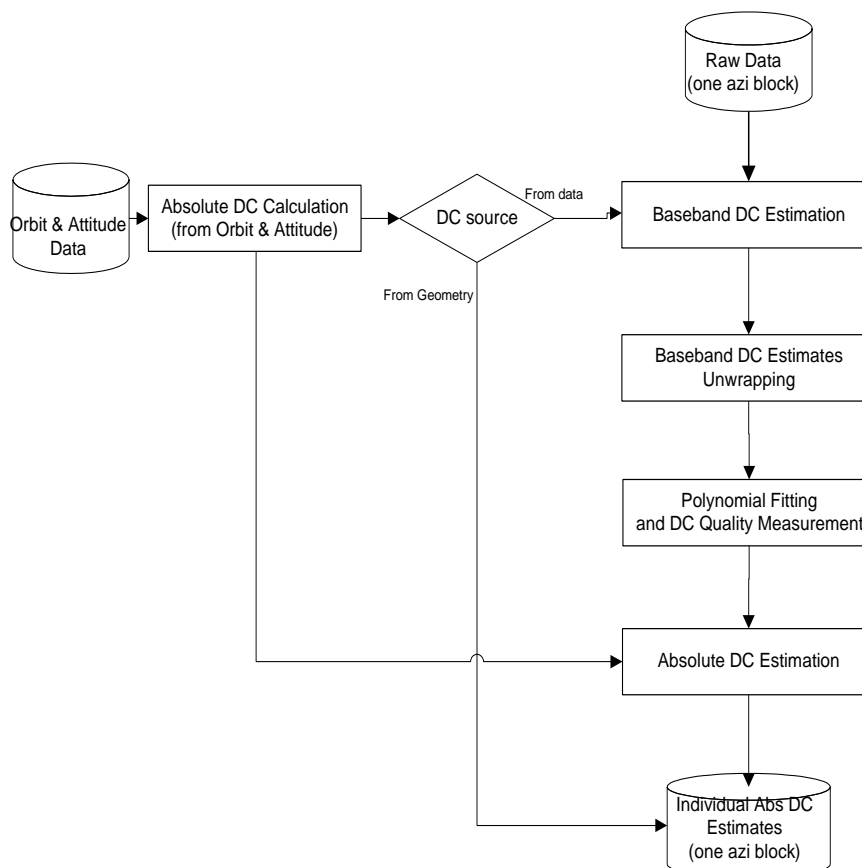


Figure 5-2 Calculation of Individual Absolute DC Estimates

The source of the DC frequency to be used in the subsequent processing steps is configurable, and can be one of the following:

- Calculated from Orbit & Attitude
- Estimated from data
- Set to a pre-defined, configurable polynomial (this case is not shown in Figure 5-2).

Note that in case of software failure or quality threshold breach, occurring in any of the steps of the algorithm (except the first one), the fall-back DC frequency is the one calculated from geometry at the beginning of the procedure.

5.1 Absolute DC Calculation (from Orbit & Attitude)

The first step is to determine the absolute DC from the spacecraft-earth geometry, using the Orbit and Attitude data. IPF will always perform this step. If the results were highly accurate this calculation would in fact completely solve the problem. At the very least, the precision of the beam pointing and measurements would allow the precise determination of the ambiguity. However, inherent measurement and beam pointing errors invariably translate into DC estimation unpredictability. In this case, the absolute DC must be estimated from the data.

In the current Sentinel-1 DCE algorithm, it is assumed that the DC frequency ambiguity can be reliably calculated from the geometry. Satellite position and velocity along the orbit, or the satellite ephemeris, is described by state vectors, which are typically estimated by on-board GPS systems and are included as engineering data in the SAR signal records.

For a platform moving with relative velocity \mathbf{v} , the Doppler frequency of a target of slant range r and view vector \mathbf{r} (the view vector is the vector from the satellite position P to the target) is given by

$$f = -\frac{2 \mathbf{v} \cdot \mathbf{r}}{\lambda r} \quad (5-1)$$

where λ is the radar wave-length.

This section describes how the Doppler frequency of a target can be calculated from:

\mathbf{p} = satellite position

\mathbf{v} = satellite relative velocity

r = slant range, and

$\boldsymbol{\varphi}$ = satellite attitude, defined by the triplet of angles (roll, pitch, yaw). (See for example [R-9] Section 2 for a definition of the attitude angles).

For a fixed orbit point (i.e. for fixed satellite position and velocity) the Doppler frequency as a function of r and $\boldsymbol{\varphi}$ will be named the *Doppler Frequency Function* (DFF). The DFF is given by

$$f(r, \boldsymbol{\varphi}) = -\frac{2}{\lambda} \cdot \mathbf{v} \cdot \mathbf{r}_0(r, \boldsymbol{\varphi}) \quad (5-2)$$

where $\mathbf{r}_0(r, \boldsymbol{\varphi})$ is the unit view vector (see Figure 5-3).

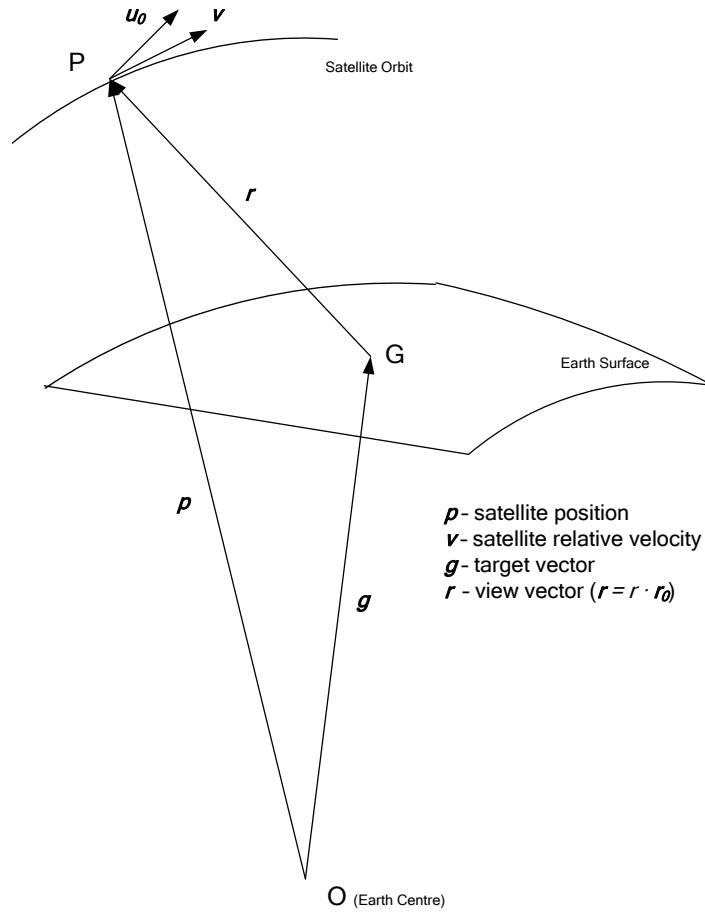


Figure 5-3 SAR Geometry

For a given slant range r and an attitude vector $\boldsymbol{\varphi}$, the unity vector $\mathbf{r}_0(r, \boldsymbol{\varphi})$ can be calculated as a solution of the following system of non-linear algebraic equations:

$$\mathbf{g}(r, \boldsymbol{\varphi}) = \mathbf{p} - r \cdot \mathbf{r}_0(r, \boldsymbol{\varphi}) \quad (5-3)$$

$$\mathbf{u}_0(\boldsymbol{\varphi}) \cdot \mathbf{r}_0(r, \boldsymbol{\varphi}) = 0 \quad (5-4)$$

$$\mathbf{r}_0(r, \boldsymbol{\varphi}) \cdot \mathbf{r}_0(r, \boldsymbol{\varphi}) = 1 \quad (5-5)$$

$$\mathbf{g}_s(r, \boldsymbol{\varphi}) \cdot \mathbf{g}_s(r, \boldsymbol{\varphi}) = 1 \quad (5-6)$$

$$\mathbf{g}_s(r, \boldsymbol{\varphi}) = \left[\frac{g_x}{a} \quad \frac{g_y}{a} \quad \frac{g_z}{b} \right]^T \quad (5-7)$$

$$\mathbf{g}(r, \boldsymbol{\varphi}) = [g_x \quad g_y \quad g_z]^T \quad (5-8)$$

where:

a = Earth's major semi-axis

b = Earth's minor semi-axis

\mathbf{g} = vector of components $(g_x \ g_y \ g_z)$ = the target point on the Earth Surface which corresponds to the given slant range r and unity vector $\mathbf{r}_0(r, \varphi)$, as illustrated in Figure 5-3)

$\mathbf{u}_0(\varphi)$ = unity vector, orthogonal to $\mathbf{r}_0(r, \varphi)$, and defined by the satellite orientation as:

$$\mathbf{u}_0(\varphi) = \mathbf{L}_0(\mathbf{p}, \mathbf{v}_a) \cdot \mathbf{L}(\varphi) \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \quad (5-9)$$

The matrices $\mathbf{L}_0(\mathbf{p}, \mathbf{v}_a)$, $\mathbf{L}(\varphi)$ can be calculated as follows:

1. The matrix $\mathbf{L}_0(\mathbf{p}, \mathbf{v}_a)$ is an orbit matrix and can be calculated from the satellite position vector \mathbf{p} and the satellite absolute velocity \mathbf{v}_a

$$\mathbf{v}_a = \mathbf{v} + \mathbf{w} \times \mathbf{p} \quad (5-10)$$

where the vector \mathbf{w} is the Earth sidereal rotational velocity (only its z-component has a non-zero value, equal to 0.00007292115833 rad/s).

Specifically, the matrix $\mathbf{L}_0(\mathbf{p}, \mathbf{v}_a)$ is defined by the following equations:

$$\mathbf{L}_0(\mathbf{p}, \mathbf{v}_a) = [\mathbf{a} \ \mathbf{b} \ \mathbf{c}] \quad (5-11)$$

$$\mathbf{c} = -\mathbf{q} / \|\mathbf{q}\| \quad (5-12)$$

$$\mathbf{q} = \begin{bmatrix} p_x \\ p_y \\ p_z / (1 - e^2) \end{bmatrix}$$

$$\mathbf{b} = (\mathbf{c} \times \mathbf{v}_a) / \|\mathbf{c} \times \mathbf{v}_a\| \quad (5-13)$$

$$\mathbf{a} = \mathbf{b} \times \mathbf{c} \quad (5-14)$$

where the scalar e is the Earth spheroid eccentricity.

2. The matrix $\mathbf{L}(\boldsymbol{\varphi})$ in equation (5-9) is the attitude matrix and can be calculated from (φ, θ, Ψ) - roll, pitch and yaw attitude angles.

$$\mathbf{L}(\boldsymbol{\varphi}) = \begin{bmatrix} \cos \Psi & -\sin \Psi & 0 \\ \sin \Psi & \cos \Psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} \cos \theta & 0 & \sin \theta \\ 0 & 1 & 0 \\ -\sin \theta & 0 & \cos \theta \end{bmatrix} \cdot \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \varphi & -\sin \varphi \\ 0 & \sin \varphi & \cos \varphi \end{bmatrix} \quad (5-15)$$

Note that in general $\mathbf{L}(\boldsymbol{\varphi})$ is not uniquely defined by a given attitude vector. In order for $\mathbf{L}(\boldsymbol{\varphi})$ to be uniquely defined, a rotation order for the (φ, θ, Ψ) triplet must be known. For Sentinel-1, the same rotation order as for the ASAR was assumed, namely the 3-2-1 rotation order.

Finally, for a given slant range r and satellite attitude vector, a value of the DFF, $f(r, \boldsymbol{\varphi})$, can be calculated using equations (5-2) through (5-15). This operation will be performed for the slant range corresponding to the centre of each of the n range blocks. The resulting collection of estimates can be expressed as a set of pairs of slant range and absolute Doppler Centroid frequency:

$$(r_i, \tilde{f}_i), \quad i = 0, 1, 2, \dots, n-1. \quad (5-16)$$

5.2 Fine DC Estimation

Typically, the DC estimate from geometry alone does not provide sufficient accuracy. As a consequence, the fine (also known as ‘baseband’, ‘fractional’ or ‘ambiguous’) DC frequency must be estimated from data.

The Doppler sampling rate, which is the PRF, limits the highest observable rate of the Doppler frequency. Thus, in the received signal, only frequencies between $-\text{PRF}/2$ and $+\text{PRF}/2$ can be observed. The component corresponding to these frequencies is referred to as the baseband PRF part of the Doppler frequencies.

Note that before actually applying the demodulation, the data must be first BAQ-decoded and raw data corrected. Also, the PG product compensation has to be applied. These operations are performed for each range line in the azimuth block:

- Perform raw data decoding** as described in Section 9.1.
- Perform raw data correction** as described in Section 9.2.
- Perform drift compensation** as described in Section 9.4.
- Perform range compression** as described in Section 9.4. **Erreur ! Source du renvoi introuvable..**

The calculation of the fine DC frequency requires the following steps:

- DCE pre-conditioning (TOPSAR only)**, as discussed in Section 5.2.1.
- Correlation DC estimator (CDCE)**, as discussed in Section 5.2.2.

Note that the fine DC frequency estimation is performed on range compressed data, and therefore, it takes place in the range-time-by-azimuth-time domain.

5.2.1 DCE Pre-Conditioning (TOPSAR only)

For performing DCE for TOPSAR mode data, before applying the CDCE algorithm, the data must be demodulated in the azimuth direction. The input data to this step is a burst of range compressed data.

The demodulation is performed azimuth line by azimuth line, as follows:

1. **Compute the de-ramping signal, $d(\eta)$** , defined by:

$$d(\eta) = \exp\left(j \cdot \pi \cdot k_s \left(\eta - \frac{T_b}{2}\right)^2\right), \quad \eta \in [0, T_b] \quad (5-17)$$

where:

η = Azimuth time (or slow time)

T_b = Burst time

k_s = the Doppler Centroid rate introduced by the scanning of the antenna (see Section 9.14 for description and calculation).

2. **Multiply each azimuth line in the burst by the de-ramping signal.**

5.2.2 Correlation DC Estimator (CDCE)

The CDCE algorithm is a phase-based algorithm (second-order statistic estimate) which derives the fine Doppler centroid from the *Average Cross Correlation Coefficient* (ACCC) at lag one, a statistical measure of the phase derivative at the beam centre crossing time.

5.2.2.1 Algorithm Overview

The algorithm is very well described in the literature (see for example [R-2], [R-5] Section 12.4.2, as well as the ENVISAT PF-ASAR Technical Note [R-10] for details and interpretation).

5.2.2.2 Algorithm Implementation

The input to this algorithm is an azimuth block of signal data of a particular swath. The CDCE consists of the following operations, performed on each range line:

1. **Calculate the ACCC at lag one for each azimuth line:**

$$c(\eta) = \sum_{\eta} s(\eta) s^*(\eta + \Delta\eta) \quad (5-18)$$

where:

$s(\eta)$ = a sample of the radar signal

$\Delta\eta$ = the azimuth sampling interval of 1/PRF, therefore

$s(\eta + \Delta\eta)$ = the next sample in the azimuth direction.

Note that this operation can be performed in the same time for all azimuth lines (or equivalently, all range samples) by performing the cross correlation on full range lines.

2. **Divide the ACCC vector in as many sub-vectors as range blocks (see Section 5.6 for the range blocks definition) and average within each sub-vector.** This gives a range-averaged ACCC value corresponding to each range block.
3. **Calculate the angle of each ACCC value derived at step 2, ϕ_{acc} .**
4. **Calculate the fine DC frequency for each ϕ_{acc} derived in step 3 (i.e. calculate the DC frequency corresponding to each range block):**

$$f_{\eta_c} = -\frac{PRF}{2\pi} \phi_{acc} \quad (5-19)$$

It is clear then, that the output corresponding to the given azimuth block of input data, is a collection of fine DC frequency estimates, each corresponding to the slant range at the centre of a range block of data. This collection of estimates can be expressed as a set of pairs of fast time and fine Doppler Centroid frequency

$$(\tau_i, f_i), \quad i = 0, 1, 2, \dots, n-1 \quad (5-20)$$

where n is the number of range blocks.

5.3 Fine DC Estimates Unwrapping

For each azimuth block, the fine DC estimates obtained as described in Section 5.2 must be unwrapped in the range direction, by removing jumps of more than PRF/2 between consecutive estimates. For best results, the block estimates that are judged to be too noisy or biased (based on results from the previous step) must be ignored or weighted down in the unwrapping process.

The conventional, simple way to implement a one-dimensional unwrapping is to ensure the DC difference between two subsequent range-block estimates is smaller than PRF/2 in absolute value. However, this approach is prone to unwrapping errors,

so the IPF employs a more robust algorithm, which was first proposed and described in [R-11].

5.3.1 Algorithm Overview

The unwrapping of fine DC estimates is carried out in a one dimensional way, along range. The fine estimates $f_{\eta_c}(kT_s)$ are supposed to be sampled along range at a sampling interval of T_s .

The unwrapping in the azimuth direction is then insured by taking into account the ambiguity derived from the absolute DC frequency, as described in Section 5.4.

The unwrapping algorithm assumes that the residuals of the estimates with respect to their linear component are not wrapped.

The first step is to estimate the coefficients a and b of the linear component of the normalized DC frequency estimates along range:

$$f_{linear}(\tau) = a\tau + b \quad (5-21)$$

where τ is the range (fast) time.

The linear model parameters a and b are derived via an FFT based estimator. First, the fine estimates are normalized to PRF and then the Fourier transform is computed:

$$F(v) = FFT(\exp(j2\pi \cdot f_{\eta_c}(kT_s) / PRF)) \quad (5-22)$$

If the contribution of the estimates is weighted by a weighting function, $w(\tau)$, this expression can be rewritten as:

$$F(v) = FFT[w(kT_s) \cdot \exp(j2\pi \cdot f_{\eta_c}(kT_s) / PRF)] \quad (5-23)$$

Let \hat{v} be the normalized frequency of the maximum of $|F(v)|^2$:

$$\hat{v} = \arg\left(\max_v |F(v)|^2\right)$$

Then

$$\begin{aligned} a &= \frac{\hat{v}}{T_s} \\ b &= \frac{\angle F(\hat{v})}{2\pi} \end{aligned} \quad (5-24)$$

where T_s is the sampling interval of the fine estimates.

The residual DCE frequency is then computed as:

$$res(kT_s) = \angle \left[\exp(j2\pi \cdot f_{\eta_c}(kT_s) / PRF) \cdot \exp(-j(akT_s + b)) \right] \cdot \frac{1}{2\pi} \quad (5-25)$$

Then the unwrapped DC estimates are derived by adding the residuals to the estimated (unwrapped) linear trend, and multiplying by PRF:

$$\hat{f}_{\eta_c}(kT_s) = (akT_s + b + res(kT_s)) \cdot PRF \quad (5-26)$$

5.3.2 Algorithm Implementation

The inputs to the unwrapping algorithm are the fine DC estimates (5-20) corresponding to the current azimuth block.

1. Calculate the vector **u** :

$$u_i = w_i \cdot \exp(j2\pi \cdot f_i / PRF), \quad i = 0, 1, 2, \dots, n-1 \quad (5-27)$$

2. **Zero-pad the vector **u** to the length of the FFT** such that the following FFT is evaluated in a dense grid in the frequency domain. The FFT length is a configurable input parameter.
3. **Take the FFT of the zero-padded vector.**
4. **Estimate the linear component of the DC frequency** (as a function of fast time) by calculating the coefficients a and b according to formulae (5-24).
5. **Calculate the residual vector:**

$$res_i = \angle(u_i \cdot \exp(-j(a\tau_i + b))) \cdot \frac{1}{2\pi}, \quad i = 0, 1, \dots, n-1 \quad (5-28)$$

6. Calculate the unwrapped DC frequency vector:

$$\hat{f}_i = (a\tau_i + b + res_i)PRF, \quad i = 0, 1, 2, \dots, n-1 \quad (5-29)$$

5.4 Absolute DC Estimation

The absolute DC estimates are calculated from the unwrapped fine DC estimates obtained as described in Section 5.3.2 and the ambiguity derived from the absolute DC calculation from orbit and attitude (Section 5.1).

- **Calculate the ambiguity A** from the absolute DC frequency calculated from orbit and attitude, of the first range block, \tilde{f}_0 :

$$A = \left\lceil \frac{\tilde{f}_0}{PRF} \right\rceil \quad (5-30)$$

- **For each azimuth block perform the following:**
 - Unwrap the FDC from data at baseband
 - Offset the unwrapped FDC from data to the ambiguity of the FDC from geometry
 - Evaluate the FDC from geometry to generate an array of FDC points across range
 - Compute the delta of the unwrapped FDC from data and the FDC from geometry
 - Perform a polynomial fit for the delta FDC
 - Add the delta FDC polynomial coefficients to the geometry FDC coefficients to calculate the data FDC polynomial for this azimuth block

5.5 Polynomial Fitting

The polynomial fitting is performed twice: once for the DC frequency calculated from orbit and attitude (Section 5.1) and once for the DC frequency estimated from data (Section 5.4). See Section 5.6 for more details on the DCE block sizes and frequency of update.

This step, which is performed for each azimuth block, depends on whether the data is single-swath or multi-swath:

1. **Polynomial fitting for single-swath** (Stripmap type modes)
 - Find the coefficients of a quadratic polynomial $p(x) = c_0 + c_1x + c_2x^2$ that fits the absolute DC frequency estimates data, $p(r_i)$ to g_i . This is a least-squares polynomial fit to the DC estimates where the fitting procedure detects and eliminates outlying DC estimates (see Section 5.5.1).
 - Find a fitting quadratic polynomial $\tilde{p}(\cdot)$, for the absolute DC frequency from Orbit and Attitude, $(r_i, \tilde{f}_i), i = 0, 1, 2, \dots, n-1$. A least-square fit is used is also used in this case.

2. **Polynomial fitting for multi-swath** (TOPSAR modes)

All the algorithms described so far are independently performed for each sub-swath (see Figure 3-1). By contrast, for a given azimuth block, the polynomial fitting for the multi-swath case is performed taking into account all individual range estimates in all sub-swaths. The procedure for fitting the polynomial for TOPSAR is the same as the one for single-swath discussed above, except that it's performed across all sub-swaths.

Once the polynomial has been derived across all sub-swaths, an adjustment is made to the first polynomial of each individual swath, using the algorithm

described in the Appendix to [R-11]. This algorithm is also implemented in the ASAR processor.

This algorithm was developed in order to capture possible jumps of the DC frequency between sub-swaths that may occur due to electronic beam pointing errors at the time of the switch. The idea exploited in the algorithm is that, in all sub-swaths, the polynomials must be defined by the same set of coefficients, with the exception of the constant term. This means that for each swath s ,

$$p_s(x) = c_{0s} + c_1x + c_2x^2, \quad s = 0, 1, \dots, S-1, \quad (5-31)$$

where S is the number of sub-swaths.

This means that only a step change can be allowed at the sub-swath boundaries, defined by $c_{0(s+1)} - c_{0s}$, $s = 0, 1, \dots, S-2$.

The problem is solved in the sense of the minimum least squares, for all DC estimation points in all sub-swaths by imposing on the polynomials coefficients the constraint described above.

The algorithm is only performed on the absolute DC estimates from data.

5.5.1 DC Quality Measurement

The DC estimates from data are often biased by various scene-content dependent factors. Therefore it is imperative to carefully assess the reliability of each block estimate.

This is done by performing a least-squares polynomial fit to the DC estimates where the fitting procedure detects and eliminates outlying DC estimates. The fitting procedure also produces an RMS error measure indicating the RMS difference between the input DC estimates and the fitted DC values (after any outliers have been eliminated). The DC estimate quality indicator used by the IPF corresponds to this RMS error measure between the selected input DC estimates and the fitted values.

5.6 Processing Blocks Dimensions

The DC estimation is performed on one azimuth block of data at a time. During the Fine DC Estimation stage, the full range extent of a block of azimuth data is further subdivided into range blocks according to input configuration parameter settings. For each of these blocks a DC estimation will be calculated.

The range and azimuth blocks can overlap in order to allow a smoother DC frequency variation. The sizes of the azimuth and range blocks are configurable input parameters. Also, for azimuth blocks, the spacing between estimates is a configurable input parameter while for range blocks, the number of blocks is an

input configurable parameter. In this way, the amount of overlapping between blocks can be controlled.

The processing block dimensions for DC estimation for the subsequent Level 1 processing stages must be carefully chosen:

- If the dimensions are too large (in either range or azimuth), the estimation can be affected by the natural variation of the DC with range and, to a lesser extent, with azimuth.
- If the dimension of the blocks is too small in azimuth, the estimation can suffer because of possible impulse response truncation. Therefore, the azimuth dimension of the blocks must be much bigger than the azimuth footprint size. In order to minimize the edge effects (due primarily to the presence of partially exposed targets), the size of an azimuth block would ideally be ten times the antenna footprint. However, this would lead to very large azimuth blocks (~ 40 km long in the C-band SAR case) and would conflict with the first constraint. A factor of minimum two is considered adequate. In TOPSAR, due to backward to forward scanning of the antenna, the footprint is reduced by a factor of approximately 3 to 4 (as described in Section 3.1.2), therefore a full burst can be used as an azimuth block. For Wave Mode, the azimuth dimension of the blocks is dictated by the natural division of the data into vignettes.

6 SLC PROCESSING ALGORITHMS

The slant-range, single-look complex (SLC) processing module of the Sentinel-1 IPF is responsible for correctly focussing the image in both range and azimuth.

The SLC processing module takes as input the following:

- Raw signal data
- Pre-processing output parameters, including orbit information
- DCE polynomials generated as described in Section 5

The SLC processing algorithm consists of the following steps, to be performed azimuth block by azimuth block:

1. **Range processing**, as described in Section 6.1
2. **Azimuth pre-processing**, as described in Section 6.2
3. **Azimuth processing**, as described in Section 6.3
4. **Azimuth post-processing (TOPSAR only)**, as described in Section 6.4

Note that throughout this section, unless otherwise specified, ‘azimuth block’ will be used in a generic way to also designate bursts (for the TOPSAR modes) or vignettes (For the WV mode).

For the TOPSAR modes each burst in each sub-swath is processed independently, resulting in an independent SLC image. The individually focused burst images are included, in azimuth-time order, into a single sub-swath image, with black-fill demarcation in between. Figure 6-1 presents the processing steps of the SLC processing algorithm.

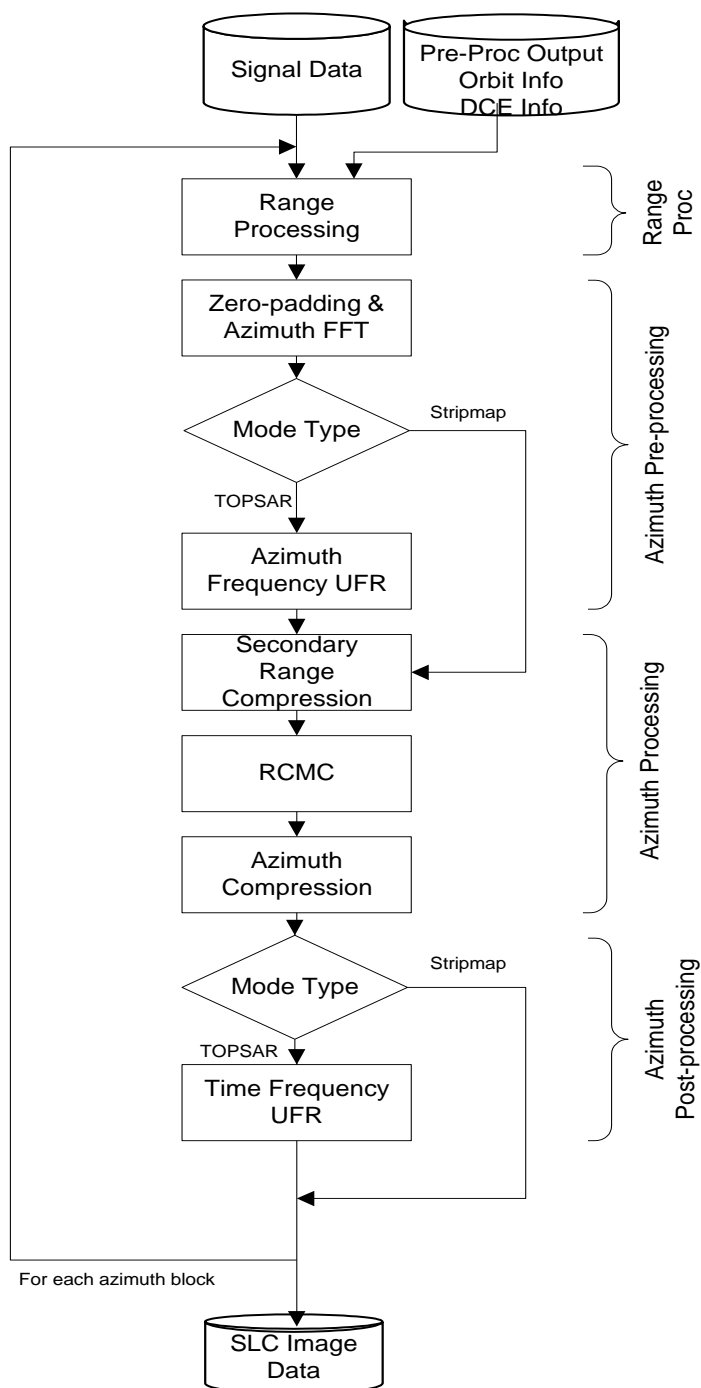


Figure 6-1 SLC Processing Algorithm

6.1 Range Processing

The range processing step of SLC processing consists of the following operations:

1. **[NORM] Calculate the Range Reference Function (RRF)**, as described in Section 6.1.1.

2. **For each range line in the current azimuth block of signal data, perform:**
 - a) **Raw data decoding**, as described in Section 9.1
 - b) **Raw data correction**, as described in Section 9.2
 - c) **[NORM] Drift compensation** as described in Section 9.4
 - d) **Range compression**, as described in Section **Erreur ! Source du renvoi introuvable.**
 - e) **[NORM] Range dependent gain correction**, as described in Section 6.1.2
3. **SWST bias correction**, as described in Section 6.1.3

6.1.1 Range Reference Function (RRF)

The matched filter used for range compression, also known as the range reference function (RRF), is generated as follows:

1. **The reference replica, $Rep(t)$, is calculated** from the amplitude and phase coefficients that are derived from the extracted chirp replica or from the nominal chirp.
 - For L1 product, the chirp amplitude is inverted such that after range compression the data spectrum will be flat. This is known as spectrum whitening. If spurious signal correction is to be applied the chirp duration will be limited to $TXPL_{nom}$ as described in section 9.2.2.2.
 - For L2 product, the reference replica bandwidth is extended to the range sampling rate. This is done by extending the chirp duration such that the bandwidth is up to the range sampling rate and the amplitude is set to be flat over the whole range sampling rate.
2. **$Rep(t)$ is zero padded to length N_{fft} and transformed to the frequency domain** using an FFT of this size, resulting in $R(f)$ of size N_{fft} .
3. **The complex conjugate, $R^*(f)$, is calculated.**
4. **Typically, a weighting window $W_r(f)$ is applied to the conjugate**, resulting in $R'(f)$. However, for the Sentinel-1 IPF the weighting window is nominally applied during post-processing as described in Section 7.1, and no weighting is applied during SLC processing.
5. **A time shift corresponding to the dual polarization misregistration, Δt is introduced** by applying a frequency domain linear phase ramp to the RRF, $\exp(-2j\pi f \Delta t)$. See section 4.2.1.4

6. [NORM] The energy, E , of $R'(f)$ is calculated as:

$$E = \frac{1}{N_{\text{fft}}} \sum_{f=0}^{N_{\text{fft}}-1} |R'(f)|^2 \quad (6-1)$$

Note that for SLC, $N_{\text{fft}} = N_{\text{fft}}$.

7. [NORM] The energy of the filter is normalized to unity by dividing $R'(f)$ by \sqrt{E} .

With the above notations, the RRF can be expressed as:

$$RRF(f) = \frac{W_r(f) \cdot (FFT\{Rep(t), N_{\text{fft}}\}(f)) \cdot \exp(-2j\pi f \Delta t)}{\sqrt{E}} \quad (6-2)$$

8. Finally, the dual polarization complex gain correction is multiplied to the RRF.

6.1.2 Range Dependent Gain Correction

To the range compressed line resulted from the range compression algorithm described in Section 6.2.2 a range dependent gain correction is applied. The range dependent gain correction has two components:

1. [NORM] **Elevation antenna pattern correction** as described in Section 9.5.1
2. [NORM] **Range spreading loss correction** as described in Section 9.6

The elevation antenna pattern corresponding to the current line is multiplied by the range spreading loss correction vector. The result is then applied to the current, range-compressed line.

The application of the elevation antenna pattern and the range spreading loss corrections are controlled by configurable, input parameters.

Note that at the time of the range dependent gain correction, the thermal noise estimation is also performed (see Section 9.17) with the purpose of annotating it to the product.

6.1.3 SWST Bias Correction

The SWST bias, $\Delta t_{\text{SWSTbias}}$, is provided as a configurable input parameter. The processor takes into account this bias by adding it to the range start time, τ_{start} :

$$\tau_{\text{start}} = \tau_{\text{start}} - \Delta t_{\text{SWSTbias}} \quad (6-3)$$

Note that the range start time is not explicitly used during range compression; this operation is performed in preparation of the azimuth processing that follows.

6.2 Azimuth Pre-Processing

The input to this stage is an azimuth block of range-compressed data represented in the 2D time domain. The azimuth pre-processing consists of the following steps, to be applied to each azimuth line in the azimuth block:

1. **Azimuth zero-padding**, as described in Section 6.2.1
2. **Azimuth forward FFT**, as described in Section 6.2.2
3. **Azimuth frequency Un-Folding and Re-sampling (UFR) (TOPSAR only)**, as described in Section 6.2.4

The calculation of the azimuth block length and block overlap size (which applies only to Stripmap data) is described in Section 9.12 and 9.13.

At the end of azimuth pre-processing, the data is represented in the range-Doppler domain, which is the required input to the Range-Doppler algorithm used in the next stage, azimuth processing.

6.2.1 Azimuth Zero-Padding

The azimuth zero-padding is performed to the length of the forward azimuth FFT, M_{fft} . For Stripmap, the size of the azimuth forward FFT is a configurable input parameter. For the TOPSAR modes, the azimuth forward FFT is the azimuth number of samples in the burst, rounded up to the next available FFT length.

6.2.2 Range Compression

Each range line is compressed using the following steps:

1. **Zero-pad the range line to N_{fft} length.**
2. **Transform the line into the frequency domain using an FFT.**
3. **Multiply the result by the RRF** (which has already been generated with the same size N_{fft} ; see Section 6.1.1).
4. **Perform the inverse FFT.**
5. **Calculate the required black fill**, as described in Section 6.2.2.1.
6. **The valid portion of the range compressed line (i.e. after matched filter throw-away) is then written to the output range processing buffer** (by

taking into account the black fill calculated as described in Section 6.2.2.1, and the range line length as described in Section 6.2.2.2).

6.2.2.1 Black-Fill Calculation

The black-fill of the compressed range lines is a consequence of (potential) changes over time in Sampling Window Start Time (SWST).

The SWST is the time offset between the start time of the transmitted pulse and the start time of the current receive sampling window. The SWST may change multiple times over a given azimuth duration, and this variation must be taken into account by the processor.

The IPF range compression algorithm handles changes in SWST by addressing separately the integer and the fractional part of the SWST change (where the SWST is expressed in range samples).

The algorithm for handling the fractional SWST change is described in Section 6.1.1, step 5. The algorithm for handling the integer range compressed sample portion of a SWST change is presented as follows:

1. **Calculate the maximum change in SWST over the entire segment**, as *maxSwstChange*, expressed in seconds.
2. **Calculate the maximum change in SWST over the entire segment**, *maxSwstChangeInSamps*, expressed in range samples:

$$\text{maxSwstChangeInSamps} = F_r * \text{maxSwstChange} \quad (6-4)$$

where F_r is the complex range sampling frequency.

3. **Calculate the integer part of the maximum SWST change:**

$$\text{intMaxSwstChange} = (\text{int})(F_r * \text{maxSwstChangeInSamps}) \quad (6-5)$$

This represents the total number of blackfill samples.

4. **For each range line, calculate the number of black fill samples required at start and end of line:**

- calculate the integer SWST change for the current line:

$$\text{intSwstChange} = (\text{int})(F_r * \text{swstChange}) \quad (6-6)$$

- calculate the black fill at the start of the line, *blackFillStart*, as:

$$\text{blackFillStart} = \text{intMaxSwstChange} - \text{intSwstChange} \quad (6-7)$$

- calculate the black fill at the end of the line *blackFillEnd* as:

$$\text{blackFillEnd} = \text{intMaxSwstChange} - \text{blackFillStart} \quad (6-8)$$

Erreur ! Source du renvoi introuvable. illustrates the concept of a range compressed buffer containing black-fill to accommodate a SWST change.

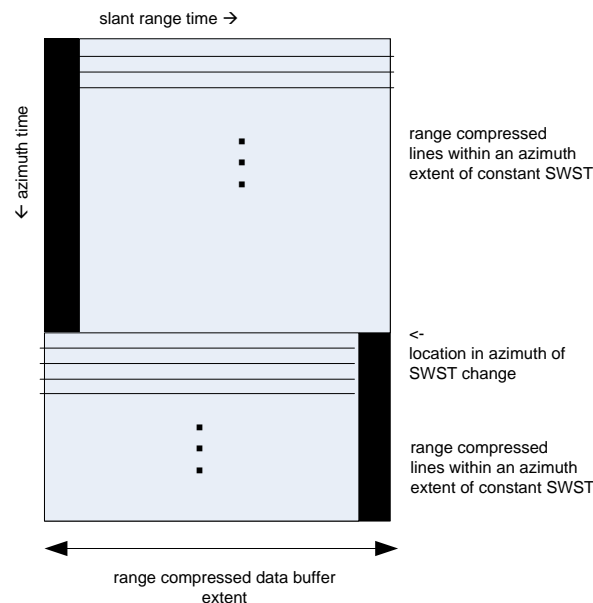


Figure 6-2 Range Compressed Data Buffer Containing Blackfill to Account for SWST Change

6.2.2.2 Range Line Length Calculation

In the input signal data, each range line is tagged with a sampling window length (SWL) and sampling window start time (SWST). For previous ESA SAR missions like ENVISAT ASAR, the SWST could change from range line to range line, but the SWL was constant within a swath. However, for Sentinel-1, both the SWL and SWST can change. During range compression, both the SWL and SWST changes must be taken into account, in order to properly align the range compressed lines in the buffer for azimuth compression.

There are three possible combinations of SWST and SWL changes which the SLC processor handles:

1. SWST changes but no SWL changes, as shown in **Erreur ! Source du renvoi introuvable.**
2. No SWST changes but with SWL changes, as shown in **Erreur ! Source du renvoi introuvable..**
3. SWST changes and SWL changes, as shown in **Erreur ! Source du renvoi introuvable.**

Note that, for the non-slicing case, the SWST and SWL changes need to be taken into account for the whole image. For the slicing case, the SWST and SWL changes need to be taken into account across all slices.

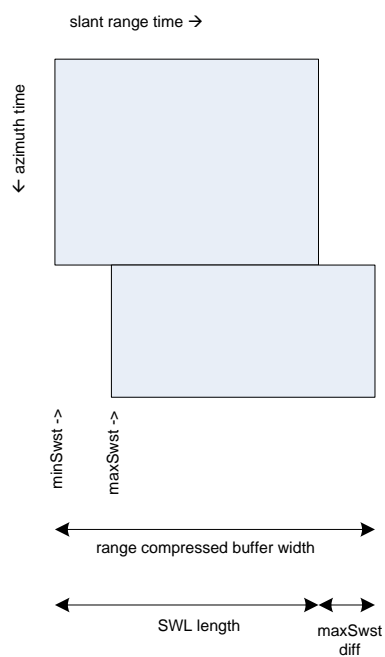


Figure 6-3 Range Lines with SWST Changes and No SWL Changes

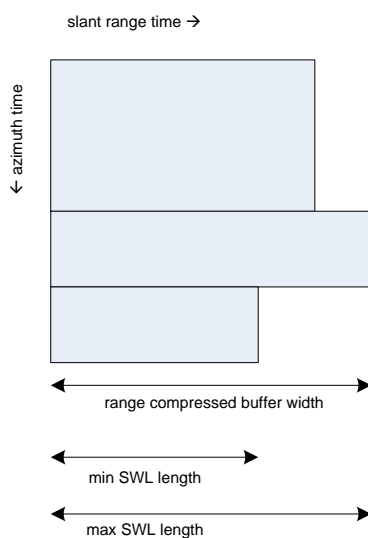


Figure 6-4 Range Lines with No SWST Changes but with SWL Changes

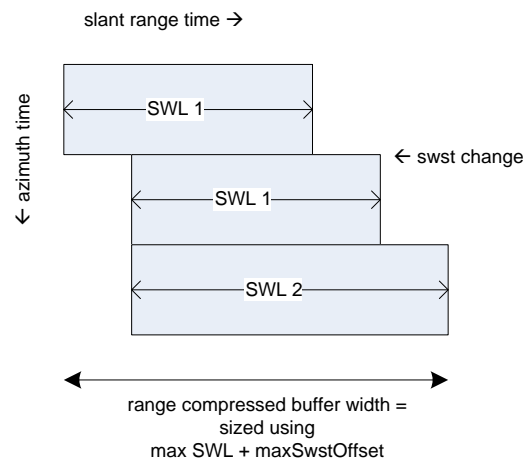


Figure 6-5 Range Lines with SWST Changes and SWL Changes

In order to account for the SWL changes, do the following when writing the range compressed results to the output buffer:

1. **Zero-fill the line buffer of to the maximum SWL for the entire segment, *maxSwlSamp***
2. **Read the current SWL size in samples, *currSwlSamp***
3. **Extract *currSwlSamp* samples from the signal data and place into the first *currSwlSamp* samples of a zero-filled line buffer of length *maxSwlSamp***
4. **When handling range compression throwaway, make sure the throwaway is with respect to *currSwlSmp*, and not with respect to *maxSwlSamp*.**

6.2.3 Azimuth Forward FFT

The operation performed in this step is a basic forward FFT of length M_{fft} . Note that, after this step:

- For the Stripmap modes, the next processing step is the Range-Doppler algorithm azimuth compression, as described in Section 6.3.
- For the TOPSAR modes, the next processing step is frequency domain UFR, as described in Section 6.2.2.

6.2.4 Azimuth Frequency UFR (TOPSAR only)

The frequency domain UFR and the time domain UFR algorithms (described in Section 6.4) are conceptually the same. A generic UFR algorithm, underlying both the frequency domain and the time domain UFR is presented in Section 9.9.

The input to the frequency UFR is a burst of range-compressed, azimuth FFTed data represented in the range time by azimuth frequency domain.

The frequency UFR algorithm is basically one-dimensional and it is applied, in the frequency domain, to each azimuth line in a burst, for all bursts. Figure 6-6 presents a schematic view of the typical time-frequency diagram of a single TOPSAR azimuth line.

Note that for clarity, in all figures in the current section as well as in Section 6.4, the time-frequency diagram is represented for a Doppler centroid frequency equal to zero, $f_{\eta_c} = 0$.

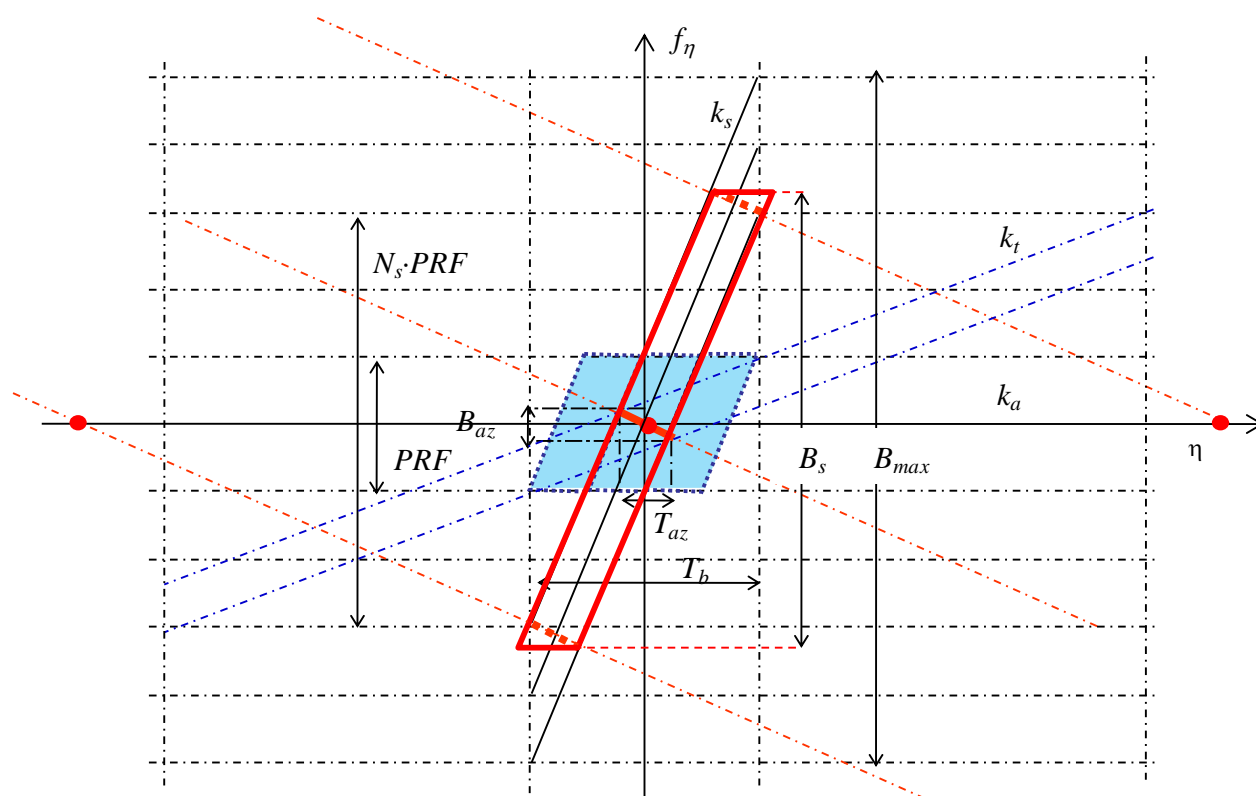


Figure 6-6 Time-Frequency Diagram of an Input TOPSAR Azimuth Line

The following notations will be used in this section:

- T_b = Burst time (or length in seconds)
- T_d = Dwell time (see Equation 6-13)
- η = Azimuth time (or slow time)
- f_{η} = Azimuth frequency
- f_{η_c} = DC frequency
- f_{ref} = DC frequency at the reference range (mid-swath)

- B_{az} = Maximum instantaneous Doppler bandwidth (see Equation 6-12)
- B_d = Target instantaneous Doppler bandwidth (configurable input parameter)
- B_{max} = Maximum acquired Doppler bandwidth (see Equation 6-11)
- B_s = Processing sweep bandwidth (see Equation (6-14))
- k_a = Doppler FM rate (see Section 9.11)
- k_s = Doppler Centroid rate introduced by the scanning of the antenna (see Section 9.14)
- k_t = Target instantaneous bandwidth rate in focused TOPSAR data

This rate is used during azimuth time UFR only (see Section 6.4.1).

Note that, in actuality, k_a and k_t are range dependent, $k_a(\tau)$ and $k_t(\tau)$ respectively. These dependencies will be omitted from the notations hereafter in order to simplify the readability of the formulae involved.

Note also the difference in notation with respect to [R-2] and [R-4], where the Doppler Centroid rate is denoted k_a rather than k_s . This change in notation was introduced in order to eliminate possible conflict with the heritage MDA documentation and software as well as with the notations in [R-5], where k_a is the notation of choice for the azimuth FM rate.

For a given azimuth line in a burst, the azimuth time range and the frequency range are given by:

$$-\frac{f_{\eta_c}}{k_a} - \frac{T_b}{2} \leq \eta \leq -\frac{f_{\eta_c}}{k_a} + \frac{T_b}{2} \quad (6-9)$$

and

$$f_{\eta_c} - \frac{PRF}{2} \leq f_{\eta} \leq f_{\eta_c} + \frac{PRF}{2} \quad (6-10)$$

The main steps for the frequency UFR algorithm are as follows:

6.2.4.1 Mosaicking

The time-frequency representation of the data is folded in frequency domain (due to the fact that the PRF is smaller than the actual Doppler bandwidth (or sweep bandwidth)). The maximum acquired Doppler bandwidth is:

$$B_{max} = k_s T_b + PRF \quad (6-11)$$

The processing sweep bandwidth, B_s , $B_s \leq B_{\max}$, is a beam dependent parameter, that must guarantee that two adjacent bursts, within the same beam, have the required spatial overlap in azimuth. Furthermore, B_s must be set in such a way as to guarantee that the contributions at the azimuth edges of the burst are effectively filtered out. (Note that in case those contributions were not discarded, an extended overlap azimuth region between bursts could be obtained; however this region would be characterised by a rapidly decreasing resolution along azimuth direction.)

In order to define the optimal sweep bandwidth, B_s , we need to introduce a design parameter that will be provided as a configurable input parameter: the target instantaneous Doppler bandwidth, B_d . This parameter directly impacts the SLC data output azimuth resolution and it is designed according to the following rules:

- B_d must be smaller than the maximum instantaneous bandwidth, B_{az} , where B_{az} is defined by:

$$B_{az} = -\frac{k_a}{k_s - k_a} PRF \quad (6-12)$$

- B_d must guarantee the required product azimuth resolution

The instantaneous Doppler bandwidth can be defined also by its time domain correspondent value T_d given by the following relation:

$$T_d = -\frac{1}{k_a} B_d \quad (6-13)$$

With these definitions, the sweep bandwidth that must be processed to guarantee coverage and the desired azimuth resolution can be defined as:

$$B_s = k_s (T_b - T_d) + B_d \quad (6-14)$$

Note that optionally, B_s can be set to be equal to a configurable input parameter, which must satisfy the condition, $B_s \leq B_{\max}$.

The actual filtering of this bandwidth can be effectively performed during the low-pass filtering and resampling step (see Section 6.2.4.3) by a proper masking window to be applied on the result of the interpolation.

Note also that since B_s is range and burst dependent, for the purpose of the implementation, the worst case must be considered in order to guarantee the azimuth coverage and azimuth overlap between consecutive bursts, in all swaths and along the entire orbit.

The sweep bandwidth is then used to define the number of spectral replicas N_s that must be replicated (put side by side). In this way the desired unfolded signal is reconstructed together with N_s unwanted copies of the same unfolded contribution.

The number of spectral replicas is computed by:

$$N_s = N_{s_pos} - N_{s_neg} \quad (6-15)$$

where:

$$N_{s_neg} = \text{floor}\left(\frac{f_{ref} - B_s/2}{PRF}\right) \quad (6-16)$$

$$N_{s_pos} = \text{ceil}\left(\frac{f_{ref} + B_s/2}{PRF}\right) \quad (6-17)$$

The effect of mosaicking is shown in Figure 6-7.

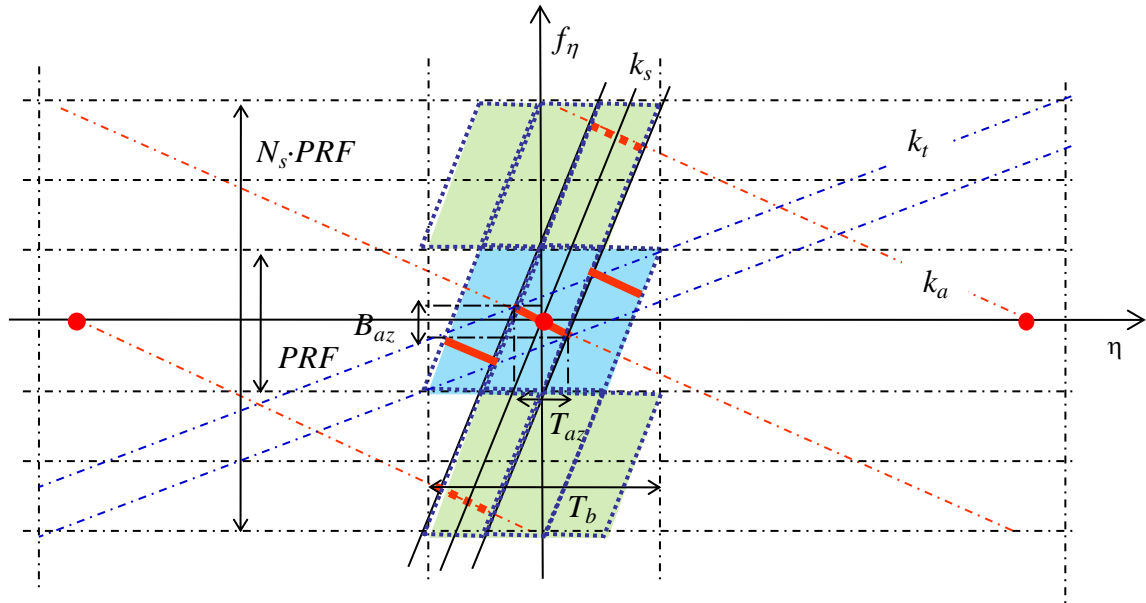


Figure 6-7 Time-Frequency Diagram of a TOPSAR Azimuth Line after Mosaicking

The frequency support of the output data line is then:

$$N_{s_neg} PRF \leq f_{\eta} \leq N_{s_pos} PRF \quad (6-18)$$

6.2.4.2 De-ramping

The desired spectral components are moved to low-pass band by a multiplication with the following chirp signal:

$$\phi(f_{\eta}) = \exp\left(j\pi \frac{1}{k_s} (f_{\eta} - f_{\eta_c})^2\right) \quad (6-19)$$

where:

$$f_{\eta_c} - \frac{B_s}{2} \leq f_{\eta} \leq f_{\eta_c} + \frac{B_s}{2}$$

6.2.4.3 Low-pass Filtering with Re-sampling

The replication adds unwanted spectral contributions that must be filtered out. The desired spectral components are selected by a low-pass filtering, performed via a convolution in frequency domain with a suitable band pass filter. Note that the operation we are describing has the time and frequency domains switched with respect to a common filtering process: here the convolution is performed in the frequency domain and the 'low-pass bandwidth' refers to an interval in the time domain around the zero time coordinate. In order to define this correct time bandwidth, we start from the maximum instantaneous bandwidth B_{az} (see Equation 6-12) in the other domain.

The low-pass filtering and resampling can be done using two different methods:

1. Perform low-pass filtering and resampling in one step. This is done using Finite Impulse Response (FIR) filter that accomplishes both low-pass filtering and resampling at the same time.
2. Perform low-pass filtering and resampling in two steps. The low-pass filtering is done using FFT, zero out unwanted data and inverse FFT. The resampling is done using a quadratic interpolation kernel.

Method 1 – Low-pass filtering and resampling in one step

In order to correctly filter-out the replicas due to mosaicking we must define a low-pass filter with 'time-domain' bandwidth T_{az} equal to:

$$T_{az} = -\frac{B_{az}}{k_a} \quad (6-20)$$

In our case we must take into account that the de-ramping step introduces a stretch of the needed time bandwidth due to the fact that the system IRF is, at this level, still unfocused. In particular, after the de-ramping step if we want to preserve the

available bandwidth B_{az} for every target, we need to select all the contribution included in the following time interval:

$$T'_{az} = T_{az} \frac{k_s - k_a}{k_s} = \frac{PRF}{k_s} \quad (6-21)$$

As already anticipated a smaller bandwidth, called instantaneous Doppler bandwidth B_d , and the correspondent time T_d , introduced in the previous section, can be considered in order to meet the azimuth resolution requirements. Applying the same scaling factor considered in the equation above, we can then define the actual time bandwidth of the required filter:

$$T'_d = T_d \frac{k_s - k_a}{k_s} \quad (6-22)$$

The filter must then be optimally flat in the time bandwidth T'_d .

Then the filter's maximum available transition bandwidth can be defined as:

$$T_{tr} = T'_{az} - T'_d \quad (6-23)$$

The filter characteristics are depicted in Figure 6-8. This plot represents half the filter bandwidth.

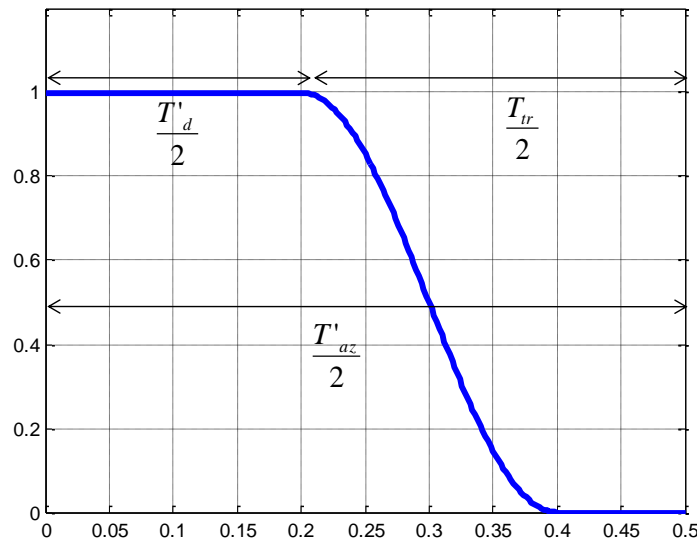


Figure 6-8 Low-pass Filter Sample Mask for Azimuth Frequency UFR

The low pass filtering step can be combined with a decimation step or, more general, can be implemented as a generic re-sampling step. The output frequency axis sampling step can be increased up to:

$$\Delta f_{\max} = \frac{1}{T_d} \quad (6-24)$$

For efficiency considerations, the output frequency sampling step, which is inversely proportional to the output time extent of the processed burst, should be chosen as large as possible. Given that the constraint on the maximum frequency step is respected, this processing step introduces a ‘controlled’ time domain aliasing without any impact on the following processing steps.

This filter method was implemented in the first release of the IPF until version 2.71 included. It was believed that this method would be advantageous throughput wise as the low-pass filtering and resampling are done in one operation. However, due to non-ideal response of the FIR filter the IPF output products suffer from azimuth ambiguity and normalization problems. As shown in Figure 6-8 the FIR filter response has transition band through which unwanted energy from the mosaicking step is not completely filtered out. This unwanted energy resulted in the azimuth ambiguity and normalization problems.

Method 2 – Low-pass filtering and resampling in two steps

Method 2 is introduced to resolve the azimuth ambiguity and normalization problems. The low-pass filter in this method has a sharp cut-off to only pass through the desired energy. Figure 6-9 shows the low pass filter responses of Method 1 and Method 2 in the same figure. The low pass filter of Method 2 (red line) only passes through the required energy and throws away the unwanted energy from the mosaicking step.

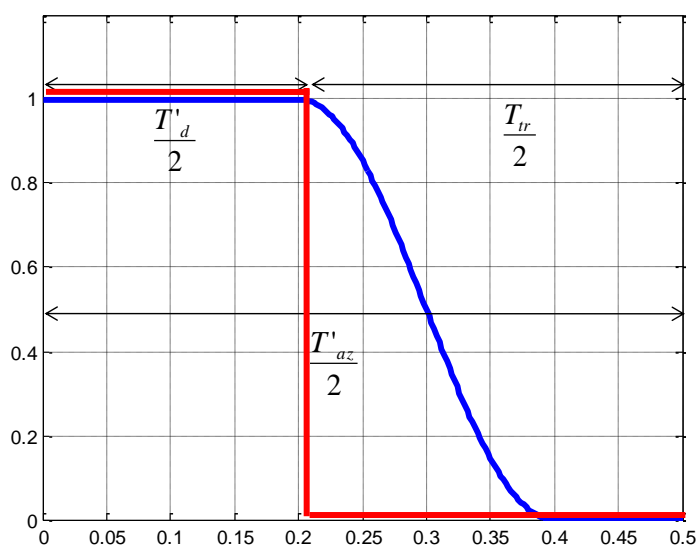


Figure 6-9 Comparison of Low-pass Filter responses for Azimuth Frequency UFR

The low-pass filter response has a sharp cut-off as it is done using FFT as follows:

1. Forward FFT
2. Zero out unwanted data
3. Inverse FFT

Following the low-pass filtering, the data is then resampled using quadratic interpolation.

6.2.4.4 Re-ramping

The phase removed by de-ramping is finally re-instated by applying the following chirp signal to the resampled azimuth data line:

$$\phi(f_\eta) = \exp\left(-j\pi \frac{1}{k_s} (f_\eta - f_{\eta_c})^2\right) \quad (6-25)$$

The following figure presents the time-frequency diagram of an azimuth line at end of the frequency unfolding step.

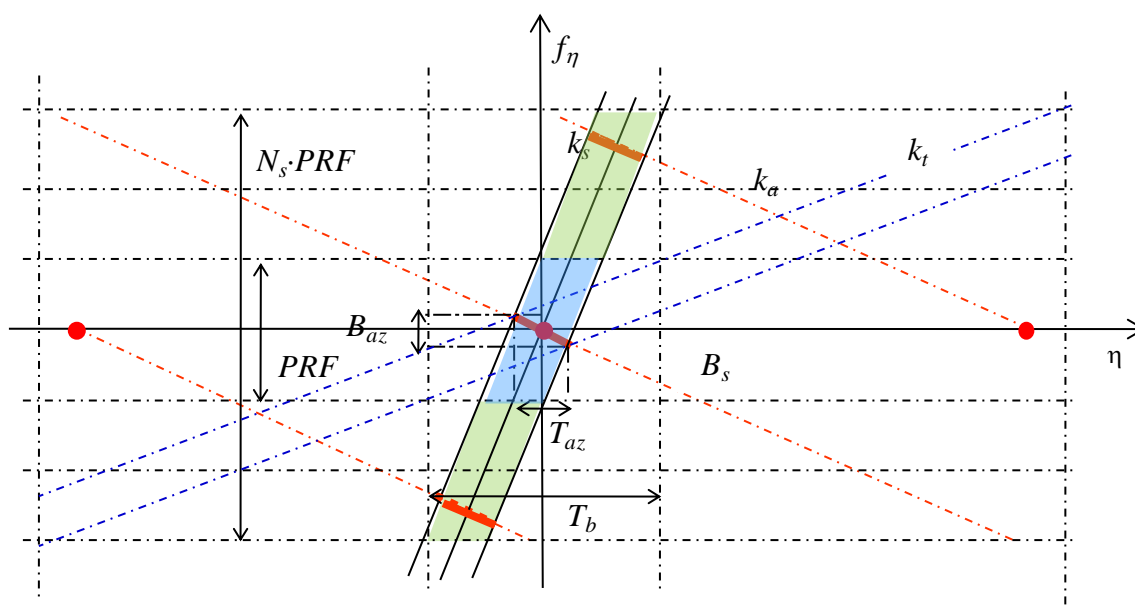


Figure 6-10 Time-frequency Diagram of a TOPSAR Azimuth Line after Frequency Unfolding (without re-sampling)

6.3 Azimuth Processing

The algorithm implemented in the Sentinel-1 IPF for azimuth processing is the Range-Doppler Algorithm (RDA), which is characterised by a hyperbolic range equation (valid over the duration of an azimuth block or burst):

$$R^2(\eta) = R_0^2 + V_r^2 \eta^2 \quad (6-26)$$

where:

- R_0 = Range of closest approach (i.e. the slant range when the radar is closest to the target)
- η = Azimuth time (or slow time). See Section 9.23 regarding bi-static delay correction of the azimuth time.
- V_r = Effective radar velocity

Note that even though the range equation (6-26) appears to assume a straight orbit, this is not the case since V_r is not the satellite velocity, but rather a pseudo-velocity, selected so that the hyperbolic range equation models the actual range equation in the general case of a curved (orbit and Earth) geometry (see more details in Section 9.10).

Note also that the algorithm updates the effective radar velocity and implicitly the reference phase with range (see Section 6.3.4.1.2 Step 4).

The RDA approach in the Sentinel-1 IPF is an enhanced version of the RDA algorithm used in the ENVISAT ASAR processor on which the Sentinel-1 IPF is based. For Sentinel-1, the IPF uses a higher accuracy secondary range compression (SRC) method (see Section 6.3.1) in order to satisfy the phase preservation requirements for Sentinel-1, in particular for TOPSAR processing. The ENVISAT ASAR processor used a simpler SRC implementation, in which the dependency of the filter on azimuth frequency was not considered. Also, the phase of the azimuth matched filter used in the IPF (equation (6-35)) is more accurate than the quadratic phase of the matched filter used in ENVISAT ASAR.

Note that the accuracy of the SRC algorithm proposed is similar with the accuracy of the SRC performed in the Space Segment (S/S) prototype (Document R-2), as part of its generalized omega-k azimuth compression block.

Steps 1 and 2 of the RDA algorithm (listed below) are applied on a range segment-by-segment basis (a segment is a configurable portion of a range line for which some range varying parameters are kept constant) and for each range line, while step 3 is applied azimuth line by azimuth line:

1. **Secondary Range Compression (SRC)**, as described in Section 6.3.1
2. **Range Cell Migration Correction (RCMC)**, as described in Section 6.3.2
3. **Azimuth compression**, as described in Section 6.3.4

6.3.1 Secondary Range Compression (SRC)

This step takes as input an azimuth block of data represented in the range time by azimuth frequency domain that is an output of the following step:

- For the Stripmap modes, the previous processing step is the azimuth FFT (described in Section 6.2.2).
- For the TOPSAR modes, the previous processing step is the frequency UFR (described in Section 6.2.4).

The SRC algorithm is applied to each range line on a range segment-by-segment basis.

The main steps of the SRC algorithm are:

1. **Range forward FFT the current segment.**
2. **Generate the SRC filter for the current segment**, as described in Section 6.3.1.1.
3. **Apply the SRC filter to the range line segment.**
4. **Range inverse FFT.**

Analysis performed for assessing the azimuth processing block in the Space Segment algorithm indicate that a segment length of up to 10 km (in ground range) is sufficient for insuring phase continuity in the range direction (see [R-2] Section 4.7.2.1). Aresys's assessment indicates that the conceptual similarities between the IPF and the Space Segment azimuth processing algorithms ensure the feasibility of a similarly sized segment for the IPF.

The number and length of the range segments for the purpose of the SRC are configurable input parameters.

6.3.1.1 SRC Filter

From the range-Doppler representation of the signal of a point target before range compression (see for example [R-5], Section 6.4), one can see that the range compression with a matched filter defined by the range chirp FM rate, K_r , does not perfectly focus the data. This is true especially for the case of higher squint angles. In this case, due to range-azimuth coupling, the range matched filter in the range-Doppler domain should have an FM rate of K_m instead of the original value of K_r , where K_m is defined by:

$$K_m = \frac{K_r}{1 - K_r / K_{src}} \quad (6-27)$$

$$K_{src}(R, f_\eta) = \frac{2V_r^2 f_0^3 D^2(f_\eta, V_r)}{cR f_\eta^2} \quad (6-28)$$

$$D(f_\eta, V_r) = \sqrt{1 - \frac{c^2 f_\eta^2}{4V_r^2 f_0^2}} \quad (6-29)$$

Note that $D(f_\eta, V_r)$ can be interpreted as the cosine of the instantaneous squint angle.

The secondary range compression is performed in the 2D frequency domain in the following way:

Since range compression has already been done using an FM rate of K_r , a filter can be applied to compensate for the difference. This filter – the SRC filter - also has a linear FM rate given by K_{src} where (see [R-5], Section 6.4):

$$H_{src}(f_\tau) = \exp\left(-j\pi \frac{f_\tau^2}{K_{src}(R, f_\eta)}\right) \quad (6-30)$$

[NORM] Multiply the filter by $\sqrt{\frac{\tilde{N}_{ifft}}{\tilde{N}_{fft}}}$ to account for the change in energy caused by

the time domain resampling which results when forward and inverse FFT sizes are different.

Note that:

- The SRC filter depends on the range frequency, f_τ .
- The SRC filter depends on the azimuth frequency, f_η through K_{src} . To account for this dependency, the filter must be updated for every range line.
- The SRC filter also depends on the slant range to the target, R , but since the data is in the 2D frequency domain, R is fixed. Therefore, the filter is assumed to be independent of range. However, the loss in accuracy due to this assumption can be mitigated by performing the SRC filtering per range segments and updating the SRC filter for each segment (i.e. regenerating it with an R = the slant range to the middle of the segment).

6.3.2 Range Cell Migration Correction (RCMC)

The range cell migration (RCM) in the range-Doppler domain is defined by the following equation (see [R-5], Section 5.3):

$$R(f_\eta) = \frac{R_0}{D(f_\eta, V_r)} \quad (6-31)$$

where $D(f_\eta, V_r)$ is defined by equation (6-29). The range migration is then equal to

$$RCMC(f_\eta, R_0) = [R(f_\eta) - R_0] \quad (6-32)$$

The slant range R_0 is varied between the slant range of the first range cell, R_{start} , and the slant range of the last range cell R_{stop} (with the increment $F_r \cdot c / 2$) and for each range cell the corresponding range cell migration is calculated using Equation 6-32.

The RCMC is implemented using a sinc-based range interpolation as described in Section 9.22.

6.3.3 Range Resampling (TOPSAR only)

For efficiency reasons, the range resampling is combined with the RCMC step (see Section 6.3.2). To this end the following have to be ensured:

- All slant ranges of the output cells are calculated, for each sub-swath relative to the same slant range origin, taken to be the first range cell in the first sub-swath, $R_{start,0}$.
- The output range sampling step is set to:

$$\hat{F}_r = \min(F_r) \quad (6-33)$$

where the minimum is taken over all sub-swaths.

- Then the slant ranges to the output points for which the RCMC offsets are calculated:

$$R_i = R_{start,0} + \frac{i \hat{F}_r c}{2}, \quad i = 0, 1, 2, \dots \quad (6-34)$$

where the last value of i is such that R_i is the maximum possible value smaller than the slant range of the last range cell in the last swath.

6.3.4 Azimuth Compression

The azimuth compression, performed for each azimuth line, consists of the following steps:

1. **Calculate the azimuth matched filter**, as described in Section 6.3.4.1.
2. **Apply the azimuth matched filter to the azimuth line.**
3. **Azimuth inverse FFT the azimuth line.**

6.3.4.1 Azimuth Matched Filter

Since after RCMC the data is in the range-Doppler domain it is convenient and efficient to implement the azimuth matched filter in this domain.

6.3.4.1.1 Algorithm Overview

The azimuth matched filter is defined by.

$$\begin{aligned} H_{az}(f_\eta) &= W(f_\eta - f_{\eta_c}) \exp\left(j \frac{4\pi R_0 D(f_\eta, V_r) f_0}{c}\right) = \\ &= W(f_\eta - f_{\eta_c}) \exp\left(j \frac{4\pi R D^2(f_\eta, V_r) f_0}{c}\right) \end{aligned} \quad (6-35)$$

Note that for the Sentinel-1 IPF the weighting window, W , is nominally applied during post-processing as described in Section 7.1, and no weighting is applied during SLC processing (W in the above formula is set to the identity).

The azimuth matched filter is then multiplied by a number of normalisation factors (see definitions and details in Section 6.3.4.1.2 below) to obtain:

$$\tilde{H}_{az}(f_\eta) = k_{proc} \cdot \frac{H_{az}(f_\eta)}{\sqrt{E}} \quad (6-36)$$

6.3.4.1.2 Algorithm Implementation

The filter is calculated, for each azimuth line, according to the following steps:

1. **Calculate the fast time, τ** , corresponding to this line (or equivalently, range cell)

$$\tau = \tau_0 + jF_r \quad (6-37)$$

where τ_0 is the range start time, j is the azimuth line number and F_r is the range sampling frequency.

2. **Calculate the slant range for this azimuth line** (or equivalently, the slant range for the current fast time, τ)

$$R = \frac{\tau c}{2} \quad (6-38)$$

3. **Calculate the DC frequency, f_{η_c}** corresponding to this line, from the DC frequency polynomial, (calculated as described in Section 5.5).
4. **Update V_r for this line**, or equivalently, for the current fast time τ (see Section 9.10 for the definition of V_r).

5. **Calculate the azimuth frequency vector** (i.e. the discrete version of (f_η)) by sampling the interval $\left[f_{\eta_c} - \frac{F_a}{2}, f_{\eta_c} + \frac{F_a}{2}\right]$, where F_a is the azimuth sampling frequency at this stage, equal to:
 - the pulse repetition frequency, for Stripmap modes
 - the azimuth sampling frequency **after** azimuth upsampling, given by $F_a = N_s PRF$, where N_s is defined in equation (6-15), for TOPSAR modes. The azimuth frequency vector is calculated as described in Section 9.8.
6. **Calculate the discrete version of $D(f_\eta, V_r): D(f_{\eta_i}, V_r)$** , $i = 0, 1, \dots, M_{fft} - 1$.
7. **Get the filter** by calculating the complex exponential defined in Equation 6-35.
8. **[NORM] Divide the matched filter by the square root of its energy, \sqrt{E}** , where E is defined by:

$$E = \frac{1}{M_{fft}} \sum_{f=0}^{M_{fft}-1} |H(f)|^2 \quad (6-39)$$

Note that if there is no weighting applied to the matched filter then the magnitude of $H(f)$ is 1 and E is also 1.

9. **[NORM] The result is then divided by $\sqrt{\frac{\sum G_{extr}^2}{\sum G_{input}^2}}$**

where:

$\sum G_{extr}^2$ = the output energy, it is the azimuth antenna pattern squared and summed over the processed bandwidth

$\sum G_{input}^2$ = the input energy, it is the azimuth antenna pattern squared and summed over the PRF

10. **[NORM] Multiply the matched filter by a processing gain, $procGain$** , a beam and polarisation dependent, SLC-specific, configurable input parameter.

6.4 Azimuth Post-processing (TOPSAR only)

The azimuth post processing consists of performing:

1. **Azimuth time domain upsampling and resampling (UFR)**, as described in Section 6.4.1.

These two operations are performed on an azimuth line-by-line basis.

6.4.1 Azimuth Time UFR

This algorithm is conceptually the same as the frequency UFR described in section 6.2.4. A generic UFR processing block diagram, underlying both the frequency domain and the time domain, is described in Section 9.9.

This step takes as input an azimuth block of azimuth compressed data (see Section 6.3.4) represented in the 2D time domain.

The algorithm is one-dimensional in its nature, and is applied, in the time domain, to each azimuth line in a burst, for all bursts. The aim of the processing is to resolve the time aliasing due to the limited support during azimuth focusing. The effect of time aliasing can be understood looking at Figure 6-11 below, which represents a single TOPSAR azimuth line at the output of the azimuth compression step.

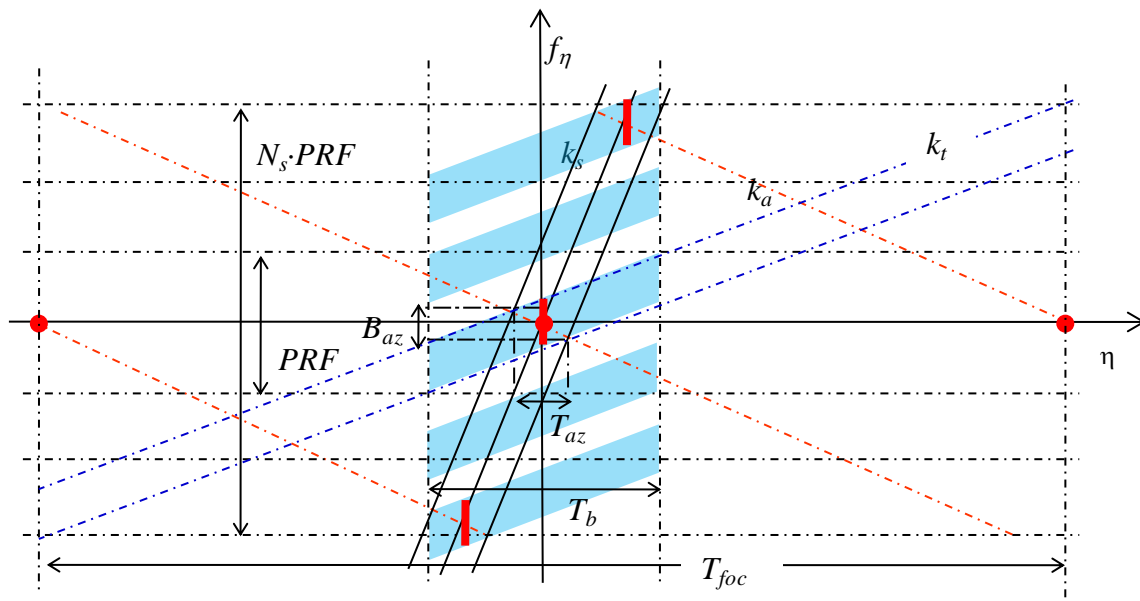


Figure 6-11 Time-frequency Diagram of a TOPSAR Azimuth Compressed Line

In addition to the notations listed in Section 6.2.4, the following notations will be used in this section:

$$\eta_c = -\frac{f_{\eta_c}}{k_a} \text{ Azimuth time corresponding to the DC frequency, } f_{\eta_c}.$$

$$\eta_{ref} = \eta_c \text{ calculated at a given reference range (mid-swath).}$$

$$T_{input} = \text{azimuth time extent of the burst input to the time UFR step (which is equal to the azimuth time extent of the burst after the frequency UFR step).}$$

$$T_{foc} = \text{the total extent of the focused data (see Equation (6-41)).}$$

For a given azimuth line in a burst, the output time support of the data is given by:

$$\eta_{ref} - \frac{T_{foc}}{2} \leq \eta \leq \eta_{ref} + \frac{T_{foc}}{2} \quad (6-40)$$

The main steps for the azimuth post-processing algorithm are as follows:

6.4.1.1 Mosaicking

The time-frequency representation of the data is folded in time domain (due to limited support during azimuth focusing). The time extent required by the signal to completely un-fold the data, T_{foc} , depends on the sweep bandwidth, B_s and on the instantaneous bandwidth B_d (both defined in Equation (6-14)) and is given by:

$$T_{foc} = -\frac{B_s - 2B_d}{k_a} + T_b \quad (6-41)$$

A number of copies of the same line, N_t , are replicated (put side by side) along the azimuth time axis.

The number of time replicas is computed by:

$$N_t = N_{t_pos} - N_{t_neg} \quad (6-42)$$

where

$$N_{t_neg} = \text{floor}\left(\frac{\eta_{ref} - T_{foc}/2}{T_{input}}\right) \quad (6-43)$$

$$N_{t_pos} = \text{ceil}\left(\frac{\eta_{ref} + T_{foc}/2}{T_{input}}\right) \quad (6-44)$$

The effect of this time domain mosaicking is represented in Figure 6-12.

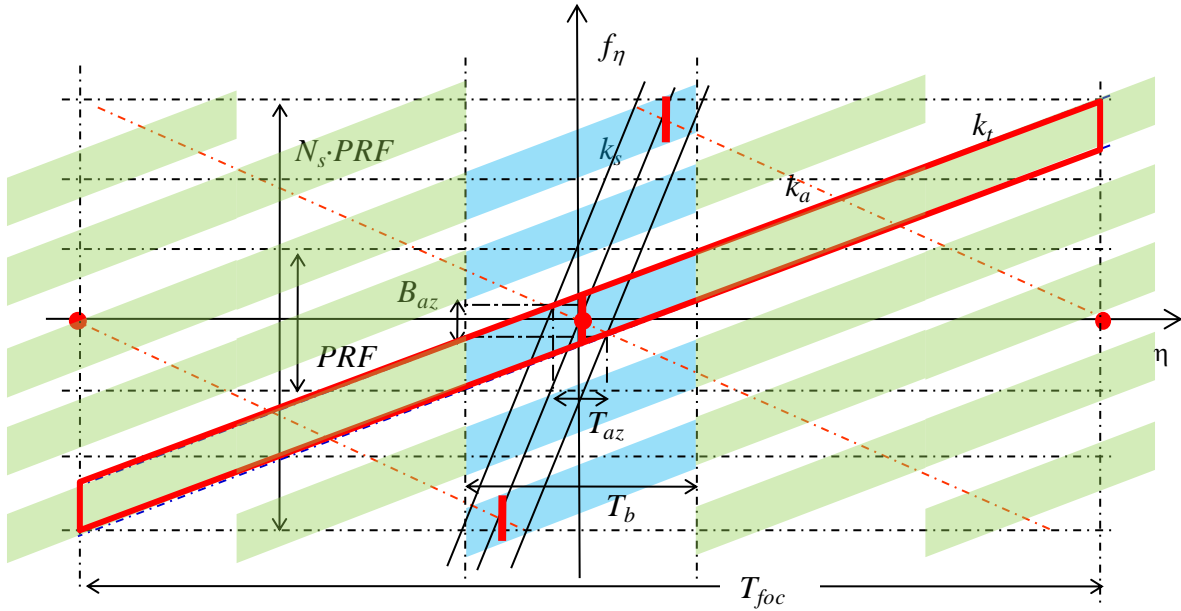


Figure 6-12 Time-frequency Diagram of a TOPSAR Azimuth Line after Time-domain Mosaicking

The time support of the data then becomes:

$$N_{t_neg} T_{input} \leq \eta \leq N_{t_pos} T_{input} \quad (6-45)$$

where T_{input} is the time extent of the data input to the mosaicking step.

6.4.1.2 De-ramping

The desired spectral components are moved to low-pass band by a multiplication in azimuth time domain with a chirp signal.

$$\phi(\eta) = \exp(-j\pi k_t \eta^2 + 2j\pi(k_t + k_a)\eta_c \eta) \quad (6-46)$$

where

$$\eta_c - \frac{T_{foc}}{2} \leq \eta \leq \eta_c + \frac{T_{foc}}{2}$$

and

$$k_t = -\frac{k_a k_s}{k_s - k_a} \quad (6-47)$$

Note that the deramp function is applied to the appropriate samples of the input data which in general, has a larger time support (as indicated in (6-42) above).

6.4.1.3 Low-pass Filtering with Re-sampling

The replication adds unwanted spectral contributions that must be filtered out. The desired spectral components are selected by a low-pass filtering, performed via a convolution in time domain with a suitable band pass filter. The maximum available Doppler bandwidth of the filter is B_{az} , defined in equation 6-7.

As in the case of frequency UFR, the low-pass filtering and resampling can be done using two methods:

1. Perform low-pass filtering and resampling in one step. This is done using Finite Impulse Response (FIR) filter that accomplishes both low-pass filtering and resampling at the same time.
2. Perform low-pass filtering and resampling in two steps. The low-pass filtering is done using FFT, zero out unwanted data and inverse FFT. The resampling is done using a quadratic interpolation kernel.

Method 1 – Low-pass filtering and resampling in one step

The optimal low pass filter is then designed with the constraint to be optimally flat in the required beam-dependent instantaneous Doppler bandwidth B_d ($B_d \leq B_{az}$).

The maximum transition bandwidth of the filter is then:

$$B_{tr} = B_{az} - B_d$$

The filter characteristics are shown in Figure 6-13. This plot represents half the filter bandwidth.

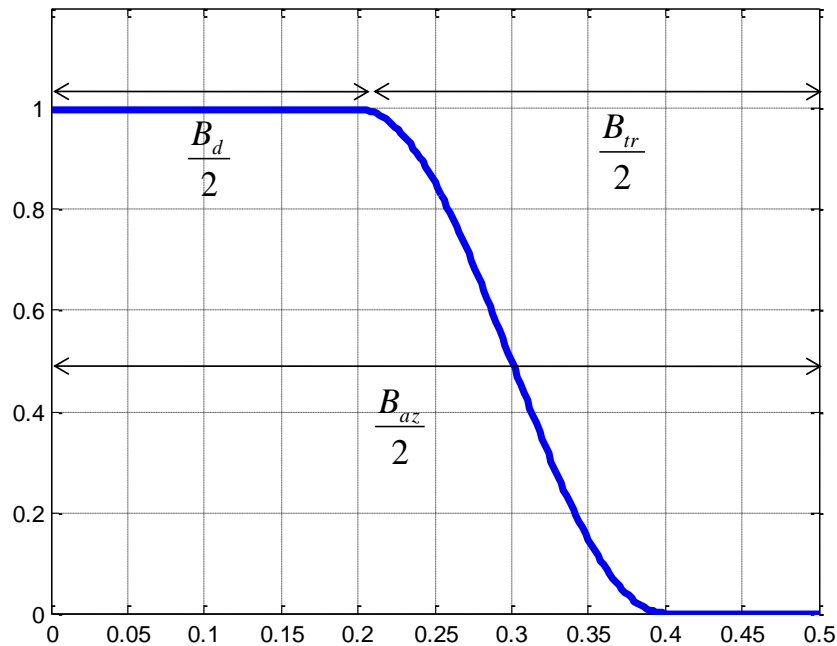


Figure 6-13 Low-pass Filter Mask for Azimuth Time UFR

The low pass filtering described above can be combined with a decimation step or, can be implemented as a generic re-sampling step as described in Section 6.2.4 for the frequency unfolding and re-sampling. The output time axis sampling step can be increased up to:

$$\Delta t_{\max} = \frac{1}{B_{az}} \quad (6-48)$$

For efficiency considerations, the output time sampling step should be chosen as large as possible. Given that the constraint on the maximum time step is respected, this processing step introduces a ‘controlled’ frequency domain aliasing without any impact on time domain characteristics of the output SLC burst. In any case, each further processing step to be carried out on the SLC burst, must take into account the effect of this induced ‘aliasing’. This effect is shown in a dedicated figure in the next section, and shows the time/frequency diagram of the output SLC data.

Due to non-ideal response of the FIR filter the IPF output products suffer from azimuth ambiguity and normalization problems. As shown in Figure 6-13 the FIR filter response has transition band through which unwanted energy is not filtered out. This unwanted energy resulted in the azimuth ambiguity and normalization problems.

Method 2 – Low-pass filtering and resampling in two steps

As in the Frequency UFR, the low-pass filter in this method has a sharp cut-off to only pass through the desired energy. Figure 6-14 shows the low pass filter responses of Method 1 and Method 2 in the same figure. The low pass filter of Method 2 (red line) only passes the required energy and throws away the unwanted energy from the mosaicking step.

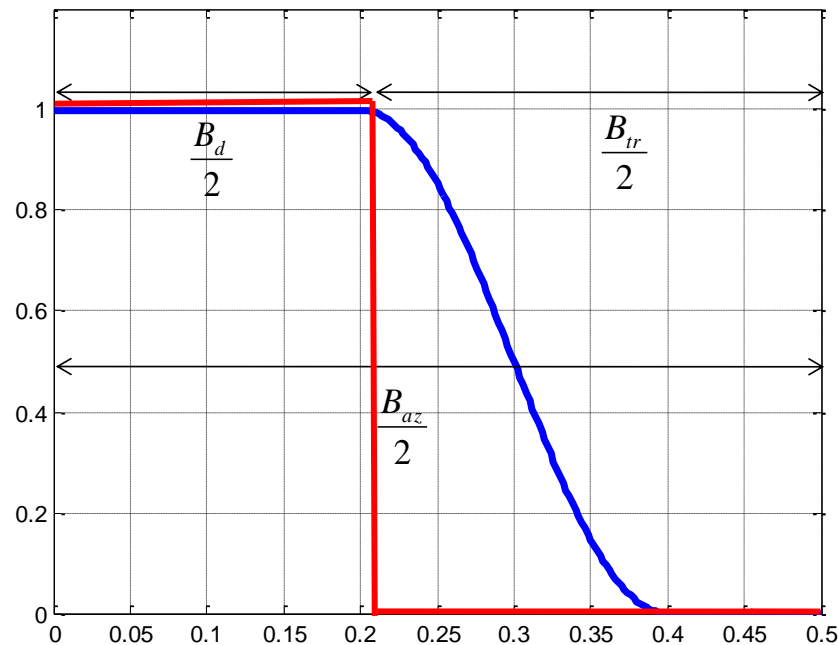


Figure 6-14 Comparison of Low-pass Filter responses for Azimuth Time UFR

The low-pass filter response has a sharp cut-off as it is done using FFT as follows:

1. Forward FFT
2. Zero out unwanted data
3. Inverse FFT

Following the low-pass filtering, the data is then resampled using quadratic interpolation.

6.4.1.4 Re-ramping

Finally, the phase removed by de-ramping (see Section 6.4.1.2) is re-instated.

$$\phi(\eta) = \exp(j\pi k_t \eta^2 - 2j\pi(k_t + k_a)\eta_c \eta) \quad (6-49)$$

where

$$\eta_{ref} - \frac{T_{foc}}{2} \leq \eta \leq \eta_{ref} + \frac{T_{foc}}{2}$$

Figure 6-15 provides the time-frequency representation of an azimuth line at the end of the time domain unfolding step (with low pass filter only).

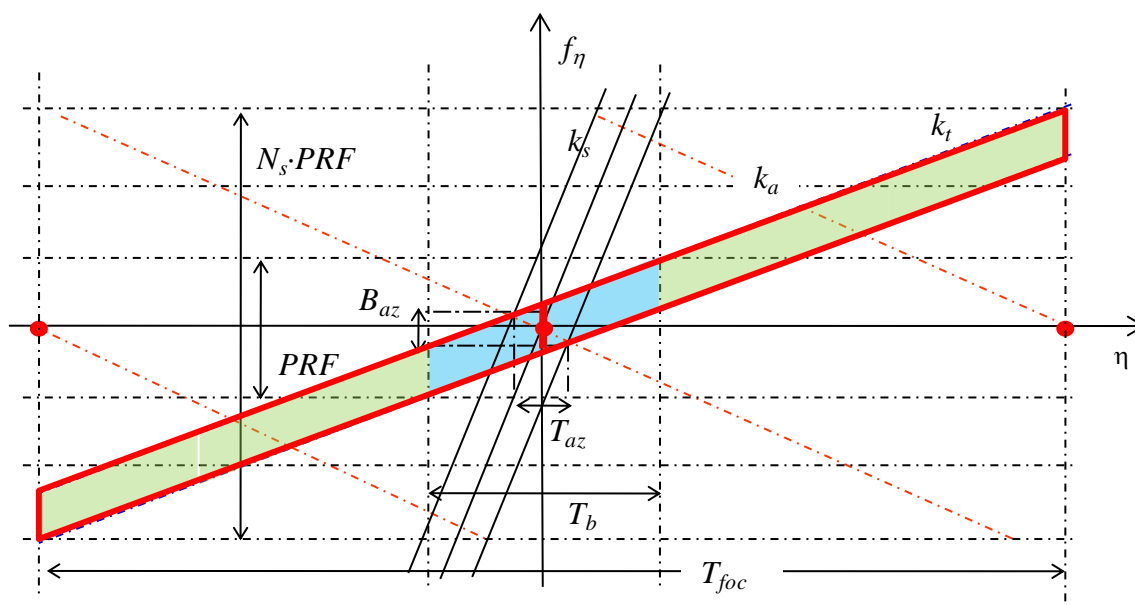


Figure 6-15 Time-frequency Diagram of a TOPSAR Azimuth Line after Time-domain Unfolding

7 POST-PROCESSING ALGORITHMS

The internal SLC product output from the SLC Processing component, generated using the algorithms described in Section 6, can be passed to the Post-Processing module for further processing. The Post-Processing module is responsible for generating both standard SLC products and ground range multi-look, detected, (GRD) images. In addition, Quick-look (QL) images can also be generated during the L1 post-processing.

The algorithm steps used within the Post-Processing module can be grouped as follows:

1. **Post-processing range processing**, as described in Section 7.1
2. **Post-processing azimuth processing**, as described in Section 7.2
3. **Post-processing output processing**, as described in Section 7.3

Figure 7-1 presents a high level block diagram of the L1 post-processing, and Figure 7-2 presents a more detailed block diagram of the three main processing stages and the way they inter-connect.

Note that throughout this section, ‘azimuth block’ will be used in a generic way to also designate bursts (for TOPSAR) or vignettes (for Wave Mode).

The range and azimuth processing steps are repeated for each azimuth block in the chronological order of acquisition (the number of sub-swaths is 1 for Stripmap and Wave modes). In particular for TOPSAR this means that the burst are processed one after another, cycle by cycles (in general, in parallel by multiple processors).

For Stripmap GRD processing, a small azimuth overlap between azimuth blocks, $T_{grdBlkOv}$, is necessary in order to avoid the edge effects induced by the subsequent azimuth multi-look filtering stage and to allow for the throwaway of incompletely azimuth interpolated cells. Therefore, $T_{grdBlkOv}$ can be written:

$$T_{grdBlkOv} = T_{ml} + T_{intKer} \quad (7-1)$$

where:

T_{ml} = Azimuth multi-looking throwaway

T_{intKer} = Azimuth interpolation throwaway

The azimuth SM GRD block size is a configurable input parameter.

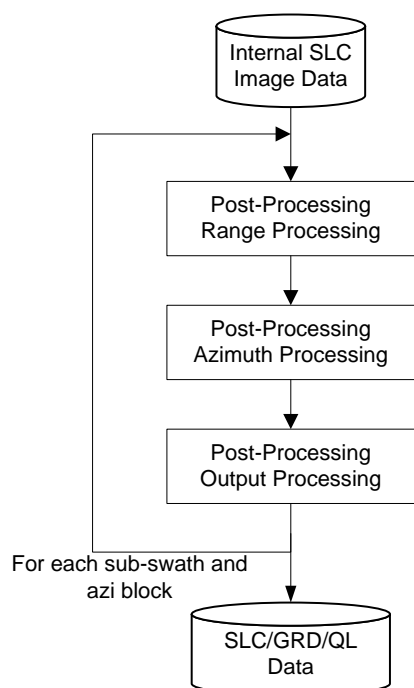


Figure 7-1 L1 Post-Processing Algorithm (High Level View)

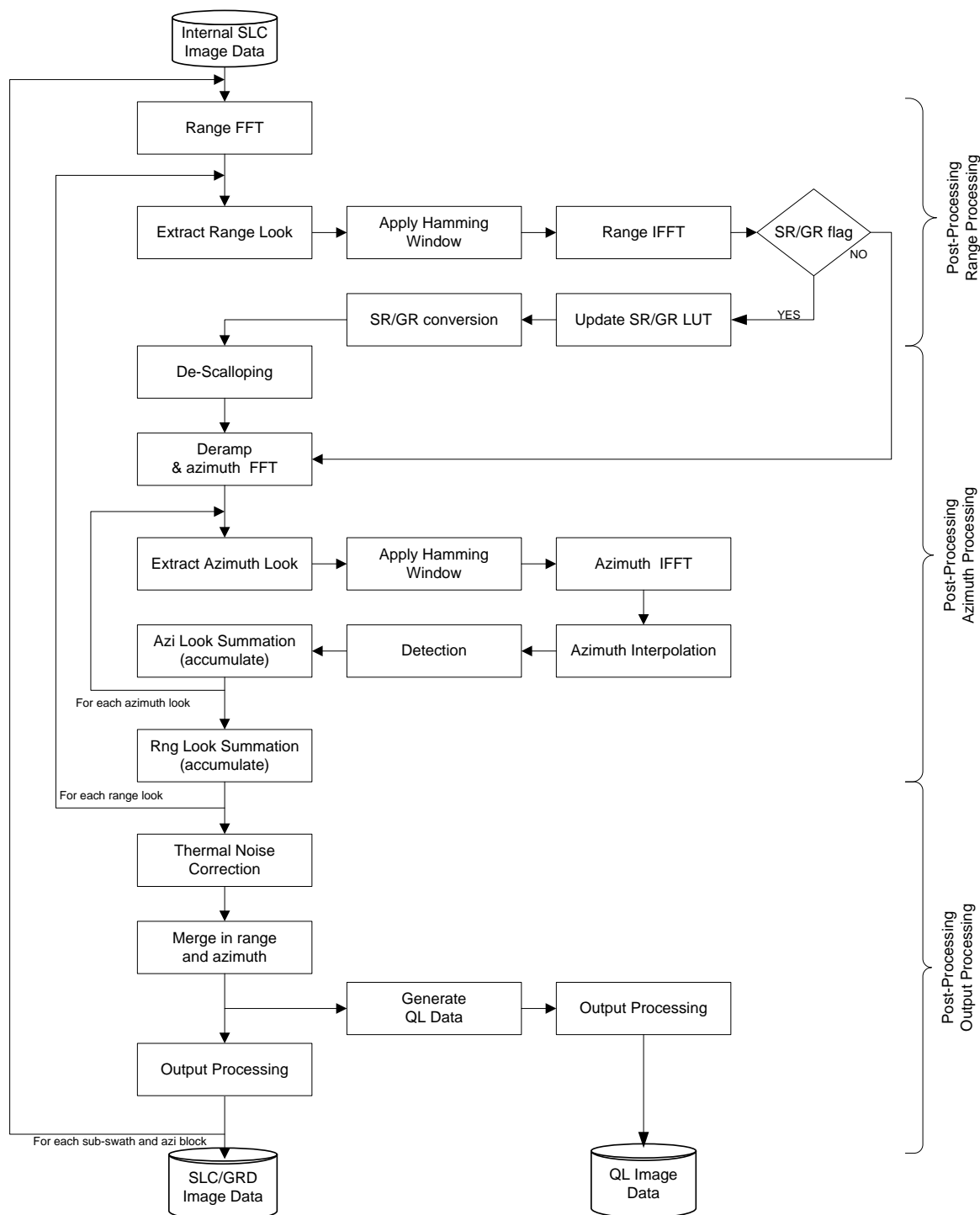


Figure 7-2 L1 Post-Processing Algorithm (Detailed View)

7.1 Post-Processing Range Processing

The post-processing range processing consists of the following operations, to be performed for each range line in the current block and sub-swath:

1. **Zero pad if necessary and perform range forward FFT of length \tilde{N}_{fft} .**
2. **Perform look extraction.**

The specified number of range looks, of length \tilde{N}_{fft} , are extracted from the range spectrum of the data (see Section 9.21). To effectively exploit the full information in the input SLC image data, the whole frequency bandwidth will be used. However, for more flexibility, the range processing bandwidth to be used is a configurable input parameter. The number of range looks is a configurable input parameter. When the output product is SLC, the number of range looks will be set to 1.

3. **Generate a range look Hamming window of length \tilde{N}_{fft} .** The window coefficient is a configurable, input parameter,

4. **[NORM] Scale the range weighting window by $\frac{1}{\sqrt{\sum W_r^2}}$ where**

W_r = the range weighting window.

5. **[NORM] Multiply the window by $\sqrt{\frac{\tilde{N}_{fft}}{\tilde{N}_{fft}}}$ to account for the change in**

energy caused by the time domain resampling which results when forward and inverse FFT sizes are different.

6. **Apply range Hamming window to each extracted look.**
7. **Range inverse FFT each extracted look.**

Since the SLC data input to range multi-looking is sampled on a common range grid of spacing Δs_{in} (for Stripmap, this is the natural slant range sampling spacing, for TOPSAR, this is obtained as described in Section 6.3.3) the data resulting after the range inverse FFT will also lie on a common range grid with the sample spacing:

$$\Delta R_{out} = \Delta R_{in} \cdot \frac{\tilde{N}_{fft}}{\tilde{N}_{fft}} \quad (7-2)$$

Note that after this step, the slant range to the first valid sample of the common grid becomes:

$$\tilde{R}_{start,0} = R_{start,0} + \Delta R_{throwaway} \quad (7-3)$$

where:

$R_{start,0}$ = Slant range to first sample of the input SLC data

$\Delta R_{throwaway}$ = Throwaway to account for the edge effects of the range filtering

8. **If conversion to ground range is required, calculate the GR/SR LUT for this line.**

The slant-range-to-ground-range conversion can be controlled by a configurable, input parameter.

The derivation and updating strategy of the GR/SR LUT is described in Section 9.16.

9. **If conversion to ground range is required, perform slant-range-to-ground-range conversion at the desired range pixel spacing.**

The slant-range-to-ground-range conversion is a range interpolation operation, performed using a sinc-based interpolator as described in Section 9.22.

7.2 Post-Processing Azimuth Processing

The post-processing azimuth processing consists of a number of steps (steps 1 to 13) to be performed for each azimuth line, for each range look, for each azimuth block and sub-swath. Finally, in step 14, the azimuth processed range looks are summed together:

1. **For TOPSAR only, perform de-scalloping**, as described in Section 7.2.1.
2. **For TOPSAR only, de-ramp the Doppler spectrum.** This de-ramping is achieved by applying the chirp signal $d(\eta)$ defined in Section 5.2.1.
3. **Perform azimuth forward FFT of length \tilde{M}_{fft} .**
4. **Perform Look extraction.**

The position of the centre of the spectrum must be determined first based on the Doppler centroid frequency corresponding to the current azimuth line. The azimuth looks, of length \tilde{M}_{iff} , are then extracted symmetrically with respect to the spectrum centre, based on the number of looks, the processing azimuth bandwidth and the look bandwidth, which are configurable input parameters (see Section 9.21).

5. **Calculate the azimuth antenna pattern correction.**

The antenna pattern correction, $G_{apc}(f_\eta)$, is calculated as described in Section 9.15.

6. **Generate the look azimuth Hamming window of length \tilde{M}_{iff} and multiply to the azimuth antenna pattern correction.** The window coefficient is a configurable, input parameter.

7. **[NORM] Scale the azimuth weighting window by** $\sqrt{\frac{\sum G_{aap}^2(f_\eta)}{\sum W_a^2}}$ where

$G_{aap}(f_\eta)$ = azimuth antenna pattern over the extracted look.

W_a = the azimuth weighting window.

8. **[NORM] Multiply the azimuth window by** $\sqrt{\frac{\tilde{M}_{fft}}{\tilde{M}_{ffr}}}$ to account for the change in energy caused by the time domain resampling which results when forward and inverse FFT sizes are different.)

9. **Apply the window to each azimuth look.**

10. **Inverse FFT each extracted look.**

The azimuth spacing of the data after this step is related to the azimuth input spacing via:

$$\Delta s_{out} = \Delta s_{in} \cdot \frac{\tilde{M}_{fft}}{\tilde{M}_{ffr}} \quad (7-4)$$

11. **Apply azimuth interpolation** to resample at the desired azimuth output spacing, which is a configurable, input parameter (GRD only).

The interpolation is performed using a sinc-based interpolator as described in Section 9.22.

12. **Detect each azimuth look** (GRD only).

The detection operation simply calculates the absolute value squared (i.e. the ‘power’) of each complex sample in an image.

13. **Perform azimuth look summation** of the looks (by accumulating to previous look).

14. **Perform range look summation.**

7.2.1 De-scalloping

In TOPSAR modes, the steering of the antenna beam in the azimuth direction causes the weighting of the echoes to vary in the azimuth direction, inducing a scalloping effect: the area at the centre of the burst is brighter than the ones at burst edges.

The azimuth scalloping for TOPSAR is shaped by the azimuth pattern of the element antennas that compose the antenna array. The scalloping extent depends on the maximum antenna steering angle within the burst and therefore it is sub-swath-dependent. The scalloping effect however, does not change from burst to burst within the same sub-swath.

The figure below shows an example of scalloping for the IW mode.

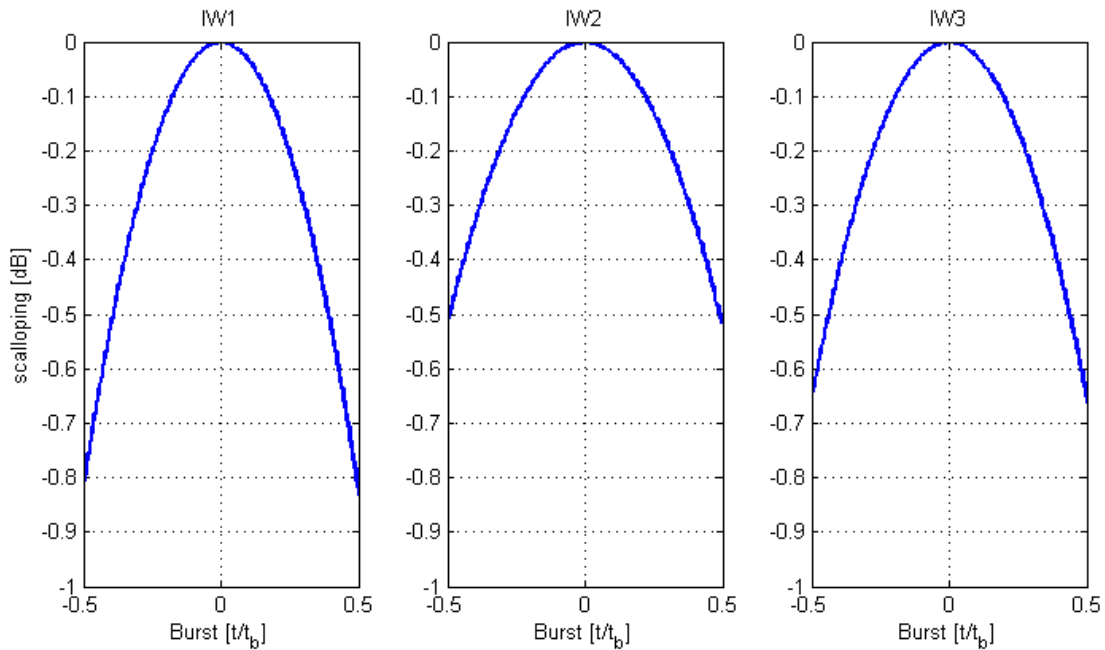


Figure 7-3 Scalloping Gain versus Focused Burst Azimuth Time for IW Mode

The de-scalloping algorithm, which is performed on 2D-time domain data, consists of the following 2 steps, to be applied to each azimuth line in each burst:

1. **Calculate the de-scalloping function** as a function of focused burst time as described in Section 7.2.1.1.
2. **De-scallop the azimuth data line** as described in Section 7.2.1.2.

7.2.1.1 De-scalloping Function

The de-scalloping function is defined by the (2-way) azimuth antenna element pattern (AAEP). The AAEP, as a function of antenna steering angle ψ , can be modeled by:

$$G(\psi) = \text{sinc}^2\left(\frac{L_{el}}{\lambda} \sin(\psi)\right) \quad (7-5)$$

where:

λ = radar wavelength

L_{el} = is the antenna element length, $L_{el} = L / N_{az}$,

L = antenna length

N_{az} = number of antenna columns ($N_{az}=14$ for Sentinel-1).

However, the IPF uses measured antenna element patterns, which in general can deviate from the model above. The Sentinel-1 antenna is composed of a total number of 14x20 antenna elements, each with its own element pattern. Generally, the pattern varies depending on the position of the element within the antenna (due to e.g. manufacturing differences between elements or coupling).

For de-scalloping, the average of all these antenna element azimuth patterns is of interest. Thus, the average azimuth element pattern is provided as input LUTs to the IPF through the input auxiliary files.

The de-scalloping as a function of the azimuth time η can be derived from the following input parameters:

- AAEP LUT (mapping azimuth steering angles to gain power, in dB)
- λ = radar wavelength
- V_s = satellite velocity
- k_t = azimuth frequency rate defined in Equation (6-47, calculated at mid-range).

Note that k_t depends on the slant range (through its dependency on the azimuth FM rate, k_a). As a consequence, the de-scalloping function also depends on the slant range. However, for each swath, the variation of the de-scalloping gain from near to far range is less than 0.05 dB. Therefore the IPF will use a de-scalloping function calculated at mid-range as this approximation will introduce an error of less than 0.025 dB (safely less than the relative radiometric accuracy requirement of 0.1 dB).

To calculate the de-scalloping function, $G_{ds}(\eta)$, as a function of focused burst azimuth time, η_i , perform the following:

1. Calculate the corresponding steering angle:

$$\psi(\eta_i) = \frac{\lambda}{2V_s} k_t \eta_i. \quad (7-6)$$

Note that the $k_t \eta_i$ factor represents the instantaneous Doppler frequency corresponding to the zero-Doppler time, η_i .

2. Identify the nearest two angles in the AAEP LUT, ψ_1 and ψ_2 such that:

$$\psi_1 \leq \psi(\eta_i) \leq \psi_2.$$

3. Compute the gain (in dB) $\tilde{G}_{ds}(\eta_i)$ for the current time sample η_i from the AAEP LUT values corresponding to ψ_1 and ψ_2 by linear interpolation:

4. Convert to linear scale:

$$G_{ds}(\eta_i) = \text{pow}\left(10, \frac{\tilde{G}_{ds}(\eta_i)}{20}\right) \quad (7-7)$$

7.2.1.2 De-scalloping Azimuth Data Line

The de-scalloped azimuth data line $X_1(\eta_i)$ is obtained by multiplying the focused azimuth line $X(\eta_i)$, by the reciprocal of the gain $G_{ds}(\eta_i)$:

$$X_1(\eta_i) = X(\eta_i)(G_{ds}(\eta_i))^{-1}. \quad (7-8)$$

7.3 Post-Processing Output Processing

The GRD post-processing consists of the following steps:

1. **If requested, remove the thermal noise** as described in Section 7.3.1.
2. **Merge the blocks** as described in Section 7.3.2.
3. **Generate Quick-look (QL)** as described in Section 7.3.3.
4. **Perform output processing** as described in Section 7.3.4.

7.3.1 Thermal Noise Removal (GRD Only)

In many detected SAR satellite images, the presence of additive noise can be noticed, especially in areas of low backscatter (like calm sea, lakes, etc.). Unlike quantization noise, which is dependent upon the signal power itself, the thermal noise can hardly be noticed, and becomes relevant only where the signal mean is low. Moreover, in multi-swath acquisition modes this noise has typically a different intensity in each sub-swath, causing an intensity step at inter-swath boundaries. During raw data focusing, data (including the noise contributions) are multiplied by several fast time-varying radiometric correction factors. The result is that noise contributions are re-shaped in a range-varying fashion.

The thermal noise level can be calculated as described in Section 9.17. According to Equation 9-44, the thermal noise removal is performed by simply subtracting the noise from the power detected image, i.e., with the notations from Section 9.17:

$$E[s(R, \eta)^2] = E[(s(R, \eta) + n(n_s; \eta))^2] \cdot G_{tot}^2(n_s; R, \eta) - \sigma_n^2(n_s, \eta) \cdot G_{tot}^2(n_s; R, \eta) \quad (7-9)$$

The thermal noise level vector corresponding to any given azimuth time will be obtained by linear interpolation between the two closest noise vectors in the bank.

The noise removal algorithm is strictly one-dimensional: for each range line, the correspondent noise vector, derived from the bank by linear interpolation, is subtracted from the power detected line.

7.3.2 Burst Merging (TOPSAR GRD Only)

The burst are merged first in the range direction and then in the azimuth direction.

7.3.2.1 Range Direction Merging

The merging of the sub-swaths is performed on a line-by-line basis, using the lines' azimuth zero-Doppler time tag. This approach is possible due to the fact that all of the bursts in all beams have already been resampled to a common grid (during azimuth post-processing, as described in Section 6.4).

One can assume without loss of generality that the line with the earliest time tag belongs to the first sub-swath. Then, the algorithm consists of the following steps:

- For each line in the first sub-swath, get its time tag, η_i
 - For each subsequent sub-swath
 - Identify the line with the same time tag, η_i .
 - If no line exists, and this is the first time cycle, fill with black pixels and continue.
 - If no line exists but this is not the first cycle, log and error message and exit.
 - If a line with the same time tag was found, merge to the line of time tag η_i in the previous sub-swath.
 - end
- end

For merging the sub-swaths, the optimal cut is determined taking half the overlap between them, considering only the valid samples of each line. Samples from 2 consecutive sub-swaths are put side-to-side according to the optimal cut position (without performing sample 'blending').

7.3.2.2 Azimuth Direction Merging

The merging is based on the azimuth time tag of the lines and the calculated overlap. Note that only a relatively small fraction of samples overlap since both the IW and the EW modes have one natural azimuth look.

To merge two consecutive bursts (the 'early' and the 'late' burst):

- Let $t_{zd,early,i}^{last}$ be the zero Doppler time of range cell i in the last line of the ‘early’ burst.
- Let $t_{zd,late,i}^{first}$ be the zero Doppler time of range cell i in the first line of the ‘late’ burst.
- With these notations, the azimuth merging time for range cell i is set to half-way between $t_{zd,early,i}^{last}$ and:

$$t_{merg,i} = \frac{t_{zd,early,i}^{last} + t_{zd,late,i}^{first}}{2} \quad (7-10)$$

- For each range cell i , the merging time, $t_{merg,i}$, is quantized to the nearest output azimuth cell.

7.3.3 Quick-Look (QL) Image Generation

Quick-Look (QL) images are low-resolution images that accompany each of the SLC and GRD products and are intended as a handy reference to the main image.

The QL images are produced one azimuth block at a time, by basic sample averaging and decimation, in both azimuth and range directions. The number of averaging samples in range and azimuth as well as the range and azimuth decimation factors are configurable, input parameters.

Note also that for dual-polarisation data, the QL image resulting from the two polarisation processing loops is written to the same tiff file. And, during the second polarisation loop, the ratio between homologous pixel values in channels 1 and channel 2 is also recorded in the file.

In the case where the output image is a TOPSAR SLC product (where the bursts and swaths are not merged), the QL image requires the burst/swath merging described in Section 7.3.2.

The QL data is subject to output processing in the same way the GRD image itself is processed for output (see Section 7.3.4) with the difference that in this case the output pixel type is 8 bit.

The QL image generation can be enabled or disabled by configurable input parameters.

7.3.4 Output Processing

The last three steps of the post-processing are:

1. **Square root extraction** (GRD only)

2. **Scaling of the data by the application LUT** as described in Section 9.18
3. **Conversion to output pixel type** (typically 16-bit for SLC/GRD images and 8-bit for QL images)

The first and last steps are controlled by configurable input parameters.

8 SLICING SUPPORT

The purpose of this section is to discuss slicing and the affects it has on various aspects of Sentinel-1 processing. The section restates the slice description provided in Section 5.1 of [R-14], which describes slicing from an external perspective outside the IPF, and then discusses additional details about the implications of slicing internally for the IPF processors.

1. **The L0 input slice definition** as described in Section 8.1
2. **The internal signal data slice definition** as described in Section 8.2
3. **The output slice definition** as described in Section 8.3.

The following notation conventions will be used in this section:

- ‘Capital T ’ notations represent azimuth time intervals
- ‘Small t ’ notations represent satellite absolute (azimuth) time values
- ‘Small t_{zd} ’ notations represent Zero-Doppler absolute (azimuth) time values
- ‘Tilde’ \tilde{t} and \tilde{T} notations represent expected values with respect to the L0 data as per the job order; for example:

\tilde{T}_{xxx} is an expected time interval

\tilde{t}_{xx}^{start} is an expected start acquisition time (of segment, slice) within the input L0 data

- ‘Simple’ t and T notation represent values with respect to the internal slices, input to the DCE/SLC processing and obtained by imposing a block-alignment constraint; for example:

T_{xxx} is a time interval relevant to the internal processing

t_{xx}^{start} is a start acquisition time of a segment, slice, block etc. input to the DCE/SLC processing (and aligned with azimuth processing blocks)

8.1 L0 Input Slice Definition

The L0 slices are defined by the following parameters provided in the job order:

- The sensing start time of the segment, \tilde{t}^{start}
- The sensing stop time of the segment, \tilde{t}^{stop}
- The slice length, \tilde{T}_{slice}

- The slice overlap, $\tilde{T}_{overlap}$
- Slice number, i
- Total number of slices, N_{slices}

With these notations, the minimum start and stop time extent of each L0 slice is:

$$\tilde{t}_i^{start} = \tilde{t}^{start} + (i-1)\tilde{T}_{slice}, \quad i = 1, \dots, N_{slices} \quad (8-1)$$

$$\tilde{t}_i^{stop} = \tilde{t}_{i+1}^{start} + \tilde{T}_{overlap}, \quad i = 1, \dots, N_{slices} - 1 \quad (8-2)$$

$\tilde{t}_i^{stop} = \tilde{t}^{stop}$, $i = N_{slices}$ (8-3) Note that each IPF instance can calculate the start and stop time of every slice; in particular its own start/stop time is obtained for $i = n$. Also note that the L0 data provided as input to the IPF for slice i can start before \tilde{t}_i^{start} and extend beyond \tilde{t}_i^{stop} , but the IPF will only process the data between those two extents for that slice. The number of slices required to process a given segment can be computed as:

$$\begin{aligned} N_{slices} &= (\tilde{t}^{stop} - \tilde{t}^{start} - \tilde{T}_{overlap}) / \tilde{T}_{slice} \\ \tilde{T}_{lastSlice} &= \text{frac}(N_{slices}) * \tilde{T}_{slice} \\ N_{slices} &\equiv \begin{cases} \text{floor}(N_{slices}) & \tilde{T}_{lastSlice} < \tilde{T}_{overlap} \\ \text{round}(N_{slices}) & \tilde{T}_{lastSlice} \geq \tilde{T}_{overlap} \end{cases} \end{aligned} \quad (8-4)$$

Note that this calculation enforces the constraint that the length of the last slice must be at least two overlaps long. If the last slice is shorter than this, then the last short slice is combined with the penultimate slice and the total number of slices is reduced.

8.2 Internal Signal Data Slice Definition

For reasons of continuity of the final concatenated slices, the internal signal data slices will be required to be aligned with:

- for Stripmap, the input azimuth processing blocks
- for TOPSAR, the burst cycle time

Therefore the L0 input slices cannot be used in their integrity and internal signal data slices have to be defined as subsets of the L0 slices:

1. The **Stripmap case** is presented in Section 8.2.1.
2. The **TOPSAR case** is presented in Section 8.2.2.

8.2.1 Stripmap Case

In the Stripmap case the signal data must be aligned to azimuth processing blocks, so the internal input signal data slices will depend on:

- The length of the of the azimuth processing blocks. The length of the blocks in seconds, T_{blk} (possibly with the exception of the very last block in the segment) is given by:

$$T_{blk} = \frac{M_{fft}}{F_a} \quad (8-5)$$

where M_{fft} is the azimuth FFT length and is a configurable input parameters.

- The overlap of the azimuth processing blocks. The azimuth block overlap, T_{blkOv} can be calculated as described in Section 9.12 and is provided as a configurable input parameter. Note that the block overlap is common to all slices.
- A time origin common (and known) to all slices, t^{start} . In particular this origin will be set to the sensing time of the first echo line in the segment after the job order sensing start time parameter, \tilde{t}^{start} .

As a consequence of the alignment of the slices with the blocks, the start time of any azimuth block in the entire segment is known to all IPF instances and can be calculated using:

$$t_{blk,p}^{start} = t^{start} + p(T_{blk} - T_{blkOv}) \quad p = 0,1,2,... \quad (8-6)$$

The sensing start and stop time of the internal input slices can be calculated as follows:

- Calculate the absolute block number of the first block in the slice, p_i^{start} :

$$p_i^{start} = \text{ceil}[(\tilde{t}_i^{start} - t^{start}) / (T_{blk} - T_{blkOv})] \quad (8-7)$$

- Calculate the acquisition start time of the slice:

$$t_i^{start} = t^{start} + p_i^{start}(T_{blk} - T_{blkOv}) \quad (8-8)$$

- Calculate the number of blocks in the slice:

$$N_{blocks,i} = \text{floor}[(\tilde{t}_i^{stop} - t_i^{start} - T_{blkOv}) / (T_{blk} - T_{blkOv})] \quad (8-9)$$

- Calculate the acquisition stop time of the slice:

$$t_i^{stop} = t_i^{start} + N_{blocks,i}(T_{blk} - T_{blkOv}) + T_{blkOv} - \frac{1}{F_a} \quad (8-10)$$

Note that if $N_{blocks,i} \leq 0$ it means that the input L0 slice was too short (i.e. shorter than $(T_{blk} - T_{blkOv})$ plus a certain margin) which will result in a processing failure.

Note that owing to the alignment of slices to azimuth blocks, the overlap between consecutive internal slices will necessary be of the form:

$$T_{sliceOv} = m(T_{blk} - T_{blkOv}) + T_{blkOv} \quad (8-11)$$

where m is the number of blocks in the overlap (a positive integer).

The m parameter can be calculated from:

$$m = \frac{t_{i+1}^{start} - t_i^{end} - \frac{1}{Fa} - T_{blkOv}}{T_{blk} - T_{blkOv}} \quad (8-12)$$

8.2.2 TOPSAR Case

For reasons of continuity, the length of each internal TOPSAR signal data slice is required to be an integer number of burst cycles. Therefore the definition of the TOPSAR internal input slices will depend on:

- The burst cycle time which is defined as:

$$T_{cycle} = \sum_{k=1}^{N_{swaths}} T_{b,k} \quad (8-13)$$

where $T_{b,k}$ is the burst time of swath.

- A time origin common (and known) to all slices, t^{start} . In particular this origin will be set to the start of the first burst in the segment that falls after the job order sensing start time parameter, \tilde{t}^{start} . The swath of this first burst is then used as the starting point of every subsequent cycle within the slices of the segment.

As a consequence of the required alignment of the slices with the burst cycles, the start time of any burst cycle in the entire segment is known to all IPF instances and can be calculated using:

$$t_{cycle,p}^{start} = t^{start} + pT_{cycle} \quad p = 0,1,2,... \quad (8-14)$$

The sensing start and stop time of the internal input slices are defined by Equations 8-3 to 8-10 in which T_{blk} was replaced with T_{cycle} and T_{blkOv} was set to zero. In a similar way, the number of cycles in the slice overlap can be calculated from Equation 8-12.

8.3 Output Slice Definition

The output slice definition will depend on the imaging mode type (Stripmap or TOPSAR) and on the product type (SLC or GRD):

1. The **Stripmap SLC** case is presented in Section 8.3.1.
2. The **TOPSAR SLC** case is presented in Section 8.3.2.
3. The **Stripmap GRD** case is presented in Section 8.3.3.
4. The **TOPSAR GRD** case is presented in Section 8.3.4.

8.3.1 Stripmap SLC Case

Each output slice will be defined in such a way as to ensure seamless concatenation between slices. To this end, the start and stop time of each output slice will lie on a common grid, uniformly spaced in time at exactly one PRI.

1. Each IPF instance calculates the zero Doppler start time of the first slice (also known as the “anchor time”) as follows:

$$t_{zd,1}^{start} = t^{start} + \frac{T_{sliceOv}}{2} + \eta_{c,nom} + \frac{T_{mf}}{2} + T_{\Delta Ov} \quad (8-15)$$

where:

t^{start} = segment start time

$\eta_{c,nom}$ = Nominal DC time offset (based on the maximum DC value among all range values, calculated from orbit and attitude at t^{start})

T_{mf} = Azimuth matched filter duration (see Section 9.12.)

$T_{\Delta Ov}$ = Extra azimuth block overlap (a configurable, input parameter; see also Section 9.12)

The time $t_{zd,1}^{start}$ is further quantized to the next PRI; without loss of generality, the notation is left unchanged.

2. Each IPF instance calculates the zero Doppler start time and stop time of its output slice i as follows:

- If the output SLC slice is the final product:

$$t_{zd,i}^{start} = t_{zd,1}^{start} + (i-1)\tilde{T}_{slice}, \quad i = 2, \dots, N_{slices} \quad (8-16)$$

$$t_{zd,i}^{stop} = t_{zd,1}^{start} + i\tilde{T}_{slice}, \quad i = 1, 2, \dots, N_{slices} - 1 \quad (8-17)$$

- If the output SLC slice is further processed into a GRD product:

$t_{zd,i}^{start}$ must be decreased by the GRD specific block overlap, $T_{grdBlkOv}$

$t_{zd,i}^{stop}$ must be increased by the GRD specific block overlap, $T_{grdBlkOv}$

$T_{grdBlkOv}$ is an configurable input parameter.

3. Each IPF searches for the first azimuth block in the slice that it should process. This is done by searching which azimuth block in the current slice can generate the first output range line of the output slice i.e. the line with zero Doppler output time $t_{zd,i}^{start}$. The search is done by translating the acquisition start time of the azimuth blocks in the input slice to zero Doppler time:

$$t_{zd,blk,p}^{start} = t_{blk,p}^{start} + \frac{T_{mf}}{2} + \max(\eta_c) \quad (8-18)$$

where the maximum is taken over all range cells.

Note that due to the quantization of the slices to azimuth blocks, $t_{blk,p}^{start}$ can be easily calculated with a formula as in Equation 8-6.

The IPF found its first azimuth block p_i when $t_{zd,i}^{start}$ is between t_{zd,blk,p_i}^{start} and $t_{zd,blk,p_{i+1}}^{start}$. Thus, when outputting the data, the IPF will throw away the first $(t_{zd,blk,p_i}^{start} - t_{zd,i}^{start}) \cdot Fa$ lines of block p_i .

4. Each IPF searches for the last azimuth block in the slice that it should process using similar steps to the calculation above. The search is done by translating the acquisition start time of the azimuth blocks in the input slice to zero Doppler time:

$$t_{zd,blk,p}^{start} = t_{blk,p}^{start} - \frac{T_{mf}}{2} + \min(\eta_c) \quad (8-19)$$

where the minimum is taken over all range cells.

The IPF found its last azimuth block, q_i , when $t_{zd,i}^{stop}$ is between t_{zd,blk,q_i}^{start} and $t_{zd,blk,q_{i+1}}^{start}$. (Thus, when outputting the data, the IPF will throw away the last $(t_{zd,blk,q_i}^{start} - t_{zd,i}^{stop}) \cdot Fa$ lines of block q_i .)

8.3.2 TOPSAR SLC Case

All the TOPSAR bursts in all swaths are already resampled (irrespective to slicing) to a common output azimuth pixel spacing (see Section 6.4.1.3). To ensure a common grid across all slices, the time origin chosen for the resampling step must be the same for all slices; for convenience the time origin will be set to t^{start} . The TOPSAR slices will then be automatically aligned to the common grid.

Furthermore:

- If the final product is an SLC product, for each swath, all bursts in a slice, with the exception of the bursts in the overlap at the end of the slice, will constitute the output slice.
- If the output SLC slice is further processed into a GRD product retain one overlapping burst and discard the rest (if any).

8.3.3 Stripmap GRD Case

All the Stripmap slices are resampled (via azimuth interpolation) to a common grid in the azimuth direction. The common grid will be defined by an origin common to all slices, which for convenience can be set to $t_{zd,0}^{start}$ (see Equation 8-15), and the fixed, output azimuth pixel spacing.

The start time of the first output GRD slice will be the time $t_{zd,1,grd}^{start}$ rounded to the next integer output pixel, where:

$$t_{zd,1,grd}^{start} = t_{zd,1}^{start} + \frac{T_{grdBlkOv}}{2} \quad (8-20)$$

The start and stop start of all the slices will be defined by Equation 8-16 and 8-17, rounded to the next integer output pixel, since the extra slice overlap provided (see Section 8.3.1 Step 2) will be thrown away.

8.3.4 TOPSAR GRD Case

All the bursts in all slices are resampled (via azimuth interpolation) to a common grid in the azimuth direction. The common grid will be defined by an origin common to all slices and the fixed, output azimuth pixel spacing.

The slices will be cut along the middle of the slice overlapping region, which contains the output from one burst cycles.

9 COMMON AND SUPPORTING ALGORITHMS

This section presents a number of lower level algorithms and algorithms common to multiple modules. In particular the following algorithms are presented:

1. **Raw Data Decoding** as described in Section 9.1
2. **Raw Data Correction** as described in Section 9.2
3. **Nominal Replica** as described in Section 9.3
4. **PG Gain Compensation** as described in Section 9.4
5. **Bank of Elevation Antenna Pattern Correction Vectors** as described in Section 9.5
6. **Range Spreading Loss Correction** as described in Section 9.6
7. **Weighting Window** as described in Section 9.7
8. **Azimuth Frequency Vector** as described in Section 9.8
9. **Generic Unfolding and Resampling (UFR) (TOPSAR only)**, as described in Section 9.9
10. **Effective Radar Velocity** as described in Section 9.10
11. **Azimuth FM Rate** as described in Section 9.11
12. **Focusing Azimuth Block Overlap (Stripmap only)** as described in Section 9.12
13. **Focusing Azimuth Block Length (Stripmap only)** as described in Section 9.13
14. **Antenna Steering Rate and DC Rate Due to Steering (TOPSAR only)** as described in Section 9.14.
15. **Azimuth Antenna Pattern** as described in Section 9.15
16. **Bank of GR/SR LUTs** as described in Section 9.16
17. **Bank of Thermal Noise Estimation Vectors** as described in Section 9.17
18. **Application Scaling** as described in Section 9.18
19. **Absolute Calibration Vectors** as described in Section 9.19
20. **Fourier Transform and Energy of a Signal** as described in Section 9.20
21. **Look Extraction** as described in Section 9.21
22. **Resampling Interpolator** as described in Section 9.22
23. **Bi-static Delay Correction** as described in Section 9.23

9.1 Raw Data Decoding

This section provides a summary of the encoding types of the Sentinel-1 raw data and the corresponding decoding algorithms. A detailed description of the Sentinel-1 data decoding can be found in Section 4 of [A-6].

9.1.1 Raw Data Decoding Overview

The input raw Sentinel-1 data processed by the IPF is encoded on board the Sentinel-1 satellite in order to optimize the dynamic range of the digitized data acquired by the SAR instrument given the restrictions on storage capacity and downlink data rate of the satellite. The main encoding algorithm used for the Sentinel-1 data is the Flexible Dynamic Block Adaptive Quantization (FDBAQ) algorithm, which is a Huffman encoding algorithm applied to the already BAQ-encoded data. In turn, the BAQ is a data reduction algorithm based on the principles of minimum mean-squared error quantization.

Note that due to the FDBAQ encoding, distortion is introduced into the data in the form of quantization noise. The goal of the algorithm is to reduce the mean squared error of the quantization noise to the minimum possible amount.

The FDBAQ decoding is performed as part of the following processing stages:

- Pre-processing: only a sub-set of the Level 0 data is decoded in order to perform the raw data analysis (see Section 4.1).
- Doppler Centroid Estimation: the Level 0 data is decoded range line by range line prior to the fine DC Frequency Estimation (see Section 5.2.2).
- SLC Processing: the Level 0 data is decoded range line by range line prior to range compression (see Section 9.1).

Note that the decompression is a fast operation that represents only a small fraction of the total SAR processing computation time. Therefore, in order to minimize the storage needed for the signal data file (especially demanding in particular for processing long strips of data), the BAQ decoding is performed in line by both the DCE and the SLC processors.

9.1.2 Raw Data Decoding Algorithms

In a broad sense, the Sentinel-1 raw data can have three types of encoding:

- No encoding
- BAQ encoding
- FDBAQ encoding

The type of encoding is determined by the BAQ Mode field in the ISP. The following table describes the various values of the BAQ Mode and their meanings. Note that the decoding algorithm is the same for the 3 different FDBAQ modes shown in the table.

Table 9-1 Raw Data Encodings

BAQ Mode Value	Description
Bypass Mode	No encoding, just sign and magnitude (10 bits, 1 sign, 9 magnitude)
BAQ 3-BIT Mode	BAQ encoding with 3 bits (1 sign, 2 magnitude)
BAQ 4-BIT Mode	BAQ encoding with 4 bits (1 sign, 3 magnitude)
BAQ 5-BIT Mode	BAQ encoding with 5 bits (1 sign, 4 magnitude)
FDBAQ Mode 0	Nominal Entropy (Huffman) Coding
FDBAQ Mode 1	Entropy (Huffman) Coding with first alternate rate selection thresholds
FDBAQ Mode 2	Entropy (Huffman) Coding with second alternate rate selection thresholds

Typically, Bypass mode will be used for calibration data, while FDBAQ will be used for echo data. For noise data, both BAQ and FDBAQ encoding options may be available.

As shown in the figure below, the BAQ decoding requires reconstructing the sample value from the magnitude code via lookup tables. FDBAQ decoding requires first performing Huffman decoding, and then sample value reconstruction.

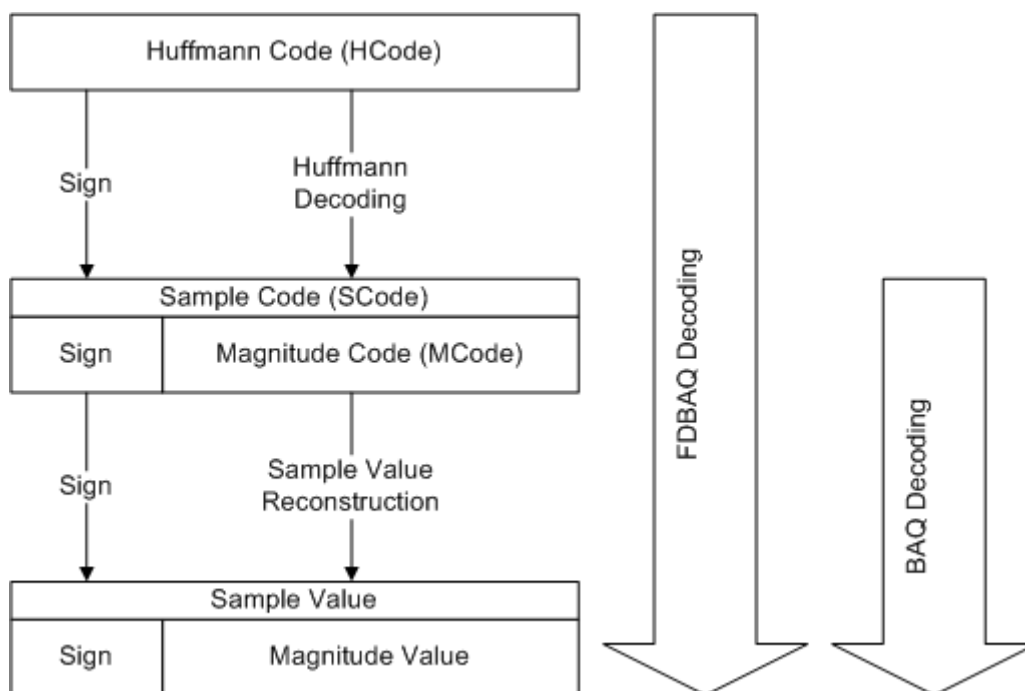


Figure 9-1 Decoding Process

Regardless of the encoding, the sample data is broken into four components, as described by the following table.

Table 9-2 Raw Data Sample Types

Channel / Section	Description
IE	In-Phase Even samples (0, 2, 4, ...)
IO	In-Phase Odd samples (1, 3, 5, ...)
QE	Quadrature Even samples (0, 2, 4, ...)
QO	Quadrature Odd samples (1, 3, 5, ...)

The data sequence within a source packet is sorted according to the channel, such that a block of IE data is followed by a block of IO data, and then QE and QO data. That is, the data channels need to be interleaved after decoding.

The following table provides the dictionary used to describe the various encodings. It should be noted that both the Bit Rate Code (BRC) and Threshold Index (THIDX) are valid only for the block in which they are sent.

Table 9-3 Encoding Terms

Term	Definition
NRL	Normalised Reconstruction Level
SF	Sigma Factor
THIDX	Threshold Index
BRC	Bit Rate Code
HCode	Huffman Code (Sign + Huffman coded Magnitude)
MCode	Magnitude Code
MValue	Magnitude Value
SCode	Sample Code
SValue	Sample Value
b	BAQ Block Index

The decoding of the three types of encoded data are further described as follows:

1. **Bypass Mode decoding** is described in Section 9.1.2.1
2. **BAQ Mode decoding** is described in Section 9.1.2.2
3. **FDBAQ mode Decoding** is described in Section 9.1.2.3

9.1.2.1 Bypass Mode Decoding

In Bypass mode the sample data is not encoded. It has 1 sign bit followed by a 9-bit magnitude code. To calculate the sample value, do the following:

$$S_{\text{value}} = (-1)^{\text{sign}} * M_{\text{code}}.$$

For example:

Sample Code (binary)	= 10 1011 1100
Sign	= 1
MCode	= 0 1011 1100 (binary) = 188 (decimal)
Sample Value	= -188

For Bypass mode, the M_{code} is the same as the M_{value} . That is, no decoding of the magnitude needs to happen.

9.1.2.2 BAQ Mode Decoding

Depending on the Threshold Index (THIDX) found in the ISP, the BAQ decoding may be the same as the Bypass Mode decoding. Normally, however, 2 multipliers need to be used to calculate the sample value. Both are determined from lookup tables. The Normalized Reconstruction Level (NRL) is set by based on the BAQ and the MCode. The Sigma Factor (SF) is determined by the THIDX. The sample value with BAQ encoding is calculated with the following formula. Note that THIDX can vary with the block index. Also, the actual thresholds used to determine whether the data is BAQ encoded or not are defined as auxiliary input parameters.

$$S_{\text{value}} = \begin{cases} (-1)^{\text{Sign}} \cdot M_{\text{Code}} & \text{for 3bit BAQ and } THIDX(b) < tbd1 \\ (-1)^{\text{Sign}} \cdot M_{\text{Code}} & \text{for 4bit BAQ and } THIDX(b) < tbd2 \\ (-1)^{\text{Sign}} \cdot M_{\text{Code}} & \text{for 5bit BAQ and } THIDX(b) < tbd3 \\ (-1)^{\text{Sign}} \cdot NRL_{BAQMOD, M_{\text{Code}}} \cdot SF_{THIDX(b)} & \text{for else} \end{cases}$$

For example, if the encoding is 3-bit BAQ, and the THIDX is 130:

Sample Code (binary)	= 110
Sign	= 1
MCode	= 2
NRL	= 1.344 (from NRL lookup table, based on 3-bit encoding and MCode)
SF	= 100.58 (from SF lookup table, based on THIDX = 130)
Sample Value	= $(-1) * 2 * 1.344 * 100.58 = -270.36$

Unlike in Bypass mode, the Magnitude is a code used to calculate a value.

9.1.2.3 FDBAQ Mode Decoding

FDBAQ encoding takes BAQ encoded data and applies a Huffman encoding to it. Each Huffman encoded value, or HCode, has a variable length, and is only determined during the decoding process. Each HCode consists of 1 sign bit, and N MCode bits. To determine the length and extract the MCode, the decoder must step through each bit of the HCode and follow the pattern in the binary Huffman decoding tree until a MCode is detected.

The Huffman decoding tree to use is determined by the Bit Rate Code (BRC) passed with the data. The BRC can be one of 0, 1, 2, 3, or 4.

The sample value with FDBAQ encoding is calculated as follows. Note that BRC and THIDX can both vary with the block index. Also, the actual thresholds used to determine whether the data is BAQ encoded or not are defined as auxiliary input parameters.

$$S_{Value} = \begin{cases} (-1)^{Sign} \cdot M_{Code} & \text{for } BRC(b) = 0 \text{ and } THIDX(b) < tbd1 \\ (-1)^{Sign} \cdot M_{Code} & \text{for } BRC(b) = 1 \text{ and } THIDX(b) < tbd2 \\ (-1)^{Sign} \cdot M_{Code} & \text{for } BRC(b) = 2 \text{ and } THIDX(b) < tbd3 \\ (-1)^{Sign} \cdot M_{Code} & \text{for } BRC(b) = 3 \text{ and } THIDX(b) < tbd4 \\ (-1)^{Sign} \cdot M_{Code} & \text{for } BRC(b) = 4 \text{ and } THIDX(b) < tbd5 \\ (-1)^{Sign} \cdot NRL_{BRC(b), M_{Code}} \cdot SF_{THIDX(b)} & \text{for else} \end{cases}$$

Once the MCode has been determined by tracing the Huffman tree, the calculation is very similar to the one used for BAQ encoded data. For FDBAQ, however, the NRL is indexed with the bit rate code (BRC) instead of the number of bits.

For example, if the BRC = 3 and the THIDX is 29:

HCode (binary)	= 0111 1110
Sign	= 0
HCode (binary) w/o Sign	= 111 1110
Mcode	= 7 (from tracing Huffman tree)
NRL	= 2.5333 (from NRL lookup table, based on BRC and MCode)
SF	= 18.17 (from SF lookup table, based on THIDX)
SValue	= 1 * 7 * 2.5333 * 18.17 = -322.21

For FDBAQ, the HCode is a code to work out the MCode. In turn, the MCode is a code to calculate the value.

9.2 Raw Data Correction

Three different types of raw data correction can be performed by the Sentinel-1 IPF: I Q bias correction, spurious signal correction, and receiver gain compensation. They are described in the following sections.

9.2.1 I Q Bias Correction

The purpose of I Q bias correction is to:

- Remove constant biases from the I and Q channels. This correction enforces a zero mean to the I and Q components of the signal.
- Correct for gain imbalance between the I and Q channels. This correction imposes the same standard deviation for the I and Q channels.
- Correct for non-orthogonality between the two channels.

For processing Sentinel-1 data the gain imbalance and non-orthogonality corrections will not be necessary (see also Section 4.1). These two corrections are however supported and controlled by configurable input parameters.

The raw data correction is always performed right after the BAQ decoding on each decoded sample and on a range line by range line basis.

Using the values of μ_I , μ_Q , ρ , and θ calculated during pre-processing as described in Section 4.1, the raw data correction is performed as follows:

1. Correct for biases

$$I_1 = I_{in} + \mu_I \quad (9-1)$$

$$Q_1 = Q_{in} + \mu_Q \quad (9-2)$$

where:

I_{in} = I channel input data

Q_{in} = Q channel input data

2. Correct for gain imbalance

$$I_2 = I_1 \quad (9-3)$$

$$Q_2 = \rho \cdot Q_1 \quad (9-4)$$

This correction is controlled by a configurable input parameter.

3. Correct the Q channel for non-orthogonality:

$$I_{out} = I_2 \quad (9-5)$$

$$Q_{out} = \frac{Q_2}{\cos(\theta)} - I_2 \cdot \tan(\theta) \quad (9-6)$$

This correction is controlled by a configurable input parameter.

9.2.2 Spurious Signal Correction

The raw echo and noise data contain spurious signals which are explained as harmonics of the 37.5 MHz system clock. They are picked up within the ADC assembly, are additive to the nominal measurement data. The spurious signals at ± 37.5 MHz and at 0 MHz are coherent and stable from pulse to pulse and this allows them to be subtracted from the imaging data. The spurious signals at 0 MHz are removed by the I/Q bias correction described above. The spurious signals at ± 37.5 MHz are removed by limiting the chirp pulse bandwidth using a configurable input parameter which has the effect of attenuating the signals outside the desire bandwidth (including the spurious signals).

The filtering of the spurious signals is controlled by the configurable parameter $TXPL_{nom}$. The spurious signal filtering will be applied if $TXPL_{nom}$ is greater than zero and less than the TXPL value extracted from the downlink.

9.2.2.1 Noise Correction

The filtering of the spurious signal from the noise data is performed by transforming each noise line $Noise(t)$ to the frequency domain and zero padding the frequency components beyond the filter bandwidth B_{filter} as follows:

1. $Noise(t)$ is zero padded to length N_{fft} and transformed to the frequency domain using an FFT of this size, resulting in $Noise(f)$ of size N_{fft} .
2. The filter bandwidth, B_{filter} , is calculated as:

$$B_{filter} = TXPL_{nom} TXPRR \quad (9-7)$$

where:

$TXPL_{nom}$ is a configurable input parameter

$TXPRR$ is the transmit chirp pulse ramp rate

3. The number of zero-fill samples, N_{zero} , is calculated as:

$$N_{zero} = N_{fft} - \frac{B_{filter}}{F_r N_{fft}} \quad (9-8)$$

4. N_{zero} samples are set to zero in the middle of $Noise(f)$ to remove the frequency components outside of B_{filter} .
5. The filtered time domain signal $FilteredNoise(t)$ is calculated by transforming the filtered $Noise(f)$ to the time domain using an inverse FFT.

9.2.2.2 Echo Correction

The filtering of the spurious signal from the echo data is controlled by the chirp duration reported by the preprocessor. If the spurious signal filtering is to be applied, the preprocessor provides the SLC component with a chirp duration of $TXPL_{nom}$ which will limit the bandwidth of the RRF applied to the echo data during range compression. If the spurious signal filtering is not to be applied then the preprocessor provides the SLC component with the full chirp duration of TXPL and the full RRF will be applied during range compression.

9.2.3 Receiver Gain Compensation

The receiver introduces a gain variation into the raw data. This can be compensated for in the IPF, as described in document [A-3]. The compensation is modelled by two terms. The first, the gain trend compensation, is a function of time within a PRI relative to the end of the transmitted pulse. The second, the gain overshoot correction, is a function of time relative to the acquired sampling window. All times below are in microseconds.

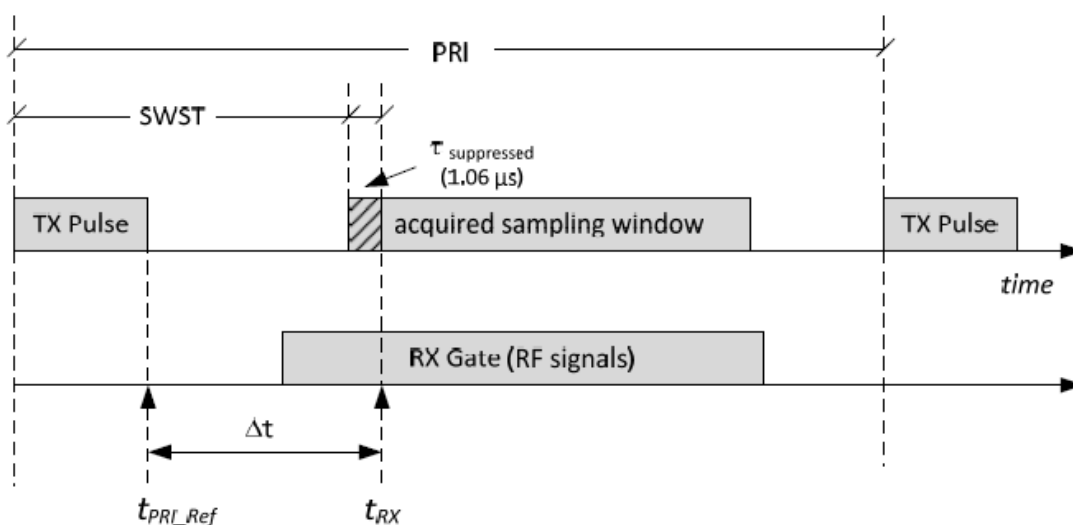


Figure 9-2 Timing Schematic for Receiver Gain Compensation

Let t_s be the time scale with origin at t_{RX} . Then the overall gain variation can be described in the form:

$$GV(t_s) = GV1(t_s + \Delta t) \cdot GV2(t_s) \quad (9-9)$$

Where Δt is the time between the end of the transmitted pulse and the start of the acquired sampling window. Note that within the IPF, the SWST provided by the pre-processor includes $\tau_{suppressed}$.

Let t_1 be the time relative to the end of the transmitted pulse.

$$t_1 = t_s + \Delta t \quad (9-10)$$

Therefore, gain variation equation can be rewritten:

$$GV(t_s) = GV1(t_1) \cdot GV2(t_s) \quad (9-11)$$

The gain trend term can be modelled as follows:

$$GV1dB(t_1) = \frac{p_4 t_1^4 + p_3 t_1^3 + p_2 t_1^2 + p_1 t_1 + p_0}{q_2 t_1^2 + q_1 t_1 + q_0} \quad (9-12)$$

The gain overshoot term can be modelled as follows:

$$GV2dB(t_s) = a \cdot \exp(-b \cdot t_s) \cdot \sin(d \cdot t_s + \varphi) + c \quad (9-13)$$

These functions can be transformed from dB to a linear scale as follows:

$$GV1(t_1) = 10^{\frac{GV1dB(t_1)}{10}} \quad (9-14)$$

$$GV2(t_s) = 10^{\frac{GV2dB(t_s)}{10}} \quad (9-15)$$

The gain correction can be applied to the complex raw signal as follows:

$$cmplxSigCorrected(t_s) = \frac{cmplxSigRaw(t_s)}{\sqrt{GV(t_s)}} \quad (9-16)$$

Note that the correction can be performed on I and Q samples separately.

9.3 Nominal Replica

The nominal (or theoretical, or ideal) replica is required in several instances like for example to extract the replicas (see Section 4.2.1.4) or to derive the reference replica in case the replica reconstruction failed or processing with the nominal chirp was explicitly requested through a configuration parameter. The nominal chirp generation for the PG and image replicas is defined in Section 4.2.1.1.

9.4 Drift Compensation

The drift compensation is performed as indicated in [A-3], Section 4.5, on a range line by range line basis, right after the raw data correction (described in Section 9.2). The drift compensation makes use of the table of PG (complex) product values (i, P_i) , computed as described in Section 4.2.1.6, where i is the absolute line number of the signal range line to which the PG value corresponds. The PG value applicable to the current line is calculated by linear interpolation as:

$$P_k = P_i + (P_j - P_i) \frac{k - i}{j - i} \quad (9-17)$$

Note that if the PG values have been calculated from the calibration pulses, (and not from the PG product model,) only the valid value will be used for interpolation (as described in Section 4.2.1.8). Furthermore, at the beginning and the end of the segment, linear extrapolation from the first (last) two valid PG values will be employed. If we denote by $s(k)$ the current (raw data corrected) range line, output from the previous step (described in Section 6.1.2) then the drift-corrected current data line is:

$$s_c(k) = s(k) \cdot P_k \quad (9-18)$$

9.5 Bank of Elevation Antenna Pattern (EAP) Correction Vectors

The EAPs are used to correct the corresponding radiometric variation of the data in the range direction (see Section 6.1.2). The EAPs are also used for the estimation (see Section 9.17) and removal (see Section 7.3.1) of the thermal noise level.

In order to preserve the radiometric continuity of the images in the azimuth direction, the EAP must be updated frequently enough to account for the variation in incidence angle and mean terrain height of the image (see Section 4.4 for terrain height calculation). The approach is then to create a bank of EAPs calculated at a given configurable input time interval. Then, the EAP corresponding to any given azimuth time will be obtained by linear interpolation between the two closest EAPs in the bank.

Each EAP correction vector in the bank can be calculated as described in Section 9.5.1.

9.5.1 EAP Correction

The EAP correction vector is derived from the following input parameters:

- Complex Elevation Antenna Pattern LUT (mapping elevation angles to elevation antenna values)
- Satellite position (ECR coordinates)
- Satellite velocity (ECR coordinates)
- Earth model
- Terrain height vector
- Elevation boresight angle
- Range time of the first range sample

The nominal elevation boresight angle must also be updated frequently enough to account for the roll steering law specific to the Sentinel-1 mission (see more details on the roll steering law in [R-12] and references therein). The frequency of update of nominal elevation boresight angle is controlled by a configuration input parameter.

Clearly, the nominal elevation boresight angle encompasses both the mechanical roll angle and the electronic elevation steering angle needed to generate a particular beam. The term ‘nominal’ should be understood in the sense of ‘theoretical’ or ‘programmed’ value as opposed to actual value; the actual value can be slightly different from the nominal value due to (small) unknown antenna mechanical or electronic pointing errors.

The elevation boresight angle corresponding to the current position and attitude of the satellite can be retrieved from the EO CFI software following the procedure defined in [R-15]. If the elevation boresight angle cannot be estimated using this approach due to lack of sufficient satellite position or attitude data, then the nominal elevation boresight angle corresponding to the current position of the satellite can be derived by linear interpolation from the values corresponding to the reference altitude position, as follows (see [R-12] Section 3.2):

$$\theta_{boresight} = \theta_{ref} - \alpha_{roll}(H - H_{ref}) \quad (9-19)$$

where:

θ_{ref} = antenna boresight angle at reference altitude = 29.45 deg

α_{roll} = roll steering sensitivity versus altitude = 0.0566 deg/km

H_{ref} = reference satellite altitude = 711.7 km

H = satellite altitude corresponding to current satellite position

To calculate the EAP correction **for each range sample** perform the following:

1. **Calculate the slant range** R_i using slant range time and sampling frequency.
2. **Calculate the ECR coordinates of the target on the ground** corresponding to the current slant range and lying in the zero-Doppler elevation plane; use the terrain height, h_i of the current sample.
3. **Calculate the view vector** (= vector from the satellite position to the target) in ECR coordinates.
4. **Calculate the elevation angle** by computing the angle between the satellite position vector and the view vector.
5. **Subtract the antenna boresight angle (computed using equation 9-9 above) from the calculated elevation angle.**
6. **Compute the complex antenna pattern LUT value** $\tilde{C}_{eap}(R_i)$ for the sample corresponding to the current elevation angle, from the input elevation pattern table, by linear interpolation.
7. **Compute the current complex antenna pattern value** for the current sample as:

$$C_{eap}(R_i) = \sqrt{|\tilde{C}_{eap}(R_i)|} \cdot \varphi(\tilde{C}_{eap}(R_i))$$

where the square root is applied only to the absolute value, while the phase is left unchanged.

8. **Compute antenna pattern correction** for the current sample as the reciprocal of the complex antenna pattern value.

9.6 Range Spreading Loss Correction

The range spreading loss is compensated in the SAR processor by the amplitude scaling of each range sample by $1/G_{rsl}(R)$ where:

$$G_{rsl}(R) = \sqrt{\left(\frac{R_{ref}}{R}\right)^3} \quad (9-20)$$

and

R = slant range of the sample

R_{ref} = reference slant range (configurable input parameter)

9.7 Weighting Window

In the SAR processor, both the range and azimuth spectrum of the data are weighted by a (generalized) Hamming window:

$$W(\alpha) = a - (1 - a) \cos(\alpha), \quad \alpha \in [0, 2\pi] \quad (9-21)$$

where a is the Hamming window coefficient. The Hamming window coefficient is in general a parameter dependent on acquisition mode, product type and beam.

The weighting has opposite effects on resolution and side lobes level therefore the coefficient is chosen in such a way as to realize the desired trade-off between these image characteristics. For more details on image characteristics and particular Sentinel-1 Hamming coefficients see Document [A-4].

The weighting window type and coefficient are input, configurable parameters for each processing step to which the window is applied.

9.8 Azimuth Frequency Vector

The azimuth frequency vector is the result of sampling the interval

$\left[f_{\eta_c} - \frac{F_a}{2}, f_{\eta_c} + \frac{F_a}{2} \right]$, to obtain a vector of length equal to the azimuth FFT length, M_{fft} . where:

F_a = azimuth sampling frequency

f_{η_c} = Doppler Centroid

The elements of the azimuth frequency vector, $f_{\eta_i}, i = 0, 1, \dots, M_{fft} - 1$, are calculated as follows:

- **Calculate** $\hat{f}_{\eta_i} = \frac{(double)i}{M_{fft}} F_a + floor\left(\frac{f_{\eta_c}}{F_a}\right)$
- **Calculate** $f_{\eta_i} = \hat{f}_{\eta_i} - int\left(2 \frac{\hat{f}_{\eta_i} - f_{\eta_c}}{F_a}\right) F_a$

9.9 Generic Unfolding and Resampling (UFR) (TOPSAR only)

The next figure presents a generic UFR block that underlies both the frequency domain UFR (Section 6.2.4) and the time domain UFR (Section 6.4).

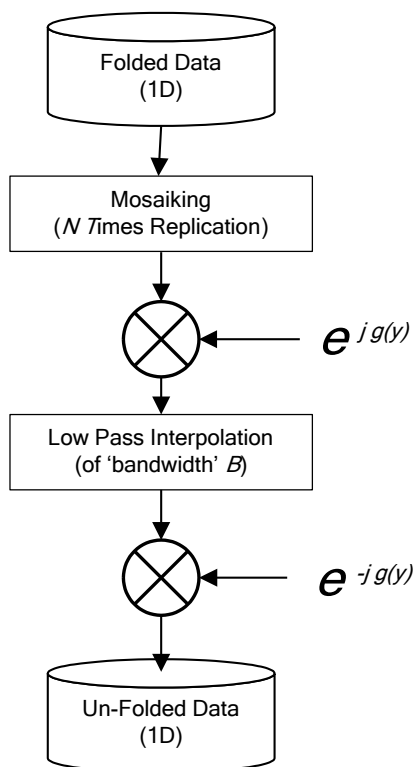


Figure 9-3 Generic Unfolding and Re-sampling

The generic variable y will become the azimuth frequency variable, f_η , for the frequency domain case and respectively the azimuth time variable η for the time domain case.

The generic number of replications, N the de-ramp phase function $g(y)$ and the low-pass filter bandwidth, B , will also be specifically derived in each of the 2 cases (frequency UFR and time UFR).

9.10 Effective Radar Velocity

The effective radar velocity, V_r , is the parameter occurring in the range equation 6.16. As explained in [R-5], Section 4.3.1, V_r is a pseudo-velocity, selected so that the hyperbola defined by equation 6-18 models the actual range equation in the general case of a curved (orbit and Earth) geometry.

In the Sentinel-1 IPF, V_r is calculated according to the detailed description in [R-5], Appendix 4A.

The precise calculation of V_r is relatively complicated, therefore it is useful to note that, as shown in [R-5], Appendix 4A, $V_r \cong \sqrt{V_g V_s}$ where,

V_g = ground velocity, which is a positive scalar representing the antenna beam footprint velocity when projected onto the ground.

V_s = satellite velocity, which is the norm of the satellite velocity vector expressed in ECR coordinates.

Note that, as V_g is range dependent, V_r will also be range dependent. The Range-Doppler azimuth focusing can fully take into account this variation.

The effective radar velocity is also azimuth dependent, as it depends on the satellite position in orbit and its height above the Earth. This dependency will be taken into account in IPF by updating V_r for each azimuth block.

9.11 Azimuth FM Rate

The azimuth FM rate is the rate of change of azimuth (or Doppler) frequency and is defined by:

$$k_a = \frac{2V_r^2 D^2(f_{\eta_c}, V_r)}{\lambda R(\eta_c)} = \frac{2V_r^2 D^3(f_{\eta_c}, V_r)}{\lambda R_0} \quad (9-22)$$

where:

- V_r = effective radar velocity
- f_{η_c} = Doppler Centroid frequency
- $D(\cdot, \cdot)$ = defined by equation (6-29)
- λ = wave length of the carrier frequency
- R_0 = Slant range of closest approach

The derivation and discussion of the azimuth FM rate can be found in [R-5] Section 4 and Appendix 5B.

The azimuth FM rate varies with range (as f_{η_c} and V_r are range dependent). Therefore, k_a is updated in range with a frequency of update defined by a configurable input parameter.

The azimuth FM rate is also updated for each azimuth block (as both V_r and f_{η_c} are updated for each azimuth block).

9.12 Focusing Azimuth Block Overlap (Stripmap only)

For Stripmap SLC processing the overlap between azimuth processing blocks has two components:

1. The first (and most significant) component of the overlap is equal to the maximum possible azimuth matched filter (or, equivalently) the maximum possible exposure time of a target) T_{mf} . can be calculated as follows:

$$T_{mf} = -\frac{Bw}{k_a^{far}} \quad (9-23)$$

where

Bw = the azimuth bandwidth (provided in the input auxiliary file)

k_a^{far} = the azimuth FM rate calculated at maximum far range s

Expressed in azimuth samples, the matched filter length becomes:

$$A = \text{ceil}(T_{mf} F_a) \quad (9-24)$$

According to the instrument mode definition, for the different swaths, A varies between approximately 900 and 1700 azimuth samples.

2. The second component of the overlap is due to the possible variation of the DC frequency from block to block.

Since for Sentinel-1 the Doppler centroid can vary from block to block, the block overlap has to be increased by the azimuth time corresponding to the maximum possible Doppler centroid variation between consecutive blocks:

$$T_{\Delta Ov} \cong -\frac{\text{maxDeltaFdc}}{k_a^{far}} + \text{smallMargin} \quad (9-25)$$

where maxDeltaFdc ($\text{maxDeltaFdc} > 0$) is the maximum expected Fdc variation between consecutive azimuth blocks and the maximum is taken over all expected values over the entire scene.

Expressed in azimuth samples, this increase in block overlap becomes:

$$\Delta Ov = \text{ceil}(T_{\Delta Ov} F_a) \quad (9-26)$$

With these notations, the necessary azimuth block overlap becomes:

$$Ov = A + \Delta Ov \quad (9-27)$$

Note that due to the small variation of k_a around the orbit (<1.5%), both A and ΔOv also have a relatively small variation around the orbit:

- The maximum matched filter length, A , has a variation of approximately 15 to 30 samples, depending on the beam.

- For a reasonable small $\max\Delta Fdc$ (for example $< 100\text{Hz}$), ΔOv has a variation of about one sample.

As a consequence, both A and ΔOv , which are beam dependent but unique for all scenes, will be specified as configurable input parameter.

This will also serve the sliced data case since Ov must be the same for all IPF instances.

9.13 Focusing Azimuth Block Length (Stripmap only)

Clearly the azimuth block length has as lower limit the azimuth block overlap length. So a choice of an azimuth block which is too short will impact the throughput performance.

On the other hand, the length of the azimuth block is limited from above by the requirement of insuring phase continuity between azimuth blocks. If the azimuth blocks are too long, the update of the V_r (and implicitly the update of the FM rate) could potentially introduce phase discontinuities between azimuth blocks.

The azimuth block length, which is a configurable input parameter, is also chosen to be equal to the FFT length to be used for azimuth compression.

9.14 Antenna Steering Rate and DC Rate (TOPSAR only)

The results in this section are referenced by Sections 5.2.1 (DCE Pre-conditioning) 6.2.4 (Azimuth Frequency UFR) and 6.4 (Azimuth Post-processing).

In TOPSAR modes, the antenna is steered from backward to forward, in the azimuth direction.

The antenna steering rate, k_ψ , can be calculated from $(\psi_i)_{i=0,1,\dots,N-1}$, the set of azimuth steering angles, one value for each PRI in a burst. The azimuth steering angles are configurable input parameters. A set of azimuth steering angles for each TOPSAR swath is provided.

The antenna steering rate can be calculated by inverting the following linear system:

$$\begin{bmatrix} \psi_0 \\ \psi_1 \\ \vdots \\ \psi_{N-1} \end{bmatrix} = \begin{bmatrix} 1 & \eta_0 \\ 1 & \eta_1 \\ \dots & \dots \\ 1 & \eta_{N-1} \end{bmatrix} \begin{bmatrix} \psi_0 \\ k_\psi \end{bmatrix} \quad (9-28)$$

The Doppler Centroid rate introduced by the antenna scanning can be calculated as:

$$k_s(\eta) = -\frac{2 \cdot V_s}{\lambda} \cos\left(k_\psi\left(\eta - \frac{T_b}{2}\right)\right) \cdot k_\psi(\eta) \approx -\frac{2 \cdot V_s}{\lambda} k_\psi \quad (9-29)$$

Note that in the Sentinel-1 mode design the maximum steering angle of the antenna beam during TOPSAR acquisition is about 1 degree. Ignoring the cosine weight in the $k_s(\eta)$ formula, an under-estimation error in the order of 0.02 % of the actual DC is introduced, which is considered negligible.

9.15 Azimuth Antenna Pattern

The azimuth antenna pattern calculation is referenced by Section 7.2.

The (2-way) azimuth antenna pattern (AAP) can be modeled, as a function of azimuth angle ψ , by:

$$G(\psi) = \text{sinc}^2\left(\frac{L}{\lambda} \sin(\psi)\right) \quad (9-30)$$

where:

λ = the wavelength

L = antenna length

However, the IPF uses measured antenna patterns, which in general can deviate from the model above. The measured, beam dependent, azimuth antenna patterns are provided as input LUTs to the IPF.

In order to be applied to the data (i.e. to the azimuth lines) the AAP is derived as a function of azimuth frequency, f_η , from the following input parameters:

- AAP LUT (mapping azimuth angles to azimuth antenna gains, in dB)
- λ = radar wavelength
- V_s = satellite velocity

The AAP LUT and λ are defined by configurable input parameters.

To calculate the azimuth pattern, for each azimuth frequency f_{η_i} perform the following:

1. **Calculate the corresponding azimuth angle as a function of azimuth frequency**

$$\Psi(f_{\eta_i}) = \arcsin\left(\frac{\lambda}{2V_s} f_{\eta_i}\right) \approx \frac{\lambda}{2V_s} f_{\eta_i} \quad (9-31)$$

Note that the small angle approximation introduces a negligible error of < 0.03% for any azimuth frequency, $|f_{\eta_i}| < 10$ kHz.

Note also that for TOPSAR data the above formula shall be modified to take into account the “shrinking factor” described in Section 3.1.2 by equation (3-5):

$$\Psi(f_{\eta_i}) \approx \frac{\lambda}{2V_s} f_{\eta_i} \cdot k = \frac{\lambda}{2V_s} f_{\eta_i} \cdot \left(1 + \frac{R_0 k_\psi}{V_r}\right)$$

2. **Identify the nearest two angles in the azimuth antenna pattern LUT, ψ_1 and ψ_2 such that**

$$\psi_1 \leq \Psi(f_{\eta_i}) \leq \psi_2 \quad (9-32)$$

3. **Compute the azimuth antenna pattern gain (in dB) for the current frequency sample, $\tilde{G}_{aap}(f_{\eta_i})$ from the azimuth antenna pattern LUT values corresponding to ψ_1 and ψ_2 by linear interpolation.**
4. **Convert to linear scale:**

$$G_{aap}(f_{\eta_i}) = \text{pow}\left(10, \frac{\tilde{G}_{aap}(f_{\eta_i})}{20}\right) \quad (9-33)$$

5. **[NORM] For the purpose of azimuth normalization (see Section 6.3.4.1.2), calculate the Azimuth Antenna Pattern Correction, $G_{aapc}(f_{\eta_i})$:**

$$G_{aapc}(f_{\eta_i}) = \frac{G_{aap}(f_{mid})}{G_{aap}(f_{\eta_i})} \quad (9-34)$$

9.16 Bank of GR/SR LUTs

The GR/SR LUT represents a mapping between the ground range pixel number and its corresponding slant range pixel value (in general, a real number). This table is used for conversion of the image from slant range coordinates to ground range coordinate, as part of the GRD product generation.

In order to preserve the geometric continuity of the GRD image in the azimuth direction, the GR/SR must be updated frequently enough to account for the variation in orbit, latitude and mean terrain height of the image (see Section 4.3.1 for mean terrain height calculation). The approach is then to create a bank of GR/SR LUTs calculated at a given configurable input azimuth time interval. Then, the GR/SR

table corresponding to any given azimuth time will be obtained by linear interpolation between the two closest GR/SR tables in the bank.

Section 9.16.1 describes how any of the GR/SR LUTs in the bank is calculated.

9.16.1 GR/SR Polynomial Coefficients

The GR/SR polynomial coefficients are derived from the following input parameters:

- Satellite Position
- Satellite Velocity
- Earth ellipsoid model
- Terrain height
- Range start time
- Range sampling frequency
- Number of input (slant range)
- Degree of GR/SR polynomial

The steps for calculating the GR/SR polynomial coefficients are the following:

1. **Calculate the number of points** to be used for the polynomial fitting based on a pre-defined slant range spacing ΔR between points.
2. **Initialize the slant range to the first pixel** from the range start time, τ_0 :

$$R_0 = \frac{\tau_0 \cdot c}{2} \quad (9-35)$$

3. **Initialize the relative ground range distance of the first sample**, $R_{g0} = 0$;
4. **For each point to be used for the polynomial fitting perform the following:**
 - Determine the slant range:

$$R_i = \frac{\tau_0 \cdot c}{2} + i \cdot \Delta R \quad (9-36)$$

- Calculate the zero-Doppler target position in ECR coordinates, \mathbf{s}_i .
- Calculate the angle between \mathbf{s}_{i-1} and \mathbf{s}_i :

$$\theta = \arccos \left(\frac{\mathbf{s}_{i-1} \cdot \mathbf{s}_i}{\|\mathbf{s}_{i-1}\| \cdot \|\mathbf{s}_i\|} \right). \quad (9-37)$$

- Set the relative ground range distance R_{gi} :

$$R_{gi} = R_{gi-1} + \frac{\theta \cdot (\|s_{i-1}\| + \|s_i\|)}{2} \quad (9-38)$$

5. **Perform a polynomial fitting** i.e. find the coefficients of a polynomial $f(t) = c_0 + c_1t + \dots + c_{n-1}t^{n-1}$ that fits the data $f(R_{gi})$ to R_i , in a least squares sense.

The output of this algorithm is then the set of the GR/SR polynomial coefficients c_0, c_1, \dots, c_{n-1} .

9.16.2 GR/SR LUT

The GR/SR table is calculated for each ground range pixel, using the:

- GR/SR coefficients calculated as described in Section 9.16.1
- Input slant range pixel spacing, ΔR_{in}
- Output ground range pixel spacing, ΔR_{out}

For each ground range pixel of index i perform the following:

1. **Calculate the relative ground range distance** (i.e. distance with respect to the first ground range pixel in the image):

$$R_{gi} = i \cdot \Delta R_{out} \quad (9-39)$$

2. **Calculate the corresponding slant range** of the pixel i by evaluating the value of the GR/SR polynomial at R_{gi} .
3. **Calculate the slant range to the first pixel**; this is precisely the c_0 coefficient of the GR/SR polynomial.
4. **Calculate the corresponding slant range index (as a real positive number):**

$$x_i = \frac{R_i - R_0}{\Delta R_{in}} \quad (9-40)$$

The GR/SR LUT is then the array of pair of indices (i, x_i) defined for all ground range pixels in an image line.

9.17 Bank of Thermal Noise Estimation Vectors

The algorithm described in this section performs the estimation of thermal noise level of Sentinel-1 SAR SLC images. The accuracy of the noise estimation is highly affected by the specific radiometric characteristics of the system (including

radiometric stability, gain losses, thermal noise power, etc.) so specific tailoring may be needed in the future.

The knowledge of the thermal noise level of an image is important for many applications. In particular, it is also required for the generation of Sentinel-1 Level 2 products. Furthermore, once estimated, the thermal noise can be removed from power detected images, as described in Section 7.3.1, improving in this way the quality of the image.

The algorithm determines the noise intensity as a function of slant-range distance, R , and slow-time, η . The noise estimation takes into account all the radiometric correction applied during SLC processing and is therefore representative for single-look images. However, if multi-looking processing preserves the radiometric characteristics of the acquired data, the estimated noise level calculated for single-look images will also apply to multi-look images. Since noise level errors introduced by multi-looking are considered negligible, the noise estimation performed as described in this section is also representative for the GRD products.

In order to reflect the correct noise level of the image, especially in the case of long strips, the noise level estimates must be updated frequently enough to account for the variations in all of all the contributing factors.

The approach is then to create a bank of noise vectors calculated at an azimuth time interval derived from the highest frequency among the frequencies of update of all the contributing factors to the thermal noise estimation.

Then, the thermal noise level vector corresponding to any given azimuth time will be obtained by linear interpolation between the two closest noise vectors in the bank.

Each thermal noise level vector in the bank can be calculated as described in Sections 9.17.1 and 9.17.2.

9.17.1 Algorithm Overview

The generic SAR SLC signal at the end of SLC processing (therefore prior to multi-looking detection and slant to ground conversion) can be modelled as follows:

$$s_r(n_s; R, \eta) = [s(R, \eta) + n(n_s; R, \eta)]G_{tot}(n_s; R, \eta) \quad (9-41)$$

where:

R	= slant range
η	= slow time
n_s	= sub-swath number
$s(R, \eta)$	= the received backscatter signal

$n(n_s; R, \eta)$ = bi-dimensional white thermal noise for a given swath

$G_{tot}(n_s; R, \eta)$ = total gain applied to the data (signal + noise) during SLC processing.

The total gain, $G_{tot}(n_s; R, \eta)$, is defined as follows:

$$G_{tot}(n_s; R, \eta) = \frac{G_{ds}(\eta)G_{pg}(n_s; \eta)}{G_{eap}(n_s; R)G_{rst}(R)} \sqrt{k_{noise}k_{proc}} \quad (9-42)$$

where:

$G_{ds}(\eta)$ = de-scalloping function (TOPSAR only)

$G_{pg}(n_s; \eta)$ = PG correction factor (see equation (4-33))

$G_{eap}(n_s; R, \eta)$ = two-way elevation antenna pattern gain calculated as the magnitude of the complex elevation antenna pattern value:

$$G_{eap}(n_s; R, \eta) = |C_{eap}(n_s; R, \eta)|$$

$G_{rst}(R)$ = range spreading loss

k_{noise} = noise calibration factor

k_{proc} = processor scaling factor (this includes the input scaling factor parameter, and the range and azimuth oversampling factors)

Note that the de-scalloping function $G_{ds}(\eta)$ is applied to the noise when the IPF removes noise from images. However, when the IPF annotates the noise vectors, this factor is not included because it has a small effect, and it varies in azimuth for every range line. This factor can be calculated using the formulas discussed in Section 6.2.4.2 and applied to the noise vectors included in the product to calculate the noise more precisely in azimuth.

Note also that for processing ASAR data, two extra factors are added to this equation:

- the droop gain, $G_{dg}(R)$, which must be accounted for in the denominator of the right-hand side of the equation
- the replica gain used in range compression, $G_{rg}(n_s, \eta)$, which must be accounted to the numerator of the right-hand side of the equation.

The calculation of the thermal noise level is based on the assumption that signal and thermal noise are un-correlated. With this assumption, the expected value of the SLC data intensity can then be expressed as:

$$E[(s(R, \eta) + n(n_s; \eta))^2] = E[s(R, \eta)^2] + E[n(n_s; R, \eta)^2] \quad (9-43)$$

Combining equations (9-41) and (9-43) and we obtain:

$$E[(s(R, \eta) + n(n_s; \eta))^2] \cdot G_{tot}^2(n_s; R, \eta) = E[s(R, \eta)^2] + E[n(n_s; R, \eta)^2] \cdot G_{tot}^2(n_s; R, \eta) \quad (9-44)$$

Let $\sigma_n^2(n_s, \eta)$ be the noise power (measured from the down-linked noise packets as described in Section 4.2.1.9), which represents an estimate of $n(n_s; R, \eta)^2$:

$$\sigma_n^2(n_s, \eta) = E[n(n_s; R, \eta)^2]. \quad (9-45)$$

The thermal noise power level of the image is then given by the second term in Equation 9-44:

$$\sigma_n^2(n_s) G_{tot}^2(n_s; R, \eta) \quad (9-46)$$

Note that the noise power $\sigma_n^2(n_s, \eta)$, which is updated periodically according to the frequency of the noise packets within the raw data, is assumed constant during the acquisition time between two noise measurements.

The thermal noise level is calculated and annotated for both SLC and GRD products. However, the thermal noise level (which only contains amplitude, and not phase, information) can be removed only from the GRD data (as described in Section 7.3.1).

9.17.2 Algorithm Implementation

The next sub-sections present the following cases of thermal noise algorithm implementation:

1. **The SLC case** as described in Section 9.17.2.1
2. **The GRD case** as described in Section 9.17.2.2

9.17.2.1 The SLC Case

Note that for SLC products, the thermal noise is estimated with the sole purpose of annotating it to the product.

For a given azimuth time $\bar{\eta}$, the steps for calculating the thermal noise levels are the following:

- **Calculate the range spreading loss vector** $G_{rsl}(R_i)$ as described in Section 9.6. (This vector is invariant in azimuth, therefore it can be calculated only once).

- **Calculate the elevation beam pattern vector** $G_{eap}(R_i, \bar{\eta})$ as described in Section 9.5.1.
- **Apply all the scalar contributing factors** described in Section 9.17.1.

9.17.2.2 The GRD Case

For the GRD case, the thermal noise vectors must be converted from slant range to ground range.

For a given azimuth time $\bar{\eta}$, the steps for calculating the thermal noise level are the following:

- **Estimate the thermal noise (in slant range coordinates)** as described in Section 9.17.2.1
- **Convert the thermal noise vector to ground range coordinates** by using the bank of GR/SR LUTs described in Section 9.16. (Note that the conversion is performed in the same way the conversion to ground range coordinates of the SLC image is performed).
- For TOPSAR, merge the sub-swaths thermal noise vectors into one vector
The merging is performed following the same strategy used for merging two nearby ground range image lines as described in Section 7.3.2.

9.18 Application Scaling

The SLC and GRD output images are scaled by a configurable, application specific, range dependent gain function called Application Look-Up Tables (LUTs). The purpose of this scaling is to:

- Optimize the radiometric scaling of the main feature of interest (while optimizing the available dynamic range in the output product)
- Compensate for changes in the radar backscatter with changing incidence angles (for the main feature of interest to the user)

Types of applications for which different LUTs could be applied include sea, land, ice, urban, mixed etc. types of scenes. Also, different product types use different output pixel types (e.g. 16 bit integer for SLC and 16 bit unsigned integer for GRD), so for each application LUT separate tables must be defined for each output pixel type.

Each output pixel type table within an application LUT is specified as an incidence angle where the LUT starts, an incidence angle increment between LUT values, and the scaling LUT values themselves. To derive a scaling LUT value S_i for a range pixel i , the LUT values are linearly interpolated in the range direction depending on

the incidence angle corresponding to range pixel i . The LUT values are applied to the output image as follows:

$$DN_i^{scaled} = S_i \cdot DN_i \quad (9-47)$$

where:

S_i = the application scaling corresponding to range pixel i

DN_i = floating point magnitude of range pixel i in image, before application scaling

DN_i^{scaled} = integer digital number of range pixel i in output image

9.19 Absolute Calibration Vectors

The IPF product annotations allow for the derivation of β^0 , σ^0 or γ radiometrically calibrated imagery from the digital numbers in the SAR images. To achieve this calibration, the user must scale the digital numbers by:

$$\beta^0 = \frac{DN^2}{K_{abs}} \quad (9-48)$$

$$\sigma^0 = \frac{DN^2}{K_{abs}} \sin(\theta) \quad (9-49)$$

$$\gamma = \frac{DN^2}{K_{abs}} \tan(\theta) \quad (9-50)$$

$$\sigma^0 = \beta^0 \sin(\theta), \gamma = \beta^0 \tan(\theta) \quad (9-51)$$

where:

θ = radar incidence angle

K_{abs} = absolute calibration constant (swath dependent configurable, input parameter)

DN = digital numbers in image without the scaling due to the application scaling LUT (described in Section 9.18)

In order to simplify process of obtaining calibrated images from IPF images, the annotated β^0 , σ^0 and γ calibration vectors included with products also compensate for the application scaling described in Section 9.18. The vectors also include range spreading loss and elevation antenna pattern correction, if these two corrections were not applied in the image processing.

To this end, three sets of related, range dependent functions are provided as annotations to each product: *lutBeta0*, *lutSigma0* and *lutGamma*:

$$lutBeta0(i) = S_i \sqrt{K_{abs}} \quad (9-52)$$

$$lutSigma0(i) = \frac{lutBeta0(i)}{\sqrt{\sin(\theta_i)}} \quad (9-53)$$

$$lutGamma(i) = \frac{lutBeta0(i)}{\sqrt{\tan(\theta_i)}} \quad (9-54)$$

where:

K_{abs} = absolute calibration constant

S_i = the application scaling corresponding to range pixel i (see description in Section 9.18).

θ_i = is the radar incidence angle corresponding to range pixel i .

Additionally, a fourth table, *lutDN* is provided, in case the user wants to recover the original DN value, which the image had before the scaling by the application LUT:

$$lutDN(i) = S_i \quad (9-55)$$

In order to convert the DN of a given range pixel in a Sentinel-1 SAR image to a calibrated value or the original DN, the user has to apply one of the following formulae:

$$value(i) = \frac{|DN_i|^2}{A_i^2} \quad (9-56)$$

where, depending on the selected LUT:

$value(i)$ = one of β_i^0 , σ_i^0 or γ_i or DN_i^{orig} .

A_i = one of $lutBeta0(i)$, $lutSigma0(i)$, $lutGamma(i)$ or $lutDN(i)$

Note that the σ^0 and γ calibrated values obtained in this way represent only estimates of the σ^0 and γ backscatters in the sense that they are expressed in terms of the radar incidence angle (i.e. the incidence angle with respect to the WSG84 ellipsoid adjusted by an average terrain height) and not in terms of the local incidence angle, which depends on the local terrain slope and is not known at the time of the processing.

Note also that due to the roll steering law (see Section 9.5) the range of incidence angles that each output range line spans varies in azimuth, causing in this way the application LUT and the calibration vectors (which are incidence angle dependent)

to vary in the azimuth direction, especially in the case of long strip images. Therefore, the calibration vectors are updated in the azimuth direction with a frequency which is a configurable input parameter.

9.20 Fourier Transform and Energy of a Signal

The following Fourier transform pair is typically used by a SAR processor. The forward transformation is:

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{(-j2\pi n k)/N} \quad (9-57)$$

The inverse transformation is:

$$x'(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) e^{(j2\pi n k)/N} \quad (9-58)$$

Note that, for efficiency reasons, most software scientific libraries (but not Matlab for instance) do NOT implement the division by N in the inverse transform formula.

If the inverse transform is implemented without $\frac{1}{N}$, the multiplication by this factor must be done explicitly in the code (typically by combining it with an already existing scalar factor). Given that this factor has a purely software significance, it will not be included in this algorithm description.

The energy of a discrete, finite, time domain signal is defined to be the sum of the magnitudes squared of the signal. Parseval's theorem states that the energy of a time domain signal equals the energy of its frequency domain equivalent:

$$\sum_{n=0}^{N-1} |x(n)|^2 = \frac{1}{N} \sum_{k=0}^{N-1} |X(k)|^2 \quad (9-59)$$

9.21 Look Extraction

The indices for a generic look extraction (in range or azimuth) are calculated from the following input parameters:

- BW = Processing bandwidth
- F = Frequency sampling
- BW_{look} = Look bandwidth
- $centreFreq$ = Centre frequency.
- N_{fft} = Forward FFT length

- N_{fft} = Inverse FFT Length
- N_L = Number of looks (> 1)

Note that the centre frequency, $centreFreq$, is always zero in the range case. In the azimuth case, $centreFreq$ is equal to the Doppler centroid, f_{η_c} , corresponding to the azimuth line for which the look extraction is performed.

To calculate the position of the looks within the spectrum perform the following:

1. **Calculate the frequency increment between frequency samples:**

$$\Delta f = \frac{F}{N_{fft}} \quad (9-60)$$

2. **Calculate the centre frequency of the first look:**

$$firstLookCentreFreq = centreFreq + \frac{BW}{2} - \frac{BW_{look}}{2} \quad (9-61)$$

3. **Calculate the start index of the first look** (note that this index can be negative and larger than the FFT length):

$$firstLookStartIndex = \left(\text{int} \left(\frac{firstLookCentreFreq}{\Delta f} - \frac{N_{fft}}{2} \right) \right) \quad (9-62)$$

4. **Calculate the start frequency of the first look:**

$$firstLookStartFreq = firstLookStartIndex \cdot \Delta f \quad (9-63)$$

5. **Readjust the look centre frequency** (due to the previous index rounding):

$$lookCentreFreq = firstLookStartIndex + \frac{N_{fft} \cdot \Delta f}{2} \quad (9-64)$$

6. **Transform $firstLookStartIndex$ to the set $\{0, \dots, N_{fft} - 1\}$:**

$$firstLookStartIndex = firstLookStartIndex \% N_{fft}, \quad (9-65)$$

If $firstLookStartIndex < 0$, then

$$firstLookStartIndex = firstLookStartIndex + N_{fft} \quad (9-66)$$

7. **Calculate the look offset index:**

$$lookStartOffset = \frac{BW - BW_{look}}{(N_L - 1) \cdot \Delta f} \quad (9-67)$$

8. **For each look of index $iLook$, $iLook < N_L$, calculate the start extraction index:**

$$iLookStartIndex = firstLookStartIndex + iLook \cdot lookStartOffset, \quad (9-68)$$

If $iLookStartIndex > N_{fft}$ then

$$iLookStartIndex = iLookStartIndex - N_{fft} \quad (9-69)$$

9. **Extract N_{fft} samples starting at index $iLookStartIndex$** (note that the samples that happen to exceed N_{fft} are extracted (circularly) from the beginning of the array).

9.22 Resampling Interpolator

Interpolation is required by various SAR processing algorithms like for example the RCMC and range resampling (Section 6.3.2 and 6.3.3), slant-to-ground-range resampling for GRD (Section 9.16.1) and azimuth pixel spacing resampling for GRD (Section 7.2).

All these operations are implemented using a sinc-based interpolation. The sinc kernel is truncated and weighted by a tapering window. For efficiency reasons, the size of the interpolator kernel should be as short as possible. However, since a short kernel can result in loss of radiometric and phase accuracy and radiometric artifacts, a trade-off between efficiency and accuracy must be achieved. A 16-point interpolator is considered adequate for realizing this trade-off.

Note that the interpolation filter is not explicitly calculated by the processor, but rather provided as a configurable input table. The location of the interpolation points is quantized to 1/64 of the input pixel spacing and therefore there are 64 interpolation kernels in the table.

[NORM] Each interpolation kernel in the table is normalized such that:

$$\sum_{i=0}^{N_k-1} h_i = 1, \quad (9-70)$$

Where $N_k = 16$ is the interpolation kernel length.

9.23 Bi-static Delay Correction

For Sentinel-1 (and SAR satellites in general), the azimuth slow time provided in the downlink for SAR echo data is the receive time of the pulse. In previous missions such as ENVISAT ASAR, this time was used in the processing as the azimuth time of the imaging. The relationship between the transmit and receive time of a pulse is as follows:

$$\eta_{rx} = \eta_{tx} + rank * pri \quad (9-71)$$

where:

- η_{rx} = Receive time (azimuth slow time), as annotated in downlink
- η_{tx} = Transmit time (azimuth slow time)
- $rank$ = The number of pulses transmitted before the first pulse is received
- pri = The azimuth slow time between pulses

For missions like ENVISAT ASAR, the processor implicitly assumes that the imaging of the earth's surface and the reception of the echo data happen simultaneously, which is not the case. The imaging of the ground occurs halfway between the transmission and reception of the pulse. This is called the "bi-static delay". In order to account for the bi-static delay in the processor and therefore improve the geolocation of the SAR image, the following correction is performed on the azimuth time prior to SLC processing:

$$\eta_{corrected} = \eta_{rx} - \frac{rank * pri}{2} \quad (9-72)$$

where:

- $\eta_{corrected}$ = Azimuth slow time, corrected for the bi-static delay
- η_{rx} = Receive time (azimuth slow time), as annotated in downlink
- $rank$ = The number of pulses transmitted before the first pulse is received
- pri = The azimuth slow time between pulses

For the Sentinel-1 IPF, the option to correct for the bi-static delay is configurable with an input parameter.

A LIST OF SYMBOLS

The following list provides the definitions of major symbols used in the document, typically in more than one section.

α	Normalized azimuth bandwidth ($0 < \alpha \leq 1$)
B_s	(TOPSAR) Processing sweep bandwidth [Hz]
η	Azimuth time (or slow time) [s]
F_a	Azimuth sampling frequency (= PRF for Stripmap SLC) [Hz]
F_r	Range complex sampling frequency [Hz]
f_η	Azimuth (Doppler) frequency [Hz]
$f_{\eta c}$	Absolute DC frequency [Hz]
$G(\cdot)$	Azimuth Antenna Pattern, as a function of azimuth angle [dB]
$G_{aap}(\cdot)$	Azimuth Antenna Pattern, as a function of either azimuth time or azimuth frequency (NOTE: The 2 functions are different, but for simplicity, the same notation will be used as the argument will discriminate between the two meanings)
k_a	Azimuth FM rate of point target signal [Hz/s]
K_{abs}	Absolute Calibration Constant
k_ψ	Antenna steering velocity [radians/s]
K_r	Range chirp FM rate [Hz/s]
k_s	Doppler Centroid rate introduced by the scanning of the antenna [Hz/s]
L	Antenna length [m]
L_{el}	Antenna element length [m]

M_{fft}	Azimuth FFT length
N_{fft}	Range FFT length
PRF	Pulse Repetition Frequency [Hz]
R_0	Slant range of closest approach (i.e. the slant range when the radar is closest to the target) [m]
R	Slant range [m]
S_i	Application scaling factor for range pixel i
τ	Range (fast) time [s]
T_b	(TOPSAR) Burst time duration [s]
V_g	Ground velocity (a positive scalar representing the antenna beam footprint velocity when projected onto the ground) [m/s]
V_r	Effective radar velocity (see Section 9.10) [m/s]
V_s	Satellite velocity (the norm of the satellite velocity vector expressed in ECR coordinate) [m/s]

B 2D DATA DOMAINS

Table B-1 2D Data Domains

	Section	Input 2D Data Domain	Working 2D Data Domain	Output 2D Data Domain
1.	Section 5 DOPPLER CENTROID ESTIMATION ALGORITHMS			
2.	Section 5.1 Absolute DC	N/A	N/A	N/A
3.	Section 5.2 Fine DC Estimation	See sub-sections below.		
4.	Section 5.2.1 DCE Pre-Conditioning (TOPSAR only)	(τ, η)	(τ, η)	(τ, η)
5.	Section 5.2.2 Correlation DC Estimator (CDCE)	(τ, η)	N/A	N/A
6.	Section 5.3 Fine DC Estimates Unwrapping	N/A	N/A	N/A
7.	Section 5.4 Absolute DC Estimation	N/A	N/A	N/A
8.	Section 5.5 Polynomial Fitting	N/A	N/A	N/A
9.	Section 6 SLC PROCESSING ALGORITHMS			
10.	Section 6.1 Range Processing	(τ, η)	(f_τ, η)	(τ, η)
11.	Section 6.2 Azimuth Pre-Processing	See sub-sections below.		
12.	Section 6.2.1 Azimuth Zero-Padding	(τ, η)	N/A	(τ, η)

	Section	Input 2D Data Domain	Working 2D Data Domain	Output 2D Data Domain
13.	<p>9.23.2 Section 6.2.2 Range Compression</p> <p>Each range line is compressed using the following steps:</p> <ol style="list-style-type: none"> Zero-pad the range line to N_{fft} length. Transform the line into the frequency domain using an FFT. Multiply the result by the RRF (which has already been generated with the same size N_{fft}; see Section 6.1.1). Perform the inverse FFT. Calculate the required black fill, as described in Section 6.2.2.1. The valid portion of the range compressed line (i.e. after matched filter throw-away) is then written to the output range processing buffer (by taking into account the black fill calculated as described in Section 6.2.2.1, and the range line length as described in Section 6.2.2.2). <p>9.23.2.1 Black-Fill Calculation</p> <p>The black-fill of the compressed range lines is a consequence of (potential) changes over time in Sampling Window Start Time (SWST).</p> <p>The SWST is the time offset between the start time of the transmitted pulse and the start time of the current range window. The SWST may change multiple times over a given azimuth duration, and this variation must be</p>	(τ, η)	N/A	(τ, f_η)
	<p>ESA Unclassified – For Official Use</p>			B-2

	Section	Input 2D Data Domain	Working 2D Data Domain	Output 2D Data Domain
14.	Section 6.2.4 Azimuth Frequency UFR (TOPSAR only)	(τ, f_η)		(τ, f_η)
15.	Section 6.3 Azimuth Processing	See sub-sections below.		
16.	Section 6.3.1 Secondary Range Compression (SRC)	(τ, f_η)	(f_τ, f_η)	(τ, f_η)
17.	Section 6.3.2 Range Cell Migration Correction (RCMC)	(τ, f_η)	(τ, f_η)	(τ, f_η)
18.	Section 6.3.4 Azimuth Compression	(τ, f_η)	(τ, f_η)	(τ, η)
19.	Section 6.4 Azimuth Post-processing (TOPSAR only)	See sub-sections below.		
20.	Section 6.4.1 Azimuth Time UFR	(τ, η)	(τ, η)	(τ, η)

A SUMMARY OF SENTINEL-1 IPF NORMALISATION

This appendix lists all of the processor normalisation scale factors applied by the IPF for the generation of Sentinel-1 L1 products. No new information is contained in this appendix; rather, a summary of information presented incrementally throughout the document is presented in a tabular format for reference purposes.

For each of the scaling factors or vectors, the following information is tabulated:

- A description of the scale factor
- The main processing stage where the scaling is applied.
- A reference to the main document section citing where the correction is applied in the context of the overall algorithm

Note that in case the scientific library employed implements the inverse Fourier transform **without** the $\frac{1}{N_{fft}}$ factor (see Section 9.20), at the time of the software implementation, every time an IFFT is performed, scaling by this factor should be added.

Table A-1 Normalisation Steps Summary

Normalisation Step	Proc. Stage	Section
Drift compensation	SLC	Section 6.1, Step 2c)
Range Matched Filter Energy Normalisation	SLC	Section 6.1.1, Steps 6 and 7
Antenna Elevation Pattern Correction	SLC	Section 6.1.2, Step 1
Range Spreading Loss Correction	SLC	Section 6.1.2, Step 2
Azimuth Matched Filter Energy Normalisation	SLC	Section 6.3.4.1.2, Step 8
Normalisation for the azimuth compression	SLC	Section 6.3.4.1.2, Step 9
Scaling by the (input) processor gain (note that the scaling value may be different depending on whether the final product is an SLC or a GRD)	SLC	Section 6.3.4.1.2, Step 10
Multiple Range Look Weighting	PostProc	Section 7.1, Step 4
Range FFT/IFFT scaling	PostProc	Section 7.1, Step 5
Multiple Azimuth Look Weighting and Antenna Pattern	PostProc	Section 7.2, Step 7
Azimuth FFT/IFFT Scaling	PostProc	Section 7.2, Step 8

Normalisation Step	Proc. Stage	Section
Interpolator Scaling	SLC PostProc	Section 9.22

B SUMMARY OF AUXILIARY AND INTERNAL PARAMETERS REFERENCED IN THE DOCUMENT

This section provides a list of the configurable parameters used by the Sentinel-1 IPF algorithm. For each parameter listed in Table B-1 the following information is also provided:

- The section in which the parameter is mentioned
- The symbol by which the parameter is referenced in the document (if any)
- The type of input file where the parameter is defined, which is abbreviated by one of the following strings:
 - AUX_PP1 - L1 Processor Parameters Auxiliary Data File
 - AUX_CAL - Calibration Auxiliary Data File
 - AUX_INS - Instrument Auxiliary Data File
 - Internal - Internal Parameter File

Note that in general, the configurable parameters are beam, mode and polarisation dependent.

Note that all satellite position, velocity and attitude vectors that are mentioned in the document are derived by interpolation either from the downlink or from an input orbit parameter file.

Table B-1 Summary of Auxiliary and Internal Parameters Referenced in the Document

Parameter	Section	Document Referenced Symbol	Source
Source of the reference replica (reconstructed or nominal)	Section 4.2.1.4 Replica Extraction and Internal Time Delay Estimation	<i>rrfSource</i>	AUX_PP1
Flag indicating that the drift compensation should be calculated from the PG product model (and not from the PG product derived from calibration pulses)	Section 4.2.1.6.2 Drift Compensation from PG Product Model	<i>getDriftFromPGmodel</i>	AUX_PP1
Deviation from the mean allowed for the PG product (amplitude and phase), measured as a fraction of the standard deviation.	Section 4.2.1.8 PG Product Validation	<i>pgAmpStdFraction</i> <i>pgPhaseStdFraction</i>	AUX_PP1
Maximum deviation allowed for a PG product value (amplitude and phase) from the corresponding PG product model value.	Section 4.2.1.8 PG Product Validation	<i>maxPgAmpError</i> <i>maxPgPhaseError</i>	AUX_PP1
Maximum fraction of total number of PG product values allowed to be invalid. If exceeded, the PG product model will be used instead.	Section 4.2.1.8 PG Product Validation	<i>invalidPgMaxFraction</i>	AUX_PP1
Source of the DC frequency calculated by IPF, to be used in the subsequent steps	Section 5 DOPPLER CENTROID ESTIMATION ALGORITHMS	None	AUX_PP1

Parameter	Section	Document Referenced Symbol	Source
Pre-defined DC frequency polynomial	Section 5 DOPPLER CENTROID ESTIMATION ALGORITHMS	None	AUX_PP1
Flag to control the application of the elevation antenna pattern correction	Section 6.1.2 Range Dependent Gain Correction	None	AUX_PP1
Target instantaneous Doppler Bandwidth	Section 6.2.4 Azimuth Frequency UFR	B_d	AUX_PP1
Processing Gain	Section 6.3.4.1.2 Algorithm Implementation Section 9.17 Bank of Thermal Noise Estimation	k_{proc}	AUX_PP1
Nominal beamwidth	Section 6.3.4.1.2 Algorithm Implementation	$nomBeamWidth$	AUX_PP1
Post-processing output range pixel spacing Post-processing range bandwidth Post-processing range look bandwidth Number of range looks Range weighting Hamming window coefficient	Section 7.1 Post-Processing Range Processing	None	AUX_PP1
Post-processing output azimuth pixel spacing Post-processing processing azimuth bandwidth Post-processing azimuth look bandwidth Number of azimuth looks Azimuth weighting Hamming window coefficient	Section 7.2 Post-Processing Azimuth Processing	None	AUX_PP1
Number of range samples to average for the purpose of producing QL images Number of azimuth samples to average for the purpose of producing QL images	Section 7.3 Post-Processing	None	AUX_PP1
Flag to enable/disable the generation of the QL image	Section 7.3.3 Quick-Look (QL) Image Generation	None	AUX_PP1 AUX_PP1
Flag to control the scaling of the image by the application LUT	Section 7.3.4 Output Processing	None	AUX_PP1
Flag to control the correction of the gain imbalance between I and Q channels	Section 9.2 Raw Data Correction	None	AUX_PP1
Flag to control the correction of the non-orthogonally between I and Q channels	Section 9.2 Raw Data Correction	None	AUX_PP1
The type and coefficient of each weighting window applied in the processor	Section 9.7 Weighting Window	a	AUX_PP1
Azimuth Antenna Element Pattern (AAEP) LUT	Section 7.2.1.1 De-scalloping Function	AAEP LUT	AUX_CAL
Elevation Antenna Pattern LUT	Section 9.5 Bank of Elevation Antenna Pattern (EAP) Correction Vectors	EAP LUT	AUX_CAL
Azimuth Antenna Pattern (AAP) LUT	Section 9.15 Azimuth Antenna Pattern	AAP LUT	AUX_CAL
Noise calibration factor	Section 9.17 Bank of Thermal Noise Estimation	k_{noise}	AUX_CAL
Absolute calibration constant	Section 9.19 Absolute Calibration Vectors	K_{abs}	AUX_CAL

Parameter	Section	Document Referenced Symbol	Source
Parameters supporting EAP correction vectors: 1. Antenna boresight angle at reference altitude 2. Roll steering sensitivity versus altitude 3. Reference satellite altitude	Section 9.5 Bank of Elevation Antenna Pattern (EAP) Correction Vectors	1. θ_{ref} 2. α_{roll} 3. H_{ref}	AUX_INS
Modeled PG product	Section 4.2.1.6 Instrument Drift Derivation	\mathbf{P}_{modZZ} where ZZ is HH, HV, VV and VH	AUX_INS
Nominal chirp phase coefficients	Section 4.2.1.1.1 Nominal Image Replica Generation	$\varphi(i), i = 0,1,2,3$	AUX_INS
Time of the transmitted pulse within calibration data packet	Section 4.2.1.4 Replica Extraction and Internal Time Delay Estimation	Δt_{guard}	AUX_INS
Time of suppressed decimation filter transients within calibration data packet	Section 4.2.1.4 Replica Extraction and Internal Time Delay Estimation	Δt_{suppr}	AUX_INS
Radar wavelength	Section 5.1 Absolute DC Section 9.11 Azimuth FM Rate Section 9.15 Azimuth Antenna Pattern	λ	AUX_INS
SWST bias	Section 6.1.1 Range Reference Function (RRF)	$\Delta t_{SWSTbias}$	AUX_INS
Degree and coefficients of the phase and amplitude defining the nominal replica	Section 9.3 Nominal Replica	None	AUX_INS
Normalized Reconstruction Levels LUT for BAQ and FDBAQ raw data decoding	Section 9.1 Raw Data Decoding	None	AUX_INS
Sigma Factors LUT for BAQ and FDBAQ raw data decoding.	Section 9.1 Raw Data Decoding	None	AUX_INS
Huffman Decoding trees for FDBAQ raw data decoding.	Section 9.1 Raw Data Decoding	None	AUX_INS
Threshold index levels for BAQ and FDBAQ Decoding	Section 9.1 Raw Data Decoding	None	AUX_INS
The set of antenna azimuth steering angles (one angle per each PRI in a burst) (One set of azimuth steering angles for each TOPSAR swath)	Section 9.14 Antenna Steering Rate and DC Rate (TOPSAR only)	$(\psi_i)_{i=0,1,\dots,N-1}$	AUX_INS
Nominal transmit pulse length (chirp duration) to use when limiting the processing bandwidth for spurious signal filtering.	Section 9.2.2 Spurious Signal Correction	$TXPL_{nom}$	AUX_INS
Thresholds for the 3dB width, PSLR and ISLR of the replica compressed using the nominal replica. Used for validating the replica.	Section 4.2.1.4 Replica Extraction and Internal Time Delay Estimation	None	Internal
Terrain Height Azimuth vector entry spacing	Section 4.4 Terrain Height Function	None	Internal
Terrain Height Azimuth block size	Section 4.3.1 Missing Line Detection	None	Internal
Length of the FFT used in the DCE unwrapping algorithm	Section 5.3 Fine DC Estimates Unwrapping	None	Internal
Azimuth direction spacing between DC estimates Sec 5.9 too.	Section 5.6 Processing Blocks Dimensions	None	Internal
Azimuth and range sizes of the blocks of data to be used for DCE estimation.	Section 5.6 Processing Blocks Dimensions	None	Internal
Number of estimates in the range direction (equal to the number of range blocks of data for DCE estimation)	Section 5.6 Processing Blocks Dimensions	None	Internal

Parameter	Section	Document Referenced Symbol	Source
Flag to control the application of the range spreading loss correction	Section 6.1.2 Range Dependent Gain Correction	None	Internal
Azimuth FFT length	Section 6.2.1 Azimuth Zero-Padding Section 9.12 Focusing Azimuth Block Overlap (Stripmap only)	M_{fft}	Internal
The amount of azimuth zero-padding expressed as a multiplicative factor for the number of azimuth samples in a burst. (TOPSAR only)	Section 6.2.1 Azimuth Zero-Padding Section 9.12 Focusing Azimuth Block Overlap (Stripmap only)	None	Internal
Number of range segments for the purpose of the SRC.	Section 6.3.1 Secondary Range Compression (SRC)	None	Internal
Length of range segments for the purpose of the SRC.	Section 6.3.1 Secondary Range Compression (SRC)	None	Internal
Filter Bank	Section 6.4.1 Azimuth Time UFR	None	Internal
GRD Block overlap (Stripmap only)	Section 7 POST-PROCESSING ALGORITHMS	$T_{grdBlkOv}$	Internal
GRD Block length (Stripmap only)	Section 7 POST-PROCESSING ALGORITHMS	None	Internal
GRD azimuth block overlap	Section 7 POST-PROCESSING ALGORITHMS	None	Internal
Flag to control the slant-range-to-ground-range conversion	Section 7.1 Post-Processing Range Processing	None	Internal
Image output pixel type QL image output pixel type	Section 7.3.4 Output Processing	None	Internal
Azimuth bloc overlap	Section 8.2 Internal Signal Data Slice Definition	T_{blkOv}	Internal
Maximum azimuth matched filter	Section 9.12 Focusing Azimuth Block Overlap (Stripmap only)	A	Internal
Maximum Delta Azimuth block overlap	Section 9.12 Focusing Azimuth Block Overlap (Stripmap only)	ΔO_v	Internal
Frequency of update of the EAP in the azimuth direction.	Section 9.5 Bank of Elevation Antenna Pattern (EAP) Correction Vectors	None	Internal
Reference slant range for the purpose of the range spreading loss compensation	Section 9.6 Range Spreading Loss Correction	R_{ref}	Input
Frequency of update of the azimuth FM rate in the range direction	Section 9.11 Azimuth FM Rate	None	Internal
Frequency of update in the azimuth direction of the GR/SR LUT	Section 9.16 Bank of GR/SR LUTs	None	Internal
Application LUTs	Section 9.18 Application Scaling	None	Internal
Frequency of update in the azimuth direction of the calibration vectors	Section 9.19 Absolute Calibration Vectors	None	Internal
Interpolator table	Section 9.22 Resampling Interpolator	None	Internal