

# NWP SAF

## *Satellite Application Facility for Numerical Weather Prediction*

Document NWPSAF-KN-UD-002

Version 2.1

11-11-2010

## SDP User Manual and Reference Guide

### *Scat group*

*Jos de Kloe, Marcos Portabella, Ad Stoffelen, Anton Verhoef, Jeroen Verspeek,  
and Jur Vogelzang*

***KNMI, De Bilt, The Netherlands***



<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

## SDP User Manual and Reference Guide

### Scat group

Jos de Kloe, Marcos Portabella, Ad Stoffelen, Anton Verhoef, Jeroen Verspeek,  
and Jur Vogelzang

KNMI, De Bilt, The Netherlands

This documentation was developed within the context of the EUMETSAT Satellite Application Facility on Numerical Weather Prediction (NWP SAF), under the Cooperation Agreement dated 16 December, 2003, between EUMETSAT and the Met Office, UK, by one or more partners within the NWP SAF. The partners in the NWP SAF are the Met Office, ECMWF, KNMI and Météo France.

Copyright 2006, EUMETSAT, All Rights Reserved.

Change record			
Version	Date	Author / changed by	Remarks
0.0	Oct 2004	Hans Bonekamp	First draft
1.0	May 2005	Hans Bonekamp	Beta release
1.1	11-01-2006	Jur Vogelzang	Beta release
1.2	27-03-2006	Jur Vogelzang	First public release
1.3	04-09-2006	Jur Vogelzang	Routines moved from SwsSupport to genscat; index types removed; some typo's corrected.
1.4	05-04-2007	Jur Vogelzang	New 2DVAR, improved inversion
1.4a	09-05-2007	Jur Vogelzang	Improved description of KNMI BUFR format and flag handling
1.5	24-07-2007	Jur Vogelzang	Rewrote use under Windows; rewrote sections 3.2
1.99	10-03-2008	Jur Vogelzang	Beta test version of SDP 2.0 featuring: - outer swath processing - flexible 2DVAR batch grid definition - improved handling of JPL rain flag - new KNMI BUFR file definition - new recommended settings of 2DVAR error model - memory leak repaired.
2.0	14-11-2008	Jur Vogelzang	Implemented comments from van Geffen's and OSI SAF beta tests; added warning regarding comparison with OSI SAF products
2.1	11-11-2010	Jur Vogelzang	Included OSI SAF ice model

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

# Contents

<b>Preface</b> .....	<b>5</b>
<b>1 Introduction</b> .....	<b>9</b>
1.1 Aims and scope .....	9
1.2 Development of SDP .....	9
1.3 Testing SDP .....	10
1.4 User Manual and Reference Guide .....	10
1.5 Conventions .....	11
<b>2 SDP User Manual</b> .....	<b>13</b>
2.1 Why using the SDP program ? .....	13
2.2 Modes of using SDP .....	18
2.3 Installing SDP .....	19
2.3.1 Directories and files .....	20
2.3.2 Environment variables .....	21
2.3.3 Installing BUFR library .....	23
2.3.4 Compilation and linking .....	23
2.4 Command line options .....	25
2.5 Scripts .....	29
2.6 Testruns .....	29
2.7 Documentation .....	31
2.8 Some remarks for Cygwin users .....	31
<b>3 SDP Product Specification</b> .....	<b>33</b>
3.1 Purpose of program SDP .....	33
3.2 Output specification .....	33
3.2.1 Flag settings .....	35
3.3 Input specification .....	35
3.4 System requirements .....	35
3.5` Details of functionality .....	36

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

3.5.1	BUFR IO and coding	36
3.5.2	Output resolution	37
3.5.3	Quality Control	37
3.5.4	Inversion	38
3.5.5	Ambiguity Removal	38
3.5.6	Monitoring	39
3.5.7	Ice model	39
3.6	Details of performance	39
3.7	Warning regarding comparison with OSI SAF products	40
<b>4</b>	<b>Program Design</b>	<b>41</b>
4.1	Top Level Design	41
4.1.1	Main program	41
4.1.2	Layered model structure	42
4.1.3	Data structure	45
4.1.4	Quality flagging and error handling	46
4.1.5	Verbosity	47
4.2	Module Design for genscat layer	47
4.2.1	Module <i>inversion</i>	47
4.2.2	Module <i>icemodel</i>	47
4.2.3	Module <i>ambrem</i>	48
4.2.4	Module <i>Bufrmod</i>	48
4.2.5	Support modules	48
4.3	Module Design for SeaWinds layer	49
4.3.1	Module <i>SwsData</i>	49
4.3.2	Module <i>SwsBufr</i>	56
4.3.3	Module <i>SwsSupport</i>	57
4.4	Module design for process layer	58
4.4.1	Module <i>SdpSupport</i>	58
4.4.2	Module <i>SdpIO</i>	59
4.4.3	Module <i>SdpPrePost</i>	59
4.4.4	Module <i>SdpInversion</i>	61
4.4.5	Module <i>SdpIcemodel</i>	62
4.4.6	Module <i>SdpAmbrem</i>	62
4.5	Flag use	63
<b>5</b>	<b>Inversion module class</b>	<b>65</b>
5.1	Background	65
5.2	Routines	66
5.3	Antenna direction	67

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

<b>6</b>	<b>Ambiguity Removal module class</b>	<b>69</b>
6.1	Ambiguity Removal	69
6.2	Module <i>Ambrem</i>	70
6.3	Module <i>BatchMod</i>	70
6.4	The KNMI 2DVAR scheme	73
6.4.1	Introduction	73
6.4.2	Data structure, interface and initialization	74
6.4.3	Reformulation and transformation	77
6.4.4	Module <i>CostFunc</i>	77
6.4.5	Adjoint method	78
6.4.6	Structure Functions	78
6.4.7	Minimisation	80
6.4.8	SingletonFFT_Module	81
6.5	The PreScat scheme	82
<b>7</b>	<b>Module <i>BufrMod</i></b>	<b>83</b>
7.1	Background	83
7.2	Routines	83
7.3	Data structures	85
7.4	Libraries	87
7.5	BUFR table routines	88
7.6	Center specific modules	88
<b>8</b>	<b>Module <i>iceModelMod</i></b>	<b>89</b>
8.1	Background	89
8.2	Routines	90
8.3	Data structures	91
8.4	Parameters	92
	<b>References</b>	<b>95</b>
	<b>Appendix A Calling tree for SDP</b>	<b>99</b>
	<b>Appendix B1 Calling tree for inversion routines</b>	<b>109</b>
	<b>Appendix B2 Calling tree for AR routines</b>	<b>113</b>
	<b>Appendix B3 Calling tree for BUFR routines</b>	<b>119</b>
	<b>Appendix B4 Calling tree for iceModelMod routines</b>	<b>123</b>

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

**Appendix C1 NOAA BUFR output file ..... 125**

**Appendix C2 KNMI BUFR output file ..... 129**

**Appendix D ECMWF BUFR data routines ..... 133**

**Appendix E Acronyms ..... 137**

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

# Preface

## Preface to version 1.0

Software code for processing satellite data may become very complex. On the one hand, it consists of code related to the technical details of the satellite and instruments, on the other hand, the code drives complex algorithms to create the physical end products. Therefore, the EUMETSAT Satellite Application Facility (SAF) project for Numerical Weather Prediction (NWP) has included some explicit activities aiming at enhancing the modularity, readability and portability of the processing code.

For several years, the KNMI observation research group has been developing processing code to supply a Near Real Time (NRT) level 2 surface wind product based on the SeaWinds Scatterometer level 1 Normalized Radar Cross Section data ( $\sigma_0$ ). This work is coordinated and supervised by Ad Stoffelen. In the beginning only an adaptation of his ERS code existed. Later Marcos Portabella and Julia Figa added modifications and extensions to improve, e.g., the wind retrieval and quality control algorithms. In 2003, John de Vries finished the first official release of a processor within the NWP SAF. This processor is called the QuikSCAT Data Processor (QDP). QDP is available for the meteorological community since spring 2004. Several users run QDP operationally. At KNMI, Anton Verhoef is running QDP and providing support for QDP as part of Initial Operational Phase (IOP) of the Ocean Sea-Ice (OSI) SAF wind product.

Meanwhile, Jos de Kloe has been updating the code for ERS and ASCAT scatterometer wind processing. For many parts of the process steps (e.g., the BUFR handling and part of the wind retrieval) a large overlap with SeaWinds Data processing coding exists. The KNMI SCAT group is working towards generic NRT scatterometer processing. As a result, a new modular processing code for SeaWinds data has been developed within the NWP SAF IOP. The working name of this code is currently the SeaWinds Data Processor (SDP). This document is the corresponding reference manual. I hope this manual will strongly contribute to the comprehension of future developers and of users interested in the details of the processing.

Hans Bonekamp, October 2004

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

### **Preface to version 1.1**

This is the first version of the SDP User Manual and Reference Guide that will be distributed to a larger audience, as deliverable in the NWP SAF project. After Hans Bonekamp left to EUMETSAT, Jos de Kloe, Marcos Portabella, and Anton Verhoef (as beta tester) continued working on the SDP code. They removed a number of bugs and made a lot of improvements: memory management was revised and the Generic Wind Section BUFR format was introduced. My role was to adapt the first draft of this document. With the help of Jos and Anton I found my way into the code. I made a number of adaptations and extensions to the original text, but left the underlying structure of the document unchanged.

The reader is kindly invited to give his comments in order to improve future versions of this document.

Jur Vogelzang, September 2005

### **Preface to version 1.2**

Version 1.2 will be the first public version of SDP. The recommendations made by EUMETSAT during the Delivery Readiness Inspection in November 2005 were all implemented. Moreover, almost all known problems have been solved. The reader is kindly invited to give his comments in order to improve future versions of this document.

Jur Vogelzang, March 2006

### **Preface to version 1.3**

Version 1.3 is an update of the first public version of SDP. Some routines in modules SwsSupport and Ambrem2DVAR were moved to genscat, which led to some differences in the program structure. The index\_type datatype is no longer needed and has been removed. The importance of setting the environment variables to their proper values during compilation and linking has been stressed. The inversion module has been improved at very low wind speed and flag management has been revised. Some typo's were corrected.

Jur Vogelzang, September 2006

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

## **Preface to version 1.4**

In version 1.4 of SDP the two dimensional variational ambiguity removal method (2DVAR) has been completely revised. Some serious errors have been corrected. As a consequence, chapter 6 has been adapted. The presentation of the equations governing 2DVAR has been moved to a separate report. The inversion has been improved for small wind speeds. The program structure and organization has been changed. Some small changes were made in the command line arguments of SDP: some unused commands are removed and a new one is added for reading the 2DVAR parameter values from file.

Jur Vogelzang, April 2007

## **Preface to version 1.4a**

The description of the KNMI BUFR output file format is improved. The differences between the NOAA file format and the KNMI format are stressed, notably regarding the solution probability. A description of the use of flags by SDP is included. Some minor changes in the code structure are documented. All references to web addresses were checked and corrected where needed.

Jur Vogelzang, May 2007

## **Preface to version 1.5**

The difficulties with installing SDP under Cygwin have been solved to a sufficient level. Though the minimalization strategy has been altered substantially, this does not have any impact on the documentation. For the sake of completeness, the `-ocf` option, a new command option for dumping the observation cost function, has been included in section 2.5. Like the `-ana` option, the `-ocf` option is intended for research purposes. A description of how the lookup tables for the expected MLE's were obtained has been added to section 2.3.2. Section 3.2 (output specification) has been updated and corrected.

Jur Vogelzang, July 2007

## **Preface to version 1.99**

This is the beta test version of SDP 2.0, the first major update of SDP. Thanks to the new NOAA BUFR file definition, it is now possible to process the outer swath via the command line option `-allswath`. However, also the definition of the MLE was changed. Therefore the processing at 50 km and 100 km resolution is now preceded by a processing step at 25 km resolution in order to determine the MLE values. This gives rise to some extra command line options. The radix 2 Fast Fourier Transform in SDP version 1.5 and older has been replaced by the mixed radix routine of Singleton converted to Fortran 90. Now both size and dimension of the 2DVAR grid can be changed using the `-par` command line option. At 25 km resolution, the JPL rain flag is now retained (it was deleted in earlier SDP versions); at 50 km and 100 km resolution the JPL rain flag is still deleted. The command line option for setting the resolution has been changed.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

Note that the default options in SDP 2.0 differ from those in the older versions. The rationales behind this are described extensively in NWPSAF Technical Reports given in the references and downloadable from the NWPSAF website. Finally, a memory leak in module SdpIO has been repaired.

Jur Vogelzang, March 2008

## **Preface to version 2.0**

The documentation was adapted in order to meet the recommendations from the beta tests by van Geffen and by the OSI SAF. Paragraph 2.8 for Cygwin users and subsection 3.2.1 on flag settings were added. All comparisons with QDP were removed since there won't be very much QDP users left. An important change is the new definition of the KNMI BUFR format. Users who employ SDP output in this format are recommended to check if their applications still refer to the correct BUFR items, see Appendix C2. A warning regarding comparison of SDP output with OSI SAF wind products is added in section 3.7: there may be differences due to different background winds and/or ice screening procedure.

Jur Vogelzang, October 2008

## **Preface to version 2.1**

Though SeaWinds on QuikSCAT ended its highly successful mission almost a year ago due to antenna failure, SDP will be maintained for reprocessing purposes. It has therefore been extended with a Bayesian ice model developed in the context of the Ocean and SeaIce Sattelite Application Facility (OSI SAF). The model is described in chapter 8.

Jur Vogelzang, November 2010

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

# Chapter 1

## Introduction

### 1.1 Aims and scope

The SeaWinds Data Processor (SDP) is a software package written in Fortran90 for handling data from the SeaWinds scatterometer instruments. Details of these instruments can be found on several sites and in several other documents. Important references are listed at the end of this section.

SDP generates surface winds based on SeaWinds data. In particular, it allows performing the ambiguity removal with the 2DVAR method and it supports the MSS scheme, as an alternative to the DIRTH scheme employed by NOAA. The output of SDP consists of wind vectors which represent surface winds within the ground swath of the scatterometer. Input of SDP are Normalized Radar Cross Section (NRCS,  $\sigma_0$ ) data. These data may be real-time. The input and output files of SDP are in BUFR format.

For SeaWinds on QuikSCAT the data are available for the period Nov 1999 to Nov 2009. Unfortunately, due to its failure after 9 months, a ready to use real-time (BUFR, see subsection 3.5.1) product for Seawinds on Adeos II is not available.

More information can be found in [*Kerkmann, 1998; Leidner et al., 2000; Portabella, 2002; Stoffelen, 1998*].

### 1.2 Development of SDP

SDP is developed within the NWP SAF IOP program as code which can be run in an operational setting. The coding is in Fortran 90 and has followed the procedures specified for the NWP SAF.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

Table 1.1 provides an overview of the persons involved in the development. Special attention has been paid on robustness and readability. SDP may be run on every modern UNIX or LINUX machine. SDP can also be run on a Windows machine under Cygwin as UNIX emulator (MinGW proved to be too limited to support SDP).

<b>Task</b>	<b>Person</b>
Coordinator	Ad Stoffelen
Lead Investigator	Hans Bonekamp, Jur Vogelzang
Development Team	Hans Bonekamp, Jos de Kloe, Anton Verhoef, Jur Vogelzang
Integrator	Hans Bonekamp, Jur Vogelzang
Project Team Leader	Ad Stoffelen
Beta testers	Ferry van Geffen (Netherlands) Marcos Portabella and Anton Verhoef (OSI SAF)
Reviewers	Ad Stoffelen, Jos de Kloe, Marcos Portabella

**Table 1.1** Overview of development tasks.

### 1.3 Testing SDP

Modules are tested by test programs and test routines. Many test routines or test support routines are part of the modules themselves. Test programs can be compiled separately. For the SDP program, the description of the test programs and the results of the testing are reported in [SCAT group, 2008]. The quality of the SDP output wind field is investigated in [Vogelzang, 2006, 2007b]. Some additional tests on the 2DVAR settings are reported by Vogelzang [2008]

### 1.4 User Manual and Reference Guide

This document is intended as the complete reference book for SDP.

Chapter 2 is the user manual (UM) for the SDP program. This chapter provides the basic information for installing, compiling, and running SDP.

Chapter 3 contains the Product Specification (PS) of the SDP program. Reading the UM and the PS should provide sufficient information to the user who wants to apply the SDP program as a black box.

The subsequent chapters are of interest to developers and users who need more specific information on how the processing is done. The Top Level Design (TLD) of the code and the Module Design (MD) of the SDP code can be found in chapter 4.

Several modules are very generic for NRT scatterometer data processing. Examples are the modules for the BUFR handling, ambiguity removal, and parts of the wind retrieval. These generic modules are part of the genscat layer and are described in chapters 5, 6 and 7.

The appendices of this document contain a complete calling tree of the SDP program up to and including the genscat layer. The appendices also contain a list of SeaWinds BUFR data

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

descriptors, a list of the ECMWF BUFR routines, and a list of acronyms.

Finally, many sections end with a remarks alinea. Mostly, the remarks contain some recommendations for future development. These remarks may be reconsidered in future versions of this reference book.

## 1.5 Conventions

Names of physical quantities (e.g., wind speed components  $u$  and  $v$ ), modules (e.g. *BufrMod*), subroutines and identifiers are printed italic.

Names of directories and subdirectories (e.g. /SDP/sdp), files (e.g. sdp.F90), and commands (e.g. sdp -f input) are printed in Courier. When addressing software systems in general, the normal font is used (e.g. SDP, genscat).

Hyperlinks are printed in blue and underlined (e.g. [www.knmi.nl/scatterometer](http://www.knmi.nl/scatterometer)).

References are in square brackets with the name of the author italic (e.g. [*Stoffelen*, 1998]).

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

## Chapter 2

# SDP User Manual

This chapter is the user manual of the SDP program. The SDP program is the follow-up of the QDP program [*de Vries et al.*, 2004]. SDP has extended capabilities, such as higher resolution and the Multi Solution Scheme (MSS).

Section 2.2 provides information on how to install, compile, and link the SDP software. The command line arguments of SDP are discussed in section 2.3. Section 2.4 gives information on some scripts for running SDP that are part of this release.

### 2.1 Why using the SDP program ?

Scatterometers provide valuable observational data over the world's oceans. Therefore, successful assimilation of scatterometer data in numerical weather prediction systems generally improves weather forecasts. The SDP program has been developed to fully exploit scatterometer data. It is meant to form the key component of the observation operator for surface winds in data assimilation systems.

The general scheme of SDP (and any other wind scatterometer data processor is given in figure 2.1. The input of the SDP program is the NOAA SeaWinds level 2b BUFR wind product. However, only the level 1 data contained in the NOAA (the  $\sigma_0$  values) are used in SDP.

The SDP processing chain contains five steps (see figure 2.1):

1. Pre-processing. The input BUFR file is decoded and the  $\sigma_0$  values are written in the data structures of SDP.
2. Inversion. The  $\sigma_0$  values are compared to the Geophysical Model Function (GMF) by means

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

of a Maximum Likelihood Estimator (MLE). The wind vectors that give the best description of the  $\sigma_0$  values (the solutions) are retained. The MLE is also used to assign a probability to each wind vector. The normal scheme allows 4 solutions at most, but in the Multi Solution Scheme (MSS) the maximum number of solutions is 144.

3. Quality Control. Solutions that lie far away from the GMF are likely to be contaminated by rain, sea ice, and/or confused sea state. During Quality Control these solutions are identified and flagged.
4. Ambiguity Removal. This procedure identifies the most probable solution using some form of external information. SDP uses a two-dimensional variational scheme (2DVAR) as default. A cost function is minimized that consists of a background wind field and all solutions with their probability, using mass conservation and continuity as constraints. The background wind field is obtained from the NCEP model winds in the NOAA SeaWinds level 2b product (the input file of SDP).
5. Quality Monitoring. The last step is to write the results in BUFR format and to output quality indicators.

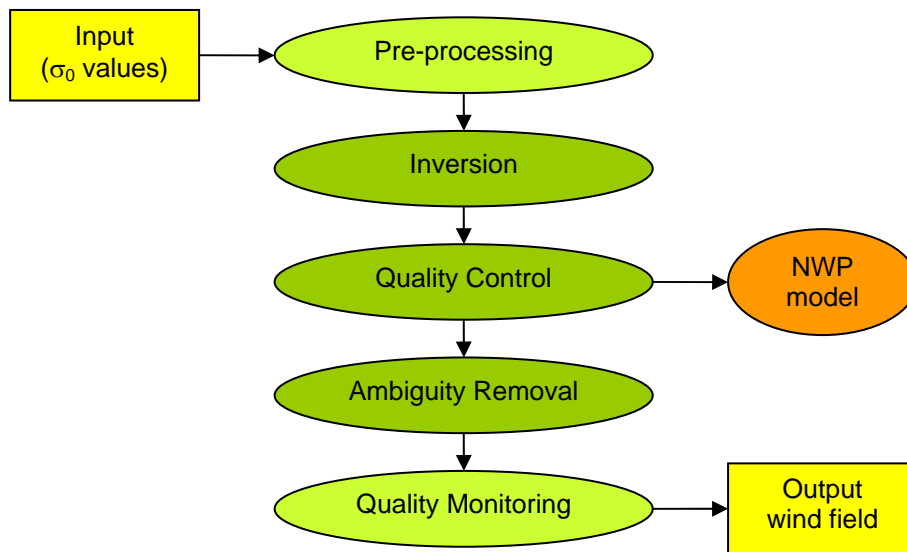


Figure 2.1 SDP processing scheme. When using MSS the wind vectors and their probabilities after Quality Control may be fed directly in the Data Assimilation step of a Numerical Weather Prediction model.

Step 1 and 5 of the processing chain are rather trivial; the real work is done in steps 2, 3, and 4. Note that an inconsistency may arise if the output wind field is assimilated into a numerical weather prediction (NWP) model: in the data assimilation step the scatterometer wind field will be checked for mass conservation and continuity, but this has already been done in the 2DVAR

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

step! Therefore it is recommended to feed the wind solutions and their probabilities directly into the NWP data assimilation step after Quality Control, as indicated in figure 2.1.

As further detailed in chapter 3, SDP profits from developments in

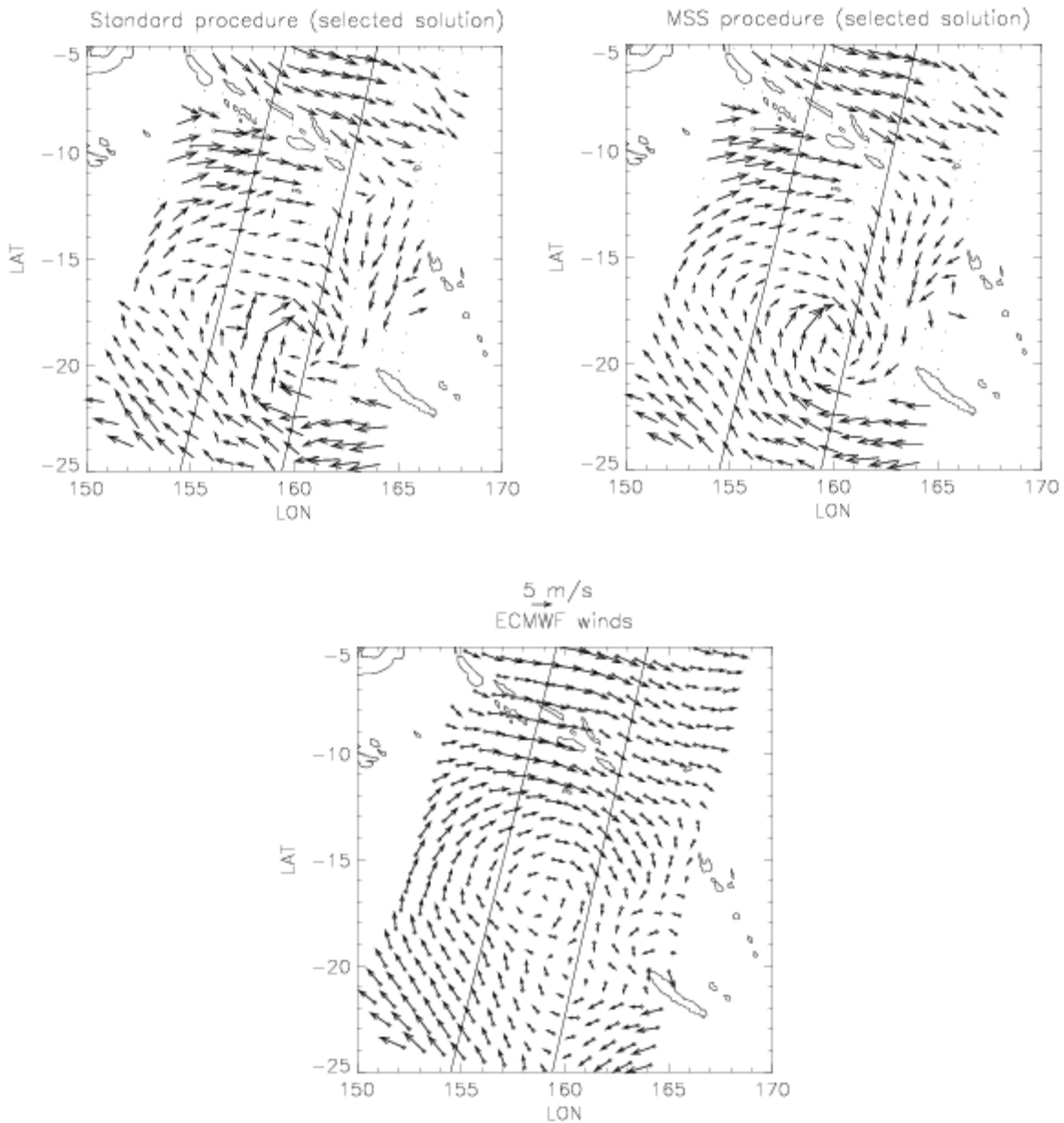
- inversion and output of the full probability density function of the vector wind (Multi Solution Scheme, MSS);
- rain detection and Quality Control (QC);
- Bayesian ice modeling;
- meteorologically balanced Ambiguity Removal (2DVAR);
- quality monitoring;
- variable resolution.

Figure 2.2 shows some example wind fields that demonstrate the improvements achievable with SDP using MSS and 2DVAR.

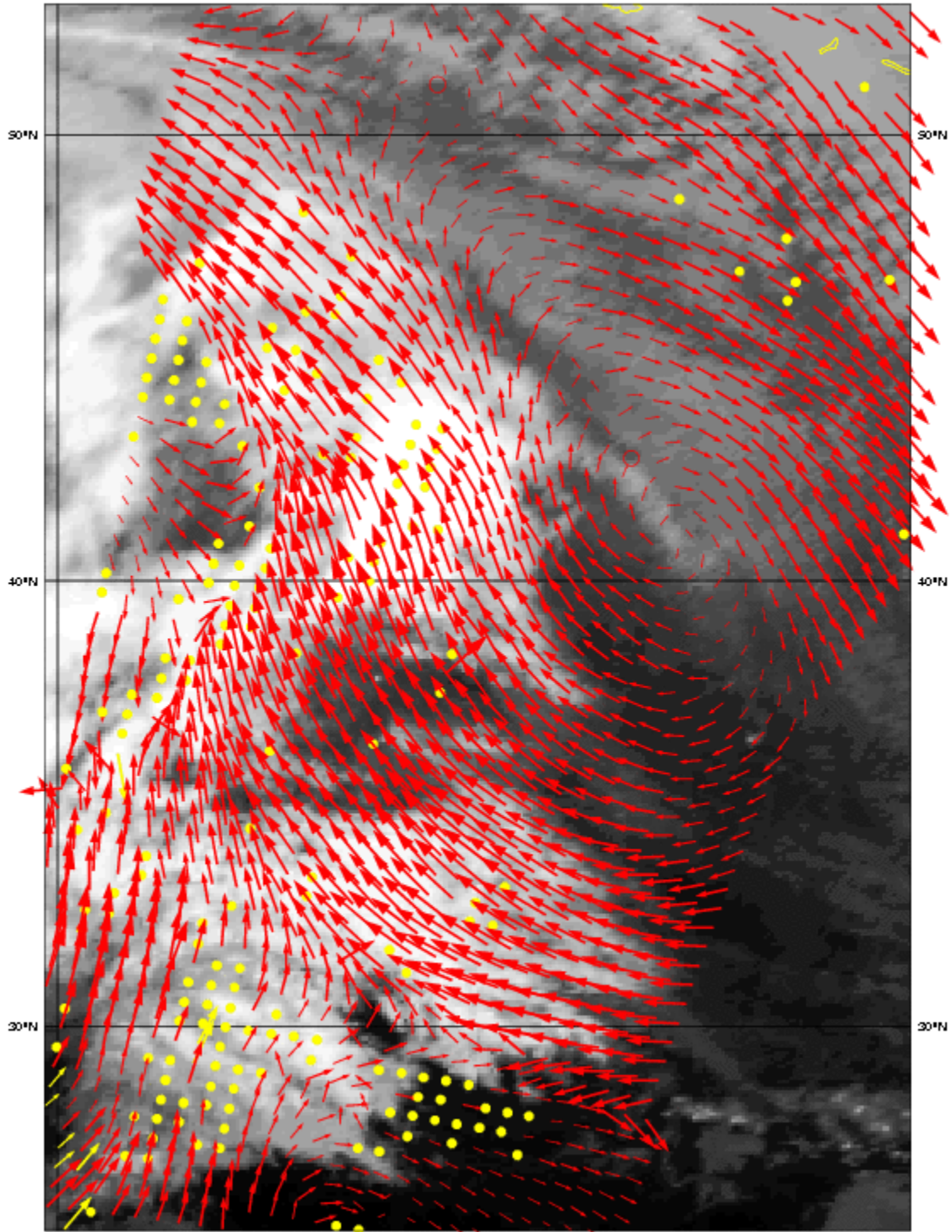
Another important aspect of the SDP program is the possibility to create an output wind product with a different resolution. Figure 2.3 shows an example of a SDP result at 25 km resolution. There is, of course, a trade-off between the output resolution and the output accuracy. The SDP program may help to process the data in the most appropriate manner for the application under consideration. More information on the quality of the SDP output wind field is given by *Vogelzang* [2006, 2007b, 2008].

SDP yields wind fields with high accuracy. Figure 2.4 shows the root mean square difference with the ECMWF First Guess at Appropriate Time (FGAT) for the SDP processed winds without and with MSS. The results of figure 2.4 were obtained using all SeaWinds data from January 2008. The NCEP winds were used as background. The MSS result is much better, especially at nadir, and further improves on the NCEP winds.

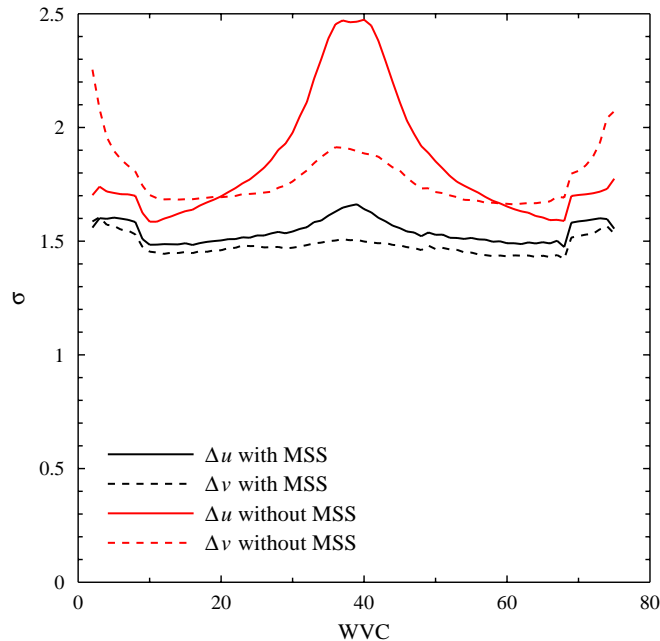
A complete specification of the SDP program can be found in the Product Specification in Chapter 4. The program is based on generic gencat routines for inversion, ambiguity removal, and BUFR file handling. These routines are discussed in more detail in chapters 5 – 7.



**Figure 2.2** An example of the advantages of SDP. The upper left image shows the wind field obtained from SeaWinds using the standard NOAA processing. The field contains some errors at low wind speeds and doesn't look smooth. The upper right image shows the wind field obtained by running SDP in MSS mode, retaining the most probable solution. As a reference, the lower image shows the ECMWF first guess winds. The scatterometer fields contain more detail and, even more important for prediction, put the structure at a different location.



**Figure 2.3** SDP wind field retrieved in MSS mode for January 31, 2005, at 25 km resolution, overlaid on an IR satellite image. Only wind arrows 50 km apart are shown. The cold front on the left of the image is clear and sharp. The yellow dots are rejected WVC's, mostly because of rain.



**Figure 2.4** Standard deviation of the difference between the SDP selected winds with NCEP background and the ECMWF prediction as a function of WVC number for the components  $u$  and  $v$ . The results are for all data from January 2008.

## 2.2 Modes of using SDP

There are several modes to assimilate the SeaWinds data in NWP models using SDP. Anyway, the first thing to assure oneself of is the absence of biases by making scatter plots between SeaWinds and NWP model first guess for at least wind speed, but wind direction and wind components would also be of interest to guarantee consistency.

The operational SDP SeaWinds product, available as a deliverable from the Ocean and Sea Ice (OSI) SAF project, could be the starting point for NWP assimilation:

1. The unique solution at every WVC may be assimilated as if it were buoys. This is the fastest way and one exploits the data to a large extent. For a small advantage, SDP could be installed to provide 2D-VAR solutions based on the local first guess.
2. The SDP software may be used to modify the 3D-VAR or 4D-VAR data assimilation system to work with the ambiguous wind solutions and their probabilities at every WVC. This is some investment, but is applicable for all scatterometer data. The advantage with respect to 1) occurs occasionally, but always in the dynamic atmospheric cases (storms/cyclones) that are really relevant.

1) and 2) can be based on SDP in standard or MSS mode, and at various resolution. MSS is somewhat more dependent on the first guess in 2D-VAR than the SDP standard, but much less

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

noisy (see above) A more noticeable advantage is thus obtained by using the local first guess and potentially the full hi-res benefit of the SeaWinds data is achieved. At the moment, the 25-km mode is experimental, since at KNMI we are now objectively evaluating the added value of MSS and 2DVAR at 25 km. Please contact the NWP SAF helpdesk if this mode will be implemented (address: [www.nwpsaf.org](http://www.nwpsaf.org)) The mode of using SDP thus depends on the opportunities, experience, and time the user has to experiment with SeaWinds in the NWP system under consideration. See also section 3.2.

The SDP program can, of course, also be used to create a stand-alone wind product. Such a stand-alone SeaWinds wind product is a deliverable of the OSI SAF project. More information on this project can be found at the project web site, [www.osi-saf.org](http://www.osi-saf.org).

### 2.3 Installing SDP

SDP is written in Fortran 90 (with a few low level modules in C) and is designed to run on a modern computer system under LINUX or UNIX. SDP needs a Fortran 90 compiler and a C compiler for installation. SDP comes along with a complete make system for compilation. The makefile contains installation scripts which are written in Bourne shell to enhance portability. When compiled, SDP requires about 88 MB disk space.

SDP may also run under Windows. However, SDP needs the BUFR library from ECMWF, and this poses some restrictions on the systems supported. Under Windows one must use Cygwin, a free UNIX emulator (see [www.cygwin.com](http://www.cygwin.com) for more information and download). An alternative UNIX emulator named MinGW/MSYS proved to be too limited to support SDP. Section 2.8 contains some extra information for Cygwin users.

To install SDP, the following steps must be taken:

1. Copy the SDP package (file `SDP_2.0.tar.gz`) to the directory from which SDP will be applied, and unzip and untar it. This will create subdirectories `SDP` and `genscat` that contain all code needed (see 2.3.1).
2. Download the ECMWF BUFR library file `bufr_000360.tar.gz` (or another version not earlier than 000240) and copy it to directory `/genscat/support/bufr`. Untarring and unzipping will be handled by the installation script; see also 2.3.3.
3. Go to the work directory (the one above directories `SDP` and `genscat` and enter `InstallSDP` (when using C shell) or `./InstallSDP` (when using Bourne or bash shell). The script will ask for the compiler used and invoke the make system for compilation and linking of the software (see also 2.3.4). For convenience, this script checks if the BUFR library file is present.

SDP is now ready for use, provided that the environment variables discussed in section 2.3.2 have the proper settings. See also 2.4 and 2.5.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

### **2.3.1 Directories and files**

All code for SDP is stored in a file named `SDP_2.0.tar.gz` that is made available in the framework of the NWP SAF project. This file should be placed in the directory from which SDP is to be run. After unzipping (with `gzip -df SDP_2.0.tar.gz`) and untarring (with `tar -xf SDP_2.0.tar`), the SDP package is extracted in subdirectories `SDP` and `genscat`, which are located in the directory where the original file `SDP_2.0.tar.gz` was located. Subdirectories `SDP` and `genscat` each contain a number of files and subdirectories. A copy of the release notes and the script `InstallSDP` can also be found in the directory containing `SDP` and `genscat`.

Tables 2.1 and 2.2 lists the contents of directories `SDP` and `genscat`, respectively, together with the main contents of the various parts.

<b>Name</b>	<b>Type</b>	<b>Contents</b>
<code>data</code>	subdirectory	Look Up Table for the SeaWinds Geophysical Model Function (GMF)
<code>docs</code>	subdirectory	Documentation, including this document
<code>execs</code>	subdirectory	Shell scripts for running SDP with various input options
<code>makefile</code>	file	Makefile for compiling SDP under LINUX or UNIX
<code>python</code>	subdirectory	Python scripts for running SDP under various operating systems
<code>readme.txt</code>	file	Readme file with some information on SDP.
<code>Sdp</code>	subdirectory	Source code for main SDP program and supporting routines
<code>Sws</code>	subdirectory	Source code for SeaWinds dependent routines
<code>tests</code>	subdirectory	Example BUFR input and output files for testing purposes.

**Table 2.1** Contents of directory `SDP`.

Directories `SDP` and `genscat` and their subdirectories contain various file types:

- Fortran 90 source code, recognizable by the `.F90` extension;
- Files and scripts that are part of the make system for compilation like `Makefile_thisdir`, `Makefile`, `use_`, `Objects.txt` and `Set_Makeoptions` (see 2.3.4 for more details);
- Scripts for the execution of SDP in directories `/SDP/execs` and `/SDP/python`;
- Look-up tables and BUFR tables needed by SDP;
- Files with information like `readme.txt`.

After compilation, the subdirectories with the source code will also contain the object codes of the various modules and routines.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

<b>Name</b>	<b>Type</b>	<b>Contents</b>
ambrem	subdirectory	Source code for ambiguity removal routines
ambrem/twodvar	subdirectory	Source code for KNMI 2DVAR routines
icemodel	subdirectory	Source code for Bayesian ice model
inversion	subdirectory	Source code for inversion routines
Makefile	file	Makefile for compiling Gencat
Objects.txt	file	Part of the makefile
Readme.txt	file	Readme file with some information on gencat
Set_Makeoptions	script file	Script needed by the make system.
support	subdirectory	Collection of general purpose routines sorted in subdirectories
support/BFGS	subdirectory	Source code for minimization routines needed in 2DVAR
support/bufr	subdirectory	BUFR tables (in subdirectories) and source code for BUFR file handling routines
support/Compiler_Features	subdirectory	Source code for handling compiler differences
support/convert	subdirectory	Source code for conversion routines
support/datetime	subdirectory	Source code for date and time conversion routines
support/ErrorHandler	subdirectory	Source code for error handling
support/file	subdirectory	Source code for file handling routines
support/num	subdirectory	Source code for numerical issues
support/singletonfft	subdirectory	Source code for mixed-radix FFT routines needed in minimization
support/sort	subdirectory	Source code for sorting routines
use_g95.bsh	script file	Script for choosing the GNU g95 Fortran compiler (Bourne shell and C shell)
use_g95.csh		
use_gfortran.bsh	script file	Script for choosing the GNU-GCC compiler collection (Bourne shell and C shell)
use_gfortran.csh		
use_ifort.bsh	script file	Script for using the Intel Fortran compiler (Bourne shell and C shell)
use_ifort.csh		
use_pgf90.bsh	script file	Script for using the Portland Fortran compiler (Bourne shell and C shell)
use_pgf90.csh		

**Table 2.2** Contents of directory *genscat*.

### **2.3.2 Environment variables**

SDP needs a number of environment variables to be set. These are listed in table 2.3 together with their possible values.

<b>Name</b>	<b>Value(s)</b>
PLATFORM	big_endian little_endian
BUFR_TABLES	genscat/support/bufr/bufr_tables/
LUT_FILENAME_KU_HH	SDP/data/\${PLATFORM}/nscat2_250_73_51_hh.dat
LUT_FILENAME_KU_VV	SDP/data/\${PLATFORM}/nscat2_250_73_51_hh.dat
EXP_MLE_FILENAME	Depends on resolution, see table 2.4

**Table 2.3** Environment variables for SDP.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

The PLATFORM variable depends on the operating system used. It should be set to `big_endian` under IRIX and SUN OS, and to `little_endian` under LINUX, OSF, and Windows. The PLATFORM variable is needed to guide SDP to the correct version of the lookup table containing the  $K_u$ -band Geophysical Model Function (GMF) needed for the inversion. These tables are in binary form, and the various operating systems have different representations of binary data.

The BUFR\_TABLES variable guides SDP to the BUFR tables needed to read the input and write the output.

The variables LUT\_FILENAME\_KU\_HH and LUT\_FILENAME\_KU\_VV point SDP to the correct  $K_u$ -band GMF lookup tables at HH and VV polarization, respectively. Note that these variables contain the PLATFORM variable already discussed.

Resolution (km)	Value of EXP_MLE_FILENAME
25	SDP/data/\${PLATFORM}/mean_bufr_lr_mle_knmi9_25_r5_mm.dat
50	SDP/data/\${PLATFORM}/mean_bufr_lr_mle_knmi9_50_r5_mm.dat
100	SDP/data/\${PLATFORM}/mean_bufr_lr_mle_knmi9_100_r5_mm.dat

**Table 2.4** Values of variable EXP\_MLE\_FILENAME for various resolutions.

The EXP\_MLE\_FILENAME variable points SDP to a lookup table containing the mean MLE's as a function of node number and wind speed [*Portabella, 2002*]. The mean MLE's are needed for quality control, see section 4.4.3. These LUT's are resolution dependent, so the value of EXP\_MLE\_FILENAME must agree with the resolution specified in the command line options of SDP (see section 2.3). The possible values of EXP\_MLE\_FILENAME are shown in table 2.4.

The lookup tables for the expected MLEs which are provided with the SDP package are obtained using the method described by *Portabella* [2002, page 39 third bullet and appendix B.4]: QuikSCAT data in the new NOAA format of January 2008 were reprocessed using SDP and the MLE values were calculated. Tables (matrices of node numbers and speed indexes) of mean MLE values were created. The data of each matrix element (node number, wind speed) were filtered by repeatedly throwing away all values higher than 5 times the mean value for that element. After 9 iterations, the data sets appeared to converge and no more values were rejected. This procedure was done for 100 km, 50 km and 25 km resolutions, yielding the three tables with expected MLE values.

**Note:** when reading the contents of BUFR tables using some KNMI BUFR to ASCII conversion program like `Bufr2Asc` or `B2A`, also an environment variable `BUFRTAB_DIR` must be set to the proper value pointing to the directory with the BUFR tables. This is required by the *BufrMod* module, see chapter 7.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

### **2.3.3 Installing BUFR library**

SDP needs the ECMWF BUFR Library for its input and output operations. Only ECMWF is allowed to distribute this software. It can be obtained free of charge from ECMWF at the BUFR web page [www.ecmwf.int/products/data/software/bufr.html](http://www.ecmwf.int/products/data/software/bufr.html). The package contains scripts for compilation and installation. The reader is referred to this site for assistance in downloading and installing the BUFR Library.

Directory `genscat/support/bufr` contains the shell script `make.bufr.lib`, which unzips, untars, and compiles the BUFR library file downloaded from ECMWF. This script is part of the `genscat` make system and is automatically invoked when compiling `genscat`. The current version assumes BUFR version 000360, but later versions (or earlier, but not earlier than 000240) can be used, though later versions may not yet be tested. The script `make.bufr.lib` looks for the most recent file with name `bufr_*.tar.gz` in directory `genscat/support/bufr`. If that version of the BUFR library is already installed nothing is done, but if not, the file is untarred, unzipped, and installed. The original tarred and zipped source file remains in `genscat/support/bufr` for future reference. Script `make.clean.bufr.lib` removes the current BUFR library.

BUFR file handling at the lowest level is difficult to achieve. Therefore some routines were coded in C. These routines are collected in library BUFRIO (see also section 7.4). Its source code is located in file `bufrio.c` in subdirectory `genscat/support/bufr`. Compilation is done within the `genscat` make system and requires no further action from the user (see 2.3.4).

### **2.3.4 Compilation and linking**

Compilation and linking of SDP under LINUX or UNIX is done in three steps by the script `InstallSDP`:

1. Set the compiler environment variables according to the choice entered on request. This is equivalent of running the appropriate `use_*` scripts in directory `genscat`;
2. Go to directory `genscat` and invoke the make system;
3. Go to directory `SDP` and invoke the make system to produce the executable `sdp` in directory `SDP/sdp`.

Before activating the make system, the installation script `InstallSDP` sets some environment variables identifying the compiler. These variables are listed in table 2.5. These environment variables can also be set by using one of the `use_*` scripts located in directory `genscat`. Table 2.6 shows the properties of these scripts. The scripts are in Bourne shell (extension `.bsh`) and in C shell (extension `.csh`). Note that all scripts select the GNU `gcc` C compiler.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

Variable	Function
GENSCAT•F77	Reference to Fortran 77 compiler
GENSCAT•F90	Reference to Fortran 90 compiler
GENSCAT•CC	Reference to C compiler
GENSCAT•LINK	Reference to linker for Fortran objects
GENSCAT•CLINK	Reference to linker for C objects
GENSCAT•SHLINK	Reference to linker for shared objects

**Table 2.5** Environment variables for compilation and linking.

Script	Fortran compiler	C compiler	Remarks
use•g95	g95	gcc	GNU compilers by A. Vaught
use•gfortran	gfortran	gcc	GNU-GCC 4.0 compiler collection
use•ifort	ifort	gcc	Intel Fortran compiler
use•pgf90	g90	gcc	Portland Fortran compiler

**Table 2.6** Properties of the four use\_\* scripts.

Example: To select the GNU g95 compiler under Bourne shell type “. use\_g95.bsh”, the dot being absolutely necessary in order to apply the compiler selection to the current shell. Under C shell the equivalent command reads “source use\_g95.csh”.

If a Fortran or C compiler not included in table 2.6 is to be employed, the user can make his own version of the InstallSDP or use\_\* script, or include the environment variables for compilation and linking in his startup file.

SDP is delivered with a complete make system for compilation and linking under UNIX or LINUX. The make system is designed as portable as possible, and system dependent features are avoided. As a consequence, some tasks must be transferred to shell scripts. The make system consists of two parts: one for SDP and one for genscat. The genscat part should be run first. For compilation and linking of the genscat part, the user should move to the genscat directory and simply enter make.

The Makefile refers to each subdirectory of genscat, invoking execution of the local Makefile and, in cases where a subdirectory contains code as well as a subdirectory containing code, Makefile\_thisdir. The makefiles need supplementary information from the files Objects.txt which are present in each directory containing code. The settings for the compilers are located in file Makeoptions in directory genscat. This file is generated by the Bourne shell script Set\_Makeoptions which is called automatically by the genscat make system. The local Makefile in subdirectory genscat/support/bufr calls the script make.bufr.lib for compilation of the BUFR library (see 2.3.3). It also contains the Fortran program test\_modules that generates the binary BUFR tables B and D from the ASCII tables already present, and is executed automatically by the make system. Program test\_modules can also be used to test the genscat BUFR module, see 2.7. The Makefile in subdirectory

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

genscat/support/bufr/bufr\_tables calls the shell scripts run\_make\_symlinks\_for\_first\_table and run\_make\_all\_needed\_symlinks\_for\_SEAWINDS. These scripts make copies of the generic binary BUFR tables B and D under different names. There are four different naming conventions in BUFR version 000240 to 000360, and binary files are generated for each of them. The copies could be replaced by symbolic links to save disk space, but this is not guaranteed to work on each system (symbolic links are not well understood by Cygwin under Windows XP). Further information on the make system is given in the inline comments in the scripts and makefiles.

Compilation and linking of the SDP part is done in a similar manner: go to the SDP directory and enter make. As with genscat, the make system will execute makefiles in every subdirectory of SDP. The result is the executable sdp in directory SDP/sdp. SDP is now ready for use. The make system of SDP doesn't need any further files except the genscat file Makeoptions. This is the reason why genscat should be compiled first.

The GMF tables in SDP/data are set to read-only, except for Cygwin that requires write permission for properly reading those tables. This is done automatically by the InstallSDP script.

When recompiling (part of) SDP or genscat with the make system, for instance when installing a new version of the BUFR library, one should be sure that the proper environment variables for compilation and linking are set. To recompile all of the software enter InstallSDP again. To recompile part of the software invoke the make system where needed. Don't forget to rerun the use\_\* commands to select the right compiler.

## 2.4 Command Line Options

The SDP program is started from directory SDP/sdp with the command

```
sdp [options] < -f BUFRfile | -fl FileList >
```

with <> indicating obligatory input, [] indicating obligatory input, and | indicating alternatives. The following command line options are available:

- f <BUFRfile>** Process a single BUFR input file with name BUFRfile. The BUFR input file should have the NOAA format. Either this option or the next one is obligatory.
- fl <FileList>** Process a list of BUFR input files in the file named FileList. Either this option or the previous one is obligatory.
- par <File>** Read the parameters of 2DVAR from a file with name (and path) File. If absent, SDP assumes default values. See 6.4 for more information. This option has been included for research purposes but may be used to modify the 2DVAR batch grid size or dimension – at your own risk, of course.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

- allswath**            Include outer swath for processing.
- mss1**                Use the Multiple Solution Scheme for Ambiguity Removal in the first processing step at 25 km resolution. If the Multiple Solution Scheme (MSS) is switched on, SDP internally works with 144 different solutions for the wind vector. If MSS is switched off, SDP calculates four solutions at most. MSS is switched off in the first step as default.
- mss2**                Use the Multiple Solution Scheme for Ambiguity Removal in the second processing step at 50 km or 100 km resolution. It is switched off by default.
- genericws1 <n>**    Produce a second BUFR file in generic wind section format in the first processing step. This option generates a second BUFR output file in the KNMI generic wind section format not yet approved by the WMO. The number *n* specifies the number of wind vector solutions written in the output file. The number *n* should not exceed 144. Without MSS switched on, it is more appropriate to set *n* equal to 4.
- genericws2 <n>**    As **-genericws1**, but for the second processing step.
- ana**                  Dump 2DVAR analysis increments  
This option dumps the 2DVAR analysis increments batch by batch in ASCII format on a file with extension `.ana`. It is included for research purposes.
- ocf**                  Dump 2DVAR observation cost function  
This option dumps the values of the observational part of the 2DVAR cost function batch by batch in ASCII format on a file with extension `.ocf`. It is included for research purposes.
- resol1 <R>**        Select output resolution for first processing step.  
The output resolution is set to value *R* equal to 25, with *R* the output resolution in km. When omitted, no output is written for the first processing step.
- resol2 <R>**        As **-resol1**, but for the second processing with *R* equal to 50 or 100. When omitted, the second processing step is not executed.  
Example: the command `sdp -f QSinp -resol1 25 -resol2 100 -mss1 -genericws2` will process the SeaWinds file `QSinp` in the first processing step with a resolution of 25 km using MSS. The output is written to file `QSinp_025`. The second processing step is done without MSS at a resolution of 100 km. The output is written to the file `QSinp_100` and to an extra output file in the KNMI BUFR format `QSinp_100.genws`.  
Warning: the environment variable `EXP_MLE_FILENAME` should have

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

the correct value corresponding to the resolution set with the `-resol` command.

- qdp** Processing in QDP mode.  
This mode of operation is equal to the old QDP scheme. MSS is switched off and the inversion uses no parabolic fitting to find the minimum in the cost function when determining the wind direction. The resolution is set to 100 km.
- tc** Perform extra minimalisation in 2DVAR for batches in the Tropics at a correlation length of 300 km in order to capture tropical cyclones. This is done for batches in which at least one of the measured wind speeds exceeds 10 m/s.
- icemodel N** Choose ice screening method to be used: 0 (default), 1 or 2.  
The value 0 results in no ice screening. The value 1 invokes a simple (non-Bayesian) ice model which does not keep history of the water or ice state of each location. Value 2 invokes the Bayesian ice model which keeps the history of each location and uses this history to determine the sea or ice state of a WVC.  
Warning: The Bayesian ice model needs a few days of data before it gives reliable results, see chapter 8.
- noinv** Switch off inversion (default switched on).
- noamb** Switch off ambiguity removal (default switched on).  
This option is useful when is run in MSS mode, and selection of the scatterometer wind is left to the data assimilation procedure of the Numerical Weather Prediction model. In other words: the NWP model is fed with a large number of solutions and their probability, and finds the best value when comparing with other data sources. This avoids too large influence of the NWP model. Such a procedure will be implemented for KNMI's HIRLAM.
- nowrite** Do not produce BUFR output (default switched on).
- mon** Switch on the monitoring function.  
The results are written on a file with the same name as the input file, but with an extension `.mon` added. As default no monitoring file is produced.
- mononly** Write the monitoring file without any processing.  
The command `sdp -mononly` has the same effect as the command `sdp -mon -noinv -noamb -nowrite`.  
Warning: the `-qdp` option is switched off by the `-mononly` option.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

**-research**

Write BUFR output in research mode.

If the BUFR output is in NOAA format, BUFR item 5 (software version) is set to zero, while items 61, 76, and 91 are replaced by the 2DVAR analysis speed, analysis direction, and the observation part of the cost function, respectively. See also Appendix C1. If the BUFR output is in KNMI format, the same applied to items 5, 36, 51, and 66. See also Appendix C2.

**Warning:** Use of this option results in BUFR output that does not satisfy the WMO standards. This option should only be used for research purposes, and the resulting BUFR files should never be disseminated.

**-verbosity <1>**

Set the verbosity level to 1.

If the verbosity level is -1 or smaller, no output is written to the standard output except error messages. If the verbosity level equals 0 only some top level processing information is written to output. If the verbosity level is 1 or greater, also additional information is given.

From the example above the SDP's output file naming conventions can already be inferred: when the input file is named `QSinp`, the output file is named

<code>QSinp_025</code>	for the normal output in NOAA BUFR format of the first processing step.
<code>QSinp_025.genws</code>	for the extra output in KNMI BUFR format of the first processing step as specified by the option <code>-genericws1</code> .
<code>QSinp_R2</code>	for the normal second output in NOAA BUFR format with <code>R2</code> specified by the <code>-resol2</code> option.
<code>QSinp_R2.genws</code>	for the extra output in KNMI BUFR format of the second processing step as specified by the <code>-resol2</code> and <code>-genericws2</code> options.

Running the command `sdp` without any command line options will yield the following output on the console:

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

```
Usage: sdp [options] < -f BUFRfile | -fl Filelist >
with [.] : free options
      <.> : mandatory options
      |   : choice between alternatives
```

Options:

```
-f <BUFRfile> - process file named BUFRfile
-fl <Filelist> - process list of BUFR files in Filelist
-par <File> - name of file with 2DVAR parameters
-mss1 - use Multiple Solution Scheme MSS in 1st process step
-mss2 - use Multiple Solution Scheme MSS in 2nd process step
-genericws1 <N> - write second BUFR file with generic wind section
containing N wind solutions in 1st process step
-genericws2 <N> - write second BUFR file with generic wind section
containing N wind solutions in 2nd process step
-resol1 <I> - set output resolution in km in 1st process step to value I = 25
-resol2 <I> - set output resolution in km in 2nd process step to value I = 50 | 100
(if resol1 and resol2 are omitted, only one output file
on 25 km will be generated)
-icemodel <IM> - 0: no icemodel (default)
1: non-bayesian icemodel
2: bayesian icemodel
-allswath - swith on outer swath processing (WVC 1-8 & 69-76)
-qdp - process 100 km output file in QDP mode
-noinv - switch off inversion
-noamb - switch off ambiguity removal
-nowrite - do not produce BUFR output
-mon - switch on monitoring
-mononly - write monitoring info without processing
-research - research mode: overwrite some BUFR items with 2DVAR analysis and
.. cost function. USE AT OWN RISK'
-ana - dump 2DVAR analysis
-ocf - dump observation cost function
-tc - perform extra minimalisation in 2DVAR for TCs
(tropical cyclones, assumed present when the maximum
observed wind speed exceeds TC_ThresholdSpeed set
in module TwoDvarData.F90)
-varqc <V> - apply variational quality control type number V
-1: double precision QDP routine (default)'
0: no VarQC
1: QDP formulation (de Vries with VarQC flag on value Jobs)
single precision, differs slightly from V = -1
2: de Vries VarQC
3: de Vries VarQC without interval check
4: traditional VarQC
5: traditional VarQC without interval check
6: Huber norm
-verbosity <L> - set verbosity level to L
```

Running the command `sdp` with an illegal option *Illegal* will produce the same output, but preceded by the error message:

```
Invalid option Illegal
```

## 2.5 Scripts

Directory `SDP/execs` contains four Bourne shell scripts for running SDP with specific input options and the correct environment variables. The reader is referred to the scripts themselves to find out their use and operation.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

This directory also calls a script named `sdp_gui.py`. This is a Python script providing a graphical user interface for entering the command line options. This script may prove helpful when experimenting with the options. Python is a freeware object-oriented programming language. It can be obtained from [www.python.org](http://www.python.org).

Directory `SDP/python` contains Python scripts for execution of the SDP program on different platforms (e.g., Linux, SGI, and Sun). The main goal of these scripts is to test the operation of the program.

A dedicated Python package called `seawindspy` contains support data and procedures, see the folder `SDP/python/seawindspy`. For example, it contains the Python Classes `SdpClass` and `QdpClass` to operate SDP or QDP in Python scripts.

It is recommended to use shell scripts in `SDP/execs` for running SDP to avoid errors caused by conflicting values of environment variables and command line options.

## 2.6 Testruns

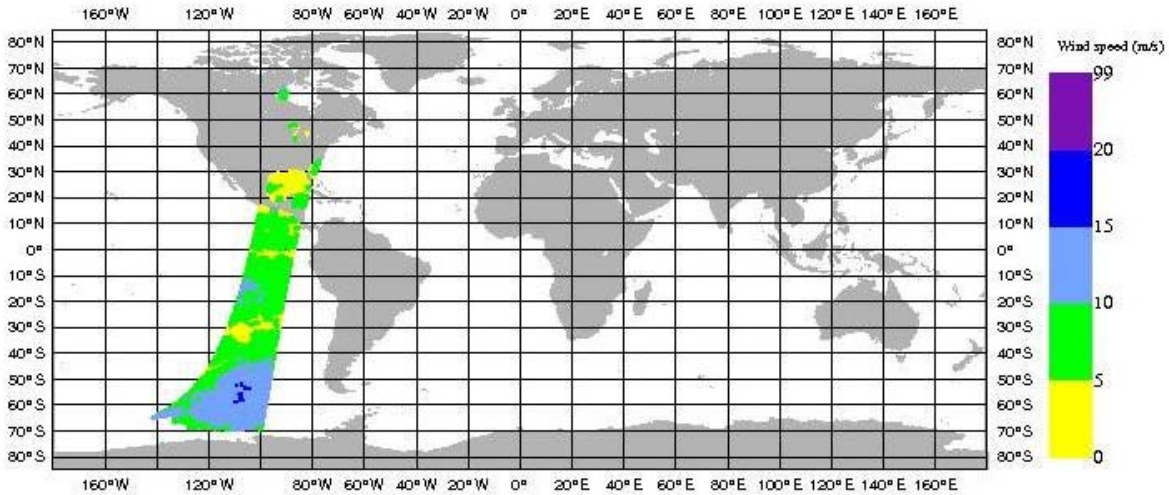
Directory `SDP/tests` contains four BUFR files for testing the SDP executable. File `QS_D08001_S0006_E0052_B4444343` is the first full orbit file of 2008 and acts as input file for SDP test run scripts `RunSDP_Test_1`, `RunSDP_Test_2`, and `RunSDP_Test_3`. Files `SDP_Test_1.bufr`, `SDP_Test_2.bufr`, and `SDP_Test_3.bufr` are their output files. See also table 2.7.

Running one of the scripts in table 2.7 will yield a BUFR output file with the default name `QS_D08001_S0006_E0052_B4444343_025` for the first and second command, or `QS_D08001_S0006_E0052_B4444343_100` for the third command. These files should contain the same results as the corresponding `SDP_Test_i.bufr` files.

<b>Script</b>	<b>Output from script</b>	<b>Output identical with</b>
<code>RunSDP_Test_1</code>	<code>QS_D08001_S0006_E0052_B4444343_025</code>	<code>SDP_Test_1.bufr</code>
<code>RunSDP_Test_2</code>	<code>QS_D08001_S0006_E0052_B4444343_025</code>	<code>SDP_Test_2.bufr</code>
<code>RunSDP_Test_3</code>	<code>QS_D08001_S0006_E0052_B4444343_100</code>	<code>SDP_Test_3.bufr</code>

**Table 2.7** SDP testruns.

Figure 2.5 shows the global coverage of the testrun. Figure 2.5 shows the results of testrun number 2, but the two other testruns will yield very similar results for the magnitude of the wind speed. More information on these tests (and other tests) is given in the SDP Test Report [SCAT group, 2005].



**Figure 2.5** Global coverage of the testruns. Wind speed results for testrun 2 are shown.

Due to round-off differences, a simple file comparison may not be appropriate to test the SDP output. It is then necessary to decode the BUFR files and compare the retrieved wind field with the one in the SDP\_Testrun file. BUFR decoding software is not part of the SDP package, but may be obtained from KNMI on request. See also below.

Directory `genscat/support/bufr` contains a test program named `test_modules`. It is invoked by the `genscat` make system to construct the BUFR tables required by SDP, but it can also be used to test the `genscat` BUFR module. The program is used as follows:

**test\_modules [BUFRinput]**

where `BUFRinput` is the BUFR input file.

If omitted, the program uses as default input the file `testreading.bufr` in directory `genscat/support/bufr`. The output is written on the BUFR file named `testwriting.bufr`. The directory also contains a shell script named `run_test_modules` that sets the environment variables required and executes the program. Further information can be found in the comment lines of the source code of `test_modules`.

Directory in <code>genscat/support</code>	Program name	Output file	Remarks
<code>bufr</code>	<code>test_modules</code>	<code>testwriting.bufr</code>	Part of make system
<code>convert</code>	<code>test_convert</code>	<code>test_convert.output</code>	Wind speed conversion
<code>datetime</code>	<code>TestDateTimeMod</code>	<code>TestDateTimeMod.output</code>	Date and time conversion
<code>file</code>	<code>TestLunManager</code>	<code>TestLunManager.output</code>	File management
<code>numerics</code>	<code>test_numerics</code>	<code>test_numerics.output</code>	Numerical issues
<code>singletonfft</code>	<code>Test_SingletonFFT</code>	<code>Test_SingletonFFT.out</code>	Mixed radix FFT

**Table 2.8** Test programs in `genscat`.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

Subdirectories `convert`, `datetime`, `file`, `num`, and `singletonfft` of directory `genscat/support` contain test programs for the module in that subdirectory. The test programs write their result to the standard output. For comparison, a copy of the output is contained in the `*.output` files. Table 2.8 gives an overview of the `genscat` test programs.

## 2.7 Documentation

Directory `SDP/docs` contains some documentation on SDP, including this document and the Test Report. Further information can be found in the `readme.txt` files, and in the comments in scripts, makefiles and source code.

## 2.8 Some remarks for Cygwin users

SDP 2.0 has been tested under Cygwin, an UNIX emulator running under Windows. Installing and running SDP under Cygwin goes almost the same as under UNIX or LINUX, but the following points may prove helpful for Cygwin users:

- The GNU `g95` compiler comes standard with Cygwin (version 1.5.25-11 and later) so it is always possible to install SDP using the `InstallSDP` script with compiler choice “1”.
- Cygwin works standard in bash shell, so all scripts should be started with `./`.
- Cygwin has its own path naming convention, for example: `C:/SDP` under Windows becomes `/cygdrive/c/SDP` under Cygwin.
- Don't forget to run the `./dos2unix` command on scripts edited under Windows, otherwise Cygwin won't recognize the files as UNIX scripts!

NWP SAF	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
---------	--	---

## Chapter 3

# SDP product specification

### 3.1 Purpose of program SDP

The SeaWinds Data Processor (SDP) program has been developed to fully exploit  $\sigma_0$  data from the SeaWinds scatterometer instruments on either the QuikScat or Adeos-II (Midori-II) satellites to generate surface winds. SDP may be used real-time. The main application of SDP is to form the core of an Observation Operator for SeaWinds Scatterometer data within an operation Numerical Weather Prediction System.

Program SDP is also a level 2 data processor. It reads data from the NOAA SWS\\_met product, see [Leidner *et al.*, 2000]. SDP applies improved algorithms for inversion, Quality Control, and Ambiguity Removal at various spatial resolutions. These methods are mainly developed and published by KNMI. The output of SDP is again a BUFR file.

More information on the SDP product can be found in the SeaWinds Product Manual [Stoffelen and Verhoef, 2008].

### 3.2 Output specification

The wind vectors generated by SDP represent the instantaneous mean surface wind at 10 m anemometer height in a 2D array of Wind Vector Cells (WVC's) with specified size (optionally  $100 \times 100 \text{ km}^2$ ,  $50 \times 50 \text{ km}^2$ , or  $25 \times 25 \text{ km}^2$ ). These WVC's are part of the ground swath of the instrument and are numbered with revolution numbers, along-track row numbers, and across-track node numbers. Therefore, every WVC is identified by a unique (lat, lon, time) triple or a unique (revolution number, row number, node number) triple.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

In conventional mode, the wind output for every WVC consists of up to 4 ambiguities (wind vector alternatives, with varying probabilities). The selected wind vector is indicated by a selection index. For every WVC additional parameters are stored. These are e.g.: latitude, longitude, and time information, revolution, row, and node numbers, background wind vector, cell quality flag, and information on the scatterometer beams including  $\sigma_0$  and  $K_p$  data. The output file is structured according to the conventions of the `SWS\_met` input product (NOAA format). A full description is given in Appendix C1.

A second output file is produced if the `genericws` option is switched on. This file is in the so-called Generic Wind Section format or KNMI format. It contains up to 144 wind vector solutions and their normalized MLE's. This format is not yet approved by the WMO. A full description is given in appendix C2.

At this point it is important to note some differences between the various BUFR output formats:

- **NOAA BUFR output without MSS applied.** Up to 4 solutions are given, the solutions being the minima of the MLE found in the inversion step. The probability of each solution is given as a number between 0 and 1 with a resolution of 0.001. The probability is normalized to 1, i.e., the sum of the probabilities over all solutions equals 1.
- **NOAA BUFR format with MSS applied.** As above, but with the first solution replaced by the solution selected by 2DVAR.
- **KNMI BUFR format without MSS applied.** Up to 4 solutions are given, like for the NOAA BUFR format, but now for each solution both the base 10 logarithm of the normalized probability and the normalized MLE are given.
- **KNMI BUFR format with MSS applied.** Up to 144 solutions with both the base 10 logarithm of their probability and their normalized MLE are written. The number of output solutions can be determined with the `-genericws` command line option.

See table 3.1 for a summary. **Note that the KNMI BUFR format has been changed!**

	NOAA BUFR format		KNMI BUFR format	
	No MSS	MSS	No MSS	MSS
Number of solutions	1-4	1-4	1-4	1-144
Probability information	Normalized probability	Normalized MLE	Base 10 logarithm of normalized probability Normalized MLE	Base 10 logarithm of normalized probability Normalized MLE
Intended use		Stand-alone product for nowcasting		Assimilation into NWP models

**Table 3.1** Differences between the various BUFR output formats.

When using MSS, SDP internally stores 144 solutions, each with a normalized probability. Therefore the probability of the solution selected by the ambiguity removal can be much lower than one may expect: a probability of 1% or lower is perfectly well possible. This applies

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

especially to cases in which the minimum of the inversion cost function is very broad. A standard procedure would select only the minimum value (with relatively high probability), whereas MSS takes all solutions around the minimum into account and divides the probability over them.

The definitions of the MLE, the normalized MLE and their relation with the probability are described by *Portabella* [2002].

### **3.2.1 Flag settings**

The most important flags set by SDP are the KNMI MLE flag and the KNMI VarQC flag. They are part of the Cell Quality Flag which is described in more detail in section 4.3.

Handling of the MLE flag depends on output resolution, WVC number, and setting of the JPL rain detection and ice detection flags. Table 3.2 gives an overview.

	<b>25 km</b>	<b>50 km or 100 km</b>
JPL rain detection flag set in outer and nadir swath	MLE flag set	--
KNMI MLE quality control rejects WVC	MLE flag set	MLE flag set
JPL ice detection flag set	MLE flag set	MLE flag set

**Table 3.2** KNMI MLE flag settings.

At 25 km resolution the JPL rain detection flag is copied, but it influences the KNMI MLE flag only in the outer and nadir swath. At higher resolution the JPL rain detection flag has no meaning in the SDP output.

The JPL ice detection flag is copied at 25 km resolution. At higher resolution it is set if any of the 25 km by 25 km cells constituting the WVC has its ice detection flag set. In this manner it is possible to detect whether a WVC has its KNMI MLE flag set due to ice (ice detection flag set) or to rain (ice detection flag not set).

More information on flag settings can be found in the SeaWinds Product Manual [*Stoffelen and Verhoef*, 2008].

## **3.3 Input Specification**

Input of SDP is the SeaWinds Scatterometer Near-Real-Time BUFR Geophysical Data Product, or shortly the *SWS\\_Met* Data Product. This product is created by NOAA. Though it is in fact already a level 2 product in itself, it should be stressed here that only the basic level 1 data from this NOAA product are used as input for the SDP program. End 2007 NOAA introduced a new definition of the *SWS\\_Met* Data Product. Only this new product is suitable for outer swath processing.

For SeaWinds on QuikSCAT the data have now been available for several years. It contains WVC-composite  $\sigma_0$  data based on slices of the scatterometer pulse footprint. Details of this product can be found in [*Leidner et al.*, 2000].

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

Unfortunately, the Adeos-II satellite collapsed after 9 months of operation. A similar data product is not (yet) available for this satellite.

Remarks:

- At KNMI, the data are gathered in a daily archive file. These `SWS\_met` files are stored in the MOS system.
- At ECMWF, the MARS system contains `SWS\_met` data stored in 6-hourly BUFR files. These files are also suitable as input for the SDP program.

### 3.4 System requirements

Table 3.3 shows the platform and compiler combinations for which SDP has been tested. SDP is designed to run on any UNIX (LINUX) based computer platform with a Fortran compiler and a C compiler. The equivalent of a modern personal computer will suffice to provide a timely NRT wind product. SDP requires about 88 MB disk space when installed and compiled.

<b>Platform</b>	<b>Fortran compiler</b>	<b>C compiler</b>
Suse LINUX work station	Portland pgf90 GNU g95	GNU gcc
SunOS UNIX	Sun Fortran	GNU gcc
Windows XP PC with Cygwin	GNU g95	GNU gcc

**Table 3.3** Platform and compiler combinations for which SDP has been tested.

SDP may also run in other environments, provided that the environment variables discussed in section 2.2 are set to the proper values, and that the BUFR library is properly installed. For Windows one needs Cygwin as UNIX emulator. MinGW proved to be too limited to support SDP.

### 3.5 Details of functionality

The SDP processing chain will be run once at 25 km resolution, but twice at 50 km or 100 km resolution. This is because the new definition of the NOAA BUFR input product introduced at the end of 2007 has a new definition of the MLE. However, this definition is incompatible with the definition of the MLE flag at 50 km or 100 km resolution. Therefore an extra run at 25 km resolution to redefine the MLE's is needed.

#### 3.5.1 BUFR IO and coding

Data sets of Near Real Time meteorological observations are generally coded in the Binary Universal Form for Representation, or shortly BUFR. BUFR is a machine independent data representation system (but it contains binary data, so care must be taken in reading and writing these data under different operating systems). A BUFR message (record) contains observational

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

data of any sort in a self-descriptive manner. The description includes the parameter identification and its unit, decimal, and scaling specifications. The actual data are in binary code. The meta data are stored in BUFR tables. These tables are therefore essential to read (write) and decode (encode) the data.

BUFR tables are issued by the various meteorological centers. The largest part of the data descriptors specified in the BUFR tables follows the official BUFR descriptor standards maintained by the World Meteorological Organization (WMO, e.g., [www.wmo.int](http://www.wmo.int)). However, for their different observational products meteorological centers do locally introduce additional descriptors in their BUFR tables.

Appendix A contains a listing of the data descriptors of the BUFR data input and the BUFR data output of the SDP program in the SWS\\_met BUFR product format (NOAA format). For more details on BUFR and the SWS\\_met BUFR product, the reader is referred to [*Dragosavac, 1994; Leidner et al., 2000*].

ECMWF maintains a library of routines reading (writing) and decoding (encoding) the binary BUFR messages. This library forms the basis of the genscat BUFR module and hence the SDP program BUFR interface, see Chapter 7.

### **3.5.2 Output resolution**

An important feature of the SDP program is that it may produce a level 2 wind product on 25 km × 25 km, 50 km × 50 km, or 100 km × 100 km resolution. Of course, there is a trade off between the output resolution and the statistical error of the mean wind vectors. Therefore KNMI has developed a SeaWinds product with 100 km resolution for assimilation in most NWP models. However, a different resolution may be optimal for a specific NWP application or use as stand-alone product.

### **3.5.3 Quality Control**

The quality of every WVC is controlled. An import aspect is the contamination of the K<sub>u</sub>-band scatterometer signals by rain. The rain flag used in the SDP program is based on the value of the normalized maximum likelihood estimator (MLE) [*Portabella and Stoffelen, 2001, 2002*]. Compared to the JPL flag, the KNMI flag accepts more non-rain winds between 10 and 20 m/s that occur in meteorologically dynamic areas. It also yields less tropical rain contaminated winds [*Portabella and Stoffelen, 2001, 2002*]. See appendices C1 and C2 for more information on how to find these flags in the BUFR output. The JPL rain flag is retained in the 25 km resolution product, but discarded at 50 km or 100 km resolution.

Late 2007, NOAA changed its processing, in particular the definition of the JPL rain flag. The new flag is set less frequently than the old one. The KNMI quality control procedure has been adapted to keep the KNMI flag setting frequency the same. However, the price to be paid is that for 50 km or 100 km output grid size the BUFR input must first be processed to 25 km resolution for initialising the quality control procedure.

NWP SAF	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
---------	--	---

### **3.5.4 Inversion**

In the inversion step of wind retrieval, the radar backscatter observations in terms of the Normalized Radar Cross Sections ( $\sigma_0$ 's) are converted into a set of ambiguous wind vector solutions. In fact, a Geophysical Model Function (GMF) is used to map a wind vector (specified in term of wind speed and wind direction) to a  $\sigma_0$  value. The GMF depends not only wind speed and wind direction but also on the measurement geometry (relative azimuth and incidence angle) and beam parameters (frequency and polarization).

For SeaWinds, a maximum likelihood estimator (MLE) is used to preselect a set of wind vector solutions and associated probabilities that yields the best match with the observed  $\sigma_0$ 's. This preselection depends on the number of independent  $\sigma_0$  values available within the wind vector cell.

The SDP program also includes the Multiple Solution Scheme (MSS). In MSS mode, a much larger preselection of wind vector solutions is produced. The wind vector solutions are ranked according to their probability based on the MLE and constitute the full wind vector probability density function. Subsequently, the 2DVAR method, see e.g., section 3.5.5 is applied with a much larger set of wind vector solutions. The output may be written in the so called Generic Wind Section BUFR format, which allows up to 144 wind vector solutions but is still to be approved by the WMO. Details on the KNMI SeaWinds inversion approach can be found in [Portabella, 2002]. MSS compares better to an independent NWP model reference than conventional four-solution schemes at 100 km resolution [Portabella and Stoffelen, 2004].

Technical information on the KNMI inversion approach can be found in Chapter 5. Details of the original JPL SeaWinds wind retrieval can be found in [Draper and Long, 2002].

### **3.5.5 Ambiguity Removal**

The Ambiguity Removal (AR) step of the wind retrieval is the selection of the most probable surface wind vector among the available wind vector solutions, the so-called ambiguities. Various methods have been developed for AR. More information on Ambiguity Removal is given in Chapter 6. The default method implemented in the SDP program is the KNMI 2DVAR scheme. A description of its implementation can be found in section 6.4. The Multiple Solution Scheme (MSS) offers the possibility to postpone AR to the NWP step in order to treat all information from models and measurements in the same manner. Further details on the algorithms and their validation can be found in the reports [de Vries and Stoffelen, 2000; de Vries et al., 2004] that may be downloaded from the EUMETSAT website, [www.eumetsat.int](http://www.eumetsat.int), or the KNMI website, [www.knmi.nl/scatterometer](http://www.knmi.nl/scatterometer). The most recent and elaborate description can be found in [Vogelzang, 2007a] that can be downloaded from the NWPSAF site [www.nwpsaf.org](http://www.nwpsaf.org).

The performance of the SDP 2DVAR with meteorological balance constraints was tested and optimized for ERS data. It was found to be superior to other schemes. Further testing for SeaWinds is described in [Vogelzang, 2006, 2007b, 2008].

Remarks:

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

- The code has been defined for easy inclusion of the PRESCAT ambiguity removal scheme.

### **3.5.6 Monitoring**

For the automatic ingestion of observations into their NWP systems meteorological centers require quality checks on the NRT products. For the Seawinds BUFR products a monitor flag is developed. This flag indicates that several measures on the level of corruption of the output BUFR files are over a specified threshold. Onset of the flag indicates that the input should be rejected for ingestion by the NWP system. Details on the monitor developed can be found in the NWP SAF document [*de Vries et al.*, 2004], downloadable from the EUMETSAT or KNMI website, [www.eumetsat.int](http://www.eumetsat.int) or [www.knmi.nl/scatterometer](http://www.knmi.nl/scatterometer), respectively.

### **3.5.7 Ice model**

By default SDP uses the ice flag from the NOAA input BUFR product. However, an ice model on a Bayesian approach has been developed in the OSI SAF and can be invoked using the `icemodel` command line option (see section 2.4).

## **3.6 Details of performance**

SDP is delivered with a BUFR input file named `QS_D08001_S0006_E0052_B4444343`, which contains the first full orbit of data recorded in 2008, and three test run scripts `RunSDP_Test_1` to `RunSDP_Test_3`, all in directory `SDP/tests`. Table 3.4 gives the approximate times needed for processing this file under various options on a personal workstation with a 2.66 GHz Pentium 4 processor under LINUX using the GNU g95 Fortran compiler.

As can be seen from table 3.4, choosing the MSS scheme results in slightly larger times needed for inversion, and much more time needed for AR. The computation time, of course, increases with decreasing resolution.

The processing times depend only little on the number of WVC's in the orbit being processed. The choice of platform, compiler, and optimization options will generate more variation.

<b>Script</b>	<b>Resolution (m)</b>	<b>MSS?</b>	<b>Inversion (seconds)</b>	<b>AR (seconds)</b>	<b>BUFR IO (seconds)</b>	<b>Total (seconds)</b>
<code>RunSDP_Test_1</code>	25	No	71	28	8	108
<code>RunSDP_Test_2</code>	25	Yes	74	66	9	150
<code>RunSDP_Test_3</code>	100	No	59	1	4	66

**Table 3.4** Approximate times needed by the SDP test runs to process BUFR file `QS_D08001_S0006_E0052_B4444343`. Note that at 100 km resolution (third test run) the input is first processed to 25 km for correct quality control.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

### **3.7 Warning regarding comparison with OSI SAF products**

The OSI SAF web site ([www.knmi.nl/scatterometer/osisaf](http://www.knmi.nl/scatterometer/osisaf)) shows wind information from various scatterometers including SeaWinds. It seems natural to compare SDP output with OSI SAF wind products. However, some care must be taken in doing so, because there may be differences between the two:

- OSI SAF wind products from SeaWinds use the ECMWF winds as background, while SDP uses NCEP winds by default. This may lead to slightly different wind fields.
- OSI SAF wind products from SeaWinds use Sea Surface Temperature (SST) data for sea ice detection, while SDP uses the NOAA ice flag by default. The SST-based ice screening is much more conservative than the NOAA ice flag, so large differences may appear in the polar regions.
- Note that the ice screening procedure may change in the future as a result of ice modeling activities within the OSI SAF.

In case of questions or problems the user is advised to contact the NWPSAF help desk at [www.nwpsaf.org](http://www.nwpsaf.org)

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

## Chapter 4

# Program Design

In this chapter, the design of the SDP program is described in detail. Readers to whom only a summary will suffice are referred to the Top Level Design (TLD) in section 4.1. Readers who really want to know the very detail should not only read the complete chapter, but also the documentation within the code.

### 4.1 Top Level Design

#### 4.1.1 Main program

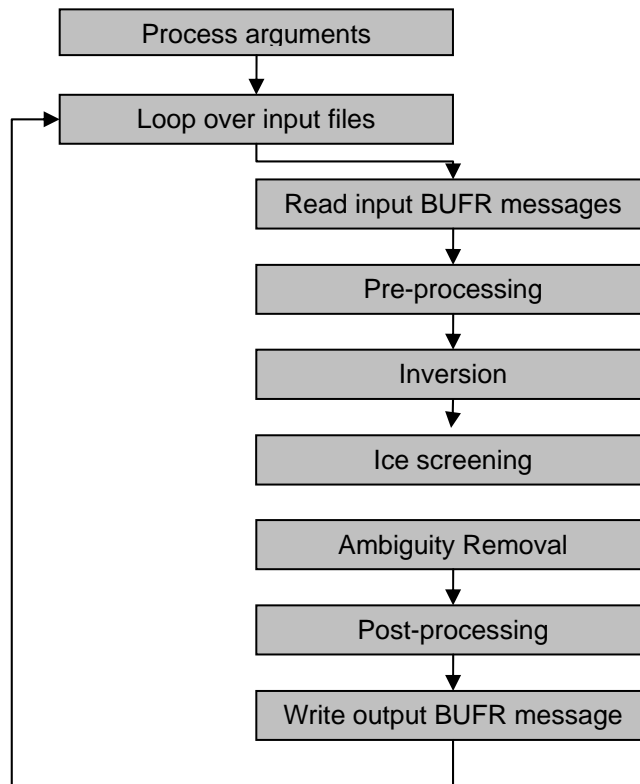
The main program, SDP, (file `sdp` in the `SDP/sdp` directory) is a UNIX (LINUX) executable which processes SeaWinds BUFR input files. The main output consists of BUFR files. The output BUFR messages have the same descriptors as the input messages. The user may provide arguments and parameters according to UNIX command line standards. The purpose of the different options is described in the User Manual (chapter 2).

When executed, the SDP program logs information on the standard output. The detail of this information may be set with the verbosity flag. The baseline of processing is described in Figure 4.1. A more detailed representation of the SDP structure is given in Appendices A and B.

The first step is to process the arguments given at the command line. Next, the SDP program loops over the input files specified in the arguments. For every input file the BUFR messages are read and mapped onto the SeaWinds data structure, see e.g., subsection 4.1.3. As part of the preprocessing a similar SeaWinds data structure is created for the output. Subsequently, the

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

output data structure is filled with level 1 ( $\sigma_0$ -related) data. The next steps are the inversion and the ambiguity removal. These steps are performed on the output data. The loop over the input files ends with the post-processing step (which includes some conversions and the monitoring) and the mapping of the output data structure onto BUFR messages of the BUFR output file. The different stages in the processing correspond directly to specific modules of the code. These modules form the process layer, see section 4.4.

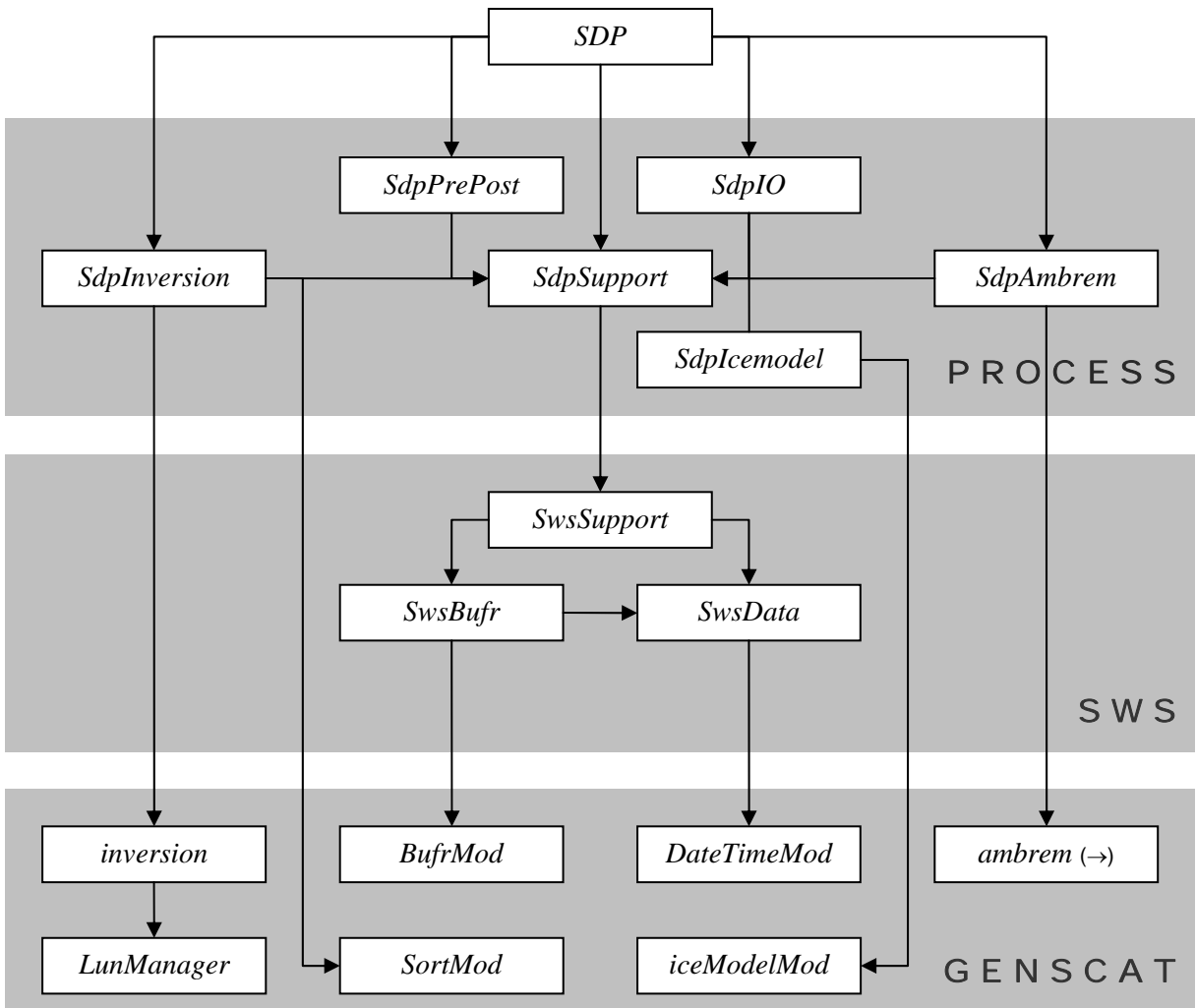


**Figure 4.1** Baseline of the Seawinds Data Processor

At 50 km or 100 km resolution, the processing chain must be run through twice. The first run is at 25 km resolution in order to redefine the MLE's in a manner consistent with the KNMI MLE flag definition. In the second run the 25 km cells are averaged and reprocessed. This must be done in this way, because the new NOAA BUFR product definition introduced at the end of 2007 has a different definition of the MLE than the old product.

#### **4.1.2 Layered model structure**

SDP is a Fortran90 program consisting of several Fortran90 modules which are linked after their individual compilation. The SPD program is set up from three layers of software modules, see Figure 4.2. The purpose of the layer structure is to divide the code with respect to its genericity. Details on the individual modules can be found in sections 4.2 to 4.4.



**Figure 4.2** Module layer and top level module dependencies. The dependencies for module *ambrem* are continued in figure 6.1

The first layer (the process layer) consists of five modules which serve the main steps of the process. These steps are:

- 1) BUFR input and output;
- 2) pre- and post-processing;
- 3) inversion;
- 4) ice screening
- 5) ambiguity removal;
- 6) support.

Each module contains code for performing one or more of the specific tasks. These tasks are shortly described in Table 4.1. A more elaborate description is given in section 4.4. The last module listed, *SdpSupport* is a general support module. This module is used by the other four

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

modules of the process layer for the inclusion of definitions of the data structures and the support routines. (Note that the names of the process modules start with the prefix *Sdp* while the source code is stored in the subdirectory with the name `sdp`).

<b>Module name</b>	<b>Tasks</b>	<b>Comments</b>
<i>SdpIO</i>	BUFR file handling Command line processing	
<i>SdpPrePost</i>	Spatial averaging Quality control Rain flagging Scale conversion Monitoring	Averaging to 50 m or 100 m resolution Usability of input data Rain flag based on normalized MLE Linear versus logarithmic Monitoring
<i>SdpInversion</i>	Inversion	Interface to <code>genscat/inversion</code>
<i>SdpIcemodel</i>	Ice modelling	Interface to <code>genscat/iceModelMod</code>
<i>SdpAmbrem</i>	Ambiguity Removal	Interface to <code>genscat/ambrem</code>
<i>SdpSupport</i>	Support for processing	Definition of data structures Interface to <code>genscat/support</code> via <i>SwsSupport</i>

**Table 4.1** SDP process modules.

The second layer (the SeaWinds layer) consists of SeaWinds Data Support modules. These modules, see table 4.2, contain the SeaWinds data structure definitions and the interface between these data structures and the (input/output) BUFR data format. The key module is *SwsData*. This module contains all the important data types that are introduced for the processing. An overview of these data structures is given in subsection 4.1.3. Details on the actual types and routines are given in section 4.3. The names of these modules start with the prefix *Sws*. The *Sws*-modules are stored in the subdirectory `SDP/sws`.

<b>Module name</b>	<b>Tasks</b>	<b>Description</b>
<i>SwsBufFr</i>	BUFR handling	Mapping of BUFR messages on SeaWinds data structure
<i>SwsData</i>	Data definitions, Data quality control	Composed type declarations Checking and flagging
<i>SwsSupport</i>	Processing support	Interface to <code>genscat/support</code> routines

**Table 4.2** SeaWinds data support modules.

Finally, the third module layer is the `genscat` layer. The `genscat` module classes (i.e., groups of modules) used in the SDP program are listed in table 4.3. `genscat` is a set of generic modules which can be used to assemble processors as well as pre-, and post-processing tools for different scatterometer instruments available for the user community. A short description of the main (interface) modules is given in section 4.2. The most important classes of modules are related to the inversion processing step (chapter 5), the Ambiguity Removal step (chapter 6), and the BUFR file handling (chapter 7). The `genscat` modules are located in subdirectory `genscat`.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

<b>Module class</b>	<b>Tasks</b>	<b>Description</b>
<i>Ambrem</i>	Ambiguity Removal	2DVAR and other schemes, see chapter 6
<i>Icemodel</i>	Ice screening	Detection of sea ice, see chapter 8
<i>Inversion</i>	Wind retrieval	Inversion in one cell, see chapter 5
<i>Support</i>	BUFR support	<i>BufrMod</i> , based on ECMWF library
	FFT, minimization	Support for 2DVAR
	Error handling	Print error messages
	File handling	Finding, opening and closing free file units
	Conversion	Conversion of meteorological quantities
	Sorting	Sorting of ambiguities to their probability
	Argument handling	Compiler independent reading of command line arguments
	Date and time	General purpose

**Table 4.3** genscat module classes.

In addition, genscat contains a large support class to convert and transform meteorological, geographical, and time data, to handle file access and error messages, sorting, and to perform more complex numerical calculations on minimization and Fourier transformation. Many routines are co-developed for ERS and ASCAT data processing.

The layer set-up facilitates a fast and comprehensive development of pre- and post-processing functionality without interfering with the code of the processor itself. In fact, the SeaWinds support layer does not contain any real processing functionality, but this layer provides the required functionality to develop applications which only need the input or output of the process.

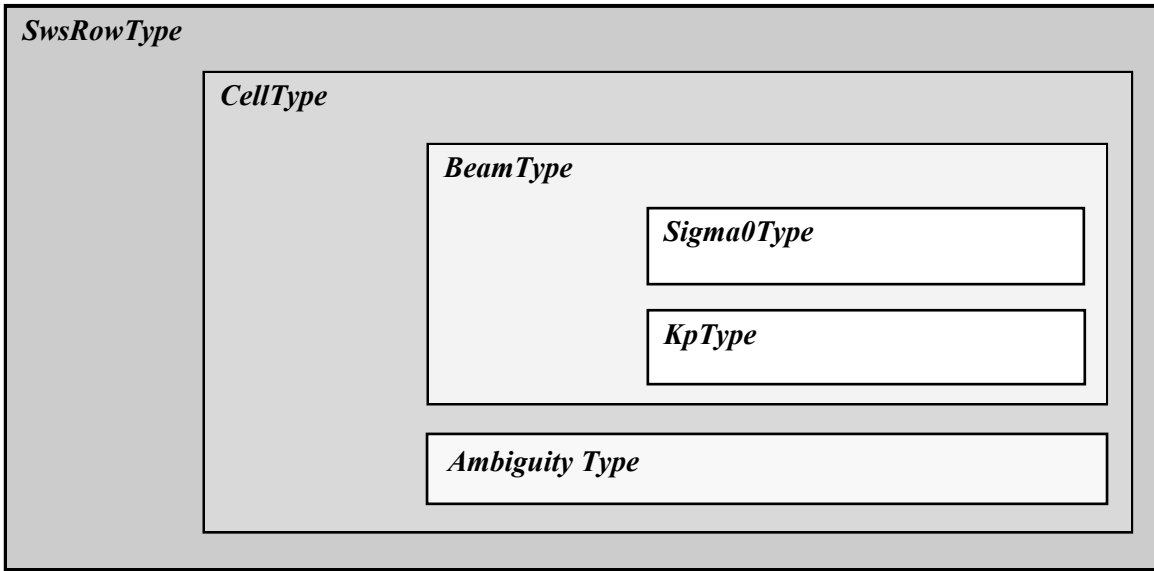
### **4.1.3 Data Structure**

Along track, the SeaWinds swath is divided into rows. Within a row (across track) the SeaWinds orbit is divided into cells, also called Wind Vector Cells (WVC) or nodes. This division in rows and cells forms the basis of the main data structures within the SDP package. In fact, both the input and the output structure are one dimensional arrays of the row data structure, *SwsRowType*. These arrays represent just a part of the swath. Reading and writing (decoding and encoding) SeaWinds BUFR files corresponds to the mapping of a BUFR message to an instance of the *SwsRowType* and vice versa.

The main constituent of the *SwsRowType* is the cell data structure, *CellType*, see figure 4.3. Since most of the processing is done on a cell-by-cell basis the *CellType* is the pivot data structure of the processor. The level 1 data of a cell are stored in a data structure called *BeamType*.

Every cell contains 4 instances of the *BeamType*, corresponding to the inner fore and aft beams and the outer fore and aft beams. The *BeamType* is further subdivided in the *SigmaOType* containing  $\sigma_0$ -related data and the *KpType*. The latter contains the  $\sigma_0$  variance coefficients.

A cell may also contain an array of instances of the *AmbiguityType* data structure. This array stores the results of a successful wind retrieval step, the wind ambiguities (level 2 data). Details of all the data structures and methods working on them are described in chapter 6.



**Figure 4.3** Schematic representation of the nested data definitions in the *SwsRowType* data structure.

Remarks:

- In SDP, the input instances of *CellType* have the name *c11* while the output instances have the name *c12*.

#### **4.1.4 Quality flagging and error handling**

Important aspects of the data processing are to check the validity of the data and to check the data quality. In the SDP program two WVC flags are set for every WVC, see table 4.4, and three flags are set for each of the four beams, see table 4.5. Therefore, 14 flags in total report on the quality and other aspects of the data in each WVC. Furthermore, the flags themselves do not address a single aspect of the data, but the flags are composed of several bits each addressing a specific aspect of the data. A bit is set to 0 (1) in case the data is valid (not valid) with respect to the corresponding aspect. In order to enhance the readability of the SDP code, each flag is translated to a data type consisting of only booleans (false = valid, true = invalid). On input and output these data types are converted to integer values by *set* and *get* routines.

<b>Flag</b>	<b>Tasks</b>	<b>Description</b>
Quality Flag	Quality checking	In BUFR output
Process Flag	Range checking	Not in BUFR output

**Table 4.4** Flags for every WVC (attributes of *CellType*).

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

<b>Flag</b>	<b>Tasks</b>	<b>Description</b>
Surf Flag	Check surface condition	In BUFR output
Mode Flag	Check mode	In BUFR output
Qual Flag	Check quality	In BUFR output

**Table 4.5** Flags for every beam (attributes of *Sigma0Type*).

### **4.1.5 Verboesity**

Every routine in a module may produce some data and statements for the log of the processor. To control the size the log, several modules contain parameters for the level of verbosity. The verbosity of the SDP program may be controlled by the verbosity command line option *verbosity*. In general, there are three levels of verbosity specified:

- $\leq -1$ : be quiet as possible;
- 0: only report top level processing information;
- $\geq 1$ : report additional information.

Of course, errors are logged in any case. Table 4.6 gives a (incomplete) list of verbosity parameters. They are not all set by the command line option as some of them serve testing and debugging purposes.

<b>Module</b>	<b>Verbosity parameter</b>
<i>Ambrem2Dvar</i>	<i>TDVverbosity</i>
<i>AmbremBGclosest</i>	<i>BGverbosity</i>
<i>BatchMod</i>	<i>BatchVerbosity</i>
<i>Ambrem</i>	<i>AmbremVerbosity</i>
<i>SwsBuf</i>	<i>BufVerbosity</i>

**Table 4.6** Verbosity parameters.

## **4.2 Module Design for genscat layer**

### **4.2.1 Module *inversion***

The module *inversion* contains the *genscat* inversion code. It is located in subdirectory *genscat/inversion*. Details of this module are described in chapter 5. In the SDP program, the inversion module is only used in the *SdpInversion* module, see subsection 4.4.5.

### **4.2.2 Module *icemodel***

The module *icemodel* contains the *genscat* part of the ice modeling. It is located in subdirectory *genscat/icemodel*. Details of this module are given in chapter 8. In the SDP program, the *icemodel* module is only used in the *SdpIcemodel* module, see subsection 4.4.4.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

### **4.2.3 Module *ambrem***

The module *ambrem* is the main module of the genscat Ambiguity Removal code. It is located in subdirectory `genscat/ambrem`. Details of this module are described in chapter 6. In the SDP program, the *ambrem* module is only used in the *SdpAmbRem* module, see subsection 4.4.6.

### **4.2.4 Module *Bufrmod***

Genscat contains several support modules. In particular, the *BufrMod* module is the Fortran90 wrapper around the BUFR library used for BUFR input and output. It is located in subdirectory `SDP/genscat/support/bufr`. Details of this module are described in chapter 7. In the SDP program, the *BufrMod* module is only used in the *SwsBufr* module, see subsection 4.3.2.

### **4.2.5 Support modules**

Subdirectory `genscat/support` contains more support modules besides *Bufrmod*. The KNMI 2DVAR method requires minimization of a cost function and numerical Fourier transformation. These routines are located in subdirectories `BFGS` and `singletonfft`, respectively, and are discussed in more detail in section 6.4.

Subdirectory `Compiler_Features` contains alternative routines for *iargc* and *getarg*. Routines *iargc* and *getarg* are not part of the Fortran standard and therefore not supported by each Fortran compiler.

Subdirectory `convert` contains module *convert* for the conversion of meteorological and geographical quantities. So far, only routine *uv\_to\_sd* is used by module *AmbremBGclosest*, but this may change in future updates of SDP.

Subdirectory `datetime` contains module *DateTimeMod* for date and time conversions. SDP only uses routines *GetElapsedSystemTime* (for calculating the running time of the various processing steps) and *julian2ymd* (for conversion of Julian day number to day, month and year). Module *DateTimeMod* needs modules *ErrorHandler* and *numerics*.

Subdirectory `ErrorHandler` contains module *ErrorHandler* for error management. This module is needed by module *DateTimeMod*.

Subdirectory `file` contains module *LunManager* for finding, opening and closing free logical units in Fortran. SDP uses only routines *get\_lun* and *free\_lun* (for opening and closing, respectively, of a logical unit) in the genscat routine *calc\_sigma0* (see figure B1.4).

Subdirectory `num` contains module *numerics* for handling missing values, for instance in the BUFR library. This module is needed by module *DateTimeMod* and is used in the test program `test_modules`.

Subdirectory `sort`, finally, contains module *SortMod* for sorting the wind vector solutions according to their probability.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

### 4.3 Module Design for SeaWinds layer

The SeaWinds layer consists of the modules *SwsData*, *SwsBufFr*, and *SwsSupport*. Table 4.7 lists the routines within these modules. A star indicates that the routine is not (yet) called in the processing chain.

#### 4.3.1 Module *SwsData*

The module *SwsData* contains all the important data types relevant for the processing. Elementary data types are introduced for the most basic data structures of the processing. These are, e.g. *WindType*, *TimeType*, and *RainType*. Using these data types (and of course the standard types as integer, real etc.), more complex (composed) data types are derived. Examples are *BeamType*, *AmbiguityType*, *CellType*, and *SwsRowType*. A complete description of all types is given below. The attributes of all these types have intentionally self-documenting names.

Example: the *KpType* has been introduced for the  $\sigma_0$  variance  $K_p$ . The common three coefficients of  $K_p$ , i.e.,  $\alpha$ ,  $\beta$ , and  $\gamma$ , are stored for every beam in the *Sws\\_met* BUFR messages. The values of these coefficients are copied into an instance of *KpType* (part of *BeamType*), respectively as the real attributes *Alpha*, *Beta* and *Gamma* (see table 4.12).

In the following the different data types are described in alphabetical order.

<i>SwsBufFr</i>	<i>SwsData</i>		
<i>OpenSwsBufFrFile</i>	<i>MergeRow</i>	<i>InitAmbi</i> (*)	<i>PrnProcessFlag</i>
<i>CloseSwsBufFrFile</i>	<i>CheckCell</i> (*)	<i>PrintAmbi</i>	<i>getCellQualFlagNOAA</i>
<i>SwsBufFrInit</i>	<i>TestCell</i>	<i>InitAntenna</i>	<i>getCellQualFlagGen</i>
<i>ReadSwsBufFrData</i>	<i>InitCell</i>	<i>PrintAntenna</i>	<i>setCellQualFlagNOAA</i>
<i>WriteSwsBufFrData</i>	<i>CopyCell</i>	<i>InitRain</i>	<i>setCellQualFlagGen</i>
<i>Values2CellNOAA</i>	<i>PrintCell</i>	<i>PrintRain</i>	<i>PrnCellQualFlag</i>
<i>Values2CellGen</i>	<i>SetDummyCell</i> (*)	<i>SetDummyWind</i> (*)	<i>getSigma0QualFlag</i>
<i>Cell2ValuesNOAA</i>	<i>InitBeam</i>	<i>InitWind</i> (*)	<i>setSigma0QualFlag</i>
<i>Cell2ValuesGen</i>	<i>PrintBeam</i>	<i>PrintWind</i>	<i>PrnSigma0QualFlag</i>
	<i>InitSigma0</i>	<i>TestWind</i>	<i>getSigma0ModeFlag</i>
	<i>TestSigma0</i>	<i>InitTime</i>	<i>setSigma0ModeFlag</i>
	<i>PrintSigma0</i>	<i>PrintTime</i>	<i>PrnSigma0ModeFlag</i>
	<i>InitKp</i>	<i>TestTime</i>	<i>getSigma0SurfFlag</i>
	<i>TestKp</i>	<i>InitProcessFlag</i>	<i>setSigma0SurfFlag</i>
	<i>PrintKp</i>	<i>getProcessFlag</i> (*)	<i>PrnSigma0SurfFlag</i>
		<i>setProcessFlag</i> (*)	

**Table 4.7** Routines in the genscat layer modules. Routines marked with (\*) are not needed for SDP. Note that module *SwsSupport* contains no routines..

**Ambiguity data:** The *AmbiguityType* data type contains information on an individual ambiguity (wind vector solution). The attributes are listed in table 4.8. The routine *InitAmbi()* sets all ambiguity data to missing. The routine *PrintAmbi()* may be used to print all ambiguity data.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

<b>Attribute</b>	<b>Type</b>	<b>Description</b>
<i>Wind</i>	<i>WindType</i>	Wind vector solution
<i>Error</i>	<i>WindType</i>	Error in wind vector solution
<i>Prob</i>	<i>Real</i>	Probability of wind vector solution

**Table 4.8** Ambiguity data structure.

**Antenna data:** The *AntennaType* data type contains additional information on the scatterometer beams, see *CellType*. The attributes are listed in table 4.9. The routine *InitAntenna()* sets all antenna data to missing. The routine *PrintAntenna()* may be used to print all antenna data.

<b>Attribute</b>	<b>Type</b>	<b>Description</b>
<i>Num</i>	<i>Integer</i>	Beam number
<i>Polarization</i>	<i>Real (integer)</i>	Polarization (H or V)
<i>Tb_Mean</i>	<i>Real (integer)</i>	Mean brightness temperature
<i>Tb_StdDev</i>	<i>Real (integer)</i>	Standard deviation of brightness temperature

**Table 4.9** Antenna data structure.

**Beam data:** Every WVC contains up to 4 beams. The information of every beam is stored in the data type *BeamType*. The attributes are listed in table 4.10. The routine *InitBeam()* sets all beam data to missing. The routine *PrintBeam()* may be used to print all beam data.

<b>Attribute</b>	<b>Type</b>	<b>Description</b>
<i>Num</i>	<i>Integer</i>	Beam number: 1 = inner fore, 2 = outer fore, 3 = inner aft, and 4 = outer aft
<i>Sigma0</i>	<i>Sigma0Type</i>	$\sigma_0$ data
<i>Kp</i>	<i>KpType</i>	$K_p$ data
<i>K_Polar</i>	<i>Real (integer)</i>	$K_p$ data

**Table 4.10** Beam data structure.

**Cell Data:** The *CellType* data type is a key data type in the SDP program, because many processing steps are done on a cell by cell basis. The attributes are listed in table 4.11.

The routine *InitCell()* sets the cell data to missing values. Also the flags are set to missing. The routine *TestCell()* tests the validity of data. This routine sets the cell process flag. The routine *PrintCell()* may be used to print the cell data.

NB. The routine *CheckCell()* may be used to select cells with a specified quality. The selection is controlled by a check flag which is an instance of the *CellProcessFlagType*.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

<b>Attribute</b>	<b>Type</b>	<b>Description</b>
<i>RevNr</i>	Integer	Revolution (orbit) number
<i>RowNr</i>	Integer	Row number (along track)
<i>NodeNr</i>	Integer	Node number (across track)
<i>Lat</i>	Real (integer)	Latitude of cell
<i>Lon</i>	Real (integer)	Longitude of cell
<i>Across_Track_Res</i>	Real (integer)	Across track resolution
<i>Along_Track_Res</i>	Real (integer)	Along track resolution
<i>Time_to_Edge</i>	Real (integer)	Time to edge
<i>TimeDiff</i>	Real (integer)	Time difference
<i>Time</i>	<i>TimeType</i>	Date and time
<i>Satellite_ID</i>	Integer	Satellite identification
<i>Sat_Motion</i>	Real (integer)	Satellite motion
<i>Instrument_ID</i>	Integer	Instrument identification
<i>GMF_ID</i>	Integer	GMF identification
<i>Software_ID</i>	Integer	Processor identification
<i>Sigma0_In_cell</i>	Integer	Number of beams for cell
<i>Rain</i>	<i>RainType</i>	Rain data
<i>Antenna(2)</i>	<i>AntennaType</i>	Brightness temperature
<i>Beam(4)</i>	<i>BeamType</i>	Beam data $\sigma_0 K_p$
<i>Num_Ambigs</i>	Integer	Number of ambiguities
<i>Selection</i>	Integer	
<i>Ambi</i>	<i>AmbiguityType</i>	Array of ambiguities
<i>Model</i>	<i>WindType</i>	Model wind
<i>EC</i>	<i>WindType</i>	ECMWF wind (KNMI)
<i>JPL</i>	<i>WindType</i>	JPL wind (KNMI)
<i>TwoDV</i>	<i>WindType</i>	2DVAR analysis wind (KNMI)
<i>Quality_Flag</i>	<i>CellQualFlagType</i>	Quality flag
<i>ProcessFlag</i>	<i>ProcessFlagType</i>	Processing flag

**Table 4.11** Cell data structure.

**$K_p$  data:** The error variance of the  $\sigma_0$  signals are specified in terms of  $K_p$  values.  $K_p$  values are generally a quadratic approximation in terms of  $\sigma_0$ . The coefficients of this approximation are stored in instances of *KpType*, see table 4.12. The routine *InitKp()* sets the  $K_p$  coefficients to missing values. The routine *TestKp()* tests the validity of coefficients specification (see also the cell process flag). The routine *PrintKp()* may be used to print the coefficients.

<b>Attribute</b>	<b>Type</b>	<b>Description</b>
<i>Alpha</i>	Real (integer)	Variance coefficient of quadratic term
<i>Beta</i>	Real (integer)	Variance coefficient of linear term
<i>Gamma</i>	Real (integer)	Variance offset coefficient

**Table 4.12** Variance ( $K_p$ ) data structure.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

**Normalized Radar Cross Section ( $\sigma_0$ ) data:** The *Sigma0Type* data type contains the  $\sigma_0$  (Normalized Radar Cross-Section) information of a specific beam. The attributes are listed in table 4.13. The data types of the flags are discussed further on in this section. The routine *InitSigma0()* sets the  $\sigma_0$  data to missing values. Also the flags are set to missing. The routine *TestSigma0()* tests the validity of the  $\sigma_0$  data (see also the cell process flag). The routine *PrintSigma0()* may be used to print the  $\sigma_0$  data.

<b>Attribute</b>	<b>Type</b>	<b>Description</b>
<i>Lat</i>	Real (integer)	Latitude
<i>Lon</i>	Real (integer)	Longitude
<i>Atten_Value</i>	Real (integer)	Attenuation value
<i>Azimuth</i>	Real (integer)	Azimuth angle
<i>Incidence</i>	Real (integer)	Incidence angle
<i>Value</i>	Real (integer)	$\sigma_0$ value
<i>Qual_Flag</i>	<i>Sigma0QualFlagType</i>	$\sigma_0$ quality flag
<i>Mode_Flag</i>	<i>Sigma0ModeFlagType</i>	$\sigma_0$ mode flag
<i>Surf_Flag</i>	<i>Sigma0SurfFlagType</i>	$\sigma_0$ surface flag
<i>Variance_QC</i>	Real (integer)	Variational quality control value

**Table 4.13** Signal  $\sigma_0$  data structure.

**Rain data:** For every WVC, information on rain is stored in the data type *RainType*. The attributes are listed in table 4.14. The routine *InitRain()* sets all rain data to missing. The routine *PrintRain()* may be used to print all the rain data.

<b>Attribute</b>	<b>Type</b>	<b>Description</b>
<i>MP</i>	Integer	
<i>NOF</i>	Integer	
<i>Rate</i>	Real (integer)	Rain rate
<i>Attenuation</i>	Real (integer)	Attenuation correction

**Table 4.14** Rain data structure.

**Row data:** The data of a complete row of the swath is stored in the data type *SwsRowType*, see table 4.15. The routine *InitRow()* sets all row data to missing. A complete row corresponds to a single BUFR message in the SDP input and output, see module *SwsBuf* in subsection 4.3.4. In some cases two messages are stored for the same row. The routine *MergeRow()* is used to combine the data.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

Attribute	Type	Description
<i>RevNr</i>	Integer	Revolution number
<i>RowNr</i>	Integer	Along track row number
<i>NrCells</i>	Integer	Actual number of WVC's
<i>FirstNode</i>	Integer	Node number of first non-empty WVC cell
<i>Cell(76)</i>	<i>CellType</i>	Array of cells

**Table 4.15** SeaWinds row data structure.

**Time data:** The *TimeType* data type contains a tuple of 6 integers representing both the date and the time, see table 4.16. The routine *InitTime()* sets the time tuple to missing values. The routine *TestTime()* tests the validity of the date and time specification (see also the cell process flag). The routine *PrintTime()* can be used to print the time tuple.

Attribute	Type	Description
<i>Year</i>	Integer	19XX or 20XX
<i>Month</i>	Integer	1 – 12
<i>Day</i>	Integer	1 – 31
<i>Hour</i>	Integer	0 – 23
<i>Min</i>	Integer	0 – 59
<i>Sec</i>	Integer	0 – 59

**Table 4.16** Time data structure.

**Wind Data:** The *WindType* data type contains the wind speed and wind direction, see table 4.17. The routine *SetDummyWind()* fills the wind data type with arbitrary values (remark: should use randomization). The routine *InitWind()* sets the wind vector to missing. The routine *PrintWind()* may be used to print the wind vector. The routine *TestWind()* tests the validity of the wind specification, see also the cell process flag.

Attribute	Type	Description
<i>Speed</i>	Real (integer)	Wind speed
<i>Dir</i>	Real (integer)	Wind direction

**Table 4.17** Wind data structure.

Some special data types are introduced for the data (quality) flags. These are discussed below.

**Cell quality flag:** Every WVC contains a flag for its quality. Therefore the *CellType* contains an instance of the *CellQualFlagType*. Table 4.18 gives an overview of its attributes and the bit number each flag occupies. Note that the bit position differs for the NOAA format and the KNMI format (generic wind section, generated with the `-genws` option): that in the KNMI format is 8 larger than in the NOAA format.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002
		Version : 2.1
		Date : 11-11-2010

The function *getCellQualFlag()* interprets an integer flag (BUFR input) to an instance of *CellQualFlagType*. The function *setCellQualFlag()* transforms an instance of *CellQualFlagType* to an integer flag.

Note that the MLE and AR flags have a different definition than the original NOAA product. The MLE flag has been modified following the procedure described in sections 3.2 and 4.3.3. The AR flag indicates the quality of the solution found by KNMI's 2D variational Ambiguity Removal procedure (chapter 6). The JPL RainDetect Flag is retained in the SDP 25 km resolution output, but discarded at 50 km and 100 km resolution.

Attribute	NOAA		KNMI		Description
	Bit	2 <sup>Bit</sup>	Bit	2 <sup>Bit</sup>	
<i>Missing</i>					Flag not set (all bits on)
<i>QualSigma0</i>	15	32768	22	4194304	Inferior quality of $\sigma_0$ data
<i>Azimuth</i>	14	16384	21	2097152	Invalid azimuth angle
<i>Reserved3</i>	13	8192	20	1048576	
<i>MonFlag</i>	12	4096	19	524288	Monitoring flag not calculated
<i>MonValue</i>	11	2048	18	262144	Monitor flag
<i>MLE</i>	10	1024	17	131072	KNMI + JPL MLE flag
<i>AR</i>	9	512	16	65536	KNMI VarQC flag
<i>Land</i>	8	256	15	32768	Land flag
<i>Ice</i>	7	128	14	16384	Ice flag
<i>Retrieval</i>	6	64	13	8192	No retrieval
<i>Large</i>	5	32	12	4096	Reported wind speed larger than 30 m/s
<i>Small</i>	4	16	11	2048	Reported wind speed smaller than 3 m/s
<i>RainFail</i>	3	8	10	1024	Rain flag not calculated
<i>RainDetect</i>	2	4	9	512	Rain detected
<i>FourBeam</i>	1	2	--		<i>Sigma0_in_Cell</i> does not equal 4

**Table 4.18** Cell quality flag bits (Fortran) in the NOAA and KNMI BUFR output formats.

**Cell process flag:** Besides a cell quality flag, every WVC contains a process flag. The process flag checks on aspects that are important for a proper processing, but are not available as a check in the cell quality flag. The cell process flag is set by the routine *TestCell*.

Table 4.19 lists the attributes of the *CellProcessFlagType*. The function *getCellProcessFlag()* interprets an integer flag (BUFR input) to an instance of *CellProcessFlagType*. The function *setCellProcessFlag()* transforms an instance of *CellProcessFlagType* to an integer flag. The routines *PrnCellProcessFlag()* and *PrnCellQualityFlag()* may be used to print the bit values of the flags.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

Attribute	Bit	$2^{\text{Bit}}$	Description
<i>Missing</i>		2147483647	Flag not set (all bits on)
<i>RevNr</i>	30	1073741824	Invalid revolution number
<i>RowNr</i>	29	536870912	Invalid row number
<i>NodeNr</i>	28	268435456	Invalid node number
<i>Lat</i>	27	134217728	Invalid latitude
<i>Lon</i>	26	67108864	Invalid longitude
<i>MLEQC</i>	25	33554432	MLE quality control set
<i>Along_Track_Res</i>	24	16777216	Invalid along track resolution
<i>Across_Track_Res</i>	23	8388608	Invalid across track resolution
<i>ModelWind</i>	22	4194304	Invalid background wind
<i>Time2Edge</i>	21	2097152	Invalid time to edge
<i>Year</i>	20	1048576	Invalid year specification
<i>Month</i>	19	524288	Invalid moth specification
<i>Day</i>	18	262144	Invalid day specification
<i>Hour</i>	17	131072	Invalid hour specification
<i>Minute</i>	16	65536	Invalid minute specification
<i>Second</i>	15	32768	Invalid second specification
<i>Beam(4)</i>	14	16384	Invalid data of outer aft beam
<i>Beam(3)</i>	13	8192	Invalid data of inner aft beam
<i>Beam(2)</i>	12	4096	Invalid data of outer fore beam
<i>Beam(1)</i>	11	2048	Invalid data of inner fore beam
<i>Sigma0_In_Cell</i>	10	1024	Invalid number of cells
	9	512	
<i>Ambiguity</i>	8	256	Invalid ambiguities
<i>Selection</i>	7	128	Invalid selection
<i>Rain</i>	6	64	Invalid rain data
<i>Tb</i>	5	32	Invalid brightness temperature

**Table 4.19** Cell process flag bits (Fortran).

**Flags for  $\sigma_0$  data:** Every beam contains an instance of *Sigma0Type*. This instance has three attributes to flag the information on  $\sigma_0$ . These attributes are of type *Sigma0QualFlagType*, *Sigma0ModeFlagType*, and *Sigma0SurfFlagType*. Table 4.20 gives an overview of the (bit) attributes of these flags.

The functions *getSigma0QualFlag()*, *setSigma0QualFlag()*, *getSigma0ModeFlag()*, *setSigma0ModeFlag()*, *getSigma0SurfFlag()*, and *setSigma0SurfFlag()* are introduced for the (backward) conversions to the corresponding integer flags. The routines *PrnSigma0QualFlag()*, *PrnSigma0ModeFlag()*, and *PrnSigma0SurfFlag()* may be used to print the bit values of the flags.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

<b>Quality flag</b>			
<b>Attribute</b>	<b>Bit</b>	<b>2<sup>Bit</sup></b>	<b>Description</b>
<i>Missing</i>		2147483647	Flag not set (all bits on)
<i>Useability</i>	15	32768	$\sigma_0$ value
<i>NoiseRatio</i>	14	16384	Azimuth diversity
<i>Negative</i>	13	8192	Negative $\sigma_0$ value
<i>Range</i>	12	4096	2DVAR
<i>Pulse</i>	11	2048	Pulse
<i>Convergence</i>	10	1024	Convergence
<i>FreqShift</i>	9	512	Frequency shift
<i>Temperature</i>	8	256	Temperature
<i>Attitude</i>	7	128	Attitude
<i>Ephemeris</i>	6	64	Ephemeris
<b>Mode flag</b>			
<b>Attribute</b>	<b>Bit</b>	<b>2<sup>Bit</sup></b>	<b>Description</b>
<i>Horizontal</i>	16	65536	
<i>Vertical</i>	15	32768	
<i>Right</i>	14	16384	
<i>Left</i>	13	8192	
<i>HoriVert</i>	12	4096	
<i>RightLeft</i>	11	2048	
<b>Surface flag</b>			
<b>Attribute</b>	<b>Bit</b>	<b>2<sup>Bit</sup></b>	<b>Description</b>
<i>Land</i>	15	32768	
<i>Ice</i>	14	16384	
<i>IceMap</i>	5	32	
<i>AttenuationMap</i>	4	16	

**Table 4.20**  $\sigma_0$  flag bits for quality, mode, and surface (Fortran).

### **4.3.2 Module *SwsBufR***

The module *SwsBufR* maps the SeaWinds data structure on BUFR messages and vice versa. A list of the BUFR data descriptors can be found in appendix A. Satellite and GMF identifiers are listed in tables 4.21 and 4.22. The module uses the genscat module *BufrMod*, see subsection 4.2.3, for the interface with the BUFR routine library. The SeaWinds data structure is defined in module *SwsData*, see subsection 4.3.1.

<b>Satellite</b>	<b>Parameter</b>	<b>Value</b>
ADEOS-1	<i>Adeos1Id</i>	280
QuikSCAT	<i>QscatId</i>	281
ADEOS-2	<i>Adeos2Id</i>	282

**Table 4.21** BUFR SeaWinds satellite identifiers.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

<b>Instrument</b>	<b>Parameter</b>	<b>Value</b>
Reserved	<i>GmfReserved</i>	0
SASS	<i>GmfSass</i>	1
SASS2	<i>GmfSass2</i>	2
NSCAT0	<i>GmfNscat0</i>	3
NSCAT1	<i>GmfNscat1</i>	4
NSCAT2	<i>GmfNscat2</i>	5
NSCAT2P	<i>GmfNscat2P</i>	6
QSCAT1	<i>GmfQscat1</i>	7

**Table 4.22** BUFR GMF identifiers.

<b>Routine</b>	<b>Call</b>	<b>Description</b>
<i>SwsBufrInit</i>		Initialize module <i>SwsBufr</i>
<i>OpenSwsBufrInit</i>		Open BUFR file
<i>CloseSwsBufrInit</i>		Close BUFR file
<i>ReadSwsBufrInit</i>		BUFR message to <i>SwsRowType</i>
<i>WriteSwsBufrInit</i>		<i>SwsRowType</i> to BUFR message
<i>Values2CellNOAA</i>	<i>WriteSwsBufrData</i>	BUFR values array to <i>CellType</i> in NOAA format
<i>Values2CellGen</i>	<i>WriteSwsBufrData</i>	BUFR values array to <i>CellType</i> in generic format
<i>Cell2ValuesNOAA</i>	<i>ReadSwsBufrData</i>	<i>CellType</i> to BUFR values array in NOAA format
<i>Cell2ValuesGen</i>	<i>ReadSwsBufrData</i>	<i>CellType</i> to BUFR values array in generic format

**Table 4.23** Routines in module *SwsBufr*

Table 4.23 provides an overview of the different routines and their calls in this module. Ex general, the SDP module *SdpIO* uses the *SwsBufr* module to set up its BUFR interface. The gencat support routines *GetCurrentDate()* and *GetCurrentTime()* are used to tag the BUFR messages with the date and time of creation.

Note that the routines *Values2Cell* and *Cell2Values*, which convert between BUFR and SDP internal representation, have two variants: one for the official NOAA BUFR format that supports up to four wind solutions, and one for the experimental generic format that supports up to 144 wind solutions. The latter format has not yet been approved by the WMO.

Remarks:

- BUFR message subset indices are fixed for *Sws\\_Met* BUFR. Therefore they are set once during the initialization of *SwsBufr* (for example in ERS processing). These indices must be computed from the BUFR data descriptors.

### **4.3.3 Module *SwsSupport***

- Module *SwsSupport* is the interface between the SWS layer and the general purpose routines in *gencat/support*. This module contains no routines or declarations, but only some use-statements referring to *gencat* routines.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

#### 4.4 Module Design for process layer

The process (SDP) layer consists of the modules *SdpAmbrem*, *SdpIcemodel*, *SdpInversion*, *SdpIO*, *SdpPrePost*, and *SdpSupport*. Module *SdpSupport* contains only declarations and initializations, no subroutines. Table 4.24 lists the routines in the other modules. Routines indicated by a star are not called in the SDP processing chain.

<i>SdpAmbRem</i>	<i>SdpIceModel</i>	<i>SdpInversion</i>	<i>SdpIO</i>	<i>SdpPrePost</i>
<i>RemoveAmbiguity</i>	<i>calcIcelineParms</i>	<i>InitInversion</i>	<i>ReadBufInput</i>	<i>ProcessInit</i>
<i>GetBatch</i>	<i>iceGMF</i>	<i>InitMeanMle</i>	<i>WriteBufOutput</i>	<i>Preprocess</i>
<i>SelectWind</i>	<i>iceLine</i>	<i>CalcSortProb</i>	<i>ProcessSwsFileName</i>	<i>CopyInputOutput</i>
<i>InitProbGross</i>	<i>coordTransform</i>	<i>InversionInCell</i>	<i>GetNwpFileNames</i>	<i>PrepareInput</i>
<i>DummyAmbRem</i> (*)	<i>scat2iceMap</i>	<i>InvertWVCs</i>	<i>GetOutputFileNames</i>	<i>SetInputMleQC</i>
	<i>iceMap2scat</i>	<i>DummyInversion</i> (*)	<i>ProcessArguments</i>	<i>PrepareOutput</i>
	<i>calcIceCoord</i>		<i>usage</i>	<i>PostProcess</i>
	<i>nonbayesianIceModel</i>			<i>Monitoring</i>
	<i>bayesianIcemodel</i>			<i>MonitoringCalculateData</i>
	<i>sdp_icemodel</i>			<i>MonitoringWriteStats</i>
				<i>MonitoringSetMonitorBits</i>
				<i>OutputConversion</i>
				<i>DummyPreProcess</i> (*)
				<i>Monitoring</i>

**Table 4.24** Routines in the process layer modules.

##### 4.4.1 Module *SdpSupport*

Module *SdpSupport* contains many support routines for the processing steps of the SDP program. The module inherits a lot of functionality (data structures and routines) from the *Sws*-modules, see section 4.3. In addition, the module contains the global definitions of the SDP program. Table 4.25 provides an overview.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

<b>Name</b>	<b>Type</b>	<b>Description</b>	<b>Remark</b>
<i>AlongRes</i>	Real	Output along track resolution	
<i>AcrossRes</i>	Real	Output across track resolution	
<i>RowStride</i>	Integer	Along track Stride	
<i>NodeStride</i>	Integer	Across track stride	
<i>NrInputRows</i>	Integer	Actual number of input rows	
<i>NrOutputRows</i>	Integer	Actual number of output rows	
<i>NrInputNodes</i>	Integer	Actual number of input WVC's	
<i>NrOutputNodes</i>	Integer	Actual number of output WVC's	
<i>VerbosityLevel</i>	Integer	Verbosity level	Default 0
<i>ResolutionIndex</i>	Integer	Index of resolution (0 – 15)	Default 0
<i>Lnwp</i>	Logical	Switch NWP	Default .false.
<i>Lqdp</i>	Logical	Switch QDP mode	Default .false.
<i>Lmss</i>	Logical	Switch MSS	Default .false.
<i>Lqc</i>	Logical	Switch quality control	Default .true.
<i>Linvert</i>	Logical	Switch inversion	Default .true.
<i>Lambrem</i>	Logical	Switch ambiguity removal	Default .true.
<i>Lmonitor</i>	Logical	Switch monitoring	Default .false.
<i>InpRow()</i>	<i>SwsRowType</i>	Input orbit rows	
<i>Outrow()</i>	<i>SwsRowType</i>	Output orbit rows	

**Table 4.25** Globals for the processing steps in the SDP program defined in module *SdpSupport*.

#### **4.4.2 Module SdpIO**

Module *SdpIO* has two tasks. The first task is to process the command line options and parameters; the second task is to read (write) BUFR messages from (to) the input (output) BUFR files. The data have to be converted from the BUFR data structures to the SeaWinds data structures and vice versa. Table 4.26 provides an overview of the different routines and their calls in this module.

<b>Routine</b>	<b>Call</b>	<b>Description</b>
<i>ReadBufrInput</i>	SDP	Read BUFR message from input file
<i>WriteBufrOutput</i>	SDP	Write BUFR message on output file
<i>ProcessSwsFileName</i>	<i>ProcessArguments</i>	
<i>GetNwpFileNames</i>	<i>ProcessArguments</i>	
<i>GetOutputFileNames</i>	SDP	
<i>ProcessArguments</i>	SDP	Process SDP command line options
<i>usage</i>	<i>ProcessArguments</i>	Report on the use of SDP

**Table 4.26** Routines of module *SdpIO*.

#### **4.4.3 Module SdpPrePost**

Module *SdpPrePost* contains the routines to do all the pre- and postprocessing. Preprocessing consists of the procedures between the reading of the BUFR input and the wind retrieval for the output product. This includes assessments of the quality of the input data, rain flagging, land and

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

ice flagging, and interpolation to the specified resolution.

<b>Routine</b>	<b>Call</b>	<b>Description</b>
<i>ProcessInit</i>	SDP	Initialization of the processing
<i>PreProcess</i>	SDP	Main routine of the preprocessing
<i>PrepareInput</i>	<i>PreProcess</i>	Preparation of the input cells for averaging
<i>SetInputMleQc</i>	<i>PreProcess</i>	Set normalized MLE quality control tag to input cells
<i>PrepareOutput</i>	<i>PreProcess</i>	Preparation of output cells (supercells) by averaging
<i>CopyInputOutput</i>	<i>PreProcess</i>	Copy $\sigma_0$ and $K_p$ data from input to output
<i>PostProcess</i>	SDP	Main routine of the postprocessing
<i>OutputConversion</i>	<i>PostProcess</i>	Convert output from internal data types to BUFR format
<i>Monitoring</i>	<i>PostProcess</i>	Monitoring
<i>MonitoringCalculateData</i>	<i>Monitoring</i>	
<i>MonitoringWriteStats</i>	<i>Monitoring</i>	
<i>MonitoringSetMonitorBits</i>	<i>Monitoring</i>	
<i>ProcessCleanUp</i>	SDP	Memory management

**Table 4.27** Routines of module *SdpPrePost*.

Table 4.27 lists the tasks of the individual routines. SDP first calls routine *ProcessInit()* to be sure that essential dependencies are set and/or initialized. Next *PrepareInput()* is called to sort the row with respect to the revolution, row and node numbers. It also checks on the appearance of double rows, that is, rows with the same revolution and row number. If *PrepareInput()* finds a double row it merges it into one row. In that case the number of input rows will be reduced. Once the input rows are initialized, *SetInputMleQC()* will set the quality flag using the normalized MLE,  $R_n$ , defined as [Portabella, 2002; Portabella and Stoffelen, 2001]

$$R_n = \frac{MLE}{\langle MLE \rangle} , \quad (4.1)$$

with  $MLE$  the maximum likelihood estimator and  $\langle MLE \rangle$  its average value, obtained from actual measurements. The MLE is defined as

$$MLE = \frac{1}{N} \sum_{i=1}^N \frac{(\sigma_0^{\text{meas},i} - \sigma_0^{\text{simul},i})^2}{K_p(\sigma_0^{\text{simul},i})} . \quad (4.2)$$

In (4.2),  $\sigma_0^{\text{meas},i}$  stands for the measured value of the radar cross section in a WVC, and  $\sigma_0^{\text{simul},i}$  for the simulated value which depends on wind speed and direction. The denominator  $K_p(\sigma_0^{\text{simul},i})$  quantifies the noise in the simulation, i.e., the estimated uncertainty in the GMF. The summation is over all beams of the scatterometer. For SeaWinds,  $N=4$  in the sweet and central swath.

The MLE can be regarded upon as the distance between an actual scatterometer measurement and the GMF in N-dimensional measurement space. The MLE is related to the probability  $P$  that the

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

GMF at a certain wind speed and direction represents the measurement by

$$P \propto e^{-MLE} \quad (4.3)$$

Therefore, wind vectors with low MLE have a high probability of being the correct solution. On the other hand, wind vectors with high MLE are not likely represented by any point on the GMF, probably because the measurements are contaminated by ice, rain, and/or confused sea state, phenomena not included in the GMF. The ratio  $R_n$  further refines this notion by taking the uncertainty of the GMF into account. *Portabella* [2002] and *Portabella and Stoffelen* [2001] derive the following threshold values for  $R_n$  as a function of wind speed  $w$

$$R_n^{\text{thres}} = \begin{cases} 4 - 0.05(w - 5)^2 & , \quad w < 15 \text{ m/s} \\ 2 & , \quad w > 15 \text{ m/s} \end{cases} \quad (4.4)$$

When  $R_n < R_n^{\text{thres}}$ , the solution is considered close enough to the measurement and is accepted. If, on the other hand,  $R_n > R_n^{\text{thres}}$ , the solution lies too far away from the measurement, probably because the measurement is not well described by the GMF. The solution is therefore rejected.

In addition, for the nadir swath (that is for node number between 28 and 49) the rain flag of the NOAA product is adopted. That is, if NOAA sets the rain flag for this region also the MLE rain flag is set. The main task of the *PrepareOutput()* is to average the input data and to produce an output orbit with a lower resolution (less rows, less nodes). The wind vector cells of the output orbit are sometimes called supercells. The averaging concerns all input data needed to define the temporal and spatial location of the output cell and the beam data ( $\sigma_0$ ,  $K_p$ ) of the output cell. In addition, *PrepareOutput()* adjusts the quality flags of the output cells and the output  $\sigma_0$  data.

Postprocessing consists of the procedure between the ambiguity removal step and the BUFR encoding of the output. Currently, postprocessing is confined to some simple conversions. It also includes the monitoring.

#### **4.4.4 Module *SdpInversion***

Module *SdpInversion* serves the inversion step in the wind retrieval. The inversion step is done cell by cell. The actual inversion algorithm is implemented in the genscat module *Inversion*, see subsection 4.2.1. Table 4.28 provides an overview of the different routines and their calls in this module.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

<b>Routine</b>	<b>Call</b>	<b>Description</b>
<i>InitInversion</i>	<i>InvertWVCs</i>	Initialization
<i>InitMeanMle</i>	<i>InitInversion</i>	Set the mean MLE
<i>CalcSortProb</i>	<i>InversionInCell</i>	Calculate the probabilities and sort the ambiguities on probability
<i>InversionInCell</i>	<i>InvertWVCs</i>	Call to the genscat inversion module
<i>InvertWVCs</i>	SDP	Loop over all output cells

**Table 4.28** Routines of module *SpdInversion*.

#### **4.4.5 Module *SdpIcemodel***

Module *SdpIcemodel* performs the ice screening of the wind product. The ice screening works on the principle that WVCs over water yield wind solutions which are close to the GMF. If a WVC is over ice, the  $\sigma_0$  quadruplets will be close to the so-called ice line. Hence, there is a possibility to discriminate between water (wind) and ice WVCs. The implementation of this principle is described in more detail in [Verspeek, 2006]. The ice screening is done directly after the inversion step. Table 4.29 provides an overview of the routines and their calls in this module.

<b>Routine</b>	<b>Call</b>	<b>Description</b>
<i>calcIcelineParms</i>	<i>nonbayesianIceModel</i> <i>calcIceCoord</i>	Calculate distance to ice line from given $\sigma_0$ 's
<i>iceGMF</i>	not used	Calculate the $\sigma_0$ values from the ice coordinates
<i>iceLine</i>	<i>iceGMF</i> (not used)	Calculate the ice line origin and slope
<i>coordTransform</i>	<i>bayesianIcemodel</i>	Calculate coordinates on an SSM/I grid
<i>scat2iceMap</i>	<i>bayesianIcemodel</i>	Update the ice map with the information in cell data
<i>iceMap2scat</i>	<i>bayesianIcemodel</i>	Update cell data structure with information in ice map
<i>calcIceCoord</i>	<i>bayesianIcemodel</i>	Calculate ice coordinates and distance to ice line
<i>bayesianIcemodel</i>	<i>awdpIcemodel</i>	Implementation of the Bayesian ice model
<i>nonbayesianIceModel</i>	<i>awdpIcemodel</i>	Implementation of the basic ice model without history
<i>SDP_icemodel</i>	SDP	Main routine of ice screening

**Table 4.29** Routines of module *awdp\_icemodel*.

#### **4.4.6 Module *SdpAmbrem***

Module *SdpAmbrem* controls the ambiguity removal step of the SDP program. The actual ambiguity removal schemes are implemented in the genscat module *ambrem*, see subsection 4.2.2. The default method is the KNMI 2DVAR scheme. Table 4.30 lists the tasks of the individual routines.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

<b>Routine</b>	<b>Call</b>	<b>Description</b>
<i>RemoveAmbiguity</i>	SDP	Main routine of ambiguity removal
<i>GetBatch</i>	<i>RemoveAmbiguity</i>	Obtain a batch of observations
<i>SelectWind</i>	<i>RemoveAmbiguity</i>	Final selection
<i>InitProbGross</i>	<i>RemoveAmbiguity</i>	Set the gross probabilities

**Table 4.29** Routines of module *SpdAmbrem*.

The ambiguity removal scheme works on a so-called batch. The batch is defined in the *GetBatch()* routine. For the SDP program a batch is just a set of rows. The size of the batch is determined by the resolution of the structure functions and the number of FFT. The genscat routine *DoAmbrem()* performs the actual ambiguity removal scheme.

Finally *SelectWind* passes the selection to the output WVC's.

## 4.5 Flag use

The NOAA BUFR input files that serve as input for SDP are in themselves already level 2 products and contain flags for quality control. Some of these are used in SDP, and some are redefined. Table 4.30 gives an overview.

Inversion is performed for all cells for which both the Cell Quality FourBeam and the Cell Quality QualSigma0 flags are not set (see table 4.18).

Ambiguity removal is performed for all cells containing ambiguities and model winds, and that have the Cell Quality MLE flag not set (see table 4.18).

More information on the structure of the BUFR output files can be found in Appendix C1 and Appendix C2.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

<b>Flag</b>	<b>Where used</b>	<b>Description</b>
<b>Cell Quality Flag</b> (see table 4.18)		
RainFail	Input MLE quality check	Must be false in order to use the Rain detect flag
RainDetect	Input MLE quality check	If set for WVC 29-48, the cell is rejected
Land	Input MLE quality check	If set, the JPL MLE quality check is not performed
Land	Preparation BUFR output	If set, the input beam information is not used
Ice	Input MLE quality check	If set, the JPL MLE quality check is not performed
Ice	Preparation BUFR output	If set, the input beam information is not used
Retrieval	Input MLE quality check	If set, the JPL MLE quality check is not performed
MLE	Quality control	Redefined
<b>Sigma0 Surface Flag</b> (see table 4.20)		
Land	Preparation BUFR output	If set, set Cell Quality land flag
Land	Preparation BUFR output	If set, the input beam information is not used
Ice	Preparation BUFR output	If set, set Cell Quality ice flag
Ice	Preparation BUFR output	If set, the input beam information is not used
<b>Sigma0 Quality Flag</b> (see table 4.20)		
Useability	Preparation BUFR output	If set, the input beam information is not used
Negative	Preparation BUFR output	If set, set sign of $\sigma_0$

**Table 4.30** Flag handling in SDP.

NWP SAF	SDP User Manual and Reference Guide	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
---------	-------------------------------------	---

## Chapter 5

# Inversion module class

### 5.1 Background

In the inversion step of the wind retrieval, the radar backscatter observations in terms of the normalized radar cross-sections ( $\sigma_0$ 's) are converted into a set of ambiguous wind vector solutions. In fact, a Geophysical Model Function (GMF) is used to map a wind vector (specified in term of wind speed and wind direction) to a  $\sigma_0$  value. The GMF further depends not only wind speed and wind direction, but also on the measurement geometry (relative azimuth and incidence angle), and beam parameters (frequency, polarisation). For SeaWinds, a maximum likelihood estimator (MLE) is used to select a set wind vector solutions that optimally match the observed  $\sigma_0$ 's. The wind vector solutions correspond to local minima of the MLE function

$$MLE = \frac{1}{N} \sum_{i=1}^N \frac{(\sigma_0^{obs}(i) - \sigma_0^{GMF}(i))^2}{K_p} , \quad (5.1)$$

With  $N$  the number of independent  $\sigma_0$  measurements available within the wind vector cell, and  $K_p$  the covariance of the measurement error. This selection depends on the number of independent  $\sigma_0$  values available within the wind vector cell.

Details on the SeaWinds inversion problem can be found in [Portabella, 2002]. Details on the original JPL inversion approach can be found in [Draper et al., 2002]. The SDP program includes the Multiple Solution Scheme (MSS), see [Portabella and Stoffelen, 2001].

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

## 5.2 Routines

The inversion module class contains only one module named *inversion*. It is located in subdirectory *genscat/inversion*. Table 5.1 lists all routines in this module. Appendix B.1 shows the calling tree for the inversion routines.

<b>Routine</b>	<b>Call</b>	<b>Routine</b>	<b>Call</b>
<i>invert_one_wvc</i>	SDP	<i>set_wind_speed_first_guess</i>	see B.1
<i>fill_wind_quality_code</i>	<i>invert_one_wvc</i>	<i>get_dynamic_range</i>	not used
<i>remove_one_solution</i>	<i>fill_wind_quality_code</i>	<i>get_GMF_version_used</i>	not used
<i>save_inv_input</i>	not used	<i>calc_sigma0</i>	not used
<i>read_inv_input</i>	not used	<i>INTERPOLATE</i>	generic
<i>save_inv_output</i>	not used	<i>interpolate1d</i>	<i>calc_sigma0</i>
<i>do_parabolic_winddir_search</i>	<i>invert_one_wvc</i>	<i>interpolated2d</i>	<i>calc_sigma0</i>
<i>calc_normalisation</i>	<i>invert_one_wvc</i>	<i>interpolate2dv</i>	<i>calc_sigma0</i>
<i>calc_sign_MLE</i>	<i>invert_one_wvc</i>	<i>interpolate3d</i>	<i>calc_sigma0</i>
<i>print_message</i>	see B.1	<i>read_LUT</i>	<i>calc_sigma0</i>
<i>init_inv_input</i>	SDP	<i>create_LUT_C_VV</i>	<i>calc_sigma0</i>
<i>init_inv_output</i>	<i>invert_one_wvc</i>	<i>test_for_identical_LUTs</i>	<i>calc_sigma0</i>
<i>init_inv_settings_to_default</i>	SDP	<i>my_mod360</i>	not used
<i>write_inv_settings_to_file</i>	not used	<i>my_mod</i>	not used
<i>get_inv_settings</i>	SDP	<i>my_min</i>	see B.1
<i>set_inv_settings</i>	SDP	<i>my_max</i>	see B.1
<i>check_input_data</i>	<i>invert_one_wvc</i>	<i>my_average</i>	see B.1
<i>find_minimum_cone_dist</i>	<i>invert_one_wvc</i>	<i>get_indices_lowest_local_minimum</i>	<i>invert_one_wvc</i>
<i>get_parabolic_minimum</i>	<i>do_parabolic_winddir_search</i>	<i>my_index_max</i>	see B.1
<i>calc_cone_distance</i>	<i>find_minimum_cone_dist</i>	<i>my_exit</i>	see B.1
<i>calc_dist_to_cone_center</i>	<i>fill_wind_quality_code</i>	<i>print_wind_quality_code</i>	see B.1
<i>convert_sigma_to_zspace</i>	<i>invert_one_wvc</i>	<i>print_input_data_of_inversion</i>	<i>check_input_data</i>
<i>get_ers_node_formfactor</i>	<i>calc_var_s0</i>	<i>print_output_data_of_inversion</i>	see B.1
<i>calc_var_s0_ers</i>	<i>invert_one_wvc</i>	<i>print_inout_data_of_inversion</i>	not used
<i>get_wind_speed_first_guess</i>	<i>find_minimum_cone_dist</i>	<i>calc_sigma0_cmod4</i>	<i>create_LUT_C_VV</i>
<i>get_ers_noise_estimate</i>	<i>calc_var_s0</i>	<i>f1</i>	<i>calc_sigma0_cmod4</i>
<i>calc_var_s0</i>	<i>calc_normalisation</i>	<i>Get_Br_from_Look_Up_Table</i>	<i>calc_sigma0_cmod4</i>
<i>get_wind_speed_first_guess</i>	<i>find_minimum_cone_dist</i>	<i>calc_sigma0_cmod5</i>	<i>create_LUT_C_VV</i>

**Table 5.1** Routines in module *inversion*.

To establish the MLE function (1), the radar cross section according to the GMF,  $\sigma_o^{GMF}$ , must be calculated. This is done in routine *calc\_sigma0*. The GMF at  $K_u$  band for HH and VV polarization needed for SeaWinds, is not known in analytical form. It is only available in the form of Look Up Tables (in directory *SDP/lut*). The value for  $\sigma_o^{GMF}$  is obtained from interpolation of these tables. The interpolation is done via symbolic routine *INTERPOLATE* which is set to *interpolate1d*, *interpolate2d*, *interpolate2dv*, or *interpolate3d*, depending on the type of interpolation needed.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

For C-band at VV polarization the GMF is given in analytical form (routines *calc\_sigma0\_cmod4* and *calc\_sigma0\_cmod5*, respectively). In order to treat all scatterometer types in the same way, the radar cross section at C-band is also calculated from interpolation of Look Up Tables (LUTs). If a C-band LUT is not present it will be created by routine *create\_LUT\_C\_VV*. This routine calls one of the routines *calc\_sigma0\_cmod4* or *calc\_sigma0\_cmod5* that contain the analytical expressions of the CMOD4 or CMOD5 algorithm. Routines *get\_lun* and *free\_lun* from module *LunManager* in subdirectory *genscat/support/file* are needed when reading and creating the LUTs.

### 5.3 Antenna direction

The output wind direction of inversion routines are generally given in the meteorological convention, see table 5.2. The inversion routine uses a wind direction that is relative to the antenna direction. The convention is that if the wind blows towards the antenna then this relative wind direction equals to 0. Therefore, it is important to be certain about the convention of your antenna (azimuth) angle.

For the SeaWinds Met product the radar look angle (antenna angle or simply azimuth) equals 0 if the antenna is orientated towards the north. The SeaWinds radar look angle increases clockwise. For ERS, however the antenna direction equals zero if the antenna directs towards the south. Therefore the final output wind direction needs a correction of 180 degrees.

<b>Meteorological</b>	<b>Mathematical</b>	<b><i>u</i></b>	<b><i>v</i></b>	<b>Description</b>
0	270	0	-1	Wind blowing from the north
90	180	-1	0	Wind blowing from the east
180	90	0	1	Wind blowing from the south
270	0	1	0	Wind blowing from the west

**Table 5.2** Meteorological conventions for the wind direction.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

## Chapter 6

# Ambiguity Removal module class

### 6.1 Ambiguity Removal

Ambiguity Removal (AR) schemes select a surface wind vector among the different surface wind vector solutions per cell for the set of wind vector cells in consideration. The goal is to set a unique, meteorological consistent surface wind field. The surface wind vector solutions per cell, simply called ambiguities, result from the wind retrieval process step.

Whenever the ambiguities are ranked, a naive scheme would be to select the ambiguity with the first rank (e.g., the highest probability, the lowest distance to the wind cone). In general, such a persistent first rank selection will not suffice to create a realistic surface wind vector field: scatterometer measurements tend to generate ambiguous wind solutions with approximately equal likelihood (mainly due to the 180° invariance of stand alone scatterometer measurements). Therefore additional spatial constraints and/or additional (external) information are needed to make sensible selections.

A common way to add external information to a WVC is to define a background surface wind vector. The background wind acts as a first approximation for the expected mean wind over the cell. In general, a NWP model wind is interpolated for this purpose. Whenever a background wind is set for the WVC, a second naive Ambiguity Removal scheme is at hand: the Background Closest (BC) scheme. The selected wind vector is just the minimizer of the distance (e.g., in the least squares sense) to the background wind vector. This scheme may produce far more realistic wind vector fields than the first rank selection, especially if the background surface wind field is

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

meteorologically consistent.

However, background surface winds have their own uncertainty. Therefore, sophisticated schemes for Ambiguity Removal take both the likelihood of the ambiguities and the uncertainty of the background surface wind into account. Examples are the KNMI Two-Dimensional Variational (2DVAR) scheme and the PreScat scheme.

The implementation of these schemes is described in sections 6.4 and 6.5.

## 6.2 Module *Ambrem*

Module *Ambrem* is the interface module between the various ambiguity removal methods and the different scatterometer data processors. Table 6.1 provides an overview of the different routines and their calls. A dummy method and the first rank selection method are implemented as part of *ambrem*. More elaborate Ambiguity Removal methods have an interface module, see table 6.2. Figure 6.1 shows schematically the interdependence of the various modules for Ambiguity Removal.

<b>Routine</b>	<b>Call</b>	<b>Description</b>
<i>InitAmbremModule</i>	SDP	Initialization of module <i>Ambrem</i>
<i>InitAmbremMethod</i>	SDP	Initialization of specified AR scheme
<i>DoAmbrem</i>	SDP	Execution of specified AR scheme
<i>Ambrem1stRank</i>	<i>DoAmbrem</i>	First rank selection method
<i>DoDummyMeth</i>	<i>DoAmbrem</i>	Dummy AR scheme for testing
<i>SetDummyMeth</i>	<i>DoAmbrem</i>	Batch definition of dummy method
<i>InitDummyMeth</i>	<i>DoAmbrem</i>	Initialization of dummy method
<i>InitDummyBatch</i>	not used	

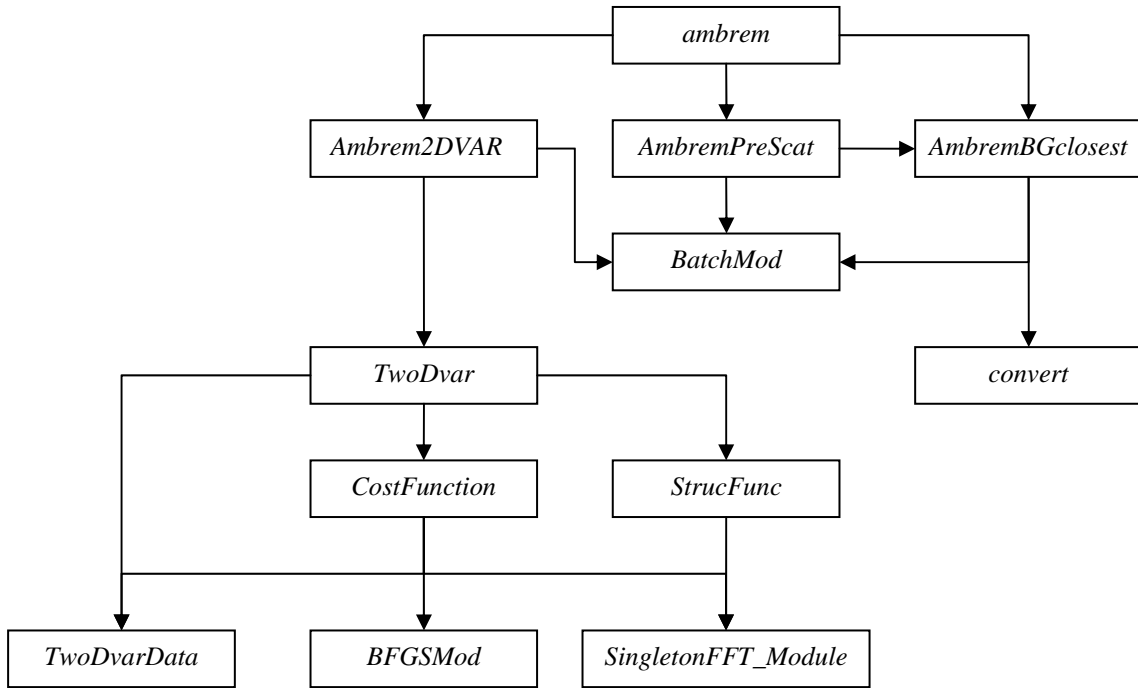
**Table 6.1** Routines of module *Ambrem*.

<b>Routine</b>	<b>Description</b>	<b>Documentation</b>
<i>Ambrem2DVAR</i>	Interface to KNMI 2DVAR method	Section 6.4
<i>AmbremBGClosest</i>	Interface to Background Closest method	Section 6.1
<i>AmbremPrescat</i>	Interface to Prescat method	Section 6.5

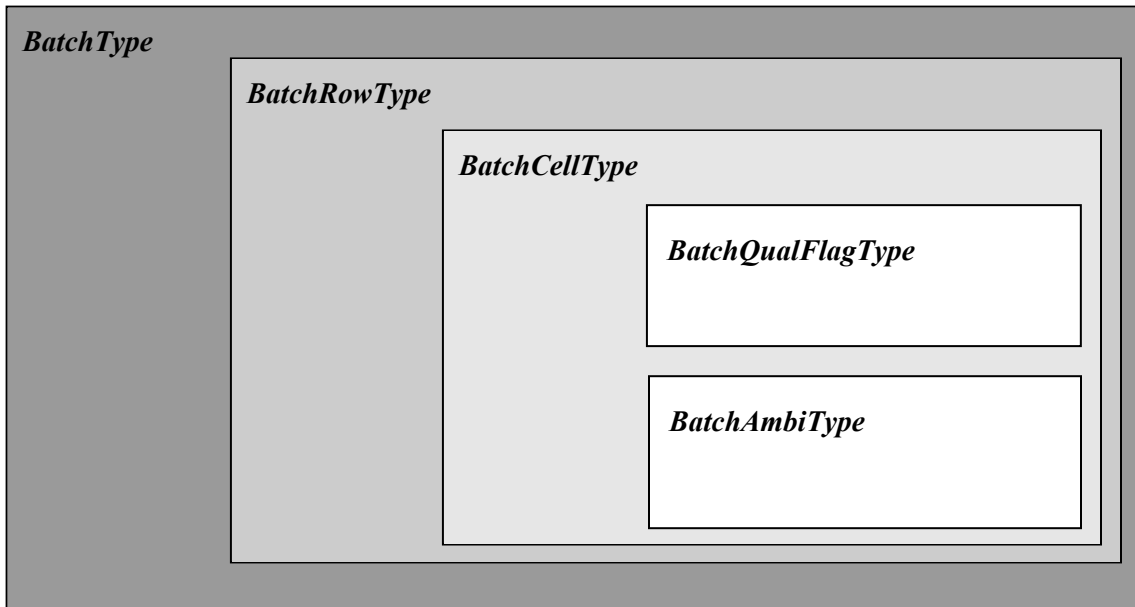
**Table 6.2** Interface modules for different Ambiguity Removal schemes.

## 6.3 Module *BatchMod*

After the wind retrieval step, the Ambiguity Removal step is performed on selections of the available data. In general, these selections are just a compact part of the swath or a compact part of the world ocean. The batch module *BatchMod* facilitates these selections of data. In fact, a batch data structure is introduced to create an interface between the swath related data and the data structures of the different AR methods. Consequently, the attributes of the batch data



**Figure 6.1** Interdependence of the modules for Ambiguity Removal. The connections from module *ambrem* to module *BatchMod* and from module *Ambrem2DVAR* to *convert* are not drawn.



**Figure 6.2** Schematic representation of the batch data structure.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

<i>BatchType</i>		
<b>Attribute</b>	<b>Type</b>	<b>Description</b>
<i>NrRows</i>	Integer	Number of rows in batch
<i>Row</i>	<i>BatchRowType</i>	Array of rows

<i>BatchRowType</i>		
<b>Attribute</b>	<b>Type</b>	<b>Description</b>
<i>RowNr</i>	Integer	Row number within orbit
<i>NrCells</i>	Integer	Number of cells in batch (max 76)
<i>Cell</i>	<i>BatchCellType</i>	Array of cells within row

<i>BatchCellType</i>		
<b>Attribute</b>	<b>Type</b>	<b>Description</b>
<i>NodeNr</i>	Integer	Node number within orbit row
<i>lat</i>	Real	Latitude
<i>lon</i>	Real	Longitude
<i>ubg</i>	Real	u-component of background wind
<i>vbg</i>	Real	v-component of background wind
<i>NrAmbiguities</i>	Integer	Number of ambiguities
<i>Ambi</i>	<i>BatchAmbiType</i>	Array of ambiguities

<i>BatchAmbiType</i>		
<b>Attribute</b>	<b>Type</b>	<b>Description</b>
<i>selection</i>	Integer	Index of selected ambiguity
<i>uana</i>	Real	u-component of analysis wind
<i>vana</i>	Real	v-component of analysis wind
<i>f</i>	Real	Contribution of this cell to cost function
<i>gu</i>	Real	Derivative of <i>f</i> to <i>u</i>
<i>gv</i>	Real	Derivative of <i>f</i> to <i>v</i>
<i>qualflag</i>	<i>BatchQualFlagType</i>	Quality control flag

**Table 6.3** Batch data structures.

To check the quality of the batch a quality flag is introduced for instances of the *BatchCellType*. The flag is set by routine *TestBatchCell()*. The attributes of this flag of type *BatchQualFlagType* are listed in table 6.4.

Module *BatchMod* contains a number of routines to control the batch structure. The calls and tasks of the various routines are listed in table 6.5. The batch structure is allocatable because it is only active between the wind retrieval and the ambiguity removal step.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

<b>Attribute</b>	<b>Description</b>
<i>Missing</i>	Quality flag not set
<i>Node</i>	Incorrect node number specification
<i>Lat</i>	Incorrect latitude specification
<i>Lon</i>	Incorrect longitude specification
<i>Ambiguities</i>	Invalid ambiguities
<i>Selection</i>	Invalid selection indicator
<i>Background</i>	Incorrect background wind specification
<i>Analysis</i>	Incorrect analysis
<i>Threshold</i>	Threshold overflow
<i>Cost</i>	Invalid cost function value
<i>Gradient</i>	Invalid gradient value

**Table 6.4** Batch quality flag attributes.

<b>Routine</b>	<b>Call</b>	<b>Description</b>
<i>AllocRowsAndCellsAndInitBatch</i>	Processor	Allocation of batch
<i>AllocAndInitBatchRow</i>	<i>AllocRowsAndCellsAndInitBatch</i>	Allocation of batch rows
<i>AllocAndInitBatchCell</i>	<i>AllocAndInitBatchRow</i>	Allocation of batch cells
<i>AllocRowsOnlyAndInitBatch</i>	not used	
<i>InitBatchModule</i>	<i>Ambrem</i>	Initialization module
<i>InitBatch</i>	<i>AllocRowsAndCellsAndInitBatch</i>	Initialization of batch
<i>InitBatchRow</i>	<i>InitBatch</i>	Initialization of batch rows
<i>InitBatchCell</i>	<i>InitBatchRow</i>	Initialization of batch cells
<i>InitbatchAmbi</i>	<i>InitBatchCell</i>	Initialization of batch ambiguities
<i>DeallocBatch</i>	Processor	Deallocation of batch
<i>DeallocBatchRows</i>	<i>DeallocBatch</i>	Deallocation of batch rows
<i>DeallocBatchCells</i>	<i>DeallocBatchRows</i>	Deallocation of batch cells
<i>DeallocBatchAmbis</i>	<i>DeallocBatchCells</i>	Deallocation of batch ambiguities
<i>TestBatch</i>	Processor	Test complete batch
<i>TestBatchRow</i>	<i>TestBatch</i>	Test complete batch row
<i>TestBatchCell</i>	<i>TestBatchRow</i>	Test batch cell
<i>TestBatchQualFlag</i>	Processor	Print the quality flag
<i>getBatchQualFlag</i>	not used	
<i>setBatchQualFlag</i>	not used	
<i>PrnBatchQualFlag</i>	not used	

**Table 6.5** Routines of module *BatchMod*.

## 6.4 The KNMI 2DVAR scheme

### 6.4.1 Introduction

The purpose of the KNMI 2DVAR scheme is to make an optimal selection provided the (modeled) likelihood of the ambiguities and the (modeled) uncertainty of the background surface wind field. First, an optimal estimated surface wind vector field (analysis) is determined based on variational principles. This is a very common method originating from the broad discipline of

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

Data Assimilation. The optimal surface wind vector field is called the analysis. Second, the selected wind vector field (the result of the 2DVAR scheme) consists of the wind vector solutions that are closest to the analysis wind vector. For details of the KNMI 2DVAR scheme formulation the reader is referred to [Vogelzang, 2007a]. Information on 2DVAR can also be found in [Stoffelen et al., 2004; de Vries et al., 2004; de Vries and Stoffelen, 2000]. These three documents may be downloaded from the EUMETSAT website, [www.eumetsat.int](http://www.eumetsat.int).

The calculation of the cost function and its gradient is rather complex matter. The reader who is only interested in how the 2DVAR scheme is assembled into the genscat module class *ambrem* is referred to subsection 6.4.2. Readers interested in the details of the cost function calculations and the minimization should also read the subsequent subsections. Subsection 6.4.3 forms an introduction to the cost function. It is recommended to first read this section, because it provides necessary background information to understand the code. Subsection 6.4.8 on the actual minimization and subsection 6.4.9 on Fast Fourier Transforms are in fact independent of the cost function itself. The reader might skip these subsections.

#### **6.4.2 Data structure, interface and initialisation**

The main module of the 2DVAR scheme is *TwoDvar*. Within the genscat ambiguity removal module class, the interface with the 2DVAR scheme is set by module *Ambrem2DVAR*. Table 6.6 lists its routines that serve the interface with *TwoDvar*.

<b>Routine</b>	<b>Call</b>	<b>Description</b>
<i>Do2DVARonBatch</i>	<i>DoAmbrem</i>	Apply 2DVAR scheme on batch
<i>BatchInput2DVAR</i>	<i>Do2DVARonBatch</i>	Fills the 2DVAR data structure with input
<i>BatchOutput2DVAR</i>	<i>Do2DVARonBatch</i>	Fills the batch data structure with output
<i>SetAlpha</i>	<i>BatchInput2DVAR</i>	Sets the observation orientation
<i>GetBatchSize2DVAR</i>		Determine maximum size of batch
<i>latlon2xyz</i>	<i>SetAlpha</i>	Coordinate transformation
<i>rotuv</i>	<i>BatchInput2DVAR</i>	Calculates the rotation of the ( <i>u,v</i> ) wind field

**Table 6.6** Routines of module *Ambrem2DVAR*.

These routines are sufficient to couple the 2DVAR scheme to the processor. The actual 2DVAR processing is done by the routines of module *TwoDvar* itself. These routines are listed in table 6.7. Figures B2.1-B2.7 show the complete calling tree of the AR routines.

The *Obs2dvarType* data type is the main data structure for the observed winds. Its attributes are listed in table 6.8. The *TDV\_Type* data type contains all parameters that have to do with the 2DVAR batch grid: dimensions, sizes, and derived parameters. Its attributes are listed in table 6.9. These data structures are defined in module *TwoDvarData* and the routines in this module are listed in table 6.10.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

<b>Routine</b>	<b>Call</b>	<b>Description</b>
<i>InitTwodvarModule</i>		Initialization of module <i>TwoDvar</i>
<i>Do2DVAR</i>	<i>Do2DVARonBatch</i>	Cost function minimization
<i>Init2DVARmeth</i>	<i>Do2DVARonBatch</i>	Initialize the 2DVAR scheme
<i>ExitTwodvarModule</i>		Deallocation of module <i>TwoDvar</i>

**Table 6.7** Routines of module *TwoDvar*.

<b>Attribute</b>	<b>Type</b>	<b>Description</b>
<i>alpha</i>	Real	Rotation angle
<i>cell</i>	Integer	Store batch cell number
<i>row</i>	Integer	Store batch row number
<i>igrd</i>	Integer	Row index
<i>jgrid</i>	Integer	Node (wind vector cell) index
<i>lat</i>	Real	Latitude to determine structure function
<i>Wll</i>	Real	Weight lower left
<i>Wlr</i>	Real	Weight lower right
<i>Wul</i>	Real	Weight upper left
<i>Wur</i>	Real	Weight upper right
<i>ubg</i>	Real	Background EW wind component
<i>vbg</i>	Real	Background NS wind component
<i>SkipForAnalysis</i>	Logical	Don't include this observation in the cost function
<i>NrAmbiguities</i>	Integer	Number of ambiguities
<i>incr()</i>	<i>AmbiIncrType</i>	Ambiguity increments
<i>uAnaIncr</i>	Real	Analysis increment
<i>vAnaIncr</i>	Real	Analysis increment
<i>selection</i>	Integer	Selection flag
<i>QualFlag</i>	<i>TwoDvarQualFlagType</i>	Quality control flag
<i>f</i>	Real	Cost function at observation
<i>gu</i>	Real	$df/du$
<i>gv</i>	Real	$df/dv$

**Table 6.8** The *Obs2dvarType* data structure.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

<b>Attribute</b>	<b>Type</b>	<b>Description</b>
<i>delta</i>	Real	2DVAR grid size in position domain
<i>delta_p</i>	Real	2DVAR grid size in frequency domain
<i>delta_q</i>	Real	2DVAR grid size in frequency domain
<i>N1</i>	Integer	Dimension 1 of 2DVAR grid
<i>H1</i>	Integer	$N1/2$
<i>K1</i>	Integer	$H1+1$ ; number of nonnegative frequencies
<i>N2</i>	Integer	Dimension 2 of 2DVAR grid
<i>H2</i>	Integer	$N2/2$
<i>K2</i>	Integer	$H2+1$ ; number of nonnegative frequencies
<i>GridExtent</i>	Integer	Size of free edge in grid points
<i>Ncontrol</i>	Integer	Size of control vector
<i>WithGEP</i>	Logical	Gross Error Probabilities applied?
<i>NWVC</i>	Integer	Number of WVC's
<i>GEP(1:84)</i>	Real	Values of GEP for WVC 1 to NWVC
<i>Verbosity</i>	Integer	Verbosity parameter

**Table 6.9** The *TDV\_Type* data structure.

<b>Routine</b>	<b>Call</b>	<b>Description</b>
<i>TDV_Init</i>	<i>InitTwodvarModule</i>	Initialization of 2DVAR grid and preparations
<i>Set_HelmholzCoefficients</i>	<i>TDV_Init</i>	Set Helmholz transformation coefficients
<i>Set_CFW</i>	<i>TDV_Init</i>	Set cost function weights
<i>Exit_TDV</i>	<i>ExitTwodvarModule</i>	Deallocate memory
<i>InitObs2dvar</i>	<i>BatchInput2DVAR,</i> <i>BatchOutput2DVAR</i>	Allocation of observations array
<i>DeallocObs2dvar</i>	<i>BatchOutput2DVAR</i>	Deallocation of observations array
<i>InitOneObs2dvar</i>	<i>InitObs2dvar</i>	Initialization of single observation
<i>TestObs2dvar</i>	<i>Do2DVAR</i>	Test single observation
<i>PrintObs2dvar</i>	<i>BatchInput2DVAR</i>	Print a single 2DVAR observation
<i>Prn2DVARQualFlag</i>	<i>Do2DVAR</i>	Print observation quality flag
<i>set2DVARQualFlag</i>	<i>TestObs2DVAR</i>	Convert observation quality flag to integer
<i>get2DVARQualFlag</i>	not used	Convert integer to observation quality flag

**Table 6.10** Routines in module *TwoDvarData*.

The quality status of an instance of *Obs2dvarType* is indicated by the attribute *QualFlag* which is an instance of *TwoDvarQualFlagType*. The attributes of this flag are listed in table 6.11.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

<b>Attribute</b>	<b>Description</b>
<i>missing</i>	Flag values not set
<i>wrong</i>	Invalid 2DVAR process
<i>Lat</i>	Invalid latitude
<i>Background</i>	Invalid background wind increment
<i>Ambiguities</i>	Invalid ambiguity increments
<i>Selection</i>	Invalid selection
<i>Analyse</i>	Invalid analysis wind increment
<i>Cost</i>	Invalid cost function specification
<i>gradient</i>	Invalid gradient specification
<i>weights</i>	Invalid interpolation weights
<i>grid</i>	Invalid grid indices

**Table 6.11** Attributes of 2DVAR observation quality flag.

### **6.4.3 Reformulation and transformation**

The minimization problem to find the analysis surface wind field (the 2D variational Data Assimilation problem) may be formulated as

$$\min_v J(v) \quad , \quad J(v) = J_{obs}(v) + J_{bg}(v) \quad , \quad (6.1)$$

where  $v$  is the surface wind field in consideration and  $J$  the total cost function consisting of the observational term  $J_{obs}$  and the background term  $J_{bg}$ . The solution, the analysis surface wind field, may be denoted as  $v_a$ . Being just a weighted least squares term, the background term may be further specified as

$$J_{bg}(v) = [v - v_{bg}]^T B^{-1} [v - v_{bg}] \quad , \quad (6.2)$$

where  $B$  is the background error covariance matrix. The  $J_{obs}$  term of the 2DVAR scheme is not simply a weighted least squares term.

Such a formulation does not closely match the code of the 2DVAR scheme. In fact, for scientific and technical reasons several transformations are applied to reformulate the minimization problem. Description of these transformations is essential to understand the different procedures within the code. The interested reader is referred to *Vogelzang* [2007a].

### **6.4.4 Module CostFunc**

Module *CostFunc* contains the main procedure for the calculation of the cost function and its gradient. It also contains the minimization procedure. Table 6.12 provides an overview of the routines.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

<b>Routine</b>	<b>Call</b>	<b>Description</b>
<i>Jt</i>	<i>minimise</i>	Total cost function and gradient
<i>Jb</i>	<i>Jt</i>	Background term of cost function
<i>Jo</i>	<i>Jt</i>	Observational term of cost function
<i>JoScat</i>	<i>Jo</i>	Single observation contribution to the cost function
<i>Unpack_ControlVector</i>	<i>Jo</i>	Unpack of control vector
<i>Pack_ControlVector</i>	<i>Jo</i>	Pack of control vector (or its gradient)
<i>Uncondition</i>	<i>Jo</i>	Several transformations of control vector
<i>Uncondition_adj</i>	<i>Jo</i>	Adjoint of <i>Uncondition</i> .
<i>minimize</i>	<i>Do2DVAR (TwoDvar)</i>	Minimization

**Table 6.12** Routines of module *CostFunc*.

#### **6.4.5 Adjoint method**

The minimization of cost function is done with a quasi-Newton method. Such a method requires an accurate approximation of the gradient of the cost function. The adjoint method is just a very economical manner to calculate this gradient. For introductory texts on the adjoint method and adjoint coding, see, e.g., [Talagrand, 1991; Giering, 1997]. For detailed information on the adjoint model in 2DVAR see Vogelzang [2007a].

#### **6.4.6 Structure Functions**

Module *StrucFunc* contains the routines to calculate the covariance matrices for the stream function,  $\psi$ , and the velocity potential,  $\chi$ . Its routines are listed in table 6.13.

<b>Routine</b>	<b>Call</b>	<b>Description</b>
<i>PrintStrcFuncPars</i>	not used	
<i>SetCovMat</i>	<i>Do2DVAR</i>	Calculate the covariance matrices
<i>InitStrcFunc</i>	<i>SetCovMat</i>	Initialize the structure functions
<i>StrcFuncPsi</i>	<i>SetCovMat</i>	Calculate $\psi$
<i>StrcFuncChi</i>	<i>SetCovMat</i>	Calculate $\chi$

**Table 6.13** Routines of module *StrucFunc*.

The default structure functions are Gaussians of the form

$$\rho_{\psi\psi}(r) = (1 - \nu)^2 L_{\psi}^2 e^{-\frac{r^2}{R_{\psi}^2}}, \quad \rho_{\chi\chi}(r) = \nu^2 L_{\chi}^2 e^{-\frac{r^2}{R_{\chi}^2}}, \quad (6.3)$$

with the parameters given in table 6.14. The length scale  $L_{\psi}^2$  is defined as [Daley, 1991]

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

$$L_{\psi}^2 = \frac{-\rho_{\psi\psi}(r)}{\nabla^2 \rho_{\psi\psi}(r)} \Big|_{r=0}, \quad (6.4)$$

with a similar definition for  $L_{\chi}^2$ . For a Gaussian one obtains  $L_{\psi}^2 = \frac{1}{2} R_{\psi}^2$  and  $L_{\chi}^2 = \frac{1}{2} R_{\chi}^2$ .

Zone	$R_{\psi}$ (km)	$R_{\chi}$ (km)	$\nu^2$
Northern Hemisphere	300	300	0.2
Tropics	600	600	0.5
Southern Hemisphere	300	300	0.2

**Table 6.14** Default structure function parameters.

Routine *InitStrucFunc* sets the structure function parameters to their default form or, when specified by the `-par` command, reads them from file. The structure of the parameter file is shown in table 6.15.

Record nr.	Parameters	Description
1	$N_1, N_2, E, \Delta$	2DVAR batch grid size and dimension
2	<i>Type</i>	Structure function type: Gaus, TOAR, or Dipo
3	$R_{\psi}, R_{\chi}, \varepsilon_{\psi}, \varepsilon_{\chi}, \nu^2$	Background error parameters for NH
4	$R_{\psi}, R_{\chi}, \varepsilon_{\psi}, \varepsilon_{\chi}, \nu^2$	Background error parameters for Tropics
5	$R_{\psi}, R_{\chi}, \varepsilon_{\psi}, \varepsilon_{\chi}, \nu^2$	Background error parameters for SH
6	$\varepsilon_t, \varepsilon_l$	Observation errors
7	<i>NewGEP</i>	New GEP definition?
8	$L_{GEP}, N_{GEP}$	Apply GEP's and number of GEP's
9	$P_{GE}(i), i=1, N_{GEP}$	GEP values per WVC

**Table 6.15** Structure of file with 2DVAR parameters read with `-par` option.

In table 6.15,  $N_1$  and  $N_2$  stands for the grid sizes (in grid points) across and along track, respectively.  $E$  denotes the size of the free edge in grid points and  $\Delta$  the batch grid size in m. Note that the number of WVC's in the batch grid plus twice the free edge size should at least equal the total number of batch grid points. This is not checked for! Further,  $N_1$  and  $N_2$  should be even and  $\Delta$  can have the values 25000, 50000 or 100000 (25 km, 50 km or 100 km, respectively).  $R_{\psi}$  and  $R_{\chi}$  are the background error correlation lengths,  $\varepsilon_{\psi}$  and  $\varepsilon_{\chi}$  the background error standard deviations, and  $\nu^2$  the ratio of the rotational over the divergent strength.  $\varepsilon_t$  and  $\varepsilon_l$  stand for the observation error standard deviation across and along the satellite path. NewGEP determines if the definition of the Gross Error Probabilities should be changed from default. If it is false reading of the file stops here. If it is true,  $L_{GEP}$  determines if the GEP's should be applied (it can be set to false!),  $N_{GEP}$  sets the number of GEP's and should be equal to the number of WVC's used. The GEP's themselves are given by  $P_{GE}$ .

The parameter *Type* is a character string of length 4 and can have the values 'Gaus' for a Gaussian structure function, 'TOAR' for a third order autoregressive structure function

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

$$\rho_{\psi\psi}(r) = (1-\nu)^2 L_{\psi}^2 \left( 1 + \frac{r}{R_{\psi}} + \frac{r^2}{3R_{\psi}^2} \right) e^{-\frac{r}{R_{\psi}}}, \quad (6.5)$$

and ‘Dipo’ for a dipole structure function

$$\rho_{\psi\psi}(r) = (1-\nu)^2 \frac{1}{1 + \frac{r^2}{R_{\psi}^2}}, \quad (6.6)$$

with similar definitions for  $\rho_{\chi\chi}$ . The latter two structure functions are included for research purposes, though the TOAR structure function has been used in practice. Note that the structure function type should be enclosed in quotes in the parameter file so that Fortran reads the characters without error in free format.

If the resolution is set to 50 km or 100 km, the 2DVAR parameter file structure in table 6.15 should be repeated twice: once for the default run at 25 km and once for the final run at 50 km or 100 km resolution.

#### **6.4.7 Minimisation**

The minimization routine used is *LBFGS*. This is a quasi Newton method with a variable rank for the approximation of the Hessian written by J. Nocedal. A detailed description of this method is given by *Liu and Nocedal* [1989]. Routine *LBFGS* is freeware and can be obtained from web page [www.netlib.org/opt/index.html](http://www.netlib.org/opt/index.html), file *lbfgs\_um.shar*. The original Fortran77 code has been adjusted to compile under Fortran90 compilers. Routine *LBFGS* and its dependencies are located in module *BFGSMod.F90* in directory *genscat/support/BFGS*. Table 6.16 provides an overview of the routines in this module.

Routine *LBFGS* uses reverse communication. This means that the routine returns to the calling routine not only if the minimization process has converged or when an error has occurred, but also when a new evaluation of the function and the gradient is needed. This has the advantage that no restrictions are imposed on the form of routine *Jt* calculating the cost function and its gradient.

<b>Routine</b>	<b>Call</b>	<b>Description</b>
<i>LBFGS</i>	<i>minimise</i>	Main routine
<i>LBI</i>	<i>LBFGS</i>	Printing of output (switched off)
<i>daxpy</i>	<i>LBFGS</i>	Sum of a vector times a constant plus another vector with loop unrolling.
<i>ddot</i>	<i>LBFGS</i>	Dot product of two vectors using loop unrolling.
<i>MCSRCH</i>	<i>LBFGS</i>	Line search routine.
<i>MCSTEP</i>	<i>MCSRCH</i>	Calculation of step size in line search.

**Table 6.16** Routines in module *BFGSMod*.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

The formal parameters of *LBFGS* have been extended to include all work space arrays needed by the routine. The work space is allocated in the calling routine *minimise*. The rank of *LBFGS* affects the size of the work space. It has been fixed to 3 in routine *minimise*, because this value gave the best results (lowest values for the cost function at the final solution).

Some of the error returns of the line search routine *MCSRCH* have been relaxed and are treated as a normal return. Further details can be found in the comment in the code itself.

Routines *daxpy* and *ddot* were rewritten in Fortran90. These routines, originally written by J. Dongarra for the Linpack library, perform simple operations but are highly optimized using loop unrolling. Routine *ddot*, for instance, is faster than the equivalent Fortran90 intrinsic function *dot\_product*.

#### **6.4.8 SingletonFFT Module**

Module *SingletonFFT\_Module* in directory *genscat/support/singletonfft* contains mixed radix FFT routines needed in the 2DVAR scheme. The software originates from the code by R.C. Singleton and is obtainable as freeware at [www.netlib.org](http://www.netlib.org). The original Fortran 77 code has been translated to Fortran 90, because the Portland Fortran compiler did not handle the original code correctly at all optimisation levels. A number of subroutines has been split off from the original routine *fft*. Routine *SingletonFFT2d* acts as an interface between 2DVAR and *fft*. In principle, the code can handle any FFT dimension, but large prime factors should be avoided for reasons of memory limitations and computational efficiency. Moreover, 2DVAR requires even grid dimensions. Table 6.17 gives an overview of the available routines. Figure B.2.11 shows the calling tree of the FT routines relevant for 2DVAR.

<b>Routine</b>	<b>Call</b>	<b>Description</b>
<i>SingletonFFT2d</i>	<i>SetCovMat, Uncondition, Uncondition_adj</i>	Interface with 2DVAR
<i>fft</i>	<i>SingletonFFT2d</i>	Main FFT routine
<i>SFT_PrimeFactor</i>	<i>fft</i>	Prime factorization
<i>SFT_Base2</i>	<i>fft</i>	Base 2 FFT
<i>SFT_Base3</i>	<i>fft</i>	Base 3 FFT
<i>SFT_Base4</i>	<i>fft</i>	Base 4 FFT
<i>SFT_Base5</i>	<i>fft</i>	Base 5 FFT
<i>SFT_BaseOdd</i>	<i>fft</i>	General odd base FFT
<i>SFT_Permute</i>	<i>fft</i>	Permutation
<i>SFT_PermuteSinglevariate</i>	<i>SFT_Permute</i>	Permutation
<i>SFT_PermuteMultivariate</i>	<i>SFT_Permute</i>	Permutation
<i>SFT_Rotate</i>	<i>fft</i>	Rotation

**Table 6.17** Fourier transform routines.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

Remarks:

- Reading in the 2DVAR structure function parameters from an external file with the `-par` command line option, thereby changing their default values, is at your own risk!
- The 2DVAR implementation can be made more efficient by using a real-to-real FFT routine rather than a complex-to-complex one as implemented now. Since SDP satisfies the requirements in terms of computational speed, this has low priority.

## 6.5 The PreScat scheme

The PreScat ambiguity removal scheme is not used within SDP. More information on this scheme can be found in [*Stoffelen et al.*, 2004].

NWP SAF	SDP User Manual and Reference Guide	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
---------	-------------------------------------	---

## Chapter 7

# Module *BufrMod*

Module *BufrMod* is part of the genscat support modules. The current version is a Fortran90 wrapper around the ECMWF BUFR library (see [www.ecmwf.int](http://www.ecmwf.int)). The goal of this support module is to provide a comprehensive interface to BUFR data for every Fortran90 program using it. In particular, *BufrMod* provides all the BUFR functionality required for the scatterometer processor based on genscat. Special attention has been paid to testing and error handling.

### 7.1 Background

The acronym BUFR stands for Binary Universal Form for the Representation of data. BUFR is maintained by the World Meteorological Organization WMO and other meteorological centers. In brief, the WMO FM-94 BUFR definition is a binary code designed to represent, employing a continuous binary stream, any meteorological data. It is a self defining, table driven and very flexible data representation system. It is beyond the scope of this document to describe BUFR in detail. Complete descriptions are distributed via the websites of WMO ([www.wmo.int](http://www.wmo.int)) and of the European Center for Medium-range Weather Forecasts ECMWF ([www.ecmwf.int](http://www.ecmwf.int)).

Module *BufrMod* is in fact an interface. On the one hand it contains (temporary) definitions to set the arguments of the ECMWF library functions. On the other hand, it provides self explaining routines to be incorporated in the wider Fortran90 program. Section 7.2 describes the routines in module *BufrMod*. The public available data structures are described in section 7.3. *BufrMod* uses two libraries: the BUFR software library of ECMWF and BUFRIO, a small library in C for file handling at the lowest level. These libraries are discussed in some more detail in section 7.4.

### 7.2 Routines

Table 7.1 provides an overview of the routines in module *BufrMod*. The most important ones are described below.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

<b>Routine</b>	<b>Call</b>	<b>Description</b>
<i>InitAndSetNrOfSubsets</i>	SDP	Initialization routine
<i>set_BUFR_fileattributes</i>	SDP	Initialization routine
<i>open_BUFR_file</i>	SDP	Opens a BUFR file
<i>get_BUFR_nr_of_messages</i>	SDP	Inquiry of BUFR file
<i>get_BUFR_message</i>	SDP	Reads instance of <i>BufrDataType</i> to file
<i>get_expected_BUFR_msg_size</i>	<i>get_BUFR_message</i>	
<i>ExpandBufrMessage</i>	<i>get_BUFR_message</i>	Convert from <i>BufrMessageType</i> to <i>BufrSectionsType</i>
<i>PrintBufrErrorCode</i>	<i>ExpandBufrMessage</i>	
<i>CheckBufrTables</i>	<i>ExpandBufrMessage</i>	Control
<i>get_file_size</i>	<i>CheckBufrTables</i>	Determine size of BUFR file
<i>get_bufrfile_size_c</i>	<i>get_file_size</i>	Support routine in C
<i>encode_table_b</i>	<i>CheckBufrTables</i>	
<i>encode_table_d</i>	<i>CheckBufrTables</i>	
<i>FillBufrSecData</i>	<i>get_BUFR_message</i>	Convert from <i>BufrSectionsType</i> to <i>BufrDataType</i>
<i>close_BUFR_file</i>	SDP	Closes a BUFR file
<i>BufrReal2Int</i>	SDP	Conversion
<i>save_BUFR_message</i>	not used	Saves instance of <i>BufrDataType</i> to file
<i>EncodeBufrData</i>	<i>save_BUFR_message</i>	Convert from <i>BufrSectionsType</i> to <i>BufrMessageType</i>
<i>CheckBufrData</i>	<i>EncodeBufrData</i>	Control
<i>FillBufrData</i>	<i>save_BUFR_message</i>	Convert from <i>BufrDataType</i> to <i>BufrSectionsType</i>
<i>bufr_msg_is_valid</i>	not used	
<i>set_bufr_msg_to_invalid</i>	not used	
<i>PrintBufrData</i>	not used	
<i>GetPosBufrData</i>	not used	
<i>GetRealBufrData</i>	not used	
<i>GetIntBufrData</i>	not used	
<i>GetRealBufrDataArr</i>	not used	
<i>GetIntBufrDataArr</i>	not used	
<i>GetRealAllBufrDataArr</i>	not used	
<i>CloseBufrHelpers</i>	not used	
<i>missing_real</i>	not used	
<i>missing_int</i>	not used	
<i>int2real</i>	not used	
<i>do_range_check_int</i>	not used	
<i>do_range_check_real</i>	not used	
<i>AddRealDataToBufrMsg</i>	not used	
<i>AddIntDataToBufrMsg</i>	not used	
<i>PrintBufrModErrorCode</i>	not used	
<i>BufrInt2Real</i>	not used	
<i>GetFreeUnit</i>	not used	There is also a copy in module <i>SwsSupport</i>

**Table 7.1** Routines of module *BufrMod*.

**Reading (decoding):** Routine *get\_BUFR\_message()* reads a single BUFR message from the BUFR file and creates an instance of *BufrDataType*.

**Writing (encoding):** Routine *save\_BUFR\_message()* saves a single BUFR message to the BUFR file. The data should be provided as an instance of *BufrDataType*. This routine is not used in SDP.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

**Checking and Printing:** The integer parameter *BufrVerbosity* controls the extent of the log statements while processing the BUFR file. The routines *PrintBufrData()* and *CheckBufrData()* can be used to respectively print and check instances of *BufrDataType*.

**Open and Close BUFR files:** The routine *open\_BUFR\_file()* opens the BUFR file for both read (*writemode=.false.*) and writing (*writemode=.true.*). Routine *set\_BUFR\_fileattributes()* determines several aspects of the BUFR file and saves these data in an instance of *bufr\_file\_attr\_data*, see table 7.5. Routine *get\_BUFR\_nr\_of\_messages()* is used to determine the number of BUFR messages in the file. Finally, routine *close\_BUFR\_file()* closes the BUFR file.

As said before, the underlying encoding and decoding routines originate from the ECMWF BUFR library, with the BUFRIO library acting as an intermediate. Appendix B3 shows the calling trees of the routines in module *BufrMod* that are used in SDP.

### 7.3 Data structures

The data type closest to the actual BUFR messages in the BUFR files is the *BufrMessageType*, see table 7.2. These are still encoded data. Every BUFR message consists of 5 sections and one supplementary section. After decoding (expanding) the BUFR messages, the data are transferred into an instance of *BufrSectionsType*, see table 7.3, which contains the data and meta data in integer values subdivided in these sections.

Attribute	Type	Description
<i>buff</i>	Integer (max_bufr_mess_size)	BUFR message, all sections
<i>size</i>	Integer	Size in bytes of BUFR message
<i>nr_of_words</i>	Integer	Idem, now size in words

**Table 7.2** Attributes for the *BufrMessageType* data type.

Attribute	Type	Description
<i>ksup</i> (9)	Integer	Supplementary info and items selected from the other sections
<i>ksec</i> (3)	Integer	Expanded section 0 (indicator)
<i>ksec1</i> (40)	Integer	Expanded section 1 (identification)
<i>ksec2</i> (64)	Integer	Expanded section 2 (optional)
<i>ksec3</i> (4)	Integer	Expanded section 3 (data description)
<i>ksec4</i> (2)	Integer	Expanded section 4 (data)

**Table 7.3** Attributes for the *BufrSectionsType* data type.

The next step is to bring the section data to actual dimensions, descriptions and values of data which can be interpreted as physical parameters. Therefore, instances of *BufrSectionsType* are transferred to instances of *BufrDataType*, see table 7.4. The actual data for input or output in a BUFR message should be an instance of the *BufrDataType* data type. Some meta information on the BUFR file is contained in the self explaining *bufr\_file\_attr\_data* data type, see table 7.5.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

<b>Attribute</b>	<b>Type</b>	<b>Description</b>
<i>Nsec0</i>	Integer	ksup ( 9) dimension section 0
<i>nsec0size</i>	Integer	ksec0( 1) size section 0
<i>nBufLength</i>	Integer	ksec0( 2) length BUFR
<i>nBufEditionNumber</i>	Integer	ksec0( 3)
<i>Nsec1</i>	Integer	ksup ( 1) dimension section 1
<i>nsec1size</i>	Integer	ksec1( 1) size section 1
<i>kEditionNumber</i>	Integer	ksec1( 2)
<i>Kcenter</i>	Integer	ksec1( 3)
<i>kUpdateNumber</i>	Integer	ksec1( 4)
<i>kOptional</i>	Integer	ksec1( 5)
<i>ktype</i>	Integer	ksec1( 6)
<i>ksubtype</i>	Integer	ksec1( 7) local use
<i>kLocalVersion</i>	Integer	ksec1( 8)
<i>kyear</i>	Integer	ksec1( 9) century year
<i>kmonth</i>	Integer	ksec1(10)
<i>kday</i>	Integer	ksec1(11)
<i>khour</i>	Integer	ksec1(12)
<i>kminute</i>	Integer	ksec1(13)
<i>kMasterTableNumber</i>	Integer	ksec1(14)
<i>kMasterTableVersion</i>	Integer	ksec1(15)
<i>ksubcenter</i>	Integer	ksec1(16)
<i>klocalinfo()</i>	Integer	ksec1(17:40)
<i>Nsec2</i>	Integer	ksup ( 2) dimension section 2
<i>nsec2size</i>	Integer	ksec2( 1) size section 2
<i>key(46)</i>	Integer	ksec2( 2: ) key
<i>Nsec3</i>	Integer	ksup ( 3) dimension section 3
<i>nsec3size</i>	Integer	ksec3( 1) size section 3
<i>Kreserved3</i>	Integer	ksec3( 2) reserved
<i>ksubsets</i>	Integer	ksec3( 3) number of reserved subsets
<i>kDataFlag</i>	Integer	ksec3( 4) compressed (0,1) observed (0,1)
<i>Nsec4</i>	Integer	ksup ( 4) dimension section 4
<i>nsec4size</i>	Integer	ksec4( 1) size section 4
<i>kReserved4</i>	Integer	ksec4( 2) reserved
<i>nelements</i>	Integer	ksup ( 5) actual number of elements
<i>nsubsets</i>	Integer	ksup ( 6) actual number of subsets
<i>nvals</i>	Integer	ksup ( 7) actual number of values
<i>nbufsize</i>	Integer	ksup ( 8) actual size of BUFR message
<i>ktklen</i>	Integer	Actual number of data descriptors
<i>ktdexl</i>	Integer	Actual number of expanded data descriptors
<i>ktlst()</i>	Integer array	List of data descriptors
<i>ktdexp()</i>	Integer array	List of expanded data descriptors
<i>values()</i>	Real array	List of values
<i>cvals()</i>	Character array	List of CCITT IA no. 5 elements
<i>cnames()</i>	Character array	List of expanded element names
<i>cunits()</i>	Character array	List of expanded element units

**Table 7.4** Attributes of the BUFR message data type *BufData*Type.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

<b>Attribute</b>	<b>Type</b>	<b>Description</b>
<i>nr_of_BUFR_mesages</i>	Integer	Number of BUFR messages
<i>bufr_filename</i>	Character	BUFR file
<i>bufr_fileunit</i>	Integer	Fortran unit of BUFR file
<i>file_size</i>	Integer	Size of BUFR file
<i>file_open</i>	Logical	Open status of BUFR file
<i>writemode</i>	Logical	Reading or writing mode of BUFR file
<i>is_cray_blocked</i>	Integer	Cray system blocked?
<i>list_of_BUFR_startpointers()</i>	Integer	Pointers to BUFR messages
<i>message_is_valid()</i>	Logical	Validity of BUFR messages

**Table 7.5** Attributes of the *bufr\_file\_attr\_data* data type for BUFR files.

Further, module *BufrMod* needs two environment variables, `BUFR_Tables` and `BUFRTAB_DIR`, that should point to directory `/genscat/support/bufr/bufr_tables`.

## 7.4 Libraries

Module *BufrMod* uses two libraries: the BUFR software library of ECMWF and BUFRIO, a small library in C for file handling at the lowest level.

The BUFR software library of ECMWF is used as a basis to encode and decode BUFR data. This software library is explained in [Dragosavac, 1994].

Appendix D provides an overview of the different routines of this library. From the calling trees in Appendix B3 it can be inferred that only a few routines of the BUFR software library are actually used.

Library BUFRIO contains routines for BUFR file handling at the lowest level. Since this is quite hard to achieve in Fortran, these routines are coded in C. The routines of BUFRIO are listed in table 7.6. The source file (`bufr.io.c`) is located in subdirectory `genscat/support/bufr`.

<b>Routine</b>	<b>Call</b>	<b>Description</b>
<i>bufr_open</i>	<i>open_BUFR_file</i>	
<i>bufr_split</i>	<i>open_BUFR_file</i>	
<i>bufr_read_allsections</i>	<i>get_BUFR_message</i>	Read <i>BufrMessageType</i> from BUFR file
<i>bufr_get_section_sizes</i>	<i>get_BUFR_message</i>	
<i>bufr_swap_allsections</i>	<i>get_BUFR_message, save_BUFR_message</i>	Optional byte swapping
<i>bufr_write_allsections</i>	<i>save_BUFR_message</i>	Write <i>BufrMessageType</i> to BUFR file
<i>bufr_close</i>	<i>close_BUFR_file</i>	
<i>bufr_error</i>	see appendix B.3	Error handling

**Table 7.6** Routines in library BUFRIO.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

## 7.5 BUFR table routines

BUFR tables are used to define the data descriptors. The presence of the proper BUFR tables is checked before calling the reading and writing routines. If absent, it is tried to create the needed BUFR tables from the text version, available in genscat.

## 7.6 Center specific modules

BUFR data descriptors are integers. These integers consist of class numbers and numbers for the described parameter itself. These numbers are arbitrary. To establish self documenting names for the BUFR data descriptors for a Fortran90 code several center specific modules are created. These modules are listed in table 7.7. Note that these modules are just cosmetic and not essential for the encoding or decoding of the BUFR data.

<b>Module</b>	<b>Description</b>
<i>WmoBufrMod</i>	WMO standard BUFR data description
<i>KnmiBufrMod</i>	KNMI BUFR data description
<i>EcmwfBufrMod</i>	ECMWF BUFR data description

**Table 7.7** Fortran90 BUFR modules.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

## Chapter 8

# Module *iceModelMod*

Module *iceModelMod* is part of *genscat*. It contains all the Bayesian statistics routines, including the routines for spatial and temporal averaging. It also contains all the routines for initialising and printing of the SSM/I grids for the North Pole and South Pole region.

### 8.1 Background

The `-icemodel` option in SDP basically fills the fields Ice Probability (BUFR item 87) and Ice Age (BUFR item 88). Also it can output graphical maps of ice model related parameters on an SSM/I grid for the North Pole and for the South Pole region.

Each time the MetOp satellite passes over the pole region the corresponding ice map is updated with the new ASCAT data. A spatial and temporal averaging is performed in order to digest the new information. After the overpass, at the end of processing an entire BUFR file, the updated information on the ice map is put back into the BUFR structure. Optionally graphical maps are plotted, which can be controlled by optional input parameters for routine `printIceMap`. The graphical filenames have encoded the North Pole/South Pole, the date/time as well as the parameter name. The most important ones are:

`print_a`: file `[N|S][yyyymmddhhmmss].ppm` contains the ice subclass and the a-ice parameter on a grey-scale for points classified as ice.

`print_t`: file `[N|S][yyyymmddhhmmss]t.ppm` contains the ice class.

`print_sst`: file `[N|S][yyyymmddhhmmss]sst.ppm` contains the sea surface temperature

`print_postprob`: file `[N|S][yyyymmddhhmmss]postprob.ppm` contains the a-posteriori ice probability.

Typically at least two days of SeaWinds data are needed to entirely fill the ice map with data and give meaningful ice model output. Because SDP handles only one BUFR file at a time, a script is needed that calls SDP several times. After each SDP-run a binary restart file is written to disk

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

containing the information of an icemap (`latestIceMapN.rst` for the North Pole and `latestIceMapS.rst` for the South Pole). With the next call of `awdp`, these restart files are read in again. Environment variable `$RESTARTDIR` contains the directory for the ice model restart files.

Optionally sea surface temperature (SST) data from GRIB files can be used to further improve the quality of the ice algorithm (the `use_sst` logical must be turned on).

The SSM/I grids are widely used for representation of ice related parameters. A good description as well as some software routines can be found on the website of the National Snow and Ice Data Centre (NSIDC): [http://www.nsidc.org/data/docs/daac/ae\\_si25\\_25km\\_tb\\_and\\_sea\\_ice.gd.html](http://www.nsidc.org/data/docs/daac/ae_si25_25km_tb_and_sea_ice.gd.html).

A more detailed description of the Bayesian statistics method and ice model is given in [*Verspeek, 2006*].

## 8.2 Routines

Table 8.1 provides an overview of the routines in module *iceModelMod*.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

<b>Routine</b>	<b>Call</b>	<b>Description</b>
<i>calcAave</i>	<i>iceMapWeighted</i>	Calculate the space-time averaged ice map parameters
<i>calcPoly3</i>	<i>calcIcelineParms,</i> <i>iceLine</i>	Calculate a 3 <sup>rd</sup> order polynomial
<i>calcSubClass</i>	<i>iceMapWeighted</i>	Calculate the ice subclass
<i>getClass</i>	<i>updateIcePixel,</i> <i>nonbayesianIceModel</i>	Get the ice class (sea or ice)
<i>printClass</i>	not used	Print the ice class (sea or ice)
<i>getLatest</i>	<i>iceMapWeighted,</i> <i>calcAave,</i> <i>calcSubclass,</i> <i>updateIcePixel,</i> <i>printIcePixel</i>	Get the indices of the latest measurement
<i>getPrevious</i>	<i>iceMapWeighted,</i> <i>calcAave</i>	Get the indices of the previous measurement
<i>GetQxGivenIce</i>	<i>updateIcePixel</i>	Quotient of wind probability and ice probability
<i>iceMapWeighted</i>	<i>bayesianIcemodel</i>	Calculate the ice a posteriori probability
<i>initIceMap</i>	<i>bayesianIcemodel</i>	Initialise ice map
<i>ExpandDateTime</i>	<i>iceMapWeighted,</i> <i>calcAave, updateIcePixel</i>	Converts a date/time to a real
<i>MAPLL</i>	<i>coordTransform</i>	Convert from lat/lon to polar stereographic coordinates
<i>MAPXY</i>	not used	Convert from polar stereographic to lat/lon coordinates
<i>wT</i>	<i>iceMapWeighted, calcAave</i>	Calculate the moving time average function
<i>logit</i>	not used	Calculate the logit of p: $\ln(p/(1-p))$
<i>inv_logit</i>	not used	Calculate the inverse of the logit of p: $1/(1+\exp(-p))$
<i>printIcePixel</i>	<i>updateIcePixel,</i> <i>scat2iceMap</i>	Print an ice pixel
<i>RW_IceMap</i>	<i>bayesianIcemodel</i>	Read or write an ice map from/to a binary restart file
<i>updateIcePixel</i>	<i>scat2iceMap</i>	Update an ice pixel with the contents of a BUFR message.
<i>printIceMap</i>	<i>bayesianIcemodel</i>	Print one or more ice map variables to graphical .ppm files
<i>printIce</i>	<i>printIceMap</i>	Print the ice parameter a and the ice classes to a .ppm file
<i>printppmcolor</i>	<i>printIceMap</i>	Print variable on ice map to .ppm file, using colour index
<i>printppmvar</i>	<i>printIceMap</i>	Print variable on ice map to .ppm file, mapped on gray scale

**Table 8.1** Routines of module *iceModelMod*.

### 8.3 Data structures

There are two important data structures defined in this module. The first contains all relevant data of one pixel on the ice map (IcePixel) as listed in table 8.2. The second one contains basically a two-dimensional array of ice pixels and represents an entire ice map (IceMapType) as listed in table 8.3. This could be either an ice map of the North Pole region or the South Pole region.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

<b>Attribute</b>	<b>Type</b>	<b>Description</b>
<i>subClass</i>	integer	Ice subclass
<i>aIceAves</i>	real	Average of the a-iceparameter
<i>aSd</i>	real	a-iceparameter standard deviation
<i>nIce</i>	integer	Number of measurement
<i>class0</i>	integer	Ice class
<i>Pice</i>	real	a-priori ice probability
<i>Qice</i>	real	a-priori odds on ice
<i>pIceGivenX</i>	real	a-posteriori ice probability
<i>qIceGivenX</i>	real	a-posteriori odss on ice
<i>sumWeightST</i>	real	Sum of weight factors
<i>Sst</i>	real	Sea surface temperature (K)
<i>aIce</i>	real(nhist)	a-iceparameter
<i>qXgivenIce</i>	real(nhist)	Odds on measurement (X) given ice
<i>timeIce</i>	DateTime(nhist)	Date/time of measurement
<i>class</i>	integer(nhist)	Ice class

**Table 8.2** Attributes for the *IcePixel* data type.

<b>Attribute</b>	<b>Type</b>	<b>Description</b>
<i>nPixels</i>	Integer	Number of pixels for the ice map
<i>nLines</i>	Integer	Number of lines for the ice map
<i>nHist</i>	Integer	Number of historical measurements that is stored (=2)
<i>pole</i>	Integer	Indicator for Northpole or Southpole
<i>timeIceNow</i>	DateTime	Expanded section 3 (data description)
<i>timeIcePrev</i>	DateTime	Expanded section 4 (data)
<i>xy</i>	IcePixel(nPixels, nLines)	Pointer to the ice map contents

**Table 8.3** Attributes for the *IceMapType* data type.

## 8.4 Parameters

There are several parameters involved that control the Bayesian statistics. They have sensible default values but most of them are made public so that their value can be overridden in the main program. The parameters are listed in table 8.4.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

<b>Parameter</b>	<b>Description</b>
<i>writeRestartFile</i>	Logical controlling the writing of a restartfile
<i>useWindInfo</i>	Logical controlling whether wind information is used
<i>useLandpoints</i>	Logical controlling whether an ice probability is calculated for land points
<i>use_only_windpnts</i>	Logical controlling whether only points with a valid wind solution are used
<i>use_sst</i>	Logical controlling wheter sea surface temperature is used in the ice model algorithm
<i>weightS</i>	Matrix given the weight factors for the spatial averaging
<i>decayTime</i>	Defines the decay time (hours) of the temporal weighting function
<i>cutoffTime</i>	Defined the cutoff time (hours) for the temporal weighting function
<i>pClimateIce</i>	This is the ice probability when no a-priori information is available
<i>qClimateIce</i>	This is the odds on ice related to <i>pClimateIce</i>
<i>Class_no_data</i>	Class: no data
<i>Class_sea</i>	Class: sea (wind)
<i>Class_ice</i>	Class: ice
<i>Class_sea_or_ice</i>	Class: sea or ice (indecisive)
<i>Class_no_sea_no_ice</i>	Class: unknown (outlier)
<i>SubClass_a2</i>	SubClass: sea
<i>SubClass_b1</i>	SubClass: sea or ice (weight < weightSTLimit)
<i>SubClass_b2</i>	SubClass: probably ice (SD(a) >= aSdLimit)
<i>SubClass_b3</i>	SubClass: ice
<i>SubClass_d</i>	SubClass: unknown (outlier)
<i>SubClass_no_data</i>	SubClass: no data
<i>nHist</i>	Number of historical measurement that are stored (=2)
<i>aSdLimit</i>	Upper limit for the standard deviation of the a-iceparameter
<i>pIceGivenXlimit</i>	Lower limit of p(ice X) for classifying a pixel as ice
<i>sumWeightSTLimit</i>	Lower limit for the total weight of all measurements involved in the temperal and spatial averaging
<i>dRefIce</i>	Upper limit for the distance to ice line
<i>dRefWind</i>	Upper limit for the distance to wind cone
<i>sstLowLimit</i>	Lower limit for sea surface temperature (K). Below this limit pixels are classified as ice
<i>sstHighLimit</i>	Upper limit for sea surface temperature (K). Above this limit pixels are classified as sea (wind)

**Table 8.4** Parameters in the Bayesian statistics.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

NWP SAF	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
---------	--	---

## References

- Dalet, R., 1991,  
*Atmospheric data analysis*. Cambridge University Press, Cambridge, USA.
- Dragosavac, M., 1994,  
*BUFR User Guide and Reference Manual*. ECMWF. (Available via the ECMWF website [www.ecmwf.int](http://www.ecmwf.int))
- Draper, D.W. and Long, D.G., 2002,  
An assessment of SeaWinds on QuikSCAT wind retrieval. *Journal of Geophysical Research*, **107**, C12, p. 3212 (2002JC001330).
- Freilich, M.H.,  
*SeaWinds algorithm theoretical basis document*. Report atbd-sws-01.
- Giering, R., 1997,  
*Tangent linear and Adjoint Model Compiler, Users manual*. Max-Planck- Institut fuer Meteorologie.
- Leidner, S.M., Hoffman, R.N., and Augenbaum, J., 2000,  
*SeaWinds Scatterometer Real Time BUFR Geophysical Data Product User's guide*. NOAA NESDIS. (Available from the KNMI site [www.knmi.nl/scatterometer/publications](http://www.knmi.nl/scatterometer/publications)).
- Liu, D.C., and Nocedal, J., 1989  
On the limited memory BFGS method for large scale optimization methods. *Mathematical Programming*, 45, pp. 503-528.
- Portabella, M., 2002,  
*Wind field retrieval from satellite radar systems*. PhD thesis, University of Barcelona. (Available via the KNMI site [www.knmi.nl/scatterometer/publications](http://www.knmi.nl/scatterometer/publications)).
- Portabella, M. and Stoffelen, A., 2001,  
Rain Detection and Quality Control of SeaWinds. *Journal of Atmospheric and Oceanic Technology*, **18**, pp. 1171-1183.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

- Portabella, M. and Stoffelen, A., 2002,  
A Comparison of KNMI Quality Control and JPL Rain Flag for SeaWinds. *Canadian Journal of Remote Sensing (special issue on Remote Sensing of Marine Winds)*, **28**, 3.
- Portabella, M. and Stoffelen, A., 2004,  
A probabilistic approach for SeaWinds Data Assimilation. *Quarterly Journal of the Royal Meteorological Society*, 130, pp. 1-26.
- SCAT group, 2008,  
*SDP Test Report, version 2.0*. Report NWPSAF-KN-TR-001, UKMO, UK. (Available via the NWPSAF web site, [www.nwpsaf.org](http://www.nwpsaf.org)).
- Stoffelen, A., de Haan, S., Quilfen, Y., and Schyberg, H., 2004,  
*ERS scatterometer ambiguity removal scheme comparison*. KNMI/ EUMETSAT Ocean and Sea Ice report. (Available from the EUMETSAT website, [www.eumetsat.int](http://www.eumetsat.int)).
- Stoffelen, A., and Verhoef, A., 2008  
*SeaWinds Product Manual*. report SAF/OSI/KNMI/TEC/MA/134. KNMI, The Netherlands.
- Stoffelen, A., de Vries, J., and Voorrips, A., 2000,  
*Towards the real-time use of QuikScat winds*. Beleidscommissie Remote Sensing, report nr. USP-2/00-26.
- Stoffelen, A.C.M., 1998,  
*Scatterometry*. PhD thesis, University of Utrecht, ISBN 90-393-1708-9. (Available via the KNMI site [www.knmi.nl/scatterometer/publications](http://www.knmi.nl/scatterometer/publications)).
- Talagrand, O., 1991,  
The use of adjoint equations in numerical modeling of the atmospheric circulation. In: *Automatic Differentiation of Algorithms: Theory, Implementation and Application*, A. Griewank and G. Corliss Eds. pp. 169-180, Philadelphia, Penn: SIAM.
- Verspeek, J., 2006,  
*Sea ice classification using Bayesian statistics*, OSI SAF report.
- Vogelzang, J., 2006,  
*On the quality of high resolution wind fields*. Report NWPSAF-KN-TR-002, UKMO, UK. (Available via the NWPSAF web site, [www.nwpsaf.org](http://www.nwpsaf.org)).
- Vogelzang, J., 2007a,  
*Two dimensional variational ambiguity removal (2DVAR)*. Report NWPSAF-KN-TR-004, UKMO, UK. (Available via the NWPSAF web site, [www.nwpsaf.org](http://www.nwpsaf.org)).
- Vogelzang, J., 2007b,  
*Validation of 2DVAR on SeaWinds data*. Report NWPSAF-KN-TR-005, UKMO, UK. (Available via the NWPSAF web site, [www.nwpsaf.org](http://www.nwpsaf.org)).

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

- Vogelzang, J., 2008,  
*SDP 2.0 validation*. Report NWPSAF-KN-TR-006, UKMO, UK.  
(Available via the NWPSAF web site, [www.nwpsaf.org](http://www.nwpsaf.org)).
- de Vries, J. and Stoffelen, A., 2000,  
*2D Variational Ambiguity Removal*. KNMI, Feb 2000. (Available from the EUMETSAT website, [www.eumetsat.int](http://www.eumetsat.int)).
- de Vries, J., Stoffelen, A., and Beysens, J., 2004,  
*Ambiguity Removal and Product Monitoring for SeaWinds*. KNMI. (Available from the EUMETSAT website, [www.eumetsat.int](http://www.eumetsat.int)).
- de Vries, J., et al., 2004,  
*QuikScat Data Processor User Manual*, KNMI. (Available from the KNMI website, [www.knmi.nl/scatterometer/publications](http://www.knmi.nl/scatterometer/publications)).
- Wentz, and Smith, D.K., 1999,  
A model function for the ocean normalized cross section at 14 GHz derived from NSCAT observations. *Journal of Geophysical Research*, **104**, C5, pp. 11499-11507.

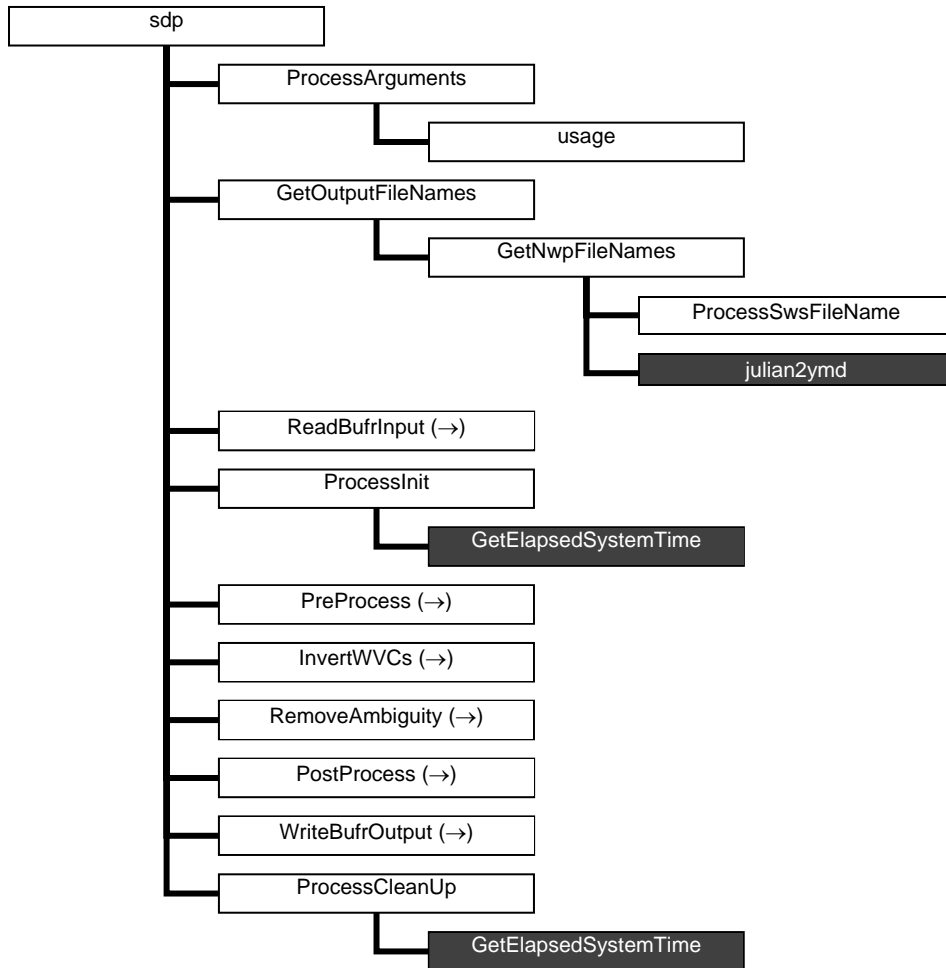
<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

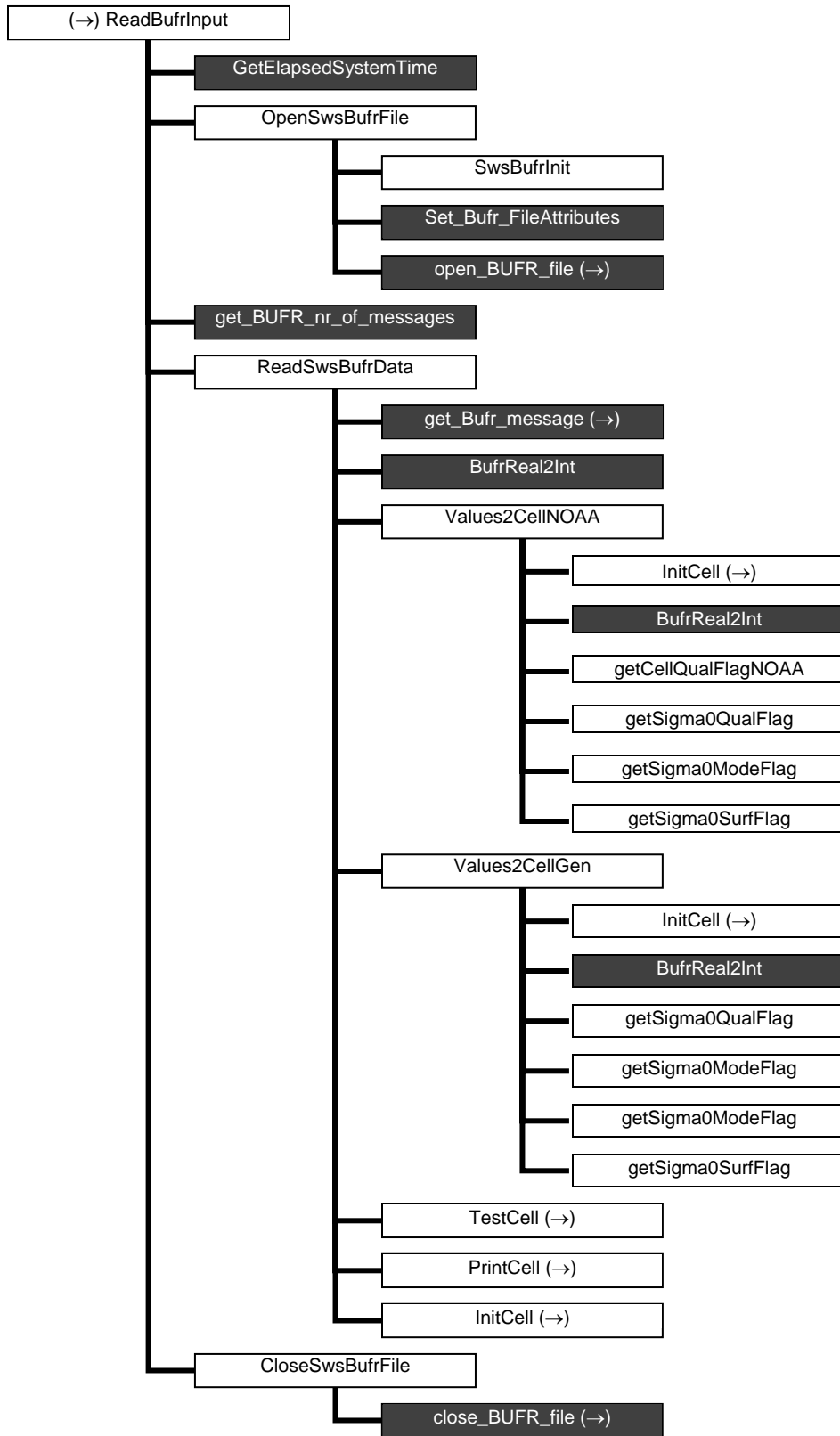
## Appendix A

# Calling tree for SDP

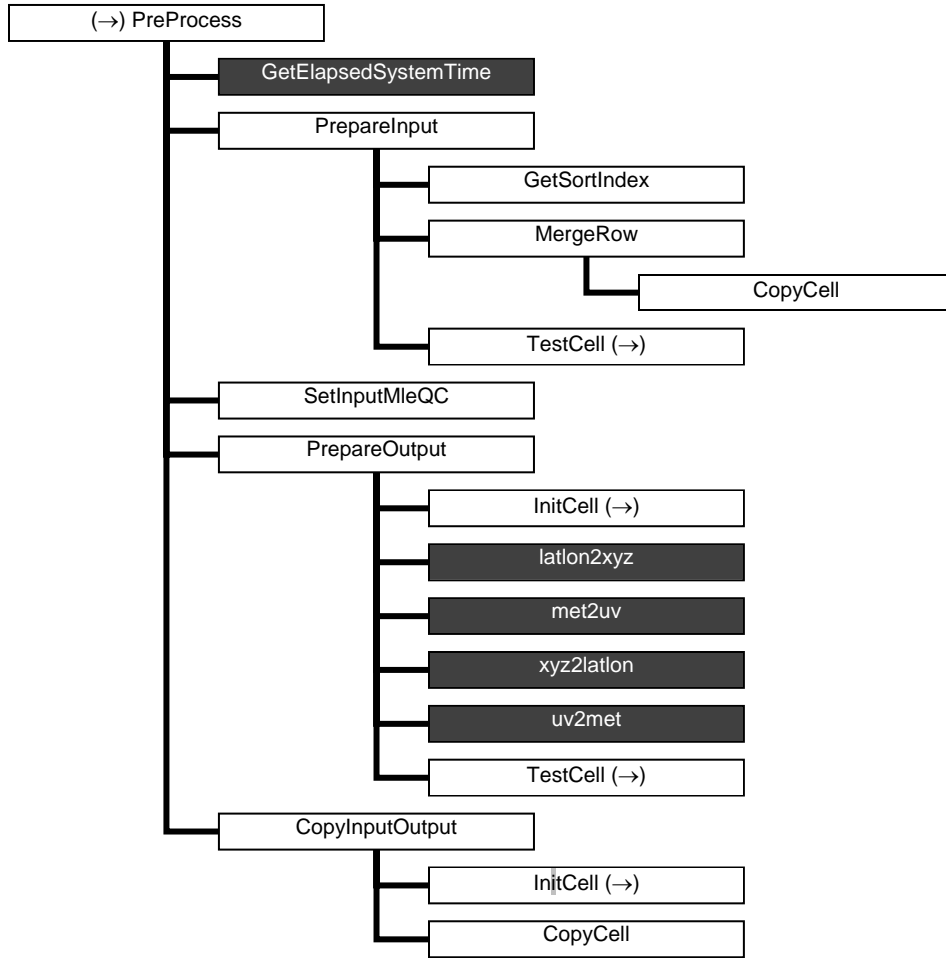
The figures in this appendix show the calling tree for the SDP program. Routines in white boxes are part of the SDP process layer and the SeaWinds Support layer. Routines in black boxes are part of genscat. An arrow (→) before a routine name indicates that this part of the calling tree is a continuation of a branch in a previous figure. The same arrow after a routine name indicated that this branch will be continued in a following figure.



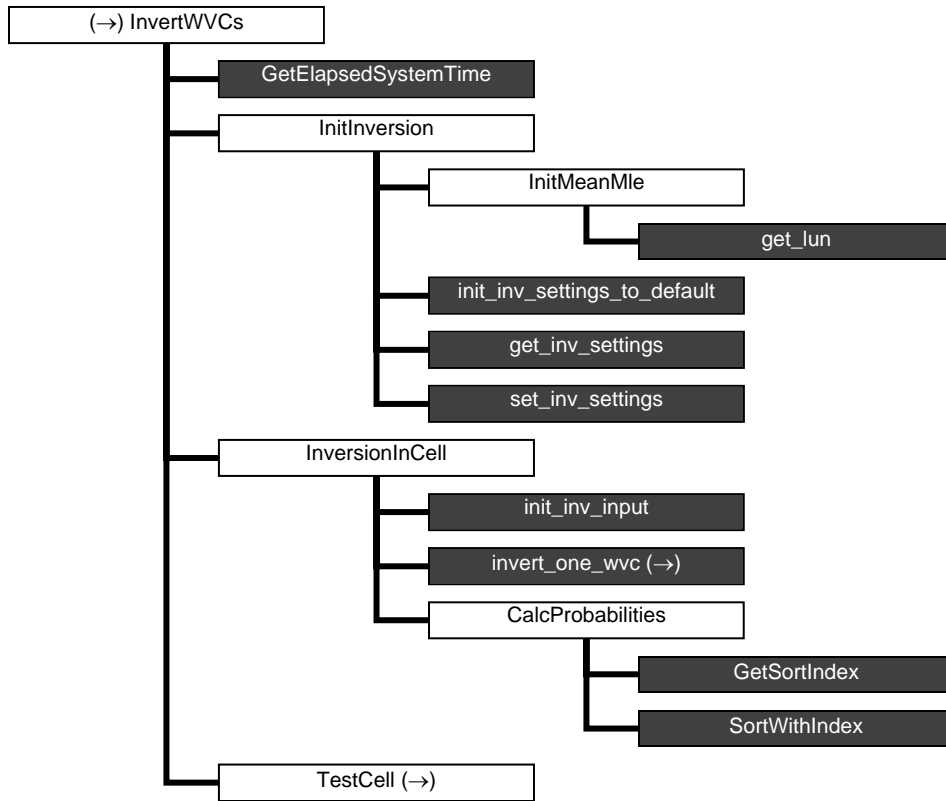
**Figure A.1** Calling tree for program *sdp* (top level). Light grey boxes are cut here and will be continued in one of the first level or second level calling trees in the next figures. Black boxes with light text indicate genscat routines.



**Figure A.2** Calling tree for routine *ReadBufInput* (first level).

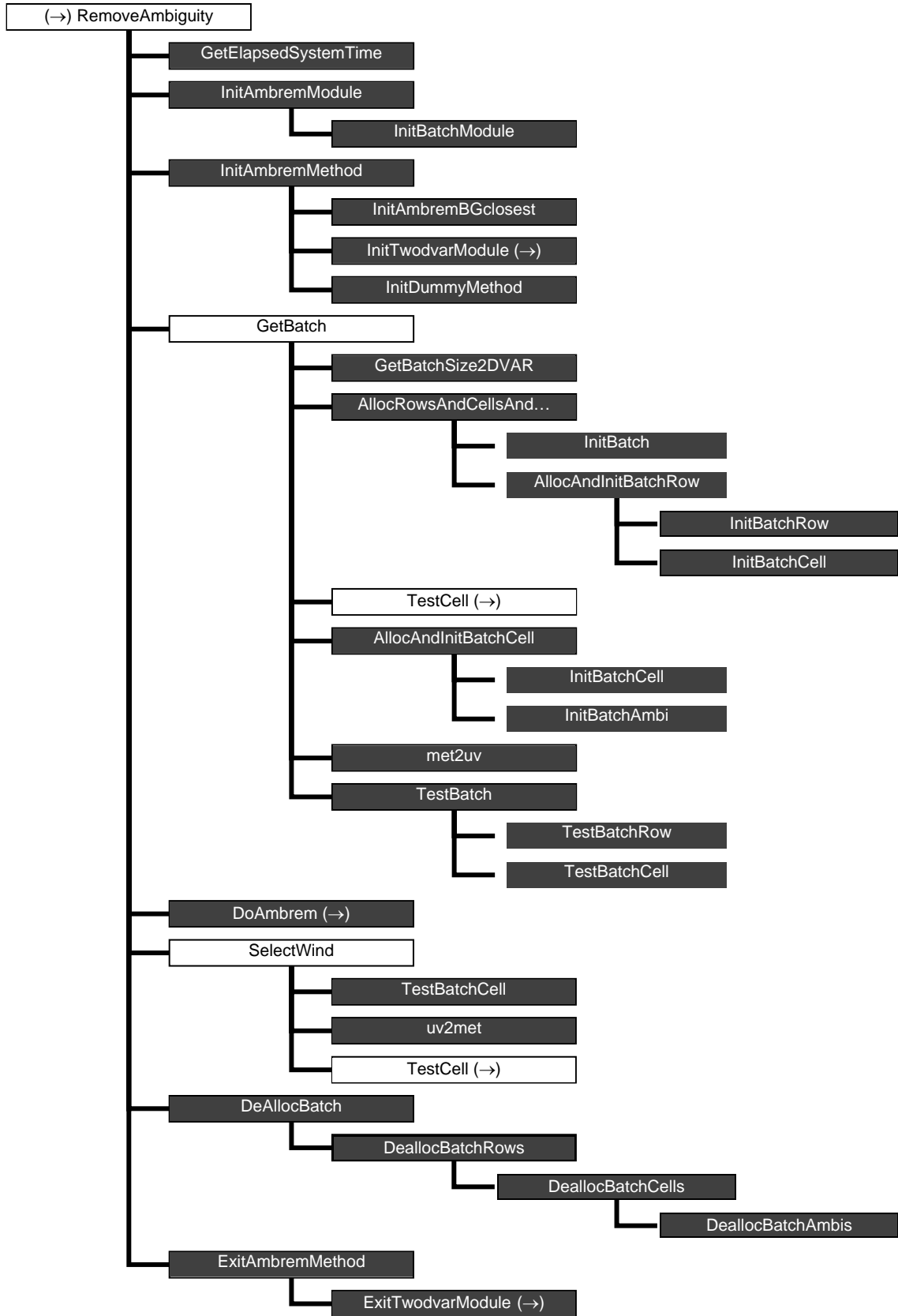


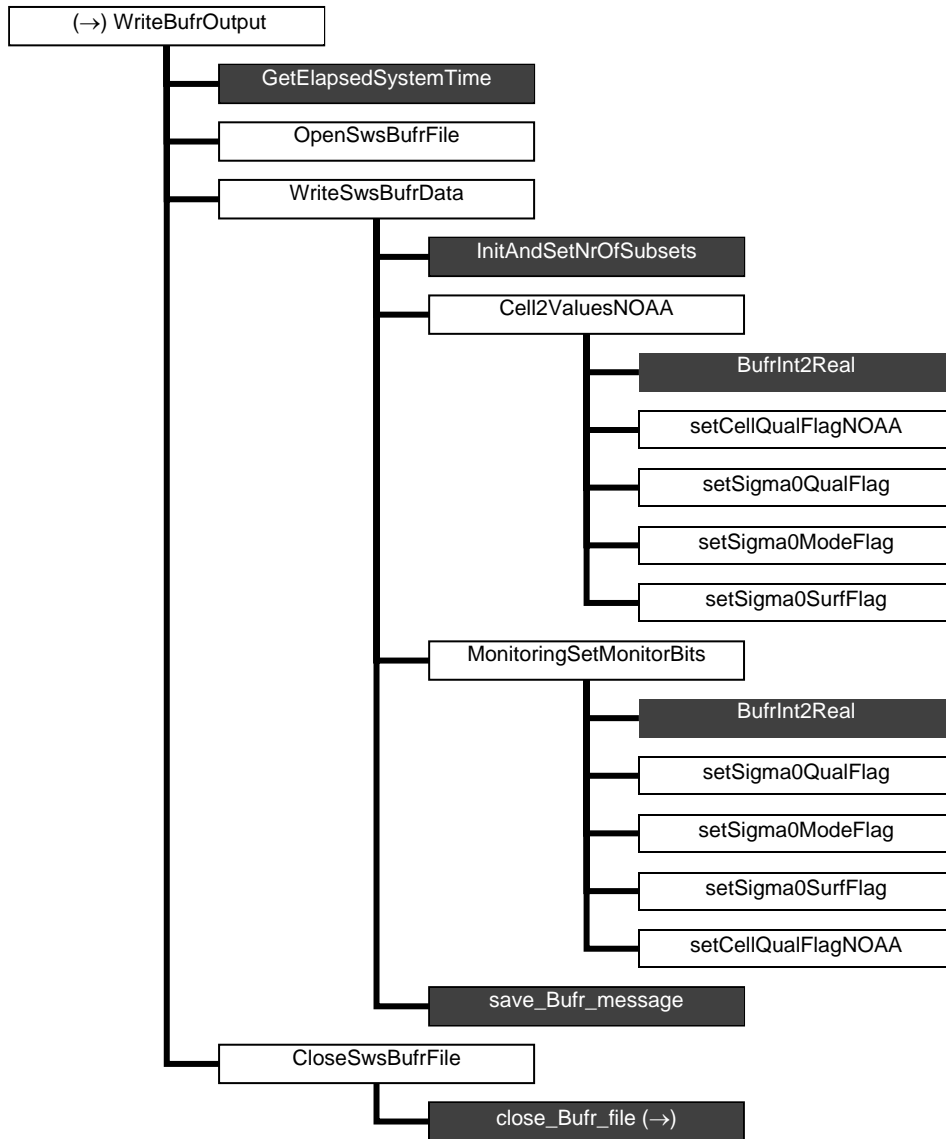
**Figure A.3** Calling tree for routine *PreProcess* (first level).



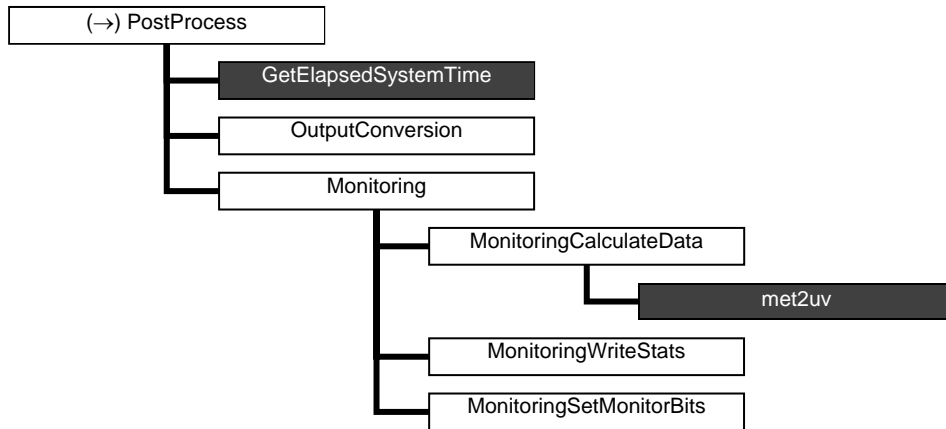
**Figure A.4** Calling tree for routine *InvertWVCs* (first level).

**Figure A.5 (next page)** Calling tree for routine *RemoveAmbiguity* (first level). The full name of the 12<sup>th</sup> routine is *AllocRowsAndCellsAndInitBatch*.

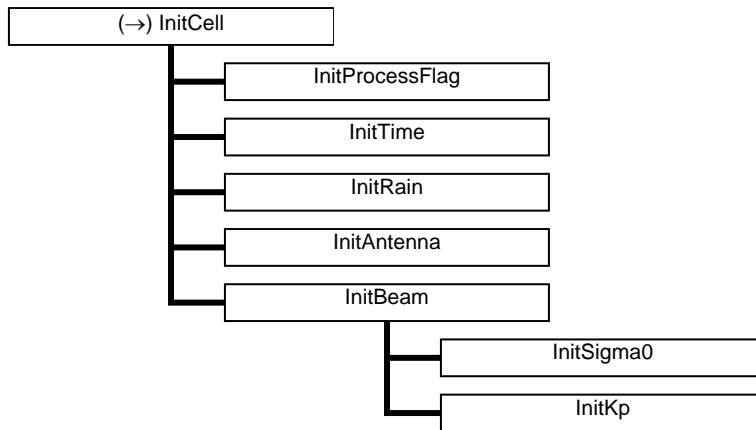




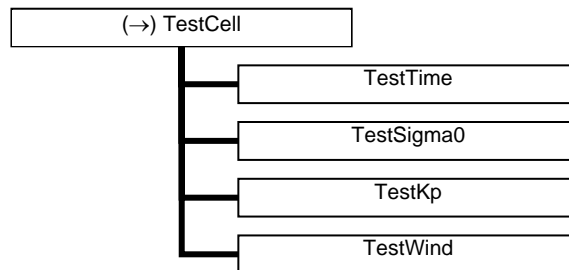
**Figure A.6** Calling tree for routine *WriteBufOutput* (first level).



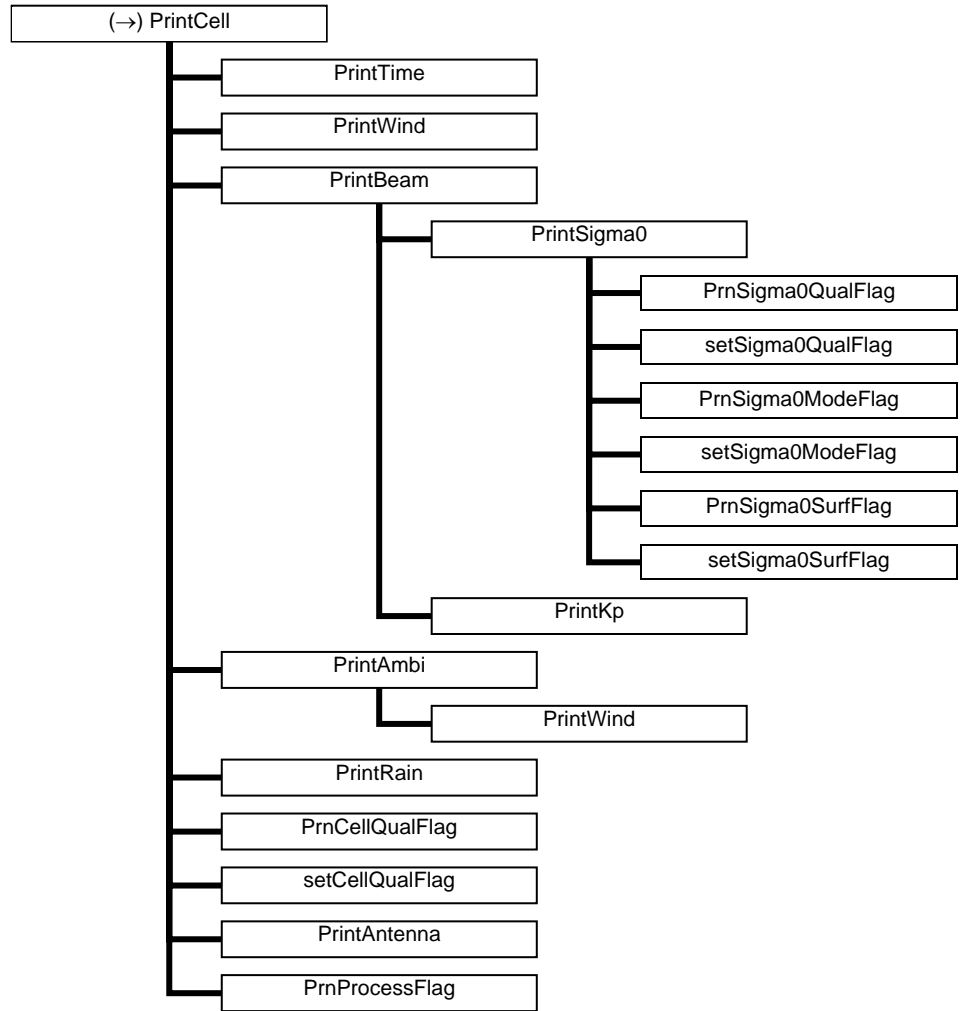
**Figure A.7** Calling tree for routine *PostProcess* (first level).



**Figure A.8** Calling tree for routine *InitCell* (second level).



**Figure A.9** Calling tree for routine *TestCell* (second level).



**Figure A.10** Calling tree for routine *PrintCell* (second level).

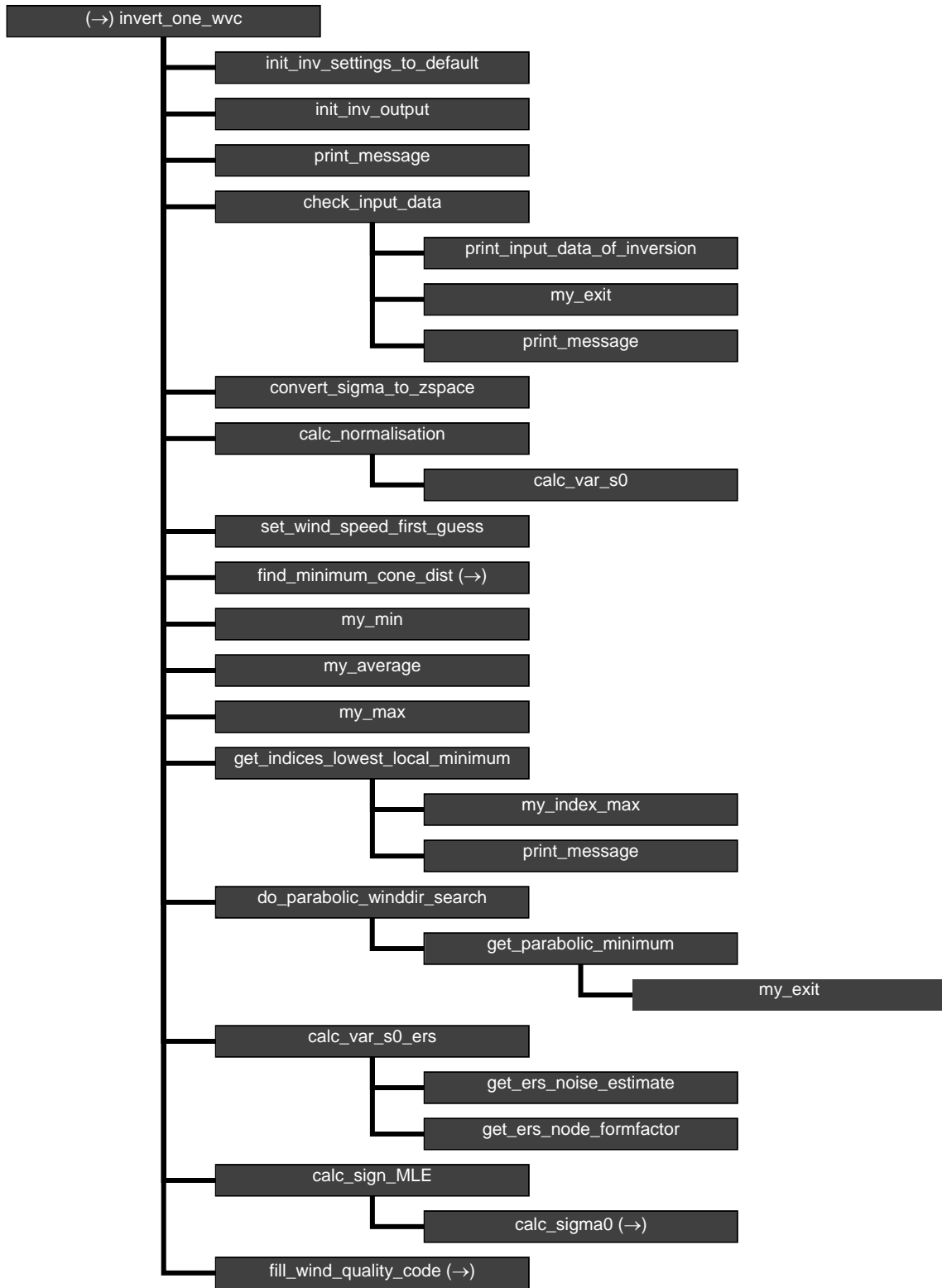
<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

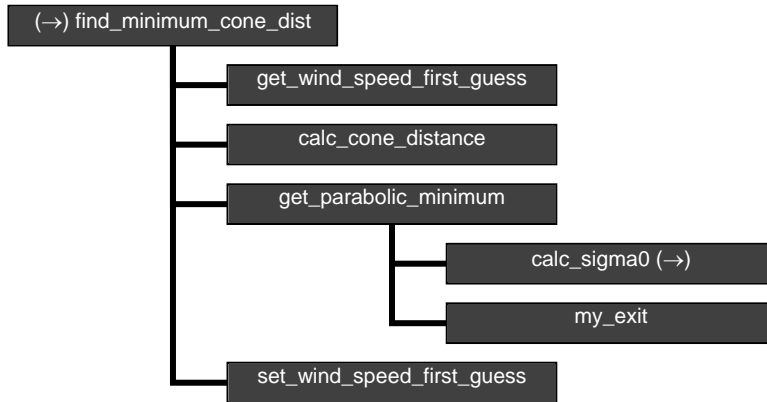
## Appendix B1

# Calling tree for inversion routines

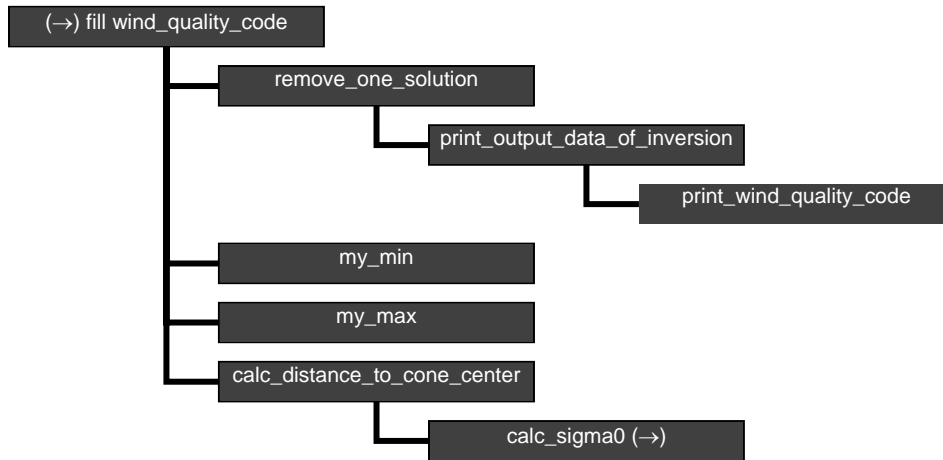
The figures in this appendix show the calling tree for the inversion routines in genscat. All routines are part of genscat, as indicated by the black boxes. An arrow (→) before a routine name indicates that this part of the calling tree is a continuation of a branch in a previous figure. The same arrow after a routine name indicated that this branch will be continued in a following figure.



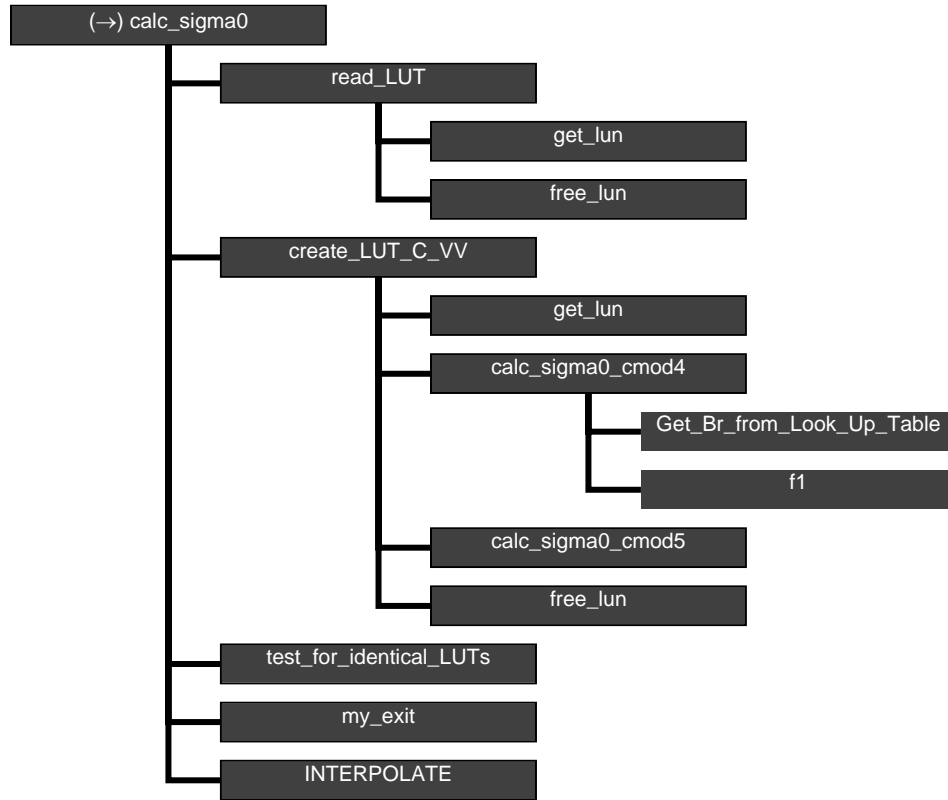
**Figure B1.1** Calling tree for inversion routine *invert\_one\_wvc*.



**Figure B1.2** Calling tree for inversion routine *find\_minimum\_cone\_dist*



**Figure B1.3** Calling tree for inversion routine *fill\_wind\_quality\_code*.



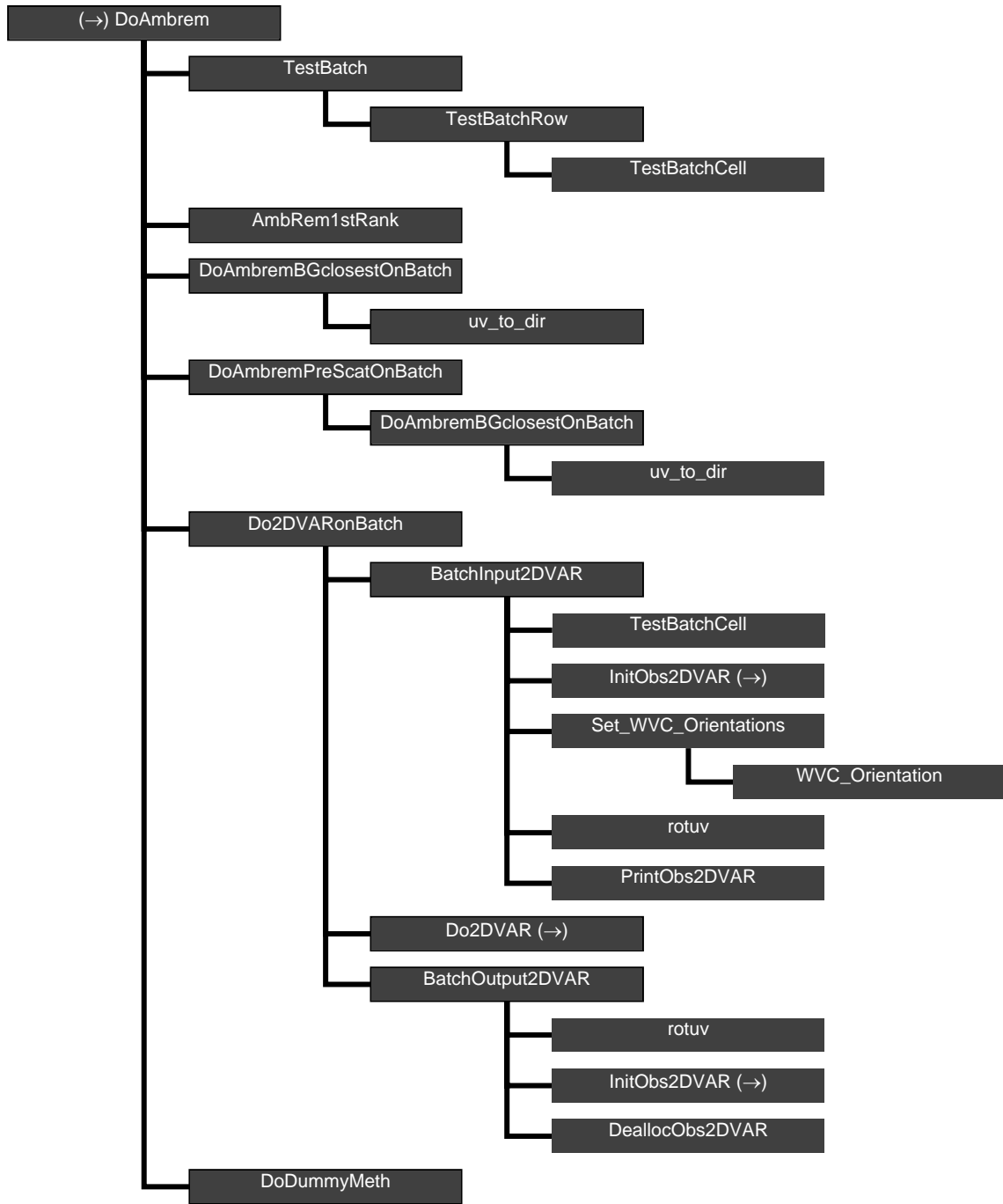
**Figure B1.4** Calling tree for inversion routine `calc_sigma0`. Routine `INTERPOLATE` is an interface that can have the values `interpolate1d`, `interpolate2d`, `interpolate2dv` or `interpolate3d`.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

## Appendix B2

# Calling tree for AR routines

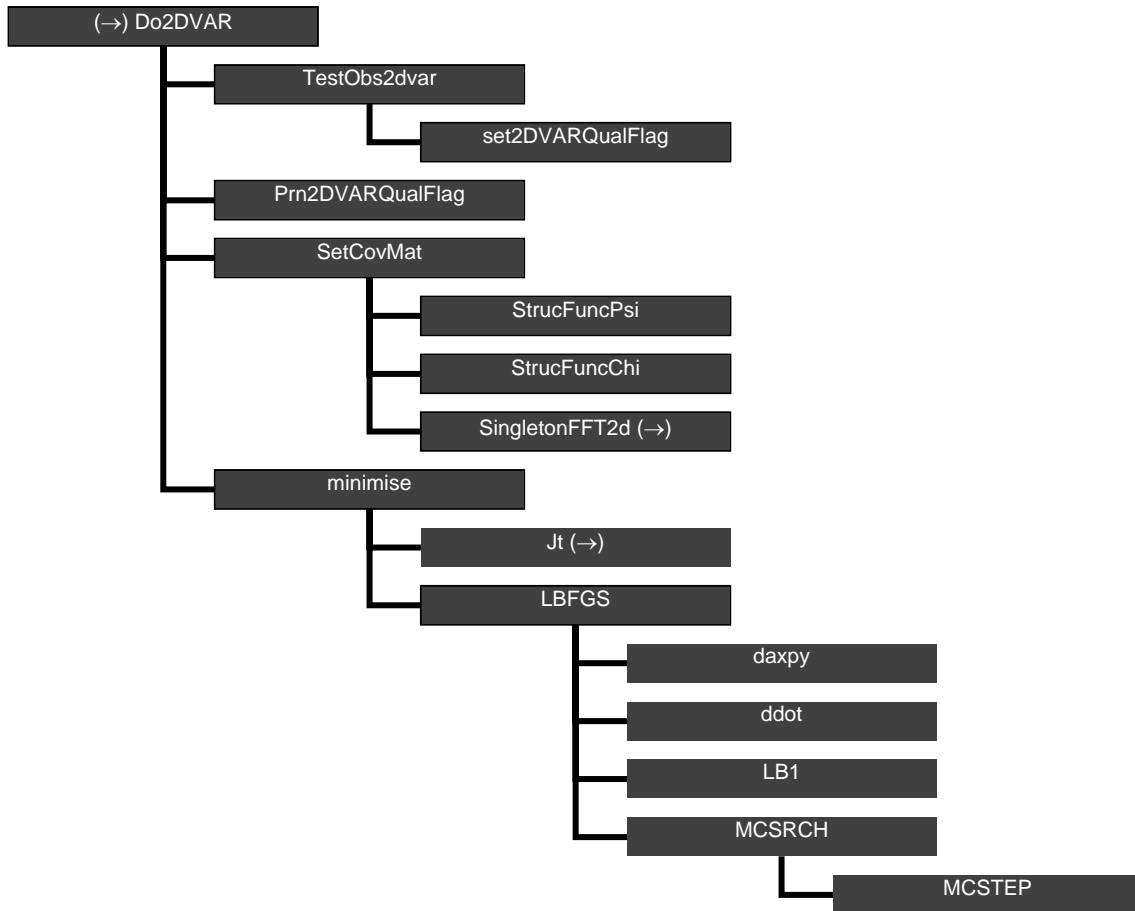
The figures in this appendix show the calling tree for the Ambiguity Removal routines in genscat. All routines are part of genscat, as indicated by the black boxes. An arrow (→) before a routine name indicates that this part of the calling tree is a continuation of a branch in a previous figure. The same arrow after a routine name indicated that this branch will be continued in a following figure.



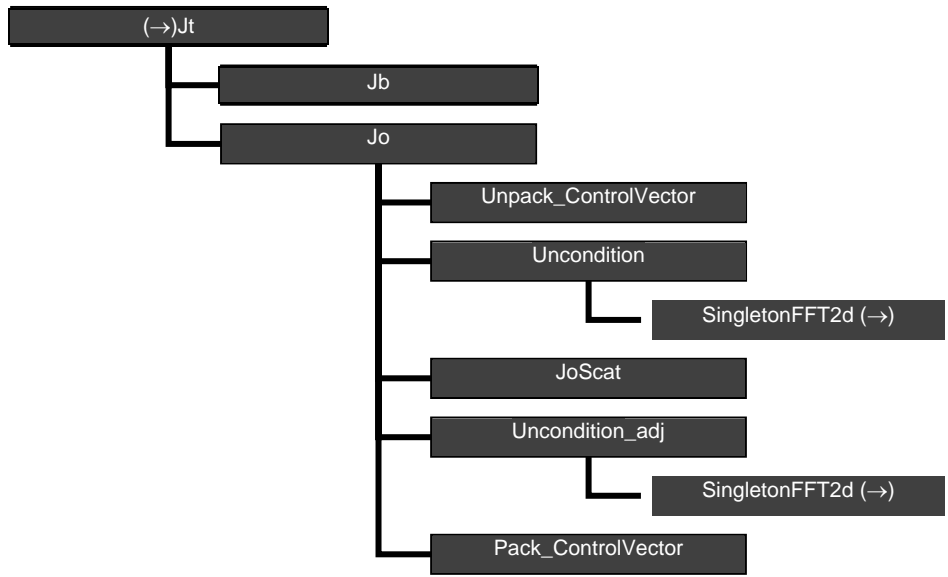
**Figure B2.1** Calling tree for AR routine *DoAmbrem*.



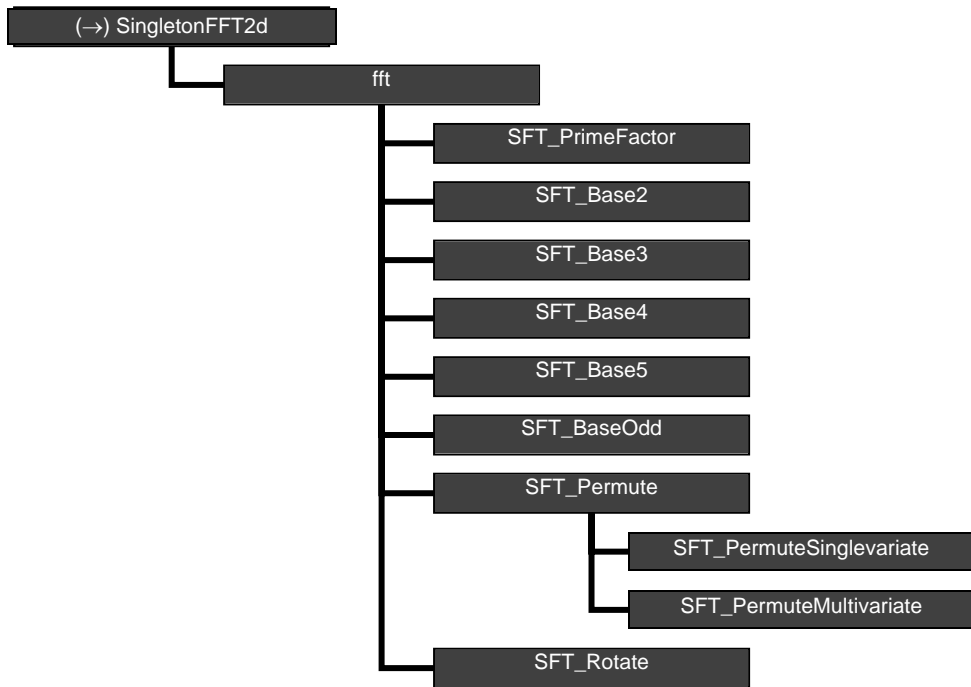
**Figure B2.2** Calling tree for AR routine *InitObs2DVAR*.



**Figure B2.3** Calling tree for AR routine *Do2DVAR*.

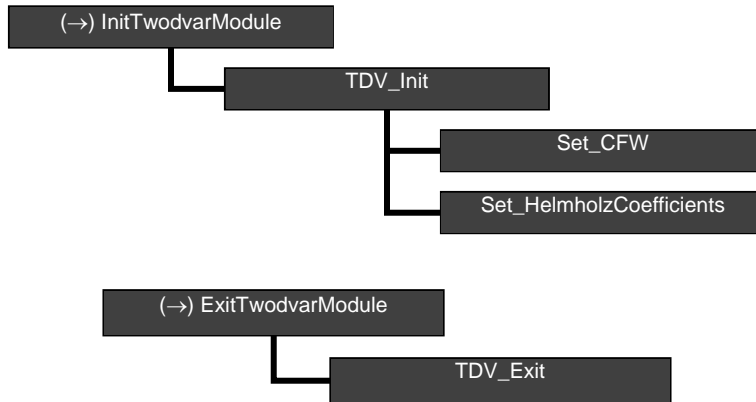


**Figure B2.4** Calling tree for AR routine *Jt* (calculation of cost function).



**Figure B2.5** Calling tree for AR routine *SingletonFFT2d*. for the forward and inverse FFT.

<p><b>NWP SAF</b></p>	<p><b>SDP User Manual and Reference Guide</b></p>	<p>Doc ID : NWPSAF-KN-UD-002  Version : 2.1  Date : 11-11-2010</p>
-----------------------	---	--



**Figure B.2.6** Calling trees for AR routines *InitTwodvarModule* and *ExitTwodvarModule*.

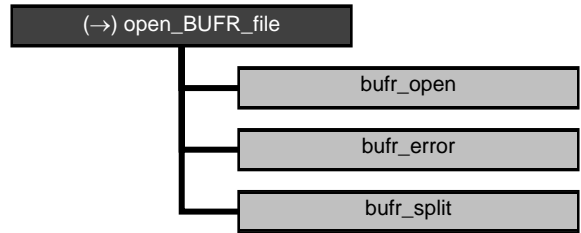
<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

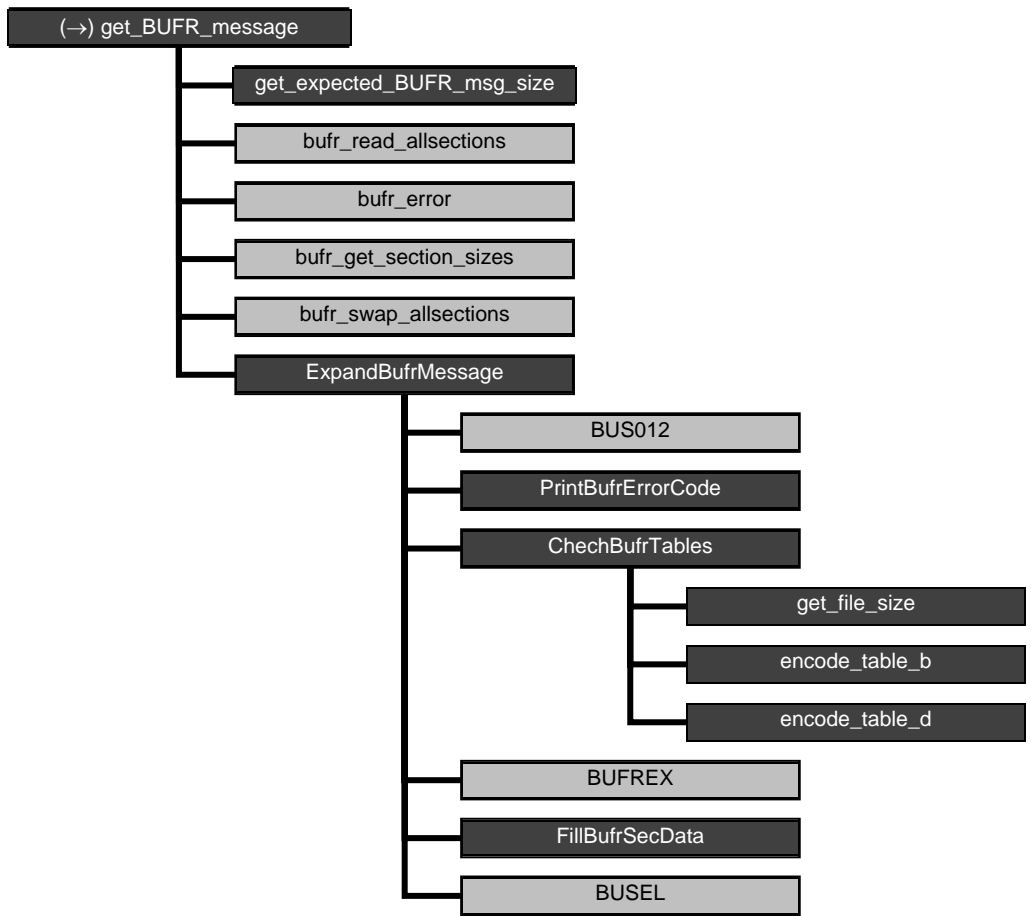
## Appendix B3

# Calling tree for BUFR routines

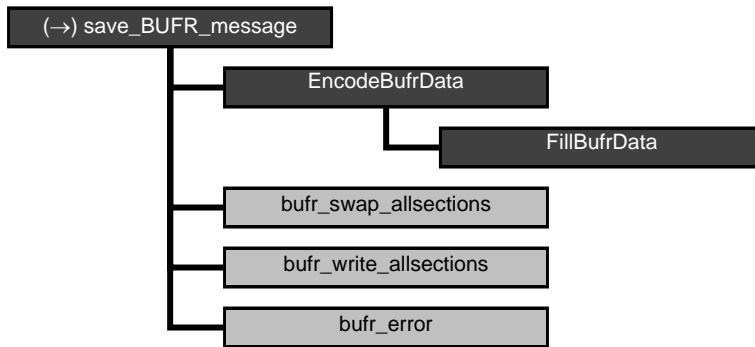
The figures in this appendix show the calling tree for the BUFR file handling routines in genscat. Routines in black boxes are part of genscat. Routines in grey boxes with names completely in capitals belong to the ECMWF BUFR library. Other routines in grey boxes belong to the BUFRIO library. An arrow (→) before a routine name indicates that this part of the calling tree is a continuation of a branch in a previous figure. The same arrow after a routine name indicated that this branch will be continued in a following figure.



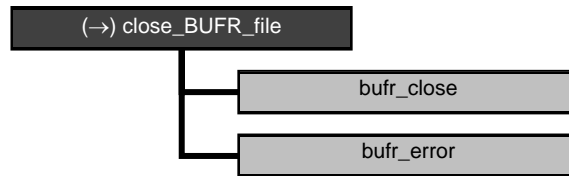
**Figure B3.1** Calling tree for BUFR file handling routine *open\_BUFR\_file*.



**Figure B3.2** Calling tree for BUFR handling routine *get\_BUFR\_message*.



**Figure B3.3** Calling tree for BUFR file handling routine *save\_BUFR\_file*.



**Figure B3.4** Calling tree for BUFR handling routine *close\_BUFR\_file*.

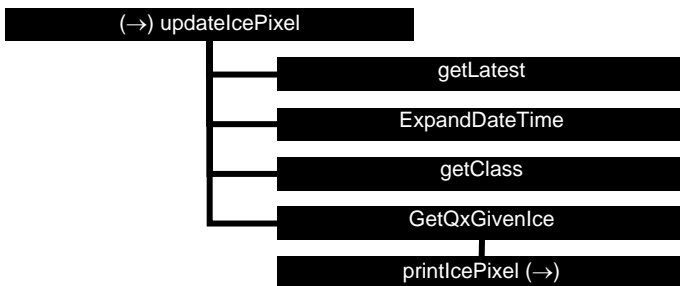
<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

NWP SAF	SDP User Manual and Reference Guide	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
---------	-------------------------------------	---

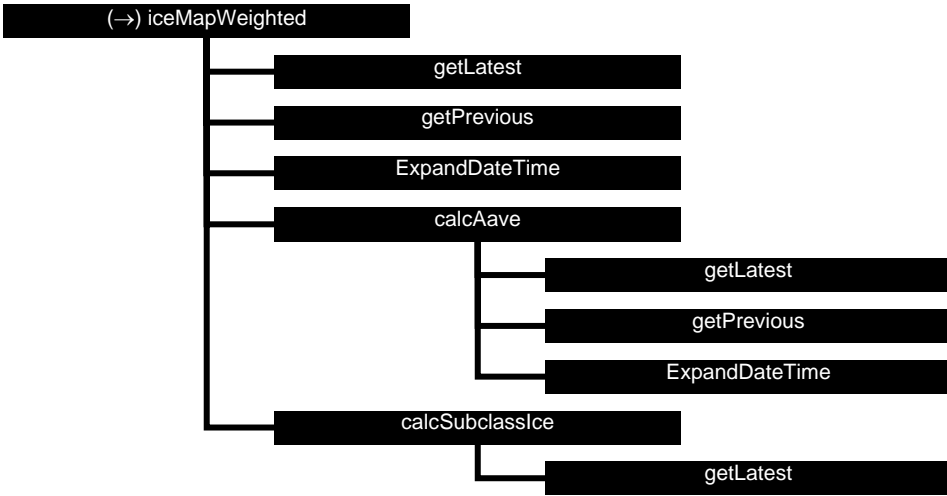
## Appendix B4

# Calling tree for ice model routines

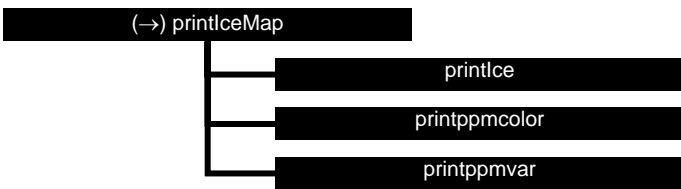
The figures in this appendix show the calling tree for the ice model routines in genscat. All routines are part of genscat, as indicated by the black boxes. An arrow (→) before a routine name indicates that this part of the calling tree is a continuation of a branch in a previous figure. The same arrow after a routine name indicates that this branch will be continued in a following figure.



**Figure B4.1** Calling tree for routine *updateIcePixel* (second level).



**Figure B4.2** Calling tree for routine *iceMapWeighted* (second level).



**Figure B4.3** Calling tree for routine *printIceMap* (second level).



**Figure B4.4** Calling tree for routine *printIcePixel* (second level).

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

## Appendix C1

# NOAA BUFR output file

<b>Number</b>	<b>Parameter</b>	<b>Unit</b>	<b>Descriptor</b>
001	Satellite Identifier	CODE TABLE	(01007)
002	Direction of Flight	DEGREE TRUE	(01012)
003	Satellite Instrument Identifier	CODE TABLE	(02048)
004	Wind Scatterometer GMF	CODE TABLE	(21119)
005	Software Identification	NUMERIC	(25060)
006	Cross Track Resolution	M	(02026)
007	Along Track Resolution	M	(02027)
008	Orbit Number	NUMERIC	(05040)
009	Year	YEAR	(04001)
010	Month	MONTH	(04002)
011	Day	DAY	(04003)
012	Hour	HOUR	(04004)
013	Minute	MINUTE	(04005)
014	Second	SECOND	(04006)
015	Latitude (Coarse Accuracy)	DEGREE	(05002)
016	Longitude (Coarse Accuracy)	DEGREE	(06002)
017	Time Difference Qualifier	CODE TABLE	(08025)
018	Time to Edge	S	(04001)
019	Along Track Row Number	NUMERIC	(05034)
020	Cross Track Cell Number	NUMERIC	(06034)
021	Seawinds Wind Vector Cell Quality Flag	FLAG TABLE	(21109)
022	Model Wind Direction At 10 M	DEGREE TRUE	(11081)
023	Model Wind Speed At 10 M	M/S	(11082)
024	Number of Vector Ambiguities	NUMERIC	(21101)
025	Index of Selected Wind Vector	NUMERIC	(21102)

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

*Continued on next page*

*Continued from previous page*

<b>Number</b>	<b>Parameter</b>	<b>Unit</b>	<b>Descriptor</b>
026	Total Number of Sigma0 Measurements	NUMERIC	(21103)
027	Seawinds Probability of Rain	NUMERIC	(21120)
028	Seawinds NOF Rain Index	NUMERIC	(21121)
029	Intensity Of Precipitation	KG/M**2/SEC	(13055)
030	Attenuation Correction On Sigma-0 (from Tb)	dB	(21122)
031	Wind Speed At 10 M	M/S	(11012)
032	Formal Uncertainty In Wind Speed	M/S	(11052)
033	Wind Direction At 10 M	DEGREE TRUE	(11011)
034	Formal Uncertainty In Wind Direction	DEGREE TRUE	(11053)
035	Likelihood Computed for Wind Solution	NUMERIC	(21104)
036	Wind Speed At 10 M	M/S	(11012)
037	Formal Uncertainty In Wind Speed	M/S	(11052)
038	Wind Direction At 10 M	DEGREE TRUE	(11011)
039	Formal Uncertainty In Wind Direction	DEGREE TRUE	(11053)
040	Likelihood Computed for Wind Solution	NUMERIC	(21104)
041	Wind Speed At 10 M	M/S	(11012)
042	Formal Uncertainty In Wind Speed	M/S	(11052)
043	Wind Direction At 10 M	DEGREE TRUE	(11011)
044	Formal Uncertainty In Wind Direction	DEGREE TRUE	(11053)
045	Likelihood Computed for Wind Solution	NUMERIC	(21104)
046	Wind Speed At 10 M	M/S	(11012)
047	Formal Uncertainty In Wind Speed	M/S	(11052)
048	Wind Direction At 10 M	DEGREE TRUE	(11011)
049	Formal Uncertainty In Wind Direction	DEGREE TRUE	(11053)
050	Likelihood Computed for Wind Solution	NUMERIC	(21104)
051	Antenna Polarisation	CODE TABLE	(02104)
052	Total Number w.r.t. accumulation or average	NUMERIC	(08022)
053	Brightness Temperature	K	(12063)
054	Standard Deviation Brightness Temperature	K	(12065)
055	Antenna Polarisation	CODE TABLE	(02104)
056	Total Number w.r.t. accumulation or average	NUMERIC	(08022)
057	Brightness Temperature	K	(12063)
058	Standard Deviation Brightness Temperature	K	(12065)
059	Number of Inner-Beam Sigma0 (fwd of sat.)	NUMERIC	(21110)
060	Latitude (Coarse Accuracy)	DEGREE	(05002)
061	Longitude (Coarse Accuracy)	DEGREE	(06002)
062	Attenuation Correction On Sigma-0	dB	(21118)
063	Radar Look (Azimuth) Angle	DEGREE	(02112)
064	Radar Incidence Angle	DEGREE	(02111)
065	Antenna Polarisation	CODE TABLE	(02104)
066	Normalized Radar Cross Section	NUMERIC	(21105)
067	Kp Variance Coefficient (Alpha)	NUMERIC	(21106)
068	Kp Variance Coefficient (Beta)	NUMERIC	(21107)
069	Kp Variance Coefficient (Gamma)	dB	(21114)
070	Seawinds Sigma-0 Quality Flag	FLAG TABLE	(21115)
071	Seawinds Sigma-0 Mode Flag	FLAG TABLE	(21116)
072	Seawinds Land/Ice Surface Flag	FLAG TABLE	(08018)
073	Sigma-0 Variance Quality Control	NUMERIC	(21117)

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

*Continued on next page*

*Continued from previous page*

<b>Number</b>	<b>Parameter</b>	<b>Unit</b>	<b>Descriptor</b>
074	Number of Outer-Beam Sigma0 (fwd of sat.)	NUMERIC	(21111)
075	Latitude (Coarse Accuracy)	DEGREE	(05002)
076	Longitude (Coarse Accuracy)	DEGREE	(06002)
077	Attenuation Correction On Sigma-0	dB	(21118)
078	Radar Look (Azimuth) Angle	DEGREE	(02112)
079	Radar Incidence Angle	DEGREE	(02111)
080	Antenna Polarisation	CODE TABLE	(02104)
081	Normalized Radar Cross Section	NUMERIC	(21105)
082	Kp Variance Coefficient (Alpha)	NUMERIC	(21106)
083	Kp Variance Coefficient (Beta)	NUMERIC	(21107)
084	Kp Variance Coefficient (Gamma)	dB	(21114)
085	Seawinds Sigma-0 Quality Flag	FLAG TABLE	(21115)
086	Seawinds Sigma-0 Mode Flag	FLAG TABLE	(21116)
087	Seawinds Land/Ice Surface Flag	FLAG TABLE	(08018)
088	Sigma-0 Variance Quality Control	NUMERIC	(21117)
089	Number of Inner-Beam Sigma0 (aft of sat.)	NUMERIC	(21112)
090	Latitude (Coarse Accuracy)	DEGREE	(05002)
091	Longitude (Coarse Accuracy)	DEGREE	(06002)
092	Attenuation Correction On Sigma-0	dB	(21118)
093	Radar Look (Azimuth) Angle	DEGREE	(02112)
094	Radar Incidence Angle	DEGREE	(02111)
095	Antenna Polarisation	CODE TABLE	(02104)
096	Normalized Radar Cross Section	NUMERIC	(21105)
097	Kp Variance Coefficient (Alpha)	NUMERIC	(21106)
098	Kp Variance Coefficient (Beta)	NUMERIC	(21107)
099	Kp Variance Coefficient (Gamma)	dB	(21114)
100	Seawinds Sigma-0 Quality Flag	FLAG TABLE	(21115)
101	Seawinds Sigma-0 Mode Flag	FLAG TABLE	(21116)
102	Seawinds Land/Ice Surface Flag	FLAG TABLE	(08018)
103	Sigma-0 Variance Quality Control	NUMERIC	(21117)
104	Number of Outer-Beam Sigma0 (aft of sat.)	NUMERIC	(21113)
105	Latitude (Coarse Accuracy)	DEGREE	(05002)
106	Longitude (Coarse Accuracy)	DEGREE	(06002)
107	Attenuation Correction On Sigma-0	dB	(21118)
108	Radar Look (Azimuth) Angle	DEGREE	(02112)
109	Radar Incidence Angle	DEGREE	(02111)
110	Antenna Polarisation	CODE TABLE	(02104)
111	Normalized Radar Cross Section	NUMERIC	(21105)
112	Kp Variance Coefficient (Alpha)	NUMERIC	(21106)
113	Kp Variance Coefficient (Beta)	NUMERIC	(21107)
114	Kp Variance Coefficient (Gamma)	dB	(21114)
115	Seawinds Sigma-0 Quality Flag	FLAG TABLE	(21115)
116	Seawinds Sigma-0 Mode Flag	FLAG TABLE	(21116)
117	Seawinds Land/Ice Surface Flag	FLAG TABLE	(08018)
118	Sigma-0 Variance Quality Control	NUMERIC	(21117)

**Table C1.1** List of data descriptors (NOAA BUFR format).

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

## Appendix C2

# KNMI BUFR output file

<b>Number</b>	<b>Parameter</b>	<b>Unit</b>	<b>Descriptor</b>
001	Satellite Identifier	CODE TABLE	(01007)
002	Direction of motion of moving observation platform	DEGREE TRUE	(01012)
003	Satellite sensor indicator	CODE TABLE	(02048)
004	Wind Scatterometer GMF	CODE TABLE	(21119)
005	Software Identification	NUMERIC	(25060)
006	Cross Track Resolution	M	(02026)
007	Along Track Resolution	M	(02027)
008	Orbit Number	NUMERIC	(05040)
009	Year	YEAR	(04001)
010	Month	MONTH	(04002)
011	Day	DAY	(04003)
012	Hour	HOUR	(04004)
013	Minute	MINUTE	(04005)
014	Second	SECOND	(04006)
015	Latitude (Coarse Accuracy)	DEGREE	(05002)
016	Longitude (Coarse Accuracy)	DEGREE	(06002)
017	Time Difference Qualifier	CODE TABLE	(08025)
018	Second	SECOND	(04006)
019	Along Track Row Number	NUMERIC	(05034)
020	Cross Track Cell Number	NUMERIC	(06034)

*Continued on next page*

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

*Continued from previous page*

021	Total Number of Sigma0 Measurements	NUMERIC	(21103)
022	Probability of Rain	NUMERIC	(21120)
023	Seawinds NOF Rain Index	NUMERIC	(21121)
024	Intensity Of Precipitation	KG/M**2/SEC	(13055)
025	Attenuation Correction On Sigma-0 (from Tb)	dB	(21122)
026	Antenna Polarisation	CODE TABLE	(02104)
027	Total Number w.r.t. accumulation or average	NUMERIC	(08022)
028	Brightness Temperature	K	(12063)
029	Standard Deviation Brightness Temperature	K	(12065)
030	Antenna Polarisation	CODE TABLE	(02104)
031	Total Number w.r.t. accumulation or average	NUMERIC	(08022)
032	Brightness Temperature	K	(12063)
033	Standard Deviation Brightness Temperature	K	(12065)
034	Number of Inner-Beam Sigma0 (forward of satellite)	NUMERIC	(21110)
035	Latitude (Coarse Accuracy)	DEGREE	(05002)
036	Longitude (Coarse Accuracy)	DEGREE	(06002)
037	Attenuation Correction On Sigma-0	dB	(21118)
038	Radar Look (Azimuth) Angle	DEGREE	(02112)
039	Radar Incidence Angle	DEGREE	(02111)
040	Antenna Polarisation	CODE TABLE	(02104)
041	SeaWinds Normalized Radar Cross Section	NUMERIC	(21105)
042	Kp Variance Coefficient (Alpha)	NUMERIC	(21106)
043	Kp Variance Coefficient (Beta)	NUMERIC	(21107)
044	Kp Variance Coefficient (Gamma)	dB	(21114)
045	Seawinds Sigma-0 Quality Flag	FLAG TABLE	(21115)
046	Seawinds Sigma-0 Mode Flag	FLAG TABLE	(21116)
047	Seawinds Land/Ice Surface Flag	FLAG TABLE	(08018)
048	Sigma-0 Variance Quality Control	NUMERIC	(21117)
049	Number of Outer-Beam Sigma0 (forward of satellite)	NUMERIC	(21113)
050	Latitude (Coarse Accuracy)	DEGREE	(05002)
051	Longitude (Coarse Accuracy)	DEGREE	(06002)
052	Attenuation Correction On Sigma-0	dB	(21118)
053	Radar Look Angle	DEGREE	(02112)
054	Radar Incidence Angle	DEGREE	(02111)
055	Antenna Polarisation	CODE TABLE	(02104)
056	Normalized Radar Cross Section	NUMERIC	(21105)
057	Kp Variance Coefficient (Alpha)	NUMERIC	(21106)
058	Kp Variance Coefficient (Beta)	NUMERIC	(21107)
059	Kp Variance Coefficient (Gamma)	dB	(21114)
060	Seawinds Sigma-0 Quality	FLAG TABLE	(21115)
061	Seawinds Sigma-0 Mode	FLAG TABLE	(21116)
062	Seawinds Land/Ice Surface Type	FLAG TABLE	(08018)
063	Sigma-0 Variance Quality Control	NUMERIC	(21117)
064	Number of Inner-Beam Sigma0 (aft of satellite)	NUMERIC	(21113)
065	Latitude (Coarse Accuracy)	DEGREE	(05002)
066	Longitude (Coarse Accuracy)	DEGREE	(06002)
067	Attenuation Correction On Sigma-0	dB	(21118)
068	Radar Look Angle	DEGREE	(02112)
069	Radar Incidence Angle	DEGREE	(02111)

*Continued on next page*

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

*Continued from previous page*

070	Antenna Polarisation	CODE TABLE	(02104)
071	Normalized Radar Cross Section	NUMERIC	(21105)
072	Kp Variance Coefficient (Alpha)	NUMERIC	(21106)
073	Kp Variance Coefficient (Beta)	NUMERIC	(21107)
074	Kp Variance Coefficient (Gamma)	dB	(21114)
075	Seawinds Sigma-0 Quality	FLAG TABLE	(21115)
076	Seawinds Sigma-0 Mode	FLAG TABLE	(21116)
077	Seawinds Land/Ice Surface Type	FLAG TABLE	(08018)
078	Sigma-0 Variance Quality Control	NUMERIC	(21117)
079	Number of Outer-Beam Sigma0 (aft of satellite)	NUMERIC	(21113)
080	Latitude (Coarse Accuracy)	DEGREE	(05002)
081	Longitude (Coarse Accuracy)	DEGREE	(06002)
082	Attenuation Correction On Sigma-0	dB	(21118)
083	Radar Look Angle	DEGREE	(02112)
084	Radar Incidence Angle	DEGREE	(02111)
085	Antenna Polarisation	CODE TABLE	(02104)
086	Normalized Radar Cross Section	NUMERIC	(21105)
087	Kp Variance Coefficient (Alpha)	NUMERIC	(21106)
088	Kp Variance Coefficient (Beta)	NUMERIC	(21107)
089	Kp Variance Coefficient (Gamma)	dB	(21114)
090	Seawinds Sigma-0 Quality	FLAG TABLE	(21115)
091	Seawinds Sigma-0 Mode	FLAG TABLE	(21116)
092	Seawinds Land/Ice Surface Type	FLAG TABLE	(08018)
093	Sigma-0 Variance Quality Control	NUMERIC	(21117)
094	Software identification	NUMERIC	(25060)
095	Generating application	NUMERIC	(01032)
096	Model wind speed at 10 m	M/S	(11082)
097	Model wind direction at 10 m	DEGREE TRUE	(11081)
098	Ice probability	NUMERIC	(20095)
099	Ice age (A-parameter)	dB	(20096)
100	Wind vector cell quality	FLAG TABLE	(21155)
101	Number of Vector Ambiguities	NUMERIC	(21101)
102	Index of Selected Wind vector	NUMERIC	(21102)
103	Delayed Description Replication Factor	NUMERIC	(31001)
104	Wind Speed at 10 m	M/S	(11012)
105	Wind Direction at 10 m	DEGREES TRUE	(11011)
106	Backscatter Distance	NUMERIC	(21226)
107	Likelihood Computed for Solution	NUMERIC	(21104)
etc.	etc.	etc.	etc.

**Table C2.1** List of data descriptors (KLNMI BUFR format). Numbers 100 to 103 (yellow background) form the first element of the generalized wind section. They may be repeated up to 144 times.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

## Appendix D

# ECMWF BUFR data routines

<b>Function</b>	<b>Description</b>
bbuprs0.F	Print BUFR section 0 (?)
bbuprs1.F	Print BUFR section 1 (?)
bbuprs2.F	Print BUFR section 2 (?)
bbuprs3.F	Print BUFR section 3 (?)
bbuprt.F	Print BUFR (?)
bbuprtbox.F	Print BUFR box (?)\\
buaug.F	Update augmented BUFR table B
bubox.F	??
bucomp.F	Pack number of subsets in a compressed form
bucrkey.F	Extract elements needed for RDB key definition(update)
bucrkey.F	Extract elements needed for RDB key definition
buedd.F	Expand section 3 of BUFR message
buens0.F	Pack section 0 of BUFRsage
buens1.F	Pack section 1 of BUFR message
buens2.F	Pack section 2 of BUFR message
buens3.F	Pack section 3 of BUFR message
buens4.F	Pack preliminary items/data of sect.4 of BUFR message
buens5.F	Pack sect.5 of BUFR message
buepmrk.F	Process marker operator, replace with table B descriptor
buepmrkc.F	Process marker operator, replace with table B descriptor
buepwt.F	Updates working tables (name, unit, scale, ref, data width)
buepwtc.F	Updates working tables (name, unit, scale, ref, data width)
buerr.F	Print error code
buetab.F	Load BUFR table B, D and C according to BUFR code
buetd.F	Expand sect.3 of BUFR message

*Continued on next page*

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

*Continued from previous page*

<b>Function</b>	<b>Description</b>
buetdr.F	Solve BUFR table D reference
buevar.F	Initialize constants and variables
buexs0.F	Expand section 0 of BUFR message
buexs1.F	Expand section 1 of BUFR message
buexs2.F	Expand section 2 of BUFR message
buexs3.F	Expand section 3 of BUFR message
buexs3p.F	Expand section 3 of BUFR message (preliminary items)
buexs4.F	Expand section 4 of BUFR message
buexs5.F	Expand section 5 of BUFR message
bufren.F	Encode BUFR message
bufrex.F	Decode BUFR message into fully expanded form
bugbts.F	Load BUFR table B, D and C according to BUFR code
bugetbm.F	Create bit map to resolve marker operators
buivar.F	Initialize constants and variables
bunexs.F	Sets word/bit pointers at the start of next BUFR sect
bunpck.F	Unpack bit string
bunpks.F	Unpack bit string of KSIZE bits
buoctn.F	Calculate number of octets from bit position
buoper.F	Process BUFR operator
buoperc.F	Process BUFR operator
bupck.F	Pack value *KS* in *KSI* bits
bupkey.F	Pack local ECMWF information (rdb key)
bupks.F	Pack bit string of KSIZE bits
bupmrk.F	Process marker operator, replace with table B descriptor
buprco.F	Process BUFR operator
buprq.F	Sets variable KPMISS,KPRUS into common block
buprs0.F	Print section 0 of BUFR message
buprs1.F	Print section 1 of BUFR message
buprs2.F	Print section 2 of BUFR message (expanded RDB key)
buprs3.F	Print section 3 of BUFR message
buprt.F	Print expanded BUFR message
buprtbox.F	Print boxed expanded BUFR message
burep.F	Resolve data descriptor replication problem
burepc.F	Resolve data descriptor replication problem
burqc.F	Create parameters needed for partial expansion of BUFR
burquc.F	Create parameters needed for partial expansion of BUFR
bus012.F	Expands sections 0, 1, and 2 of BUFR message
busel.F	Returns Data Descriptors as in Section 3 of BUFR
buset.F	Set flags in common block (?)
busrp.F	Resolve data descriptor replication problem
busrq.F	Set BUFR table B references for partial expansion
bustr.F	Solve BUFR table D reference
buuatb.F	Update augmented BUFR table B
buukey.F	Expands local ECMWF information from sect.2
buunp.F	Unpack bit string of KSIZE bits
buunps.F	Unpack bit string of KSIZE bits
buupwt.F	Updates working tables (name, unit, scale, ref, data width)
buxdes.F	Expand data descriptors to show user's template

*Continued on next page*

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

*Continued from previous page*

<b>Function</b>	<b>Description</b>
fmmh.F	Find max/min latitude/longitude
setlalo.F	Return indices for latitude and longitude

**Table D.1** List of ECMWF BUFR routines.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---

## Appendix E

# Acronyms

<b>Name</b>	<b>Description</b>
AR	Ambiguity Removal
BUFR	Binary Universal Form for the Representation of data
C-band	Radar wavelength at about 5 cm
ECMWF	European Center for Medium-range Weather Forecasts
EUMETSAT	European Organization for the Exploitation of Meteorological Satellites
GMF	Geophysical model function
HIRLAM	High resolution Local Area Model
KNMI	Koninklijk Nederlands Meteorologisch Instituut (Royal Netherlands Meteorological Institute)
Ku-band	Radar wavelength at about 2 cm
L1b	Level 1b product
LUT	Look up table
MLE	Maximum Likelihood Estimator
MSS	Multiple Solution Scheme
NRCS	Normalized radar cross-section ( $\sigma_0$ )
NWP	Numerical Weather Prediction
QC	Quality Control
RFSCAT	Rotating fan beam scatterometer
RMS	Root mean square
SAF	Satellite Application Facility
WVC	Wind vector cell, also called node or cell

**Table E.1** List of acronyms.

<b>NWP SAF</b>	<b>SDP User Manual and Reference Guide</b>	Doc ID : NWPSAF-KN-UD-002 Version : 2.1 Date : 11-11-2010
----------------	--	---