NWP SAF

Satellite Application Facility for Numerical Weather Prediction

Document NWPSAF-KN-TV-005
Version 1.0.14
October 2008

AWDP Test Report

Anton Verhoef, Jur Vogelzang, Jeroen Verspeek and Ad Stoffelen

KNMI, De Bilt, the Netherlands

AWDP Test Report

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.

Change record			
Version	**Date**	**Author / changed by**	**Remarks**
1.0.14	Oct 2008	Anton Verhoef	First version

Contents

Preface

Preface to version 1.0.14

This document is the test report for the ASCAT Wind Data Processor (AWDP) program. It is set up according to the guidelines of the NWP SAF; see the NWP SAF Development Procedures for Software Deliverables. Parts of the AWDP developments are in fact genscat developments. The tests for genscat modules are also included in this document.

Anton Verhoef, October 2008

Chapter 1

Introduction

1.1 Aims and scope

The ASCAT Wind Data Processor (AWDP) is a software package written in Fortran 90 for handling data from the Advanced Scatterometer (ASCAT) and European Remote Sensing satellite (ERS) scatterometer instruments. Details of these instruments can be found on several web sites and in several other documents, see e.g. [*Portabella*, 2002; *Stoffelen*, 1998] and information on the ESA and EUMETSAT web sites.

AWDP generates surface winds based on ASCAT and ERS data. It allows performing the ambiguity removal with the Two-dimensional Variational Ambiguity Removal (2DVAR) method and it supports the Multiple Solution Scheme (MSS). The output of AWDP consists of wind vectors which represent surface winds within the ground swath of the scatterometer. Input of AWDP is Normalized Radar Cross Section (NRCS, σ_0) data. These data may be real-time. The input files of AWDP are in BUFR or Product Format Specification (PFS, native MetOp) format. BUFR input may be provided using the BUFR templates for ERS or ASCAT; output is always written using the ASCAT BUFR template. Moreover, AWDP needs Numerical Weather Prediction (NWP) model winds as a first guess for the Ambiguity Removal step. These data need to be provided in GRIB.

1.2 Development of AWDP

AWDP is developed within the Numerical Weather Prediction Satellite Application Facility (NWP SAF) and Ocean and Sea Ice Satellite Application Facility (OSI SAF) programs 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. Special attention has been paid on robustness and readability. AWDP may be run on every modern Unix or Linux machine. In principle, AWDP can also run on a Windows machine if a Unix emulator like Cygwin is installed. Details on the AWDP program can be found in [*Verhoef et. al.*, 2007].

The AWDP code is based on code developed for the ERS scatterometers, NSCAT scatterometer, and the simulations of the ESA Rotating Fan beam Scatterometer (RFSCAT). The common code of these projects is consolidated in the generic scatterometer (genscat) layer. In each development step, following from the heritage, the output of the code developments has been compared to the output of the original code. Moreover, KNMI runs an experimental suite in the framework of the

OSI SAF, where AWDP, in different modes, is routinely compared to the operational OSI SAF suite at http://www.knmi.nl/scatterometer/. This comparison is both field-wise and statistical.

Several developers work with and on AWDP at KNMI, and even more with the genscat layer for SeaWinds, ERS or ASCAT projects. Improvements to the code follow the test procedures as described in this document. The effort of maintaining a unique reference code greatly improves robustness and reliability of the code, i.e., sharing results and enjoying the benefits.

1.3 Testing AWDP

This section describes the Test Plan of the AWDP deliverable. Tests have been carried out in all stages of the development of AWDP. The inversion module is not tested for the AWDP program, because such a test has already been made for the QuikSCAT Data Processor (QDP) development. AWDP contains several methods for Ambiguity Removal within module *ambrem* and its sub modules. Only modules needed for the KNMI 2DVAR scheme for Ambiguity Removal are tested within this project.

Compilation is done on several platforms (operating systems) and with different Fortran 90 compilers. The integration and validation tests were done on both a LINUX work station and a SUN machine.

Chapter 2 contains the tests for a number of individual modules. In general, modules are tested with the associated test programs that are located in the folder containing the module under consideration. The output of the test programs is always the standard output (screen) which may be redirected to any test log file or to some output files which are stored in the associated folders. Chapter 3 describes the AWDP integration test. A test folder containing some sample data is provided with AWDP and some of the resulting wind fields from these data are shown. Chapter 4 discusses the validation tests. AWDP has been compared with the Prescat wind processing software using ERS data, and the results of both programs have been compared for identical output. Chapter 5 describes the portability tests. It contains an overview of platform/operating systems and Fortran compilers for which AWDP is supported. Finally, Chapter 6 is devoted to testing the user documentation.

1.4 Test folders

The Test folder of the AWDP Program is located in subdirectory `awdp/tests`. This subdirectory contains several input files for AWDP that are discussed in more detail in Chapter 3. The scripts for executing these tests are located in directory `awdp/execs`. It is recommended to use these scripts (or a modified version) also for normal AWDP operation, as the environment variables needed by AWDP are set in these scripts.

As stated before, most test programs are located in the same directory as the module to be tested. See Chapter 2 for detailed information.

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. `awdp/src)`, files (e.g. `awdp.F90`), and commands (e.g. `awdp -f input`) are printed in Courier. Software systems in general are addressed using the normal font (e.g. AWDP, genscat).

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

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

Chapter 2

Module tests

In this chapter the various tests to individual modules within AWDP are presented. The tests are listed alphabetically in the module name. Table 2.1 gives an overview of the modules tested, their location and the name of the associated test programs.

Module tests have been included in AWDP if the following conditions were satisfied:

1. The test does not require additional software.
2. The output of the test program is self explanatory enough to judge the outcome of the test.

Module name	Location	Test program
awdp_data	`awdp/src`	*awdp_data_test*
BFGSMod	`genscat/support/BFGS`	*Test_BFGS*
BufrMod	`genscat/support/bufr`	*test_modules*
convert	`genscat/support/convert`	*test_convert*
CostFunction	`genscat/ambrem/twodvar`	*Test_SOS*
StrucFunc	`genscat/ambrem/twodvar`	*Test_SOS*
DateTimeMod	`genscat/support/datetime`	*TestDateTimeMod*
ErrorHandler	`genscat/support/ErrorHandler`	*TestErrorHandler*
gribio_module	`genscat/support/grib`	*test_read_GRIB1*, *test_read_GRIB2*, *test_read_GRIB3*
LunManager	`genscat/support/file`	*TestLunManager*
numerics	`genscat/support/num`	*test_numerics*
SingletonFFT	`genscat/support/singletonfft`	*TestSingleton*
SortMod	`genscat/support/sort`	*SortModTest*

Table 2.1 Overview of module tests.

2.1 Module *awdp_data*

Module *awdp_data.F90* in directory `awdp/src` contains the data structure definitions for the AWDP program. It is tested by program *awdp_data_test*, the output of which is listed in table 2.2.

```
 === CELL INFO: Level 1b data ===
Originating centre:    2147483647
Origin sub-centre:     2147483647
Software id:           2147483647
Satellite id:          2147483647
Satellite instr:       2147483647
Dir of motion:         1.7000000E+38
Year:****** month:**** day:****
Hour:******   min:**** sec:****
Latitude:              1.7000000E+38
Longitude:             1.7000000E+38
Pixel size on hor:     1.7000000E+38
Orbit number:          2147483647
Node number:           2147483647
Height of atmosph:     1.7000000E+38
Loss per unit len:     1.7000000E+38
Beam collocation:   missing
Beam colloc value:     2147483647

 BEAM nr. 1
Identifier:            2147483647
Incidence angle:       1.7000000E+38
Azimuth angle:         1.7000000E+38
Sigma0:                1.7000000E+38
Noise value:           1.7000000E+38
Kp estimate qual:   missing
Kp est qual value:     2147483647
Sigma0 usability:      2147483647
Synt data quantity:    1.7000000E+38
Synt data quality:     1.7000000E+38
Orbit quality:         1.7000000E+38
Solar refl contam:     1.7000000E+38
Telemetry pres/qua:    1.7000000E+38
Extrapol ref pres:     1.7000000E+38
Land fraction          1.7000000E+38

 BEAM nr. 2
Identifier:            2147483647
Incidence angle:       1.7000000E+38
Azimuth angle:         1.7000000E+38
Sigma0:                1.7000000E+38
Noise value:           1.7000000E+38
Kp estimate qual:   missing
Kp est qual value:     2147483647
Sigma0 usability:      2147483647
Synt data quantity:    1.7000000E+38
Synt data quality:     1.7000000E+38
Orbit quality:         1.7000000E+38
Solar refl contam:     1.7000000E+38
Telemetry pres/qua:    1.7000000E+38
Extrapol ref pres:     1.7000000E+38
Land fraction          1.7000000E+38

 BEAM nr. 3
Identifier:            2147483647
Incidence angle:       1.7000000E+38
Azimuth angle:         1.7000000E+38
Sigma0:                1.7000000E+38
Noise value:           1.7000000E+38
Kp estimate qual:   missing
Kp est qual value:     2147483647
Sigma0 usability:      2147483647
Synt data quantity:    1.7000000E+38
Synt data quality:     1.7000000E+38
Orbit quality:         1.7000000E+38
Solar refl contam:     1.7000000E+38
Telemetry pres/qua:    1.7000000E+38
Extrapol ref pres:     1.7000000E+38
Land fraction          1.7000000E+38

 === CELL INFO: Level 2 soil moisture data ===
Software id:           2147483647
Database id:           2147483647
Surf soil moisture:    1.7000000E+38
```

```
Error in surface sm:   1.7000000E+38
Sigma0 at 40 deg:      1.7000000E+38
Error in sigma0_40:    1.7000000E+38
Slope at 40 deg:       1.7000000E+38
Error in slope_40:     1.7000000E+38
Sm sensitivity:        1.7000000E+38
Dry backscatter:       1.7000000E+38
Wet backscatter:       1.7000000E+38
Mean surface sm:       1.7000000E+38
Rain fall detection:   1.7000000E+38
Sm correction flag:    2147483647
Sm processing flag:    2147483647
Sm quality:            1.7000000E+38
Snow cover fraction:   1.7000000E+38
Frozen land frac:      1.7000000E+38
Inundat/wetland fr:    1.7000000E+38
Topogr complexity:     1.7000000E+38

 === CELL INFO: Level 2 wind data ===
Software id:           2147483647
Generating appl:       2147483647
 MODEL WIND
Wind speed:            1.7000000E+38
Wind direction:        1.7000000E+38
Ice probability:       1.7000000E+38
Ice age A-param:       1.7000000E+38
 WVC QUALITY
WVC quality: missing
WVC quality value:     2147483647
Num of ambiguities:             0
Index of sel wind:     2147483647
Skill for AR:          1.7000000E+38

 === CELL INFO: Process information ===
Process flag: POOR satellite id
Process flag: POOR satellite instrument
Process flag: POOR satellite dir of motion
Process flag: POOR time
Process flag: POOR lat/lon
Process flag: POOR pixel size on horizontal
Process flag: POOR node number
Process flag: POOR beam            1
Process flag: POOR beam            2
Process flag: POOR beam            3
Process flag: POOR model wind
Process flag: POOR ambiguity
Process flag: POOR selection
Level 1/2 of input:    2147483647

 === CELL INFO: Level 1b data ===
Originating centre:            99
Origin sub-centre:              5
Software id:                    1
Satellite id:                   3
Satellite instr:              100
Dir of motion:          180.0000
Year:  2005 month:  10 day:   6
Hour:    10   min:   3 sec:  33
Latitude:               50.00000
Longitude:              12.00000
Pixel size on hor:      25000.00
Orbit number:               12345
Node number:                    4
Height of atmosph:      5000.000
Loss per unit len:     9.9999997E-06
Beam collocation:     T
Beam colloc value:              1

 BEAM nr. 1
Identifier:                     1
Incidence angle:        40.00000
Azimuth angle:          45.00000
Sigma0:                -10.00000
Noise value:            5.000000
```

```
  Kp est. qual. flag:   F
  Kp est qual value:              0
  Sigma0 usability:               0
  Synt data quantity:    0.1000000
  Synt data quality:     0.2000000
  Orbit quality:         0.3000000
  Solar refl contam:     0.4000000
  Telemetry pres/qua:    0.5000000
  Extrapol ref pres:     0.6000000
  Land fraction          0.7000000

   BEAM nr. 2
  Identifier:                     2
  Incidence angle:        30.00000
  Azimuth angle:          90.00000
  Sigma0:                -7.000000
  Noise value:            2.000000
  Kp est. qual. flag:   T
  Kp est qual value:              1
  Sigma0 usability:               1
  Synt data quantity:    0.2000000
  Synt data quality:     0.3000000
  Orbit quality:         0.4000000
  Solar refl contam:     0.5000000
  Telemetry pres/qua:    0.6000000
  Extrapol ref pres:     0.7000000
  Land fraction          0.8000000

   BEAM nr. 3
  Identifier:                     2
  Incidence angle:        40.00000
  Azimuth angle:          135.0000
  Sigma0:                -10.00000
  Noise value:            5.000000
  Kp est. qual. flag:   F
  Kp est qual value:              0
  Sigma0 usability:               0
  Synt data quantity:    0.3000000
  Synt data quality:     0.4000000
  Orbit quality:         0.5000000
  Solar refl contam:     0.6000000
  Telemetry pres/qua:    0.7000000
  Extrapol ref pres:     0.8000000
  Land fraction          0.9000000

   === CELL INFO: Level 2 soil moisture data ===
  Software id:                    2
  Database id:                   15
  Surf soil moisture:     12.30000
  Error in surface sm:    4.500000
  Sigma0 at 40 deg:      -12.34000
  Error in sigma0_40:     4.560000
  Slope at 40 deg:      -0.2300000
  Error in slope_40:    -0.1200000
  Sm sensitivity:         6.780000
  Dry backscatter:        7.890000
  Wet backscatter:        8.900000
  Mean surface sm:        45.60000
  Rain fall detection:    78.90000
  Sm correction flag:             1
  Sm processing flag:             5
  Sm quality:             34.50000
  Snow cover fraction:    1.230000
  Frozen land frac:       2.340000
  Inundat/wetland fr:     3.450000
  Topogr complexity:      5.670000

   === CELL INFO: Level 2 wind data ===
  Software id:                    3
  Generating appl:               99
   MODEL WIND
  Wind speed:             5.000000
  Wind direction:         234.0000
  Ice probability:       0.1000000
```

```
Ice age A-param:        10.00000
 WVC QUALITY
WVC quality: KNMI Quality Control fails
WVC quality: some portion of WVC over land
WVC quality value:         163840
Num of ambiguities:             2
Index of sel wind:              1
Skill for AR:          1.7000000E+38

 AMBIGUITY nr.   1
Wind speed:             6.000000
Wind direction:         222.0000
Probability:           0.9000000
Cone distance:         0.1000000

 AMBIGUITY nr.   2
Wind speed:             5.000000
Wind direction:         15.00000
Probability:           0.1000000
Cone distance:         0.2000000

 === CELL INFO: Process information ===
Process flag: OK satellite id
Process flag: OK satellite instrument
Process flag: OK satellite dir of motion
Process flag: OK time
Process flag: OK lat/lon
Process flag: OK pixel size on horizontal
Process flag: OK node number
Process flag: OK beam            1
Process flag: OK beam            2
Process flag: OK beam            3
Process flag: OK model wind
Process flag: OK ambiguity
Process flag: OK selection
Level 1/2 of input:             2
```

Table 2.2 Output of program *awdp_data_test*.

2.2 Module *BFGSMod*

Directory `genscat/support/BFGS` contains program `Test_BFGS`. This program tests the minimization routine LBFGS and its associated routines in module *BFGSMod*. The routines in *BFGSMod* are slightly modified versions of the freeware routine `LBFGS` and its subroutines. `LBFGS` was written by J. Nocedal, see [*Liu and Nocedal* 1989].

Program `Test_BFGS` finds the minimum of the function

$$f(x) = \sum_{i=1}^{100000} (x-i)^4$$

The minimum is the point (1, 2, …, 100000). The search starts at the origin. The typical output is shown in table 2.3.

```
Program Test_BFGS testing routine LBFGS

Routine LBFGS completed succesfully
  Number of iterations                 :      87
  Dimension of problem                 :  100000
  Number of corrections in BFGS update :       5
  Cost function at start               :  0.20001D+25
  Cost function at end                 :  0.30995D-16
  Precision required                   :  0.10D-19
  Norm of final X                      :  0.18258D+08
  Norm of final G                      :  0.97625D-13
  Minimum and Maximum error in solution :  0.000003  0.000005
  Time needed                          :  2.744  seconds
Program Test_BFGS completed succesfully.
```

Table 2.3 Output of program Test_BFGS.

2.3 Module *BufrMod*

Directory `genscat/support/bufr` contains program *test_modules*. This program is compiled and called automatically by the genscat make system, since it is needed to translate the ASCII BUFR tables to binary form. It will also read in a small BUFR test file, decode it, encode the data again and write them to an output BUFR file. Hence, the program can be used to check the BUFR library. Table 2.4 shows the output generated by *test_modules*. The program can be invoked by calling the shell script *run_test_modules*, which sets the environment variable `$BUFR_TABLES` and calls *test_modules*.

```
nr of BUFR messages in this file is:     1
                   ECMWF

      BUFR DECODING SOFTWARE VERSION -  7.2
            1 APRIL  2007.



 Your path for bufr tables is :
 ./bufr_tables/
BUFR TABLES TO BE LOADED  B0000000000210000001.TXT,D0000000000210000001.TXT
 tbd%nelements =            44
 pos_lat =            25
 pos_lon =            26
 latitude  range:    -3.630000        1.260000
 longitude range:     2.850000        7.690000
                   ECMWF

      BUFR ENCODING SOFTWARE VERSION -  7.2
            1 April  2007.



 Your path for bufr tables is :
 ./bufr_tables/
BUFR TABLES TO BE LOADED  B0000000000210000001.TXT,D0000000000210000001.TXT
```

Table 2.4 Output of program *test_modules*.

2.4 Module *convert*

Directory `genscat/support/convert` contains module *convert.F90*, a number of routines for the conversion of meteorological and geographical quantities. Its associated test program is *test_convert*, and part of its output is listed in table 2.5. Program *test_convert* produces quite a lot of output.

It starts with checking some conversions between different wind vector representations and transformations between different geographical coordinate systems, followed by a check of the transformation from orbit angles (p,a,rot(z)) to three-dimensional position (x,y,z).

Only the results for p = 0° and 90° are (partly) shown in table 2.5; those for p = 10°, 45°, and 70° are omitted. Program *test_convert* ends with some trigonometric calculations on a sphere.

```
===========================================
 u =     5.000000      v =    -7.000000
 uv_to_speed, uv_to_dir ====> sp =     8.602325      dir =     324.4623
 ===========================================
 sp =     8.602325      dir =     324.4623
 speeddir_to_u, speeddir_to_v ====> u =     5.000002      v =    -6.999999
 ===========================================
 met2uv: sp =     10.00000      dir =     135.0000
 met2uv: ====> u =    -7.071068      v =     7.071068
 uv2met: u =    -7.071068      v =     7.071068
 uv2met: ====> sp =     10.00000      dir =     135.0000
 ===========================================
 lat,lon =     55.00000        5.000000
 latlon2xyz: ====> x,y,z =    0.5713938       4.9990479E-02   0.8191521
 x,y,z =    0.5713938       4.9990479E-02   0.8191521
 xyz2latlon: ====>lat,lon =     55.00000        5.000000
 ===========================================
          p          a      rot_z          x          y          z         a1     rot_z1         a2     rot_z2
    0.00000  -90.00000    0.00000    0.00000    0.00000   -1.00000  -90.00000  106.16298  270.00000    0.00000
    0.00000  -90.00000   15.00000    0.00000    0.00000   -1.00000  -90.00000  105.59795  270.00000    9.72975
    0.00000  -90.00000   30.00000    0.00000    0.00000   -1.00000  -90.00000  103.95005  270.00000   27.91061
    0.00000  -90.00000   45.00000    0.00000    0.00000   -1.00000  -90.00000  101.35209  270.00000   43.81981
    0.00000  -90.00000   60.00000    0.00000    0.00000   -1.00000  -90.00000   98.00070  270.00000   59.32336
    0.00000  -10.00000    0.00000    0.98481    0.00000   -0.17365  -10.00000    0.00000  190.00000  180.00000
    0.00000  -10.00000   15.00000    0.95125    0.25489   -0.17365  -10.00000   15.00000  190.00000 -164.99998
    0.00000  -10.00000   30.00000    0.85287    0.49240   -0.17365  -10.00000   30.00000  190.00000 -149.99998
…
   90.00000   45.00000   30.00000    0.25882    0.96593    0.00000   74.99999    0.00000  105.00000    0.00000
   90.00000   45.00000   45.00000    0.00000    1.00000    0.00000   90.00000    0.00000   90.00000    0.00000
   90.00000   45.00000   60.00000   -0.25882    0.96593    0.00000   74.99999    0.00000  105.00000    0.00000
   90.00000   90.00000    0.00000    0.00000    1.00000    0.00000   90.00000    0.00000   90.00000    0.00000
   90.00000   90.00000   15.00000   -0.25882    0.96593    0.00000   74.99999    0.00000  105.00000    0.00000
   90.00000   90.00000   30.00000   -0.50000    0.86603    0.00000   59.99999    0.00000  120.00000    0.00000
   90.00000   90.00000   45.00000   -0.70711    0.70711    0.00000   45.00000    0.00000  135.00000    0.00000
   90.00000   90.00000   60.00000   -0.86603    0.50000    0.00000   30.00000    0.00000  149.99998    0.00000
 ===========================================
 latlon1 =     5.000000        5.000000     latlon2 =     6.000000
    5.000000
 angle distance =     1.000000
 km distance    =     111.3188
 latlon1 =     55.00000        5.000000     latlon2 =     56.00000
    5.000000
 angle distance =     1.000000
 km distance    =     111.3188
 latlon1 =     85.00000        5.000000     latlon2 =     86.00000
    5.000000
 angle distance =     1.000000
 km distance    =     111.3188
 ===========================================
 latlon1 =     5.000000        5.000000     latlon2 =     5.000000
    6.000000
 angle distance =    0.9961947
 km distance    =     110.8952
 latlon1 =     55.00000        5.000000     latlon2 =     55.00000
    6.000000
 angle distance =    0.5735765
 km distance    =     63.84987
 latlon1 =     85.00000        5.000000     latlon2 =     85.00000
    6.000000
```

```
angle distance =    8.7155804E-02
km distance    =     9.702084
===========================================

Test WVC_Orientation
  WVC1 coordinates (Lam1,Phi1) =   -115.2000       -18.61000
  WVC2 coordinates (Lam2,Phi2) =   -123.6500       -17.52000
  WVC1 orientation Alfa1 =    173.5995        (Should equal 173.5994720)
  WVC2 orientation Alfa2 =    170.9747        (Should equal 170.9747467)
===========================================
```

Table 2.5 Output of program *test_convert*

2.5 Modules *CostFunction* and *StrucFunc*

Module *CostFunc.F90* in directory `genscat/ambrem/twodvar` contains the cost function definition of the 2DVAR method. Module *StrucFunc* in the same directory contains the error covariance model of the background field. Large parts of these modules are tested in the single observation solution test implemented in program *Test_SOS*. Table 2.6 lists its output.

The main idea behind this test is that the 2DVAR analysis increment can be calculated analytically in case of one single observation with unit probability. Starting with zero background increment and an observation increment (t_o, l_o) on the 2DVAR grid at the position with indices (1,1), the initial total cost function equals

$$J_t^{init} = \frac{t_o^2 + l_o^2}{\varepsilon^2}$$

where ε stands for the standard deviation of the observation error, which is set to 1.8 in *Test_SOS*. The 2DVAR problem now reduces to a simple optimal interpolation problem. If the standard deviation of the background error is set to the same value as that of the observation error, the final solution has $J_t^{fin} = J_o^{fin} + J_b^{fin} = \frac{1}{2} J_t^{init}$ with $J_b^{fin} = J_o^{fin}$. This allows construction of the final solution and its gradient, see *Vogelzang* [2007] for more detailed information and a complete description of the 2DVAR method.

Program *Test_SOS* reads the observation increment and the structure function parameters from an input file with default name *Test_SOS.inp*, see below. The Helmholz transformation coefficients are set according to option JV, which is the default option standing for sampled continuum (the other option is for periodic boundary conditions but these do not reproduce the correct scaling, see *Vogelzang* [2007] for more details. The program copies the structure function parameters into the *SF*-struct, and the observation increments in the *TwoDvarObs*-struct. The structure function parameters are printed by routine *PrnStrucFuncPars*.

The error covariances are calculated numerically in module *StrucFunc*. For Gaussian structure functions, they can also be calculated analytically. The two methods are compared and the relative precision is printed. In table 2.6 it is 0.00345 for the stream function ψ and 0.0 for the velocity potential χ, since the latter quantity is identically zero in this example. The precision of the covariances depends on the correlation lengths R_ψ and R_χ.

The total cost function and its gradient is evaluated by routine *JoScat* in module *CostFunction*. From this the cost function components and gradients at the final solution are calculated and checked against their analytical value. The (absolute) precision is printed. Finally, *Test_SOS* checks the packing and unpacking routines of the control vector in both directions.

As stated before, program *Test_SOS* reads its input from an input file. The name (and path) of that file must be given as command line argument of *Test_SOS*. When omitted, the program assumes *Test_SOS* as input file. Table 2.7 gives the structure and contents of the input file. It is in free format.

```
================================================================
PROGRAM Test_SOS - Single Observation Soluton Check
================================================================

2DV: 2DVAR parameters read from file
2DV: Batch grid dimension           32 by           32 points
2DV: Batch grid free edge            5 points
2DV: Batch grid size    100000.0     m

Input read from file        : Test_SOS.inp
Helmholz coefficients type  : JV

Parameters inside the StructFunc module:
  Grid size in position domain delta =     100000.0
  Northern hemisphere:
    SF(LatBandNorthern)%rpsi  =     300000.0
    SF(LatBandNorthern)%rchi  =     300000.0
    SF(LatBandNorthern)%epsi  =     2.000000
    SF(LatBandNorthern)%echi  =     2.000000
    SF(LatBandNorthern)%nu_sq =    0.1000000
  Tropics:
    SF(LatBandTropical)%rpsi  =     300000.0
    SF(LatBandTropical)%rchi  =     300000.0
    SF(LatBandTropical)%epsi  =     1.800000
    SF(LatBandTropical)%echi  =     1.800000
    SF(LatBandTropical)%nu_sq =     1.000000
  Southern hemisphere:
    SF(LatBandSouthern)%rpsi  =     300000.0
    SF(LatBandSouthern)%rchi  =     300000.0
    SF(LatBandSouthern)%epsi  =     2.000000
    SF(LatBandSouthern)%echi  =     2.000000
    SF(LatBandSouthern)%nu_sq =    0.1000000

CheckCovMat - checking precision of Covariances
  Relative precision in covariances of psi:   0.0000000
  Relative precision in covariances of chi:   2.1050793E-04

Number of observations      :            1
Number of control variables :         2046

Obs2dvar after initialization:
 i  j  Namb    u    v           Jo           gu           gv
------------------------------------------------------------
 1  1     1  1.0  0.0  0.77161E-01 -0.30864E+00  0.00000E+00

The gradient velocity fields duo and dvo (nonzero components only):
 i  j          duo          dvo
-------------------------------
 1  1 -0.30864E+00  0.00000E+00

The cost function of the solution:
  Observation part :   7.7160500E-02
  Background part  :   7.7160493E-02        precision   7.4505806E-09

The background velocity field:
  u(1,1)         :   0.5000000
  Expected value :   0.5000000                 precision   0.0000000
  v(1,1)         :   3.1561532E-20
  Expected value :   0.0000000                 precision   3.1561532E-20

Check background cost function
  Direct calculation from psi and chi   :   7.7160493E-02
  Calculation by Jb from control vector :   7.7160500E-02        precision
```

```
  7.4505806E-09

Check observation cost function
  Expected value                        :   7.7160500E-02
  Calculation by Jo from control vector :   7.7160500E-02        precision
  0.0000000
  Precision in gradients better than    1.9753911E-10

Check packing/unpacking:
  Precision in packing/unpacking of xi   0.0000000
  Precision in packing/unpacking of psi   0.0000000
  Precision in packing/unpacking of chi   0.0000000

Program Test_SOS completed.
================================================================
```

Table 2.6 Output of the single observation solution test.

Record	Item nr.	Name	Meaning
1	1	u0_ini	Initial observation increment in transversal direction (m/s)
1	2	v0_ini	Initial observation increment in longitudinal direction (m/s)
2	1	R_psi	Correlation length of stream function (m)
2	2	R_chi	Correlation length of velocity potential (m)
2	3	Div2Rot	Divergence-to-rotation ratio
2	4	MH	Character selector of length 2 for Helmholz coefficient type, MH=JV (default): continuum boundary conditions; MH=HB: periodic boundary conditions.

Table 2.7 Input file for *Test_SOS*.

2.6 Module *DateTimeMod*

Module *DateTimeMod.F90* in directory `genscat/support/datetime` contains general purpose date and time help functions. These are tested by program *TestDateTimeMod*, the output of which is listed in table 2.8.

```
time-tests
time: 14:22:03.70
time_real        = 51723.70
time_real + 77.2 = 51800.90
time: 14:23:20.90
 time2 is valid
 time1 =
time: 14:22:03.70
 time2 =
time: 14:23:20.90
 time 1 .ne. time2
 date-tests
date: 15-12-1999
 date_int =      19991215
 date_int + 1 =      19991216
date: 16-12-1999
 date2 is valid
 date1 =
date: 15-12-1999
 date2 =
date: 16-12-1999
 date 1 .ne. date2
 date-stepping-tests
 ERROR: The date      21000101  is outside the range
 19000101...20991231, this is not implemented at this time
 ERROR: Julian routines differ from my own routines
date: 31-12-2099
 next_date_int =    2147483647
```

```
date: 01-01-2100
 next_julian_date_int =      21000101
 all OK
 before:
time: 23:59:57.70
date: 31-12-1999
after incrementing by: 5.22 seconds
time: 00:00:02.92
date: 01-01-2000
 valid time
 test of function date2string: 19991231
 test of function date2string_sep: 1999-12-31
 test of function time2string: 235957
 test of function time2string_sep: 23:59:57
 before convert_to_derived_datetime:
date: 28-02-2005
time: 52:00:00.00
 after convert_to_derived_datetime:
date: 02-03-2005
time: 04:00:00.00
 Current date and time:
date: 29-04-2008
time: 17:12:26.58
```

Table 2.8 Output of program *TestDateTimeMod*.

2.7 Module *ErrorHandler*

Module *ErrorHandler.F90* in directory `genscat/support/ErrorHandler` contains routines for handling errors during program execution. The module is tested by program *TestErrorHandler*, the output of which is listed in table 2.9.

```
The Error Handler program_abort routine is set to
return after each error,
in order to try and resume the program...
testing: report_error
an error was reported from within subroutine: dummy_module_name1
error while allocating memory
testing: program_abort (with abort_on_error = .false.)
an error was reported from within subroutine: dummy_module_name2
error while allocating memory
==> trying to resume the program ...
The Error Handler program_abort routine is set to
abort on first error...
testing: program_abort (with abort_on_error = .true.)
an error was reported from within subroutine: dummy_module_name2
error while allocating memory
```

Table 2.9 Output of program *TestErrorHandler*.

2.8 Module *gribio_module*

Module *gribio_module.F90* in directory `genscat/support/grib` contains routines for reading and decoding GRIB files. The module is tested by programs *test_read_GRIB1*, *test_read_GRIB2* and *test_read_GRIB3*, the output of which is listed in tables 2.10 to 2.12.

```
date of grib field =    20031112.00
time of grib field =          24.00

    lat     lon      10u      10v
  54.00   4.00   -4.576    8.006
  54.00   4.50   -5.143    7.764
  54.00   5.00   -5.034    7.520
  54.00   5.50   -4.925    7.276
  54.50   4.00   -4.849    8.455
  54.50   4.50   -5.139    8.315
  54.50   5.00   -5.200    8.426
  54.50   5.50   -5.261    8.537
  55.00   4.00   -5.267    8.577
  55.00   4.50   -5.398    8.454
  55.00   5.00   -5.416    8.620
  55.00   5.50   -5.434    8.786
  55.50   4.00   -5.686    8.699
  55.50   4.50   -5.657    8.594
  55.50   5.00   -5.632    8.814
  55.50   5.50   -5.606    9.034
```

Table 2.10 Output of program *test_read_GRIB1*.

```
retrieve grib field par_id_t
lat of first gridpoint =   89.142
lat step               =   -1.121
number of lat points   =      160
lon of first gridpoint =    0.000
lon step               =    1.125
number of lon points   =      320

    i    j  field(i,j)
   80  160     302.663
   80  161     302.445
   80  162     302.148
   80  163     301.560
   81  160     301.999
   81  161     302.298
   81  162     301.808
   81  163     301.708
   82  160     302.056
   82  161     302.117
   82  162     301.490
   82  163     301.888
   83  160     302.214
   83  161     302.001
   83  162     301.796
   83  163     302.361
```

Table 2.11 Output of program *test_read_GRIB2*.

```
retrieve grib field par_id_10u
date of grib field =    20031112.00
time of grib field =          24.00
 WARNING: lattitude dimension of field is too small to contain
 WARNING: the read data; truncating the array !!!!!
 original:  nr_lat_points =          160
 truncated: nr_lat_points =           50
 WARNING: longitude dimension of field is too small to contain
 WARNING: the read data; truncating the array !!!!!
 original:  nr_lon_points =          320
 truncated: nr_lon_points =           50

    i    j  field(i,j)
   48   48      -0.414
   48   49       0.477
   48   50      -0.111
   49   48       3.330
   49   49       2.899
   49   50       3.252
   50   48       3.503
   50   49       2.408
```

```
    50   50       3.212
```

Table 2.12 Output of program *test_read_GRIB3*.

2.9 Module *LunManager*

Module *LunManager.F90* in directory `genscat/support/file` contains routines for file unit management. It is tested by program *TestLunManager*, the output of which is listed in table 2.13.

```
Starting fileunit test program
 ===== lun_manager ======
 fileunit:            31  was not in use !!!
 free_lun returns without freeing any fileunit
 fileunit:            88  was not in the range that is handled
 by this module ! (           30  -            39 )
 free_lun returns without freeing any fileunit
 fileunit:            88  was not in the range that is handled
 by this module ! (           30  -            39 )
 enable_lun returns without enabling any fileunit
 fileunit:            88  was not in the range that is handled
 by this module ! (           30  -            39 )
 disable_lun returns without disabling any fileunit
 fileunit:            21  was not in the range that is handled
 by this module ! (           30  -            39 )
 disable_lun returns without disabling any fileunit
 unit:            31  is used?:   F
 unit:            31  is used?:   T
 start of inspect_luns
  lun             0  is open
  lun             0  has a name: stderr
  lun             5  is open
  lun             5  has a name: stdin
  lun             6  is open
  lun             6  has a name: stdout
  lun            31  is open
  lun            31  has a name: TestLunManager.F90
 end of inspect_luns
 fileunit:            31  is still in use !
 disabling it is only possible if it is not used !
 disable_lun returns without disabling any fileunit
 fileunit:            30  is in use
 fileunit:            31  is in use
 fileunit:            32  is still available
 fileunit:            33  is still available
 fileunit:            34  is still available
 fileunit:            35  is still available
 fileunit:            36  is still available
 fileunit:            37  is still available
 fileunit:            38  is still available
 fileunit:            39  is still available
 fileunit:            21  was not in the range that is handled
 by this module ! (           30  -            39 )
 enable_lun returns without enabling any fileunit
 fileunit:            22  was not in the range that is handled
 by this module ! (           30  -            39 )
 enable_lun returns without enabling any fileunit
```

Table 2.13 Output of program *TestLunManager*.

2.10 Module *Numerics*

Module *numerics.F90* in directory `genscat/support/num` contains routines for checking and handling numerical issues like variable sizes and ranges. These are tested by program *test_numerics*, the output of which is listed in Table 2.14.

```
 Starting numerics test program
 ===== representation tests ======
 REALACC(6)
 r4: digits                24
 r4: epsilon       1.1920929E-07
 r4: huge          3.4028235E+38
 r4: minexponent         -125
 r4: maxexponent          128
 r4: precision              6
 r4: radix                  2
 r4: range                 37
 r4: tiny          1.1754944E-38
 ENDREALACC
 REALACC(12)
 r8: digits                53
 r8: epsilon       2.2204460492503131E-016
 r8: huge          1.7976931348623167E+308
 r8: minexponent        -1021
 r8: maxexponent         1024
 r8: precision             15
 r8: radix                  2
 r8: range                307
 r8: tiny          2.2250738585072010E-308
 ENDREALACC
 ===== numerics tests ======
 int1 =    127
 int2 =    32767
 int4 =    2147483647
 int8 =       9223372036854775807
 huge(int1) =           127
 huge(int2) =         32767
 huge(int4) =    2147483647
 huge(int8) =       9223372036854775807
 REALACC(6)  r4   =    1.7000000E+38  ENDREALACC
 REALACC(12) r8   =    1.7000000000000000E+038  ENDREALACC
 ===== check variable sizes  ======
 Variable sizes are as expected
 ===== detect and print variable sizes ======
var_type nr_of_words range precision
       i           4     9
     i1_           1     2
     i2_           2     4
     i4_           4     9
     i8_           8    18
      dr           4    37         6
      s_           4    37         6
      l_           4    37         6
      r_           4    37         6
     r4_           4    37         6
     r8_           8   307        15
 ===== dB conversion test ======
 REALACC(6)
 input test number:           1.2300001E-04
 converted to dB:             -39.10095
 converted back to a real:    1.2299998E-04
 ENDREALACC
 ===== done ======
```

Table 2.14 Output of program *test_mumerics*.

2.11 Module *SingletonFFT*

Module *SingletonFFT* in directory `genscat/support/singletonfft` contains routines for Fast Fourier Transforms. The associated test program is *TestSingleton*. Part of its output is shown in table 2.15.

```
=========================================================================
 PROGRAM TestSingleton
 Test of SingletonFFT routines by comparing with analytical FT
 =========================================================================

 Spreading times grid size in dimension 1:   0.1000000      (should be ~ 0.1)
 Spreading times grid size in dimension 2:   0.1000000      (should be ~ 0.1)
 =========================================================================

  1D             F O R W A R D           B A C K W A R D
             P r e c i s i o n         P r e c i s i o n
  N1         Real         Imag         Real         Imag
 -------------------------------------------------------
  32  0.89206E-06  0.10286E-04  0.11938E-06  0.53646E-07
  34  0.66905E-06  0.78932E-05  0.71246E-07  0.14503E-07
  36  0.89206E-06  0.12215E-04  0.11921E-06  0.90160E-07
  38  0.27877E-06  0.20358E-05  0.35763E-06  0.31126E-07
  40  0.83631E-06  0.12143E-04  0.11921E-06  0.57708E-07
  42  0.39028E-06  0.56252E-05  0.11921E-06  0.10509E-06
  44  0.12900E-06  0.37786E-07  0.11921E-06  0.38596E-07
  46  0.94782E-06  0.13554E-04  0.35763E-06  0.40079E-07
  48  0.89206E-06  0.14143E-04  0.11921E-06  0.66032E-07
  50  0.44603E-06  0.66967E-05  0.17881E-06  0.48369E-07
 =========================================================================

  2D                F O R W A R D   F F T          B A C K W A R D   F F T
                 P r e c i s i o n                P r e c i s i o n
  N1  N2         Real         Imag   Time         Real         Imag   Time
 -------------------------------------------------------------------------
  32  32  0.12516E-05  0.20572E-04 0.0000  0.11921E-06  0.63015E-07 0.0000
  32  34  0.11473E-05  0.18179E-04 0.0000  0.11921E-06  0.41598E-07 0.0000
  32  36  0.12516E-05  0.22501E-04 0.0000  0.11921E-06  0.56660E-07 0.0010
  32  38  0.88658E-06  0.82503E-05 0.0010  0.29802E-06  0.41553E-07 0.0000
  32  40  0.11473E-05  0.22430E-04 0.0000  0.17881E-06  0.52022E-07 0.0010
  32  42  0.99089E-06  0.15911E-04 0.0010  0.11921E-06  0.12113E-06 0.0000
  32  44  0.88658E-06  0.10286E-04 0.0000  0.11921E-06  0.56563E-07 0.0010
  32  46  0.12516E-05  0.23840E-04 0.0000  0.41723E-06  0.37254E-07 0.0010
  32  48  0.12516E-05  0.24430E-04 0.0010  0.17881E-06  0.65104E-07 0.0000
  32  50  0.99089E-06  0.16983E-04 0.0010  0.23842E-06  0.58744E-07 0.0000
  34  32  0.11473E-05  0.18179E-04 0.0000  0.11921E-06  0.94071E-07 0.0010
.....
  48  50  0.99089E-06  0.20840E-04 0.0010  0.23842E-06  0.73236E-07 0.0010
  50  32  0.99089E-06  0.16983E-04 0.0010  0.17881E-06  0.49138E-07 0.0000
  50  34  0.83443E-06  0.14590E-04 0.0010  0.23842E-06  0.53570E-07 0.0010
  50  36  0.10430E-05  0.18912E-04 0.0000  0.23842E-06  0.70452E-07 0.0010
  50  38  0.41722E-06  0.46609E-05 0.0010  0.29802E-06  0.41385E-07 0.0010
  50  40  0.93873E-06  0.18840E-04 0.0000  0.35763E-06  0.47009E-07 0.0000
  50  42  0.52152E-06  0.12322E-04 0.0020  0.29802E-06  0.10955E-06 0.0010
  50  44  0.41722E-06  0.66967E-05 0.0010  0.23842E-06  0.49293E-07 0.0010
  50  46  0.99089E-06  0.20251E-04 0.0010  0.23842E-06  0.44801E-07 0.0010
  50  48  0.99089E-06  0.20840E-04 0.0000  0.23842E-06  0.57817E-07 0.0000
  50  50  0.57367E-06  0.13393E-04 0.0010  0.41723E-06  0.63718E-07 0.0010
 =========================================================================

 Program TestSingleton: Resume
 Worst case accuracies

                  F O R W A R D           B A C K W A R D
              Real         Imag         Real         Imag
 --------------------------------------------------------
 1D    0.94782E-06  0.14143E-04  0.35763E-06  0.10509E-06
 2D    0.13038E-05  0.28287E-04  0.65565E-06  0.23791E-06

 Program TestSingleton: Normal termination.
 =========================================================================
```

Table 2.15 Output of program *TestSingleton*

2.12 Module *SortMod*

Module *SortMod* in directory `genscat/support/sort` contains two routines for sorting the wind vector solutions found in the inversion step to their probability. The associated test program is *SortModTest*. Its output is shown in table 2.16.

```
Test program for the SortMod module
Unsorted array
10.0  9.0  8.0  7.0  6.0  5.0  4.0  3.0  2.0  1.0
After GetSortIndex
 1.0  2.0  3.0  4.0  5.0  6.0  7.0  8.0  9.0 10.0
Sorted array, after SortWithIndex
 1.0  2.0  3.0  4.0  5.0  6.0  7.0  8.0  9.0 10.0
```

Table 2.16 Output of program *SortModTest*

Chapter 3

AWDP integration test

Directory `awdp/tests` contains two ASCAT BUFR files to test the AWDP executable. File `ascat_20070426_test_250.l1_bufr` contains ASCAT level 1b data from 26 April 2007, 9:51 to 10:29 UTC with 25 km cell spacing. The same data, but on 12.5 km cell spacing, is available in file `ascat_20070426_test_125.l1_bufr`. The files `ECMWF*.grib` contain the necessary NWP data (SST, land-sea mask and wind forecasts) to perform the NWP collocation step.

The user can test the proper functioning of AWDP using the files in the `awdp/tests` directory. To do this, first create a small file containing a list of NWP files:

```
ls -1 ECMWF_200704260000_0* > nwpflist
```

Then run AWDP on 25-km and 12.5-km cell spacing:

```
../execs/awdp_run -f ascat_20070426_test_250.l1_bufr -nwpfl nwpflist -mon -calval
```

```
../execs/awdp_run -f ascat_20070426_test_125.l1_bufr -nwpfl nwpflist -mon -calval
```

The result should be two ASCAT level 2 files in BUFR format, called `ascat_20070426_095102_metopa_02681_srv_o_250_ovw.l2_bufr` and `ascat_20070426_095100_metopa_02681_srv_o_125_ovw.l2_bufr`, respectively.

Directory `awdp/tests` also contains an ERS file in ESA BUFR format, called `scatt_20070426_test_250.l1_bufr`. The data are from the same date as the ASCAT data in this directory and they can be processed using the same ECMWF files. Before switching between ASCAT and ERS processing, remove the CMOD Geophysical Model Function (GMF) lookup tables in the current directory:

```
rm -f *.dat *.dat.zspace
```

```
../execs/awdp_run -f scatt_20070426_test_250.l1_bufr -nwpfl nwpflist -mon
```

The result should be an output file in ASCAT BUFR format, called `scatt_20070426_063627_ers2___00000_srv_o_250_ovw.l2_bufr`

Note that by default, the ASCAT winds will be calculated using the CMOD5.5 GMF, whereas for

ERS the CMOD5 GMF will be used. The CMOD5.5 is the same as CMOD5, but 0.5 m/s is added to the wind speeds.

3.1 ASCAT test data

Figure 3.1 shows the global coverage of the ASCAT test run on 25 km. The colours show the magnitude of the wind speed as indicated by the legend. The result on 12.5 km should be very similar to this. The figures 3.2 and 3.3 show detailed wind vector plots over the Atlantic near the UK, with 25 km and 12.5 km cell spacing, respectively. In the detail plots, a magenta marker on top of the wind arrow denotes land presence. Yellow wind arrows indicate that the Variational Quality Control flag is set, i.e. the Wind Vector Cell is spatially inconsistent. A yellow dot means that the KNMI Quality Control Flag is set.

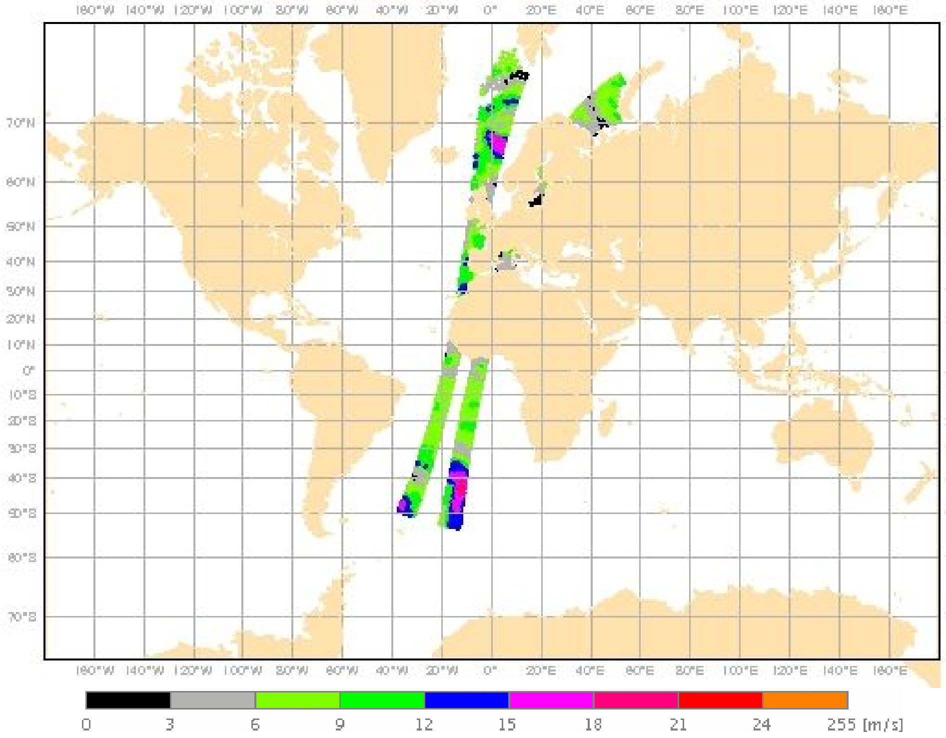

Figure 3.1 Global coverage of the ASCAT test run. Wind speed results for the 25 km product are shown.

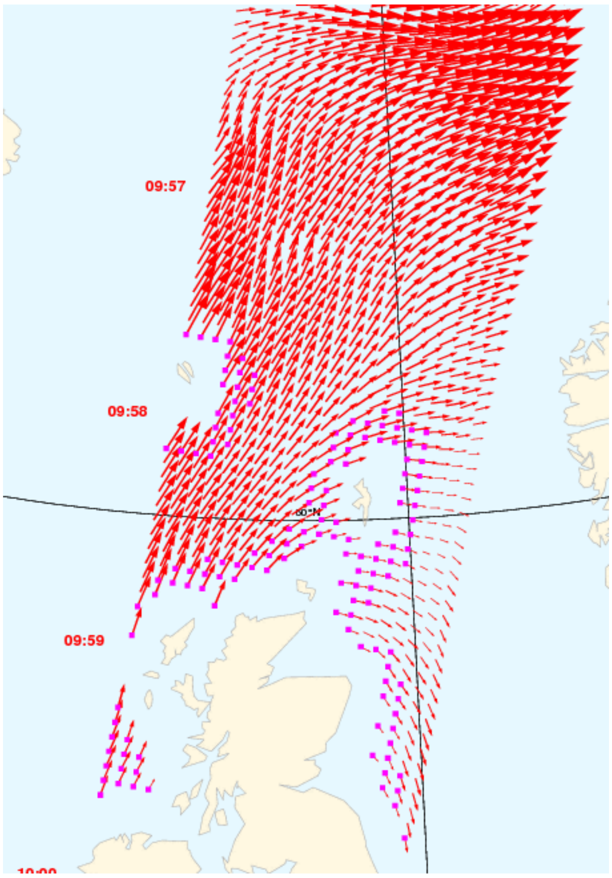

Figure 3.2 Detail plot of the ASCAT test run. Wind vectors for the 25 km product are shown.

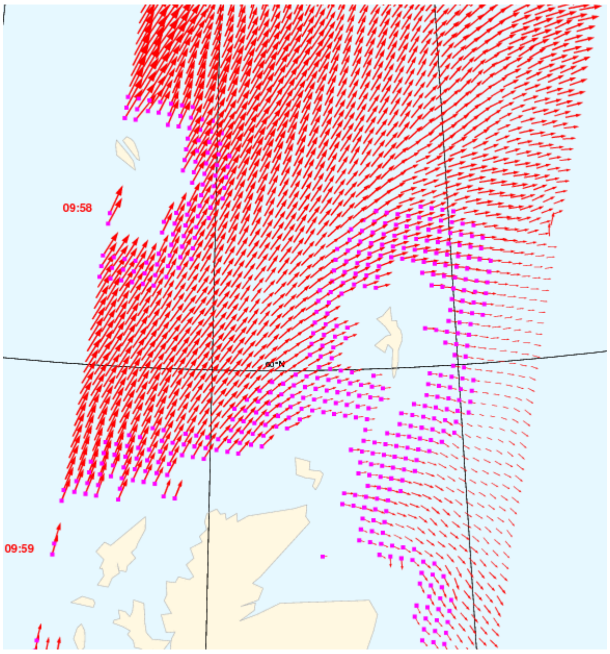

Figure 3.3 Detail plot of the ASCAT test run. Wind vectors for the 12.5 km product are shown.

3.2 ERS test data

Figure 3.4 shows the global coverage of the ERS test run on 25 km. The colours show the magnitude of the wind speed as indicated by the legend. Since the ERS data are only available when the satellite is in sight of a ground station, only a limited spatial coverage is obtained. Figure 3.5 shows detailed wind vector plots over the Pacific near the US west coast. In the detail plots, a magenta marker on top of the wind arrow denotes land presence. Yellow wind arrows indicate that the Variational Quality Control flag is set, i.e. the Wind Vector Cell is spatially inconsistent. A yellow dot means that the KNMI Quality Control Flag is set.

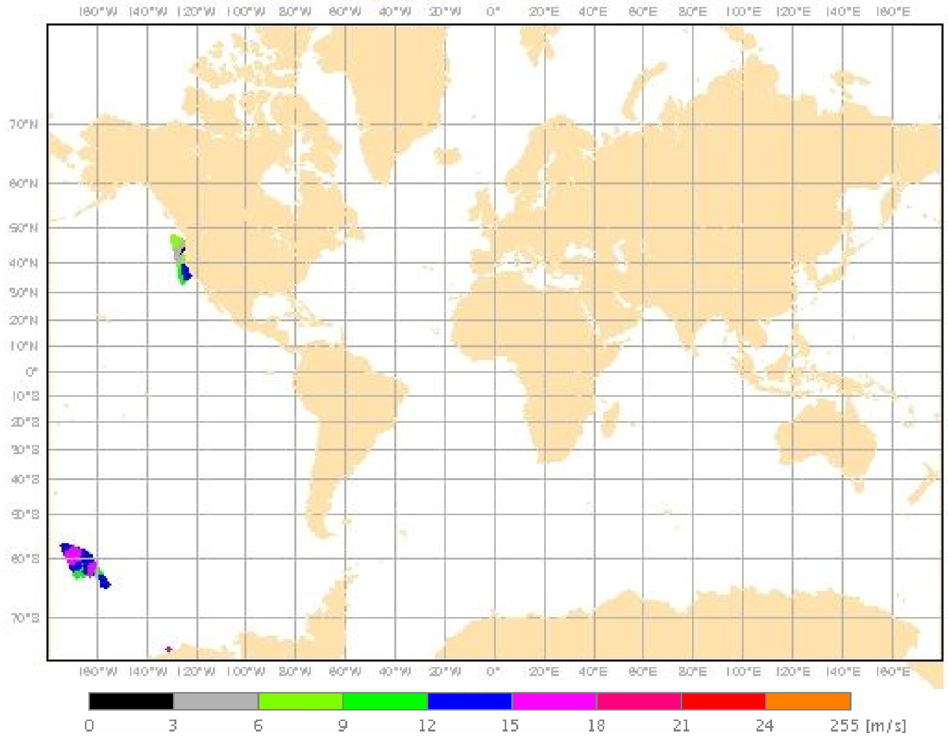

Figure 3.4 Global coverage of the ERS test run.

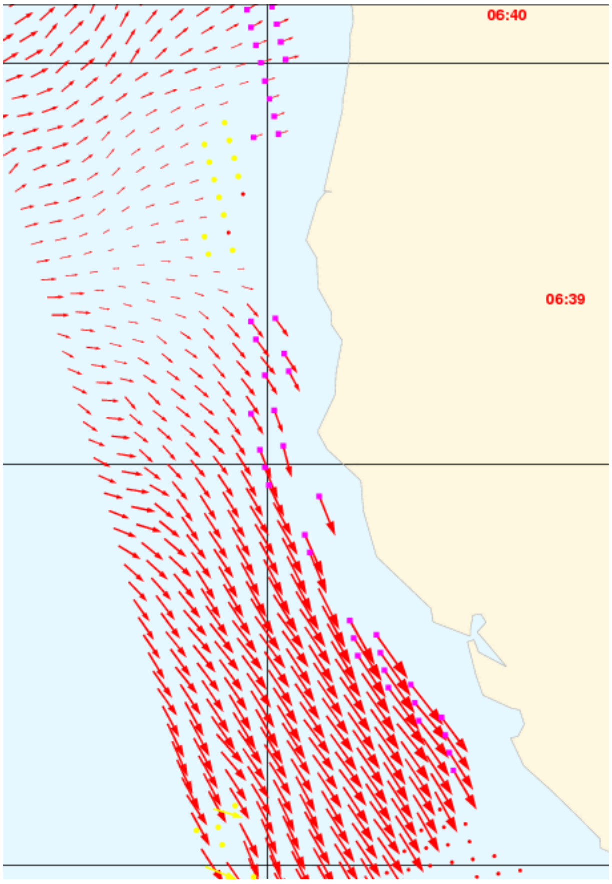

Figure 3.5 Detail plot of the ERS test run.

Chapter 4

Validation tests

Since there is no other wind processing software available for ASCAT data, no ASCAT validation tests have been done in this scope. ASCAT winds from AWDP are routinely compared with NWP and buoy data in the OSI SAF project. See http://www.knmi.nl/scatterometer/osisaf/ for more information.

On the other hand, there is ERS wind processing software available: the Prescat package [*Stoffelen*, 1998] has been used to routinely process ERS winds at KNMI for many years. In the next section we compare ERS winds computed with Prescat with those computed by AWDP.

4.1 AWDP versus Prescat

Figure 4.1 shows the collocations of the ERS winds computed by Prescat and those computed by AWDP. Contoured histograms are shown for wind speed, wind direction and u and v wind components. In the wind direction plots, only those wind vectors where the Prescat wind speed is at least 4 m/s are taken into account. The bin sizes for the histograms are 0.5 m/s for wind speed, u and v, and 5° for wind direction.

The ERS data are those from 26 April 2007 which are also used in the previous section. It is clear from the plots that the results are almost spot on and no biases are discernable. Although a limited data set is used in this experiment, it is obvious that the AWDP winds are equivalent to the Prescat winds.

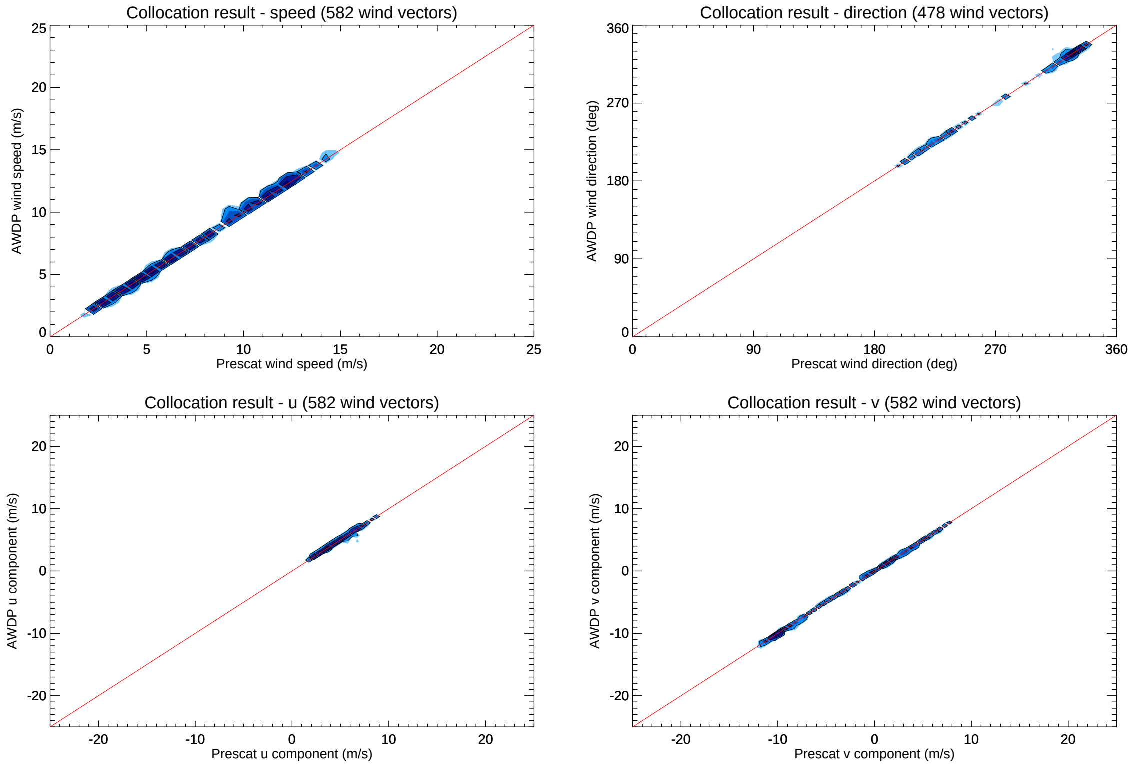

Figure 4.1 Collocation results of ERS winds from Prescat and AWDP.

Chapter 5

Portability tests

The AWDP program inherits its portability by using strict Fortran 90 code (with a few low level routines for reading and writing binary in C). AWDP is delivered with a complete make system. The Makeoptions include file of genscat takes care of the different settings needed under various platforms. This Makeoptions file is also used for the SeaWinds scatterometer processor SDP.

The default platform for development is a LINUX work station. Different Fortran 90 compilers were used to compile both genscat and AWDP. Table 5.1 provides an overview of the platforms and compilers on which AWDP was tested successfully. Note that AWDP can be run under Windows when the LINUX emulator Cygwin is installed.

Platform	Operating system	Fortran compiler
Intel-based workstation	SuSe LINUX	GNU g95, Portland f90, gfortran, Intel Fortran
SUN	SUN OS UNIX	Sun Fortran
DEC/Compaq Alpha	Tru64 UNIX	Compaq Fortran
SGI	IRIX64 UNIX	MIPSpro
PC	Windows XP with Cygwin	GNU g95

Table 5.1 Supported platforms and compilers for AWDP.

Chapter 6

User documentation tests

The user documentation (readme files within the software package and the AWDP User Manual and Reference Guide) has been and will be provided to beta testers for review. The beta tester's comments are implemented in newer versions of the user documentation.

References

- 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 on http://www.knmi.nl/scatterometer/publications/).
- Verhoef, A., Vogelzang, J., Verspeek, J. and Stoffelen, A., 2007,
  AWDP User Manual and Reference Guide, Report NWPSAF-KN-UD-005, UKMO, UK.
- Stoffelen, A.C.M., 1998,
  Scatterometry. PhD thesis, University of Utrecht, ISBN 90-393-1708-9. (Available on http://www.knmi.nl/scatterometer/publications/).
- Vogelzang, J., 2007,
  Two dimensional variational ambiguity removal (2DVAR). Report NWPSAF-KN-TR-004, UKMO, UK. (Available on http://www.knmi.nl/scatterometer/publications/).

Appendix A

Acronyms

Name	Description
AMI	Active Microwave Instrument, scatterometer on ERS-1 and ERS-2 satellites
AR	Ambiguity Removal
ASCAT	Advanced SCATterometer on MetOp
BUFR	Binary Universal Form for the Representation of data
C-band	Radar wavelength at about 5 cm
ERS	European Remote Sensing satellites
ECMWF	European Centre for Medium-range Weather Forecasts
EUMETSAT	European Organization for the Exploitation of Meteorological Satellites
genscat	generic scatterometer software routines
GMF	Geophysical model function
KNMI	Koninklijk Nederlands Meteorologisch Instituut (Royal Netherlands Meteorological Institute)
Ku-band	Radar wavelength at about 2 cm
L1b	Level 1b product
LSM	Land Sea Mask
LUT	Look up table
MetOp	Meteorological Operational Satellite
MLE	Maximum Likelihood Estimator
MSS	Multiple Solution Scheme
NRCS	Normalized Radar Cross-Section (σ_0)
NWP	Numerical Weather Prediction
OSI	Ocean and Sea Ice
PFS	Product Format Specification (native MetOp file format)
QC	Quality Control
RFSCAT	Rotating Fan beam Scatterometer
RMS	Root Mean Square
SAF	Satellite Application Facility
SSM	Surface Soil Moisture
SST	Sea Surface Temperature
WVC	Wind Vector Cell, also called node or cell

Table A.1 List of acronyms.