Solar Backscattered Ultraviolet Radiometer (SBUV)
Operational Ozone Product System Version 8

System Maintenance Manual

(Documentation Version 1.0)

December 2006

Prepared for:
National Environmental Satellite, Data, and Information Service,
Office of Satellite Data Processing and Distribution,
Information Processing Division

Prepared by:
Qiang Zhao (IMSG Inc.) and Joel Selekof (Science & Technology Corp.)
NOAA/NESDIS/ORA & SSD
World Weather Building
Camp Springs, MD 20746-4304

Last Revision: June 03, 2008

Summary of Changes

The following is a list of documented changes made to this document to reflect changes in the Version 8 software or processing methodology.

Version 1.1 01/05/07 – TEK

Description: Additions to section 7 and 7.1 to include information about the automatic email monitoring procedure and web page monitoring checks. Added section 4.5 to describe the automated daily "endofday" monitoring process and also added the Summary of Changes section to this document. Renumbered old section 4.5 and 4.6 to 4.6 and 4.7

Version 1.2 05/30/07 – VRA

Description: Modification to sections 4.5.a, 4.5.b, 4.7 and 7.0 to include information about the Zonal average daily datasets and comparison plots for SBUV/2 V8 (CCR 3142).

Version 1.3 06/03/08 – TEK

Description: Modifications to section 2.1 and 4.6.2 to remove the monthly tar files since CLASS is now using the daily files for archiving. Also changed all occurrences of Metop5 to North since that system replaced Metop5 some time ago. (CCR 3644)

Version 1.4 12/01/08 – VRA

Description: Modifications to section 5.1 to change the names of few sub programs by adding "v8" at the end. (PR000032).

1 APPLICATION OVERVIEW

The Solar Backscattered Ultraviolet radiometer (SBUV) algorithm Version 8 was developed at NASA Goddard Space Flight Center (GSFC) by a science team from Science Systems and Application, Inc. (SSAI). The operational implementation of the algorithm was carried out at NESDIS/ORA.

The SBUV instruments are flown in polar orbits to obtain global coverage. There are about 14 orbits per day with 26 degrees of separation at the equator. Each SBUV instrument consists of a nadir-viewing double monochromator of the Ebert-Fastie type, and a cloud cover radiometer (CCR, a filter photometer). A photomultiplier tube detects light exiting the monochromator. They are designed to measure the ratio of backscattered ultraviolet solar radiance to solar irradiance. During normal operation, the backscattered radiance from the nadir is measured at 12 near-UV wavelengths from 252 to 340 nm with a band-pass of 1.1 nm about 100 times per orbit on the sunlit portion. The CCR makes measurements at 379 nm wavelength with a band-pass of 3 nm, and is used to detect scene reflectivity changes during a scan.

Radiation at the near–UV wavelengths is absorbed by ozone, such that the difference between the incoming and outgoing radiation can be related to the amount of ozone in the atmosphere. Radiation at the eight shortest wavelengths is absorbed by atmospheric ozone before reaching the surface, implying that radiation scattered back to the instrument came from a particular altitude region. Radiation at successively longer wavelengths penetrates deeper into the atmosphere before being completely absorbed by ozone, allowing for a measure of the ozone profile. In the Version 8 algorithm, the total ozone is calculated as the sum of the retrieved profile ozone, rather than from measurements at the four longest wavelengths, which do penetrate to the surface. This makes the total ozone less sensitive to the variations in surface reflectivity and scattering processes in the troposphere.

2 OPERATING ENVIRONMENT

2.1 Hardware Environment

The SBUV Version 8 operational system operates on IBM POWER based machines. The system does not demand a large amount of RAM (less than 20 megabytes), nor intensive CPU usage.

The needed hard drive space depends on the system configuration. Currently, the Version 8 system processes SBUV data from three satellites (NOAA-16, NOAA-17 and NOAA-18) and keeps a week worth of output data files on the local disk, which requires about 250 megabytes of disk space.

The daily V8 binary files are sent to CLASS for archiving and are kept locally for a period of 72 hours. The orbital V8 binary files are over-written each day. The daily V8 BUFR files are over-written each day. The orbital V8 BUFR files are retained for 72 hours.

2.2 Software Environment

2.2.1 The system runs under the IBM AIX 5L operating system, a version of UNIX, and uses default shell (sh) scripts.

2.2.2 XL FORTRAN for AIX Version 7.1 or higher is required. Small Perl scripts are also used for manipulating date and time formats.

3 COMPILATION AND INSTALLATION PROCEDURES

3.1 Compilation

The Version 8 processing program **v8pmf1day.x** is made from more than 100 FORTRAN code units. The make utility is used to organize and compile/link the program. All of the source files comprising the V8 system have been placed under the Revision Control System (RCS) on North.

3.1.1 BUFR Compilation

Software was developed to produce V8 BUFR files (both daily and orbital) from the V8 binary files. There is a main "driver" program, called **pmfbufrv8.f**, which has been placed under the Revision Control System (RCS) on North. The other software components of the V8 BUFR program include a Subroutine called **getFileName.f** (a system routine contained within the common utility library), and also a library of NCEP BUFR Subroutines, contained in a file called **libncepbufr.a**, which is a library of object modules. The library of BUFR object modules (**libncepbufr.a**) is located in the directory called $ANCHOR/apps/ozone/ver8/bufr on North.

When running the V8 BUFR program, there is also a BUFR table, called **bufrtabl,** which gets read in, but this table is not needed for compiling and linking the program. The file **bufrtabl** is located in $ANCHOR/apps/ozone/ver8/data.

3.2 Makefile information

The makefile **make_v8pmf1day** has been created for compiling and linking the Version 8 program. To recompile and create a new executable, simply submit the command:

make –f make_v8pmf1day

And then the executable **v8pmf1day.x** will be generated under the current directory.

The V8 make file has been placed under RCS on North.

3.2.1 BUFR Makefile information (making changes to BUFR software)

The V8 BUFR make file is called **make_pmfbufrv8**, and this has been placed under RCS on North. There are only two software modules that should ever need to be modified:

1) the driver program, **pmfbufrv8.f**
2) the BUFR table, **bufrtabl**.

To change the driver program:

- check out **pmfbufrv8.f** and **make_pmfbufrv8** from RCS, bringing them into your directory.
- modify **pmfbufrv8.f** as desired
- create a new executable called **pmfbufrv8.x**, and test it out to verify it works as expected (do this by running the make file as follows):
  make –f make_pmfbufrv8 (this command will create the executable)
- after testing is completed, have the new version of pmfbufrv8.f placed into RCS
- copy the new executable from your directory to the operational directory $ANCHOR/apps/ozone/ver8/bin.
- copy the new executable over to the same directory on Emerald and Diamond

To change the BUFR table:

The BUFR table, called **bufrtabl**, is located in the directory $ANCHOR/apps/ozone/ver8/data.

- first make a backup of the current BUFR table
- make the desired changes to the file **bufrtabl**
- copy the new **bufrtabl** over to the same directory on Emerald and Diamond.

3.3 Directory Description

The Version 8 system includes the following subdirectories under the V8 main directory $ANCHOR/apps/ozone/ver8/:

bin/:
FORTRAN executables

control/:
Calibration constants and control files

data/:
Auxiliary input and output data files
This directory also contains the BUFR table, called **bufrtabl**

data/v8snocld:
Snow/ice cover and cloud climatology

logs/:
Log files

scripts/:
Shell and Perl scripts

tmp/dist:
Used for testing purposes, so that products made while testing will not get distributed. Actual operational V8 products are moved into $ANCHOR/data/POES/ozone and then sent to the Data Distribution Server (DDS).

bufr:
NCEP BUFR library (object modules), called libncepbufr.a. (These object modules are linked in by the BUFR make file).

3.4 Installation

The recompiled executable can be simply copied or moved into $ANCHOR/apps/ozone/ver8/bin/ directory of the new system.

4 PROCESS DESCRIPTION

The Version 8 system processes daily data sets with an integrated FORTRAN program **v8pmf1day.x**. A group of Unix Shell scripts and Perl scripts are needed to organize input, output and auxiliary data files and other control parameters.

4.1 Daily data processing

The Version 8 daily processing system is invoked from the OOPS Product Processor run script, **run_trXpprod.sh**. After the 1-day Version 6 PMF gets produced, the script will then invoke the V8 software, as follows:

4.1a **run_n*xx*v8pmf1day.sh**

4.1a.1 Run string

run_n*xx*v8pmf1day.sh

4.1a.2 Description

This is a satellite specific driver script, which simply sets the satellite ID to an environment variable SATNAME and calls a more generic main script **run_v8pmf1day.sh** to do the data processing.

4.1a.3 Sub scripts and program called

run_v8script.sh
run_v8pmf1day.sh

4.1a.4 Modifications needed for a new satellite

The only modification to this driver script is to set the environment variable **SATNAME** to the new satellite id, e.g. n18.

Note: Before the system is capable to process data from a new satellite, a set of satellite specific calibration parameters must be provided and some source code may need to be modified. The calibration information and source code modifications are expected to be received from the NASA science team.

4.1b **run_v8pmf1day.sh**

4.1b.1 Run string

run_v8pmf1day.sh

4.1b.2 Description

This is the main script, which sets up a control file and calls the data processing program **v8pmf1day.x** to do the daily data processing.

4.1b.3 Sub scripts and program called

run_v8script.sh
distver8.sh
day.pl
julian.pl
month.pl
year.pl
yy.pl
v8pmf1day.x

4.1b.4 Modifications needed for a new satellite

A satellite dependent case segment needs to be added to this script.

Note: Before the system is capable to process data from a new satellite, a set of satellite specific calibration parameters must be provided and some source code may need to be modified. The calibration information and source code modifications are expected to be received from the NASA science team.

4.2 Orbital data processing

4.2.a ROPES run script **run_trXorbit.sh**
→ invokes **run_pmf1orbX.sh**
-- runs program pmf1day.x
-- runs program dirtoseq.x
→ invokes **run_nXXv8pmf1orb.sh**
→ invokes **run_v8pmf1orb.sh**
-- runs program v8pmf1day.x

4.2.b Description

Version 8 orbital processing is invoked from the ROPES run script, called **run_trXorbit.sh**. After the V6 orbital Product Processor is run (program pprob.x) and the latest orbit has been added to the orbital PMF (ORBOPMxOP), the run script called **run_pmf1orbX.sh** is invoked. This script runs program pmf1day.x that will extract the latest orbit from ORBOPMxOP and create a file with just the latest orbit. Then, program dirtoseq.x is run which converts the V6 orbital file from direct access format to sequential format, which is what the V8 program

requires.

At this point, control is returned back to the main ROPES run script, **run_trXorbit.sh**. This script now invokes the run script called **run_nXXv8pmf1orb.sh** that sets up the satellite name and in turn invokes the V8 script called **run_v8pmf1orb.sh**. This script runs the V8 program, v8pmf1day.x, to create the orbital V8 product.

4.2.c Subscripts called

run_v8script.sh

4.2.d Modifications needed for a new satellite

The following new run scripts would need to be set up:

run_tr**3**orbit.sh (main ROPES run script; "3" is currently not used)
run_pmf1orb**3**.sh ("3" is currently not used)
run_n**19**v8pmf1orb.sh (the next SBUV satellite will be NOAA-19)

4.3 BUFR Daily data processing

The Version 8 daily BUFR processing is invoked from the run script **run_v8pmf1day.sh**. After the daily Version 8 PMF (binary format) gets produced, the script will then invoke the V8 BUFR software to produce a BUFR version of the daily V8 product, as follows:

4.3a **run_pmfbufrv8.sh**

4.3a.1 Run string

run_pmfbufrv8.sh <satellite name> <name of V8 daily PMF>

4.3a.2 Description

This script will read in the satellite name (argument 1) and the name of the daily binary V8 PMF (argument 2). It will run the V8 BUFR program and produce a daily V8 PMF product in BUFR format. The product will have a name that looks like this:

PRD.OZONE.PMFV8.Nxx.BUFR

4.3.b Subscripts called

run_v8script.sh

4.3.c Modifications needed for a new satellite

No new scripts would need to be set up:

4.4 BUFR Orbital data processing

4.4.a V8 orbital run script **run_v8pmf1orb.sh**
- runs program **orjclborb.x**
-- fills in skeleton script with name of orbital V8 BUFR file
→ invokes **run_pmfbufrv8_orb.sh**
- runs program **pmfbufrv8.x** to create V8 orbital BUFR file

4.4.b Description

The V8 orbital BUFR processing is invoked from the run script **run_v8pmf1orb.sh**. After the V8 orbital product (binary format) has been made, the run script calls program **orjclborb.x**. This program reads the orbital V6 PMF, gets the date, start and stop hour/min, and orbit number, and fills in a skeleton script with the name of the orbital V8 BUFR file to be produced. It then invokes the script **run_pmfbufrv8_orb.sh** (actually a temporary copy of this with the name filled in) which then runs the BUFR program, **pmfbufrv8.x,** to create the V8 orbital BUFR product.

The V8 orbital BUFR product will have a name that looks like this:

PRD.OZONE.PMFV8.Nxx.D-----.S----.E----.B-----

4.4.c Subscripts called

run_v8script.sh

4.4.d Modifications needed for a new satellite

No new run scripts would need to be set up:

4.5 End of day monitoring processing

The Version 8 daily end of day monitoring is invoked from the run script **run_v8endofday.sh**. This script is run at the end of the V6 run_endofday.sh script and uses the V8 daily PMF file to update several datasets used for monitoring.

4.5.a OOPS run script **run_endofday.sh**

→ invokes **run_v8endofday.sh**

→ invokes **run_pmfv8zone[1,2,4].sh**

-- runs program v8pmfzone.x

→ invokes **run_pmfv8pct[1,2,4].sh**

-- runs program v8pmfpct.x

→ invokes **run_pmfv8sbuvoz[1,2,4].sh**

-- runs program v8pmfsbuvoz.x

→ invokes **run_pmfv8laymon[1,2,4].sh**

-- runs program v8pmflaymon.x

→ invokes **run_pmf[1,2,4]v8toye.sh**

-- runs program pmfv8zstat.x

4.5.b Description

Version 8 end of day processing is invoked from the OOPS run script, called **run_endofday.sh**. Once the V8 script is called, it in turn runs five sets of processing scripts (per satellite) to update monitoring datasets used in updating the monitoring webpages.

4.2.c Subscripts called

run_v8script.sh, run_pmfv8zone[1,2,4]

4.2.d Modifications needed for a new satellite

The following new run scripts would need to be set up:

run_tr**3**orbit.sh (main ROPES run script; "3" is currently not used)
run_pmf1orb**3**.sh ("3" is currently not used)
run_n**19**v8pmf1orb.sh (the next SBUV satellite will be NOAA-19)

4.6 Cleaning up old files

The OOPS run script, **run_cleanup.sh**, also invokes the V8 cleanup script, called **run_v8cleanup.sh**. The **run_cleanup.sh** script is a "time of day" job that runs every day at 2359Z.

4.6.1 Run string

run_v8cleanup.sh

4.6.2 Description

This script cleans out the V8 log files and daily files, retaining one week of files.

4.6.3 Sub scripts and programs called

run_v8script.sh

4.6.4 Modifications needed for a new satellite

When a new satellite is added to or an old satellite is dropped off the system, the number of data and log files need to be adjusted for local storage.

4.7 Other supporting scripts

distver8.sh

Distributing daily SBUV Version 8 data to users

run_v8script.sh

Setting up environment variables for system

day.pl

Printing yesterday's date in format DD

julian.pl

Printing yesterday's Julian day of year

month.pl

Printing yesterday's month in format MM

year.pl

Printing yesterday's year in format YYYY

yy.pl

Printing yesterday's year in format YY

yyyymmdd.pl

Printing yesterday's year month day in format YYYYMMDD

5 PROGRAM DESCRIPTION

5.1 SBUV Version 8 processing program

The daily Version 8 processing is conducted with a single integrated FORTRAN program **v8pmf1day.x**. There are three functional parts in the program: initialization, processing and termination. In the initialization part, some constants and look-up tables are loaded or calculated. The termination part simply terminates the daily job by closing opened files. As the main part of the system, the processing part is a big loop. In each looping cycle, the program reads in one data record of SBUV V6 daily output, calculates total ozone, calculates profile ozone and writes out one data record of new retrievals. These sequences of processing loop through all the valid input V6 data records. Before the first data record, two header records are written out; And after the last data record, a trailer record with brief statistics is written out to conclude the whole daily data file. See Figure 1 for the hierarchical diagram of Version 8 program routines.

5.1.1 Make file

make_v8pmf1day

5.1.2 Main program

sbvmain.f

5.1.3 Sub programs

altitude.f
: Subroutine for calculating physical height at specified pressure levels

aprsbo.f
: Subroutine for determining a priori ozone profile for a given day and latitude

aprsbt.f
: Subroutine for determining a priori temperature profile for a given day and latitude

blockdata.f
: Block data for initializing various parameters in common blocks

blwcld.f
: Subroutine for calculating column ozone below cloud layer

cnstnts.f
: Subroutine for initializing various constant parameters used in the retrieval

coffs.f
: Subroutine for assigning optical coefficients and weights for SBUV sub-channels

coffs_c.f
: Subroutine for recalculating optical coefficients and weights in anomalous cases

convert20.f
: Subroutine for converting fine (80)-layer retrieved, a priori, and first guess ozone amounts, from Rodgers-type solution into corresponding coarse (20)-layer quantities. It also determines estimated measurement errors in 20-layer and total ozone, determines 10-channel x 20-layer kernel, and, if asked for, determines averaging kernel in 20 layers

Convrtv8.f
: Subroutine for converting an integer number to a character string

convtc.f
: Subroutine for converting the first NCHAR digits of an integer into a character or a string

cpystr.f
: Subroutine for copying a string into a character array

datimx.f
: Subroutine for fetching system local time and returning various time elements

delnbyt.f
: Subroutine for computing the change in N-value due to temperature difference for each of 11 Umkehr layers and each of 8 wavelengths using a chain rule relating dN/dT to dN/dX

dproc.f

Sub-driver for controlling the data processing

dtailp.f
: Subroutine for writing detailed data records for profile

dtailt.f
: Subroutine for writing detailed data records for total ozone

factor.f
: Subroutine for performing an LU factorization of a general matrix of dimension N x N

fxphot.f
: Subroutine for performing nimbus-7 photometer for along track miss-alignment of 6 km using linear interpolation and extrapolation

get_lun.f
: Subroutine for keeping logical unit numbers straight and issuing new ones when necessary. The entry point free_lun closes file and allows the logical unit number to be reused

getcov.f
: Subroutine for reading snow-ice cover and cloud pressure data for current and previous months

getmsr.f
: Subroutine for computing the multiple scattered and surface reflected part of the Jacobian dN/dX for each of 11 Umkehr layers and each of 8 wavelengths by using finite differencing between N-values interpolated from unperturbed and perturbed ozone tables. 12 tables are contained in common: 1 unperturbed followed by 11 with perturbations in Umkehr layers 0–9 and the sum of layers 10-12

getrng.f
: Subroutine for interpolating appropriate Ring corrections in reflectivity and pressure

getshp.f
: Subroutine for determining merged a priori ozone profile given ozone, day and latitude

getsur.f

Subroutine for reading terrain height pressure data and surface category code

incovr.f

Subroutine for computing the snow-ice thickness, covered on the earth, in tenths of inches and computing the cloud pressure

insurf.f

Subroutine for computing the terrain height pressure and finding surface category code at a given latitude and longitude on the surface of the earth

inter.f

Subroutine for finding terrain, cloud pressure, snow/ice, and surface category code from appropriate data bases for a given date, latitude and longitude

interp.f

Subroutine for performing table interpolations for parameters i_0, T_r, and s_b, which are used in the calling routine to compute n-value for a given table pressure, profile albedos, and zenith angle

interpol_kern.f

Subroutine for performing interpolation of multiple scattering kernel from Umkehr to fine layers

interpol_qo3.f

Subroutine for performing interpolation of layer ozone from Umkehr to fine layers

interpol_t.f

Subroutine for performing interpolation of temperature from Umkehr to fine layers

inverse.f

Subroutine for solving $a * x = b$

iztrsb.f

Subroutine for performing interpolations of the table parameters i_0, T_r, and s_b, which are used by the calling routines to calculate table radiances or N-values. It uses Lagrangian interpolation except for high clouds for which a linear extrapolation is used

jd.f
: Function for computing the Julian date corresponding to a given calendar date

linear_fit.f
: Subroutine for performing linear fit of a set of points

lodprf.f
: Subroutine for loading buffer 'bufout' with retrieved profile data, starting at location 101

lodtoz.f
: Subroutine for loading buffer 'bufout' with total ozone data

mixratio.f
: Subroutine for determining ozone mixing ratio and its estimated percentage error at prescribed pressure levels

n7hyst.f
: Subroutine for computing hysteresis correction for n7 SBUV

newflxv8.f
: Subroutine for calculating radiance/flux adjustment for N11

nvbrac.f
: Subroutine for performing N value bracketing

o3_retrieval.f
: Subroutine for performing ozone retrieval using Rodgers method

o3_snglscatt.f
: Subroutine for computing ozone single scattering approximation

oproc.f
: Sub-driver to control the processing of one orbit of data from input data file

order.f
: Adjunct to subroutine spline

oznot.f

Subroutine for computing initial ozone estimate using the 317.5 channel

ozonev8.f
: Subroutine for computing best ozone using first guess and a priori

pintrp.f
: Function for performing Lagrange interpolation in pressure

prfind.f
: Subroutine for computing table indices

prflecv8.f
: Subroutine for initializing ground reflectivity to 15% and cloud reflectivity to 80%

profile.f
: Sub-driver for profile ozone retrieval

rdatarv8.f
: Subroutine for reading one record of data from the V6 PMF file

rdcons.f
: Subroutine for reading in instrument dependent constants

rdsbtbl.f
: Subroutine for reading in tables for calculation of forward model quantities: N-value, and dN / dX_{msr}, the set of 21 standard profiles, and the a priori ozone and temperature profiles

reflecv8.f
: Subroutine for computing cloud fraction and reflectivity for a given (331 nm or 340 nm) wavelength

reflp.f
: Subroutine for computing cloud fraction and reflectivity for photometer

reflpmn.f
: Subroutine for using monochromator reflectivity and cloud fraction to replace the lost CCR

reflpzz.f
: Subroutine for computing cloud fraction and reflectivity for two monochromator channels and interpolate to emulate CCR reflectivity then calculate adjustments to monochromator reflectivity for each FOV

resadj.f
: Subroutine for adjusting residues after calculation of best ozone

residue.f
: Subroutine for computing residues for each of the multiply scattered wavelengths

rfnams.f
: Subroutine for reading and storing a list of input/output file names

sbdndx.f
: Subroutine for performing table lookups for computation of dN/dX for a given table pressure, profile, and zenith angle

scaninv8.f
: Subroutine for performing initialization tasks for a single SBUV retrieval

seterr.f
: Subroutine for setting error flags

splinev8.f
: Subroutine for performing spline interpolation

start.f
: Subroutine for performing job initialization tasks

stnprfv8.f
: Subroutine for determining a standard profile associated with a specific total ozone amount and a single latitude band

terminv8.f
: Subroutine for terminating SBUV daily processing

terpv8.f
: Function for performing linear interpolation

totalv8.f
: Version 8 driver for processing one scan of data from the level 1 SBUV record and computing the total ozone used as first guess in the profiling algorithm

update.f
: Subroutine for updating ozone profile using C. Rodgers' method

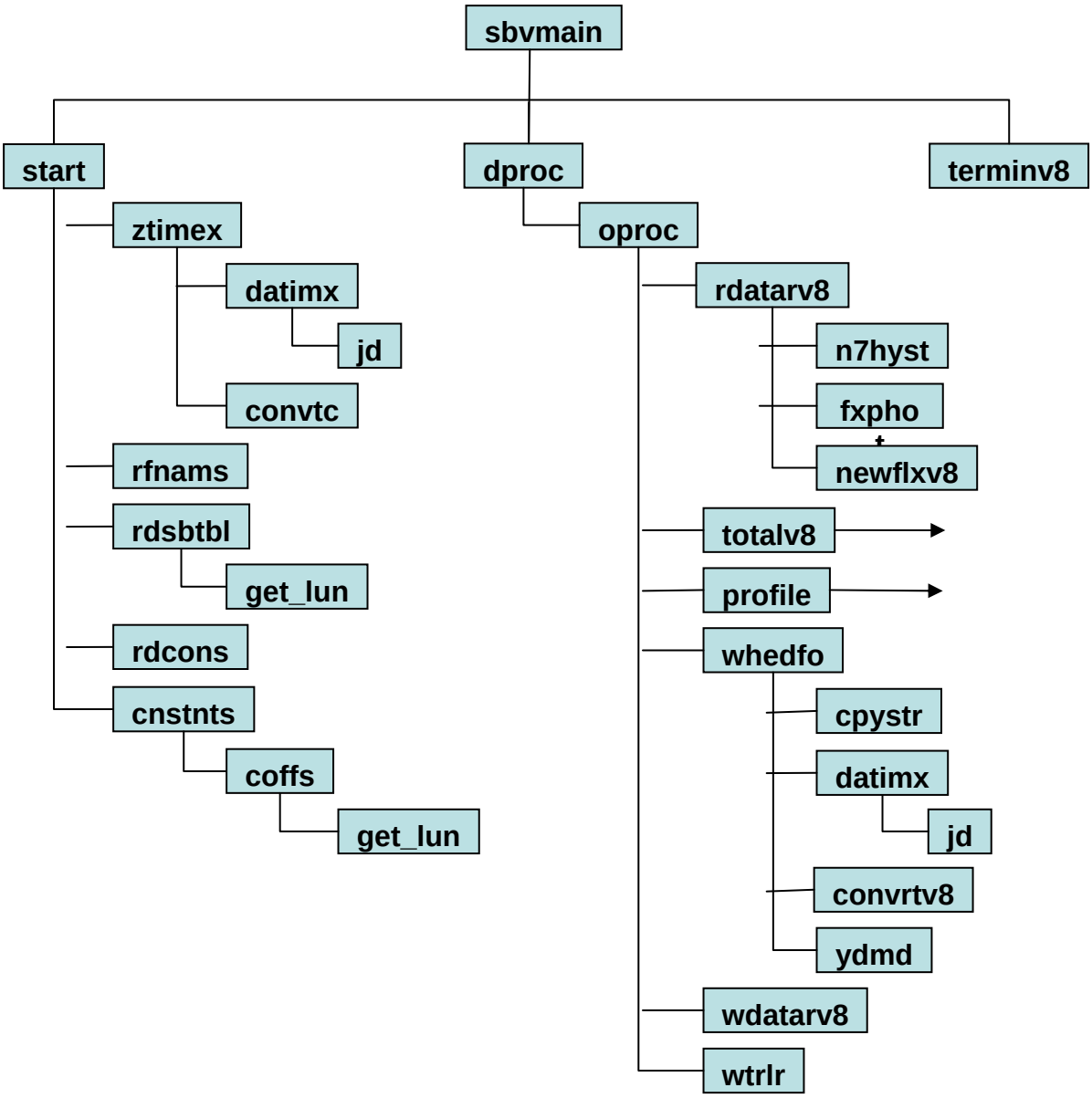
wdatarv8.f
: Subroutine for writing out a new data record to the output file

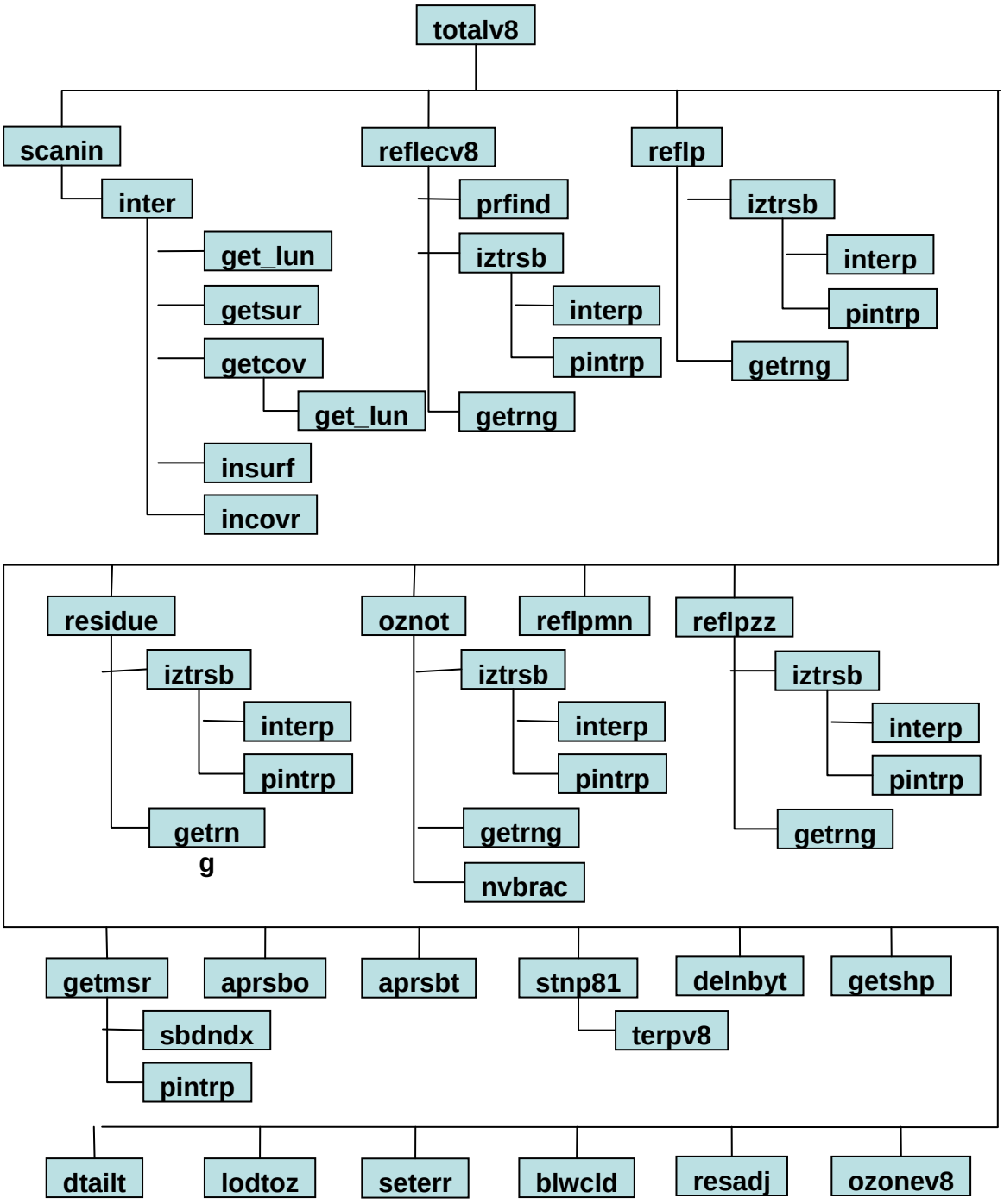
whedfo.f
: Subroutine for writing out a header record to the output file

wtrlrv8.f
: Subroutine for writing out a daily summary record to the output file

ydmd.f
: Subroutine for computing month and day of month from year and day of year

ztimex.f
: Subroutine for converting local system date and time into character string in the format of DOW MON DD, YYYY HH.MM.SS


sbvmain
start
dproc
terminv8
ztimex
datimx
jd
convtc
rfnams
rdsbtbl
get_lun
rdcons
cnstnts
coffs
get_lun
oproc
rdatarv8
n7hyst
fxpho
newflxv8
totalv8
profile
whedfo
cpystr
datimx
jd
convrtv8
ydmd
wdatarv8
wtrlr


totalv8
scanin
inter
get_lun
getsur
getcov
get_lun
insurf
incovr
reflecv8
prfind
iztrsb
interp
pintrp
getrng
reflp
iztrsb
interp
pintrp
getrng
residue
iztrsb
interp
pintrp
getrn
g
oznot
iztrsb
interp
pintrp
getrng
nvbrac
reflpmn
reflpzz
iztrsb
interp
pintrp
getrng
getmsr
sbdndx
pintrp
aprsbo
aprsbt
stnp81
terpv8
delnbyt
getshp
dtailt
lodtoz
seterr
blwcld
resadj
ozonev8

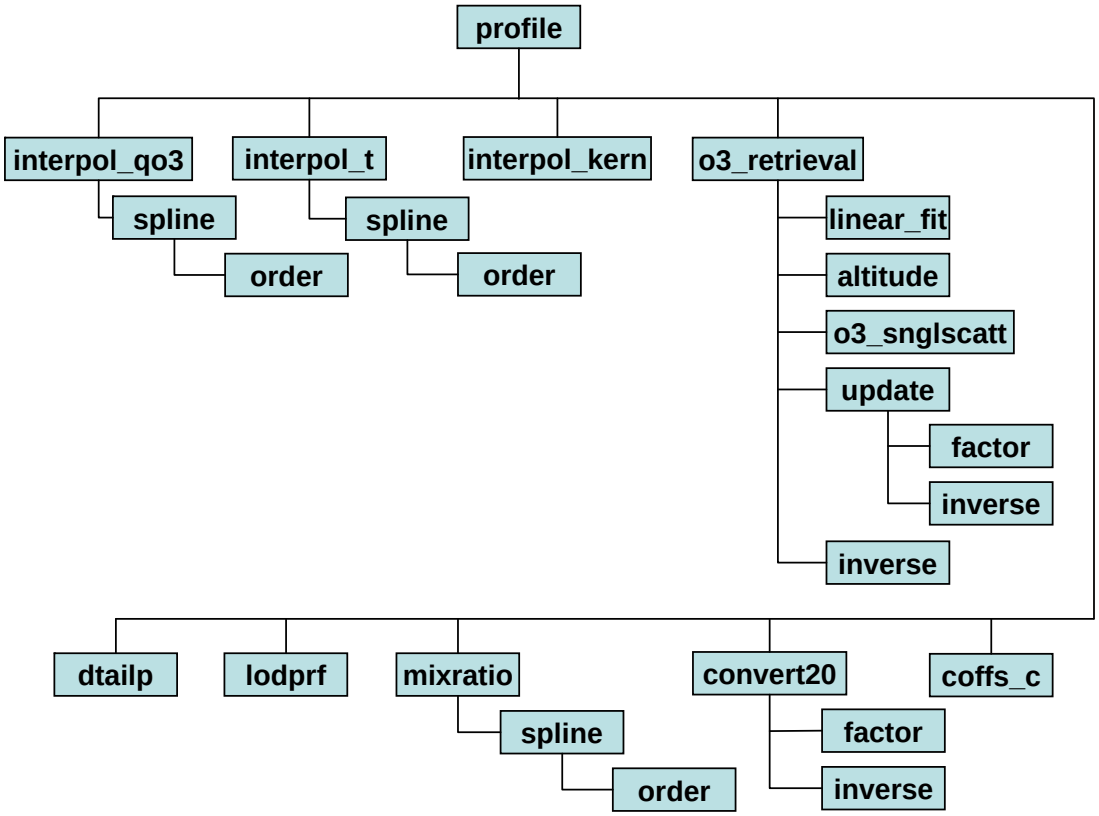


Figure 1 Hierarchical diagram of SBUV V8 program routines

5.2 SBUV Version 8 reader program

A small FORTRAN program **pmfv8reader.f** is developed for examining contents of SBUV Version 8 daily data output.

5.2.1 Make file

None

The executable can be made simply through FORTRAN compiler:

f77 pmfv8reader.f –o pmfv8reader.x

5.2.2 Main program

pmfv8reader.f

5.2.3 Sub programs

None

6 ALGORITHM

The SBUV Version 8 algorithm works as a follow-on processing system to the existing Version 6 algorithm and uses the Version 6 daily output as input. Functionally, the Version 8 processing system can be divided into three main parts: job initialization, total ozone retrieval and profile ozone retrieval. In the initialization part look-up tables for radiative transfer calculations and a priori ozone and temperature climatological data sets are read in and various parameters and coefficients are determined. This part only needs to be done once for a whole daily job. The total and profile ozone retrieving parts, on the other hand, are applied repeatedly on each and every data record through the input daily file. Step by step summaries for these three parts are given in the following sections. Some nonessential details may be omitted.

6.1 Initialization (START)

The subroutine START reads in control file and performs the following job initialization tasks:

i. Calls RDSBTBL to read in the ancillary data needed for the algorithm. This includes:

- N-value look-up tables as functions of 13 grating position, 4 pressure levels, 9 channels, 21 profiles and 10 solar zenith angles.
- Sensitivity dN / dX_{msr} look-up tables as functions of 4 pressure levels, 9 channels, 21 profiles, 10 solar zenith angles and 12 perturbation situations (unperturbed case followed by 11 Umkehr layers' perturbation).
- 21 standard profiles in 81 layers.
- Merged a priori 11-layer ozone profile climatology as function of 10 total ozone values, 18 latitude zones and 12 months.
- A priori 13-layer ozone profile climatology as function of 18 latitude zones and 12 months.
- A priori 13-layer temperature climatology as function of 18 latitude zones and 12 months.

ii. Calls RDCONS to read in instrument dependent constants from file CONST.***sat***, where ***sat*** is the satellite ID such as n16, n17 or n18.

iii. Calls CONSTNTS to initialize various parameters used in total and profile ozone retrievals. These include temperature dependent coefficients of absorption coefficient, denominators of Lagrangian coefficients for solar zenith angle and pressure interpolations, and radiance measurement error covariance matrix. The subroutine COFFS is called to assign component wavelengths within each SBUV channel, determines ozone absorption coefficients, Rayleigh scattering coefficients, and phase function coefficients at those wavelengths, and also determines the weights of the components for computation of the average channel intensity.

6.2 Total ozone retrieval (TOTAL)

The Version 8 SBUV operational system adopts the total column ozone retrieval algorithm developed for TOMS instruments (*Bhartia and Wellemeyer, 2004*), though necessary modifications have been made to fit SBUV instruments. The retrieval algorithm is a three-step process of successive estimation improvement. Step 1) the algorithm uses a pair of wavelengths 331 nm and 318 nm (340 nm and 331 nm for high solar zenith angle) to derive reflectivity and total ozone as a first guess linearization point. Step 2) adjustments due to seasonal and latitudinal variations in ozone and temperature profiles are made. Step 3) a simple procedure based on N-value residues is followed to correct errors due to aerosol, sea glint and profile shape deviation.

i. Calls SCANIN to perform initialization tasks for a single SBUV total and profile ozone retrievals:
   1) Sets latitude flag for table look-up: ILAT=1 [0, 30]; 2 (30, 60] or 3 (60, 90].

2) Calls subroutine INTER to read in 0.5 degree resolution terrain height pressure and surface category and 1 degree resolution snow-ice thickness and cloud top pressure fields. Interpolate these climatological data for the date, latitude and longitude of the observation.
3) Determines solar zenith angle for each wavelength channel.
4) Computes Lagrange coefficients for solar zenith angle interpolation.

ii. Detects wild radiances and skip the retrieval for bad observations.

iii. Assumes a nominal total ozone amount according to the latitude of observation: GUESOZ = 260 [0, 45]; 340 (45, 75] or 360 (75, 90].

iv. Calls REFLEC to calculate total ozone sensitivity at 340 nm wavelength channel:

1) Coverts the reflectivity channel N-value to albedos.
2) Initializes ground and cloud reflectivities to 0.15 and 0.80, respectively, at the first time routine call.
3) Perturbs measured reflectivity channel N-value by 0.2%.
4) Calls PRFIND to determine indices of bracketing standard ozone profiles based on total ozone estimation and latitude flag ILAT at the first time routine call.
5) For each of the bracketing ozone profiles, calls IZTRSB to perform look-up table pressure interpolations for i_0, T_r and s_b, and then calculates radiance from ground and cloud.
6) Computes terrain height corrections to the total ozone of bracketing profiles and bracketing interpolation coefficient.
7) Calls GETRNG to calculate Raman corrections
8) Interpolates logarithm of table parameters to current total ozone estimate and converts to ground and cloud radiances.
9) Calculates cloud fractions for unperturbed and perturbed cases. Assumes clear sky if snow is present.
10) Calculates reflectivities for both unperturbed and perturbed cases: if the cloud fraction is less than or equal to 0.0 or greater than or equal to 1.0, then the reflectivity is calculated using table parameters interpolated to the appropriate pressure level as

$$R = \frac{1}{\frac{T_r}{I_m - I_0} + s_b}$$

If the cloud fraction is greater than 0.0 and less than 1.0, then the reflectivity is

$$R = R_g + f_c * (R_c - R_g)$$

Total ozone sensitivity is calculated during the first call of the REFLEC routine.

v. Calls REFLEC to calculate cloud fraction, reflectivity and total ozone sensitivity at reflectivity channel (see above).

vi. For satellite other than NOAA-14, calls REFLP to calculate cloud fraction and reflectivity for each scene from the photometer measurements. Do reflectivity channel first as a reference. (For NOAA-14, where CCR measurements are not available, calls REFLPZZ or REFLPMN to emulate photometer measurements with monochromator channels observations. Detailed descriptions are omitted here).

1) Converts photometer N-value to albedos.
2) Initialize ground and cloud reflectivities with ones from monochromator reflectivity channel.
3) Perturbs measured photometer N-value at the reflectivity channel by 0.2%.
4) For each of the bracketing ozone profiles, calls IZTRSB to perform look-up table pressure interpolations for i_0, T_r and s_b, and then calculates radiance from ground and cloud.
5) Computes terrain height corrections to the total ozone of bracketing profiles and bracketing interpolation coefficient.
6) Calls GETRNG to calculate Raman corrections
7) Interpolates logarithm of table parameters to current total ozone estimate and converts to ground and cloud radiances
8) Calculates cloud fractions for unperturbed and perturbed cases. Assumes clear sky if snow is present.
9) Calculates reflectivities for both unperturbed and perturbed cases: if the cloud fraction is less than or equal to 0.0 or greater than or equal to 1.0, then the reflectivity is calculated using table parameters interpolated to the appropriate pressure level as

$$R = \frac{1}{\frac{T_r}{I_m - I_0} + s_b}$$

If the cloud fraction is greater than 0.0 and less than 1.0, then the reflectivity is

$$R = R_g + f_c * (R_c - R_g)$$

Total ozone sensitivity is calculated during the first call of the REFLEC routine.

10) Calculates residue and reflectivity sensitivity at reflectivity channel.
11) Recalculates surface model parameters based on CCR reflectivity

vii. Calls OZNOT to compute initial total ozone estimate using implied ozone channel (318 nm for regular and 331 nm for some extreme cases. See step viii.)

1) For the first call to OZNOT the iterative technique is used to find total ozone amounts that provide table N-value that bracket the measured N-value:
   a. For current ozone index, calls IZTRSB to calculate look-up table parameters i_0, T_r and s_b.
   b. Calls GETRNG to calculate Raman corrections.
   c. Determines calculated N-value for ozone wavelength channel
   d. Calls NVBRAC to determine whether the N-value for the current ozone index is above, below, or completes a bracketing of the measured N-value. If bracketing is not complete, repeats steps a-d until it is.
2) On subsequent calls to OZNOT, the bracketing profiles determined in the first call are retained even if extrapolation is required. For each of the two bracketing ozone profiles:
   a. Calls IZTRSB to calculate look-up table parameters i_0, T_r and s_b.
   b. Calls GETRNG to calculate Raman corrections.
   c. Determines calculated N-value for ozone wavelength channel
3) For all calls to OZNOT, estimates total ozone using the local slope of the N-Ω curve, $dN/d\Omega$ as
$$\Omega = \Omega_1 + (N_m - N_1)/(dN/d\Omega)$$

viii. After the first iteration, if
$$\frac{\frac{dN}{d\Omega_{318}} - \frac{dN}{d\Omega_{331}}}{\frac{dN}{d\Omega_{331}} - \frac{dN}{d\Omega_{340}}} < 1.25$$
then resets for computing total ozone using C pair (331 nm and 340 nm) in stead of B pair (318 nm and 331 nm) and start over from Step v.

ix. Iterates Step v-vii until the updated ozone estimate changes by less than 1.0 DU.

x. Calls RESIDUE to compute N-value residue for each of the multiple scattered wavelengths:
1) Calls IZTRSB to compute look-up table parameters i_0, T_r and s_b for bracketing profiles.
2) Calculates terrain adjusted bracketing total ozone amounts and ozone interpolation coefficient.
3) Calls GETRNG to calculate Raman corrections.
4) Calculates N-values for unperturbed and perturbed reflectivities.

5) Computes N-value residues, ozone sensitivities and reflectivity sensitivities.
6) Adjust observed N-values with Raman correction for profile retrieval

xi. Calls GETMSR to compute multiple scattered and surface reflected part of the Jacobian dN/dX for each of 11 Umkehr layers and each of 8 wavelengths using finite differencing between N-values interpolated from unperturbed and perturbed ozone tables.
1) Calls SBDNDX to perform table lookups for computation of dN/dX .
2) Calculates radiance from ground and cloud top for 12 layers (1 unperturbed followed by 11 with perturbations in 11 Umkehr layers) and 4 pressure levels.
3) Performs pressure interpolations through function PINTRP.
4) Combines ground and cloud radiances according to cloud fraction.
5) Calculates N-value perturbations and unperturbed N-values.
6) For perturbed layers, computes layer sensitivities for total, SS and MSR. Assigns MSR kernel as MSR sensitivities.
7) For the unperturbed case, computes multiple scattered and reflected part of total radiance.

xii. Calls APRSBO to determine 13-layer a priori ozone profile and integrated total ozone for the day and latitude.

xiii. Calls APRSBT to determine 13-layer a priori temperature profile for the day and latitude.

xiv. Defines 11-layer a priori temperature profile from 13-layer profile.

xv. Calls STNP81 to determine 81-layer first guess ozone profile for profile retrieval.

xvi. Calls DELNBYT to compute MSR dN due to temperature difference for MSR radiances each of 11 Umkehr layers and each of 8 wavelengths using chain rule relating dN/dT to dN/dX

xvii. Adjusts MSR radiances for a priori temperature profile.

xviii. Calls GETSHP to determine merged a priori ozone profile given total ozone, day of year and latitude.

xix. Calls DELNBYT to determine temperature adjustments again for dN/dX .

xx. Calls OZONE to calculate Step 2 total ozone, which improves the Step 1 total ozone based on a priori information about the seasonal and latitudinal variations in ozone profile shape and in temperature. The solution profile is adjusted due to the change in total ozone for estimation of column ozone below the cloud.

xxi. Calls RESADJ to adjust N-value residues after calculation of the Step 2 total ozone.

xxii. Computes Step 3 total ozone There are three possible corrections:

- A correction for the impact of tropospheric aerosol and sun glint with photometer residue at reflectivity channel.
- A correction with 313 nm wavelength N-value residue for the deviations of the true vertical distribution of ozone relative to the a priori climatology assumed in step 2
- A correction for non-zero N-value residues at ozone and reflectivity channels if C-pair is used.

xxiii. Calls BLWCLD to calculate column ozone below the cloud layer.
xxiv. Calls SETERR to set error flags.
xxv. Calls LODTOZ to load output buffer with total ozone data.

6.3 Ozone profile retrieval (PROFILE)

The inverse problem of deriving an ozone profile from SBUV measurements is mathematically ill-posed, some kind of constrain must be applied in order to achieve a physically reasonable solution. Based on the optimal estimation technique (*Rodgers, 1990*), the version 8 SBUV ozone profile retrieval algorithm combines SBUV measurements and a priori profile information to achieve the maximum likelihood estimate. The solution of the ozone profile is achieved through an iteration procedure following the equation:

$$q_{n+1} = q_a + S_a * K_n^T * (K_n * S_a * K_n^T + S_e)^{-1} * ((N - F(q_n)) + K_n * (q_n - q_a))$$

where $n+1$ and n refer to the current and previous iterations, q is the ozone profile vector, N is the N-value measurement vector, q_a is the a priori ozone profile vector, $F(q)$ is the forward model simulating N, K is the kernel (dF/dq), S_e and S_a are the measurement and a priori profile covariance matrices, respectively.

Ozone amounts in the top layers are assumed to be distributed according to the power law: q proportional to $p^{1/\sigma}$, with σ determined from several layers below the topmost layer.

The profile retrieval is carried out in fine (81) layers and converted to coarse (21) layers for output.

i. Calls INTERPOL_QO3 to interpolate a priori ozone profile from 13 Umkehr layers to 81 fine layers.
ii. Calls INTERPOL_T to interpolate a priori temperature profile from 13 Umkehr layers to 81 fine layer s
iii. Calls INTERPOL_KERN to interpolate multiple scattered kernels to 81 fine layers for clear and cloud cases separately, and then calculates the MSR kernel as a weighted average of clear and cloud kernels according to the cloud fraction.
iv. Defines a priori covariance matrix S_a.

v. Calls O3_RETRIEVAL to retrieve ozone profile using Rodgers method.

1) Calls O3_SNGLSCATT to run single scattering forward model to estimate radiance and kernel.
2) Computes single scattering radiances and kernels at cloud and terrain levels and combine according to the cloud fraction.
3) Converts single scattering radiance and kernel into N-value and dN / dX .
4) Adds multiple scattering contributions to N-value and kernel.
5) Defines aerosol related processing algorithm, i.e., includes measurements only from channels up to where the kernel for a layer near 100 hPa is maximum by assigning S_e for longer wavelength channels to a very large value.
6) Computes N-value residues and exit loop if desired limit is attained.
7) Calls UPDATE to update ozone profile using Rodgers' method:
   a. Computes $K_{aux} = K_n * S_a$.
   b. Computes $S_{aux} = K_{aux} * K_n{}^T + S_e$.
   c. Calls FACTOR to perform LU factorization of S_{aux} .
   d. Computes mismatches $diff = (N - F(q_n)) + K_n * (q_n - q_a)$.
   e. Calls INVERSE to compute $S^{-1}{}_{aux} * diff$
   f. Updates ozone profile $q_{n+1} = q_n + K_{aux}^T * S_{aux}^{-1} * diff$
8) Performs linear fit of logarithm of ozone in the top layers against pressure-scaled height to determine power law coefficient σ
9) Adjusts top layers ozone amounts according to the power law.
10) Integrates profile ozone to obtain total ozone.
11) Exits iteration if desired accuracy is met; otherwise repeats above procedures.
12) Calculates ozone layer partial pressure and mixing ratio.
13) Calls INVERSE to calculate the transpose of the Rodgers contribution function $D_n^T = S_{aux}^{-1} * K_{aux}$
14) Calculates absolute averaging kernel $AK = D_n * K_n$
15) Calculates solution covariance $S = S_a - D_n * K_{aux}$

vi. Calls CONVERT20 to convert fine (80)-layer retrieved, a priori, and first guess ozone amounts from Rodgers-type solution into corresponding coarse (20)-layer quantities. Also determines estimated measurement errors in 20-layer ozone and total ozone, 10-channel by 20-layer kernel and 20-layer averaging kernel in its relative (fractional) form.

vii. Calls MIXRATIO to determine ozone mixing ratio and its estimated percentage error at each of prescribed pressure levels.

viii. Calls LODPRF to load output buffer with retrieved ozone profile data.

7 MONITORING PROCEDURES

The Version 8 SBUV processing system is fully automated and no regular maintenance required. Pre-defined error messages with the location of the error will be sent to the OPUS operators via the "op_msg" utility in case of execution failure. Log files, which are saved in logs/ subdirectory for about a week, will provide more detailed job status information. In addition to operator monitoring, daily group monitoring is recommended for this product through monitoring two web pages (http://foehn-inter.nesdis.noaa.gov/PSB/OZONE/monitor_v8/sbuv_mon.html and http://foehn-inter.nesdis.noaa.gov/PSB/OZONE/monitor_v8/sbuv_graphs.html) and by checking the twice daily processing email (see section 7.1).

Besides "op_msg" and log files, a small program **pmfv8reader.x** is provided as a handy tool for checking up the content of a Version 8 SBUV output file. The usage of the program is:

>**./pmfv8reader.x < inpfile_pmfv8reader > out_pmfv8reader**

where **inpfile_pmfv8reader** is the input control file and **out_pmfv8reader** is the output file. Here is a sample of the input control file:

```
# PMF V8 data file name
/shared/DEV/apps/ozone/ver8/data/ORBOPM4OP_1ORB_V8
# Header record I (1: yes  0: no)
0
# Header Record II (1: yes  0: no)
0
# Data Records (1: yes  0: no)
1
# Range of Data Record Numbers (*: all)
*
# List of Words in Data Records (*: all)
2,36,40,184
# Trailer Record (1: yes  0: no)
0
```

And here is a fragment of the corresponding output:

```
 ============
 Data Records
 ============

Rec. No.   Word 0002         Word 0036         Word 0040         Word 0184

  0001   29013.00000       269.0666809       229.6900330       284.9313965
  0002   29045.00000       267.0005493       239.8625641       278.6076050
  0003   29077.00000       262.7921448       244.0287170       268.9415588
  0004   29109.00000       268.1286926       250.5328064       268.9781189
  0005   29141.00000       277.3322449       260.2087708       273.7097473
  0006   29173.00000       300.6136169       281.4735718       294.8785095
```

```
0007   29205.00000       303.5533447       289.1433105       299.3996887
0008   29237.00000       305.0122070       288.1808777       297.1367188
0009   29269.00000       306.8438416       291.6522217       298.0161133
0010   29301.00000       299.8998718       287.7087402       291.5357056
0011   29333.00000       297.1053162       295.8614807       298.3782349
0012   29365.00000       294.9260864       294.0593262       296.7482910
0013   29397.00000       288.1501465       287.6651611       292.4661865
0014   29429.00000       288.1649780       286.3543701       289.5664978
0015   29461.00000       296.5725098       297.2887268       293.1799927
0016   29493.00000       281.5940247       282.0793152       283.0365906
0017   29525.00000       287.2089233       287.9460449       289.4122620
0018   29557.00000       270.6221313       270.9651184       269.1062012
0019   29589.00000       268.9542847       269.6148376       268.8035278
0020   29621.00000       269.7262573       272.9600220       270.0823059
```

An IDL program **pmfv8compare.pro** is provided as a product validation tool to compare any 2 daily PMFs (pmf[124]tody, pmf[124]yest) and to create zonal average comparison plots (20 layers) for SBUV/2 V8 on Balmy.

A script ($ANCHOR/apps/ozone/chk_v8_processing_mail.pl) has been set up on the operational system to create an email sent to members of the Ozone group with all information on binary and BUFR product production twice daily (run from system cron at 12 and 18 UTC). This prevents having to access the operational system as well as makes monitoring much quicker and easier. In the event that this script does not run or email goes down, the manual procedures are listed in sections 7.1 and 7.2 below.

7.1 Version 8 binary product monitoring

After the OOPS daily system has run on the V8 operational machine, sign on the operational machine and go to $ANCHOR/data/POES/ozone. Look for the daily binary V8 products:

ozone.sbuv.YYYYMMDD.nXXv8.oper

These should have today's creation date and the "YYYYMMDD" should show yesterday's date. These files will normally be about 10,400,000 bytes in size (smaller for a NOAA-17 Tuesday file, due to sweep mode for the first 4 orbits).

7.2 Version 8 BUFR product monitoring

In the early morning, as part of the normal Ozone monitoring, sign on the V8 operational machine, and go to $ANCHOR/data/POES/ozone. Look for the daily BUFR V8 products:

PRD.OZONE.PMFV8.Nxx.BUFR

These should have today's creation date. They will normally be about 1,300,000 bytes in size (smaller for a NOAA-17 Tuesday file, due to sweep mode for the first 4 orbits).

At the same time, check for the V8 orbital BUFR products, in the same directory:

PRD.OZONE.PMFV8.Nxx.D-----.S----.E----.B-----

You should see about 2 or 3 of these (depending on how early you do this) for each satellite, with today's date in the name. (If it's Tuesday, for NOAA-17, you may only see none or none of these due to sweep mode for the first 4 orbits). These files will normally be about 100,000 bytes in size.

References

Bhartia, P.K. and Wellenmeryer, C. W., TOMS-V8 Total O_3 Algorithm., *TOMS V8 ATBD (http://toms.gsfc.nasa.gov/version8/v8toms_atbd.pdf),* Chapter 2, NASA, 2004.

Rodgers, C. D., The Characterization and Error Analysis of Profiles Retrieved from Remote Sounding Measurements, *J. Geophys. Res.*, **95**, 5587-5595, 1990.