

**NAME**

avhArea – determines the file extents of a NESDIS Level–1B data file which covers an image map.

**SYNOPSIS**

**avhArea** [OPTIONS] mapname avhrr.1b

**DESCRIPTION**

Determines the file extents (start/stop pixel/line) of a NESDIS Level–1B file (still in sensor projection) that covers a map.

**AvhArea** begins by reading in the map from the mapfile. If the file can not be opened or the named map is not in the file, a diagnostic is printed and the program will exit.

Next, the AVHRR file is opened and the navigation information initialized. If unable to open the file or get the navigation information from the file, the program will print a diagnostic and exit.

Once the navigation has been set, **avhArea** reads in every 50th scan line, and using every 50th sample, determines if that point falls within the desired map. From this, the smallest box (modulo 50) that will cover the box will be determined. These file extents will be printed to the screen. These are printed to stdout: starting pixel, space, ending pixel, space, starting line, space, ending line.

If the entire file covers the image map, then "Complete coverage" will be written to stdout. If no part of the file covers the image map, then "No coverage" will be written to stdout.

**OPTIONS**

- f type Set the format type of the input AVHRR file. Valid responds are 1 for NESDIS Level–1B format (NOAA–6 to NOAA–14), 2 for NESDIS KLM format (NOAA–15 to NOAA–16), 3 for NESDIS Level–0 format
- l Don't output start/stop line locations
- M mapFile Use the given map file to find mapName. Defaults to \$AUTO\_DATA/maps.hdf
- n # Set the number of points across and down the image used to search for data coverage. The default is 41 points which yields a control point roughly every 50 pixels and lines. For small regions – that might “fall between the cracks” – this can be set to a higher number to create a finer grid.
- p Don't output start/stop pixel locations
- help Display program help.
- version Display program name version and time of compilation.

**ENVIRONMENTAL VARIABLES**

AUTO\_DATA

The location of the APS data directory.

**EXAMPLES**

The examples below show the same input file run against two different geographical areas. The last example shows the result of trying to use an invalid input.

```
$ avhArea GulfOfMexico noaa-14.970205200124.lac
1 1638 1021 2551
$ avhArea -p -M my_maps.hdf GulfOfMexico noaa-14.970205200124.lac
1021 2551
$ avhArea EastSea noaa-14.970205200124.lac
No coverage
$ avhArea Junk noaa-14.970205200124.lac
Map (Junk) does not exist in file ($AUTO_DATA/maps.hdf).
$ echo $?
1
```

**SEE ALSO**

**avhIngest(1)**

**NAME**

avhClouds – produces a bit image of tests used to determine cloud contaminated pixels

**SYNOPSIS**

**avhClouds** [-1 albedo\_ch1.spk] [-2 albedo\_ch2.spk] [-3 btemp3.spk] [-5 btemp5.spk] l2flags.spk  
btemp\_ch4.spk

**DESCRIPTION**

**avhClouds** will determine the cloud cover using several different tests based on the cloud detection algorithm of Saunder and Kriebel (1988). These tests require brightness temperatures from channels 3, 4, and 5. If the scene is a daytime scene, then the albedo from channels 1 and 2 are also required. Each test will set a bit in the output file. The tests are divided into day and night tests as well as land/sea/coast test.

The input/output file (l2flags.spk) will be a 32-bit image with bits 17 to 26 being set. This file should be the output from the **avhIngest** (1) program as some bits set by that program (for example, land flag) are used by this program.

These tests are performed on a 3x3 box with all edge pixels being marked as cloud automatically. The input l2flags file is read in an used to determine on a pixel by pixel basis some of the characteristics needed by the algorithm for that pixel.

If all the surrounding pixels are determined to be land, the center pixel is marked as land. If all the surrounding pixels are determined to be sea (not land), the center pixel is marked as sea. Otherwise, the center pixel is marked as coast. The program **avhIngest**(1) uses a landmask file to set the LAND flag (bit 2).

For the day or night characteristic of a pixel, **avhClouds**(1) will set all pixels to day or night, if upon initialization, the code determines that all pixels are day or night. This is determined by examining the Sun elevation for the four corner points and center of the input image. If all five points are defined as day, then all pixels are marked as day. If all four points are defined as night, then all pixels are marked as night. A Sun elevation greater than 15 degrees implies day and less than or equal to 15 implies night.

If this gross day/night check fails, then the DAY\_TIME flag set in the l2flags.spk file by **avhIngest**(1) will be consulted. In a similiar manner to land/sea/coast, the DAY\_TIME flag (bit 3) for surrounding pixels is examined to determine if the center pixel is DAY or NIGHT. If all surrounding pixels are defined as DAY\_TIME, then the center pixel is marked as DAY. If all surrounding pixels are defined as NIGHT\_TIME, then the center pixel is marked as NIGHT. The

**Minimum Channel 5 Temperature**

If channel 5 temperature is too low, it is assumed that these are cloud-top temperatures. All values over sea that are less than 273.15 degrees Kelvin (0 degrees Celsius) are flagged as clouds. For land and coast pixels, 263.15 degrees Kelvin (-10 degrees Celsius) are used. For land/coast pixels, bit 17 (CLD\_LAND\_GROSS\_CLOUD\_CHECK) is set. For sea pixels, bit 18 (CLD\_SEA\_GROSS\_CLOUD\_CHECK) is set.

**Channel 4 Spatial Coherence Test**

The standard deviation in a 3x3 box surrounding the pixel in question for channel 4 is determined. If the deviation for sea (day or night) or land (night only) are greater than the thresholds (0.25 and 1.0), then the pixel is masked as cloudy. For land pixels, bit 19 (CLD\_LAND\_SPATIAL\_COHERENCE) is set. For sea pixels, bit 20 (CLD\_SEA\_SPATIAL\_COHERENCE) is set.

**NIS/VIS Test**

During the day, the ratio of the NIR (channel 2) over VIS (channel 1) may indicate clouds. For land pixels, bit 21 (CLD\_LAND\_NIR\_VIS) is set. For sea pixels, bit 22 (CLD\_SEA\_NIR\_VIS) is set.

**Channel 2 Spatial Coherence Test**

For a day pixel, the standard deviation in a 3x3 box surrounding the pixel in question of channel 2 is determined. If the deviation for sea (day or night) or land (night only) are greater than the thresholds (0.25 and 1.0), then the pixel is masked as cloudy. For land pixels, bit 23 (CLD\_SEA\_DAY\_SPATIAL\_COHERENCE) is set.

**Low Fog and Uniform Stratus Check**

For a night pixel, the channel 4 - channel 3 difference greater than a given threshold indicates low fog or uniform stratus clouds. For these pixels, bit 24 (CLD\_LOW\_FOG\_UNIFORM\_STRATUS) is set.

**Medium and High Cloud Check**

If the pixel in question is a night pixel, then the difference in ch3 - ch5 will indicate medium and/or high clouds. For these pixels, bit 25 (CLD\_MEDIUM\_HIGH\_CLOUD) is set.

**Thin cirrus cloud check**

If the pixel in question is a night pixel, then the difference in ch4 - ch5 is consulted against a 2-D table that will vary based on ch4 and the satellite zenith angle. If the channel 4-5 difference is greater than that threshold, the pixel is flagged as clouds and bit 26 (CLD\_THIN\_CIRRUS) is set.

**OPTIONS**

-v      Verbose mode.

--help    Display program help.

--version

        Display program name version and time of compilation.

**REFERENCE**

Saunders, R.W. and Kriebel, K.T., 1988, *An improved method for detecting clear sky and cloudy radiances from AVHRR data* Int. Journal of Remote Sensing, 1988, Vol 9, No. 1, pp 123-150.

**SEE ALSO**

**avhIngest(1)**

**NAME**

avhDump – dumps AVHRR video data from a NESDIS Level-1B file.

**SYNOPSIS**

**avhDump** -c *n* [-f 1] lv11b chan.bin

**DESCRIPTION**

**avhDump** is used to dump AVHRR video data from a NESDIS Level-1B file. Currently, it only supports the dumping of AVHRR video data which must be selected with the -c option.

The output file is written as 16-bit integers in a flat binary format which has 2048 (LAC/HRPT) or 409 (GAC) columns by *n* number of rows. The number of columns and rows are printed by this program. Also, if the user knows the file type, the number of rows is therefore known and the number of columns can be computed by dividing the size of the file by 2 times the number of columns.

**OPTIONS**

- c *n*     Select a channel to output. Must be between 1 and 5.
  
- f *n*     Select format of the input file. Use 1 for NESDIS Level-1B, 2 for NESDIS KLM Level-1B, and 3 for NESDIS HRPT Level-0 format.
  
- v        Turn on verbose mode.
  
- help    Display program help.
  
- version        Display program name version and time of compilation.

**SEE ALSO**

**avhInfo(1)**, **avhScan(1)**, **avhImage(1)**

**NAME**

avhImage – creates a simple graphics image file from a NESDIS Level-1B file.

**SYNOPSIS**

**avhImage** [-c n] [-f] lvl1b image.[format]

**DESCRIPTION**

**avhImage** is used to make a quick image from an AVHRR data file. By default, the program will read channel four counts (10-bits) and shift two bits to the right two for an output of the top 8 bits. If the input file is a LAC or HRPT file every fourth line and sample are written. If the input file is a GAC file, then every line/sample are written to the file.

The output file may be written as a PNM grayscale raw file, TIFF, PNG, or SGI RGB file depending on the compilation of the program. The available types may be obtained by running `avhImage --help`. The type is selected based on the extension of the output file. See **OPTIONS**.

**OPTIONS**

- c n     Select a channel to output. Must be between 1 and 5.
  
- f       Full output. If the input file is a LAC/HRPT file, this option writes the entire image to a file.
  
- t type   Set the format type of the output image file. Valid responds are based on the compilation of the program. The `type` given is given as an image “extension”. For example, `tiff`.
  
- T type   Set the format type of the input AVHRR file. Valid responds are 1 for NESDIS Level-1B format (NOAA-6 to NOAA-14), 2 for NESDIS KLM format (NOAA-15 to NOAA-16), 3 for NESDIS Level-0 format
  
- help   Display program help.
  
- version     Display program name version and time of compilation.

**EXAMPLES**

To determine which formats are available, we run `avhImage` using the `--help` option.

```
$ avhImage --help
```

```
Usage: avhImage [OPTION] INPUT OUTPUT
```

```
Creates a subsampled grayscale image of
channel 4 from the AVHRR data stored in a
NESDIS Level-1B file, KLM Level-1B file, or
an HRPT Level-0 file.
```

**OPTIONS**

- c n       use channel n for output
- f       do full output
- t type    set output file type, see below
- T type    set input file type
  - 1=NESDIS Level-1B file

```
2=NESDIS KLM Level-1B file
3=NESDIS Level-0 file
--help      this output
--version   version information
```

INPUT must be a one of the formats above  
OUTPUT is a graphics file specified by its  
extension. It can one of the following:

pnm	for PNM Format
png	for PNG Format
tiff	for TIFF Format

Since this version allows us to create PNG files, we create a quick-look PNG image of our input file using the following command:

```
$ avhImage NSS.HRPT.NL.D02142.S2044.E2056.B0858282.MO chan4.png
```

### SEE ALSO

**avhInfo(1), avhIngest(1)**

**NAME**

avhInfo – queries information about a NESDIS Level-1B file(s).

**SYNOPSIS**

**avhInfo** file1 file2 file3 .. ..

– or –

**avhInfo** option file

**DESCRIPTION**

Run without options, **avhInfo** will write a report for each input file indicating satellite id, data type, etc. It may also be run with a single option and print the input file(s) value for that option. The first method is intended for interactive use at the shell prompt and the second method is intended for use within a shell program.

**OPTIONS**

- year 4-digit year of input file.
- doy Day of year of input file.
- dsn The NOAA defined Data Set Name from the TBM/ARS header.
- month 3-character month of input file. Months are 'jan', 'feb', 'mar', 'apr', 'may', 'jun', 'jul', 'aug', 'sep', 'oct', 'nov', 'dec'
- time 4-digit time (HHMM) of input file.
- type 1-digit code for datatype, where: 1=LAC, 2=GAC, 3=HRPT
- sat 3-character satellite name. Names are 't-n', 'n06', 'n07', 'n08', 'n09', 'n10', 'n11', 'n12', 'n14', 'n15', or 'n16'.
- sat\_code 3-character satellite name. Names are 'TN', 'NA', 'NC', 'NE', 'NF', 'NG', 'ND', 'NJ', 'NK', or 'NL'.
- name Generate a file name in the following format as SSS.YYYY.MMDD.HHMM.T, where T is 'l' for LAC, 'h' is for HRPT and 'g' is for GAC.
- help Display program help.
- version Display program name version and time of compilation.

**EXAMPLES**

Here is how a Bourne shell script function might use **avhInfo** to set the name of the output filenames:

```
set_name ()
```

```
{
  sat=`avhInfo -sat $1`
  yr=`avhInfo -year $1`
  jday=`avhInfo -doy $1`
  time=`avhInfo -time $1`
  file=$sat.$yr$jday.$time.11b
}
```

Here is an interactive use of **avhInfo**:

```
$ avhInfo *.hrpt
Filename:n950622.2233.n14.hrpt
Format:NESDIS
Starting Time:06/22/95 22:32, 173
Ending Time:06/22/95 22:44, 173
Satellite:n14
Datatype:LAC (Local Area Coverage)
Total Scans:4012 with no gaps

Filename:n950623.0256.n12.hrpt
Format:NESDIS
Starting Time:06/23/95 02:56, 174
Ending Time:06/23/95 03:08, 174
Satellite:n12
Datatype:LAC (Local Area Coverage)
Total Scans:4331 with no gaps
```

## SEE ALSO

**avhImage(1), avhDump(1)**

**NAME**

avhIngest – produces calibrated albedo and brightness temperature image from AVHRR data.

**SYNOPSIS**

**avhIngest** avhrr.1b ofile

**DESCRIPTION**

This program reads the AVHRR data from a NESDIS Level-1B file, NESDIS KLM Level-1B, or NESDIS Level-0 or TeraScan “hrpt” Level-0 file and writes out calibrated data to a modified PC-SEAPAK file format. The albedo data is calibrated using the slope and intercept specified in the input file. The brightness temperatures are calibrated using an algorithm by Brown or by the calibration coefficients specified in the input file for KLM.

The user must select the output channels and the scaling method. The channel is selected by indicating its channel number as an option, after which the user must supply one or two arguments to define how the data will be written to the output files. For KLM, Channel 3A is designated as channel 6 for this program; Channel 3B retains the channel 3 value. The user has four options. If the first argument is an ‘R’, then it is the only required argument and the calibrated data for that channel will be written out as floating point data. If the first argument is a ‘C’, then it is also the only required argument and the (10-bit) digital counts will be written out as a 16-bit data file. If the first argument is an ‘S’, then it must be followed by two additional arguments which will hold the slope and intercept used to convert the calibrated data into signed 16-bit integers. If the first argument is neither ‘R’, ‘C’, or ‘S’, then it must be the slope and intercept needed to convert the calibrated data to unsigned 8-bit integers. The slope and intercept are used as:  $\text{gray} = (\text{data} - \text{intercept}) * \text{slope}$ .

Here are some examples:

```
-1 R           Output channel 1 albedo as floating point
-2 C           Output the 10-bit counts for channel 2
-3 10 275     Output channel 3 (in Kelvins) as 8-bit image with a slope of 20 and intercept of 0.
-3 S 200 200  Output channel 3 (in Kelvins) as 16-bit image with a slope of 200 and intercept of 200.
-6 R           Output channel 3A albedo as floating point
```

In the examples above, the third line will create an image such that the each change in the gray value will represent a change of 0.1 degrees Kelvin. The data will range from 275 degrees Kelvin to 300.5 degrees Kelvin. Note, this image is a byte image and the integers range from 0 to 255.

For the fourth example, which is the default scaling set up by **avhScripts(1)**, each change in the integer value will represent a change of 0.005 degrees Kelvin. The data will range from 36.165 degrees Kelvin to 363.385 degrees Kelvin. Note, this image is a signed 16-bit integer image.

The argument ofile is the base name of the output file. If ofile is set to ‘avhrr.spk’ and all outputs are requested then the following files will be created:

```
avhrr.avh     avhIngest log file.
avhrr.ctl     Navigation file
avhrrL2.spk   Level-2 Flags image
avhrr1.spk    Percent albedo image for visible channel 1
avhrr2.spk    Percent albedo image for visible channel 2
avhrr3.spk    Brightness energy temperature (deg Kelvin) image for IR channel 3.
avhrr4.spk    Brightness energy temperature (deg Kelvin) image for IR channel 4.
avhrr5.spk    Brightness energy temperature (deg Kelvin) image for IR channel 5.
avhrr6.spk    Percent albedo image for visible channel 3A.
```

Note, if the user

## OPTIONS

-angle *angle*

If *angle* is defined then it is used to reduce the swath of the input image. This option can only be used for LAC/HRPT files and will calculate the number of pixels to reduce the image. It can be used to prevent the large pixels from the edge of the swath to be output. If *angle* is less than 1.1, then it is assumed to be given in radians. Otherwise it is given in degrees. A negative angle will be converted to a positive one.

-full Do the entire image. Default.

-box *isp iep isl iel isp iel irp irl*

These set up the subsection of the NESDIS Level-1B file to extract the data from.

*isp* is the starting pixel number (1 to 2048).

*iep* is the ending pixel number (1 to 2048, greater than *isp*).

*isl* is the starting line (1 to *n*).

*iel* is the ending line (1 to *n*, greater than *isl*).

*irp* is the pixel subsampling factor.

*irl* is the line subsampling factor.

-v Verbose output.

--help Display program help.

--version

Display program name version and time of compilation.

## EXAMPLES

This example will produce floating point outputs of albedo channels 1 and 2 for input to the turbidity program **avhTurbid(1)**.

```
$ avhIngest -1 R -2 R NJ1999360192854.L1B_HNAV avhrr.spk
$ ls -l avhrr*
-rw-r----- 1 aps      aps           1837 Jan  6 10:45 avhrr.avh
-rw-r----- 1 aps      aps          252020 Jan  6 10:45 avhrr.ctl
-rw-r----- 1 aps      aps          16671232 Jan  6 10:45 avhrr1.spk
-rw-r----- 1 aps      aps          16671232 Jan  6 10:45 avhrr2.spk
$ more *.avh
avhIngest 2.4 12/07/1999.
```

### Input Parameter Information

#### Information from Header

Satellite ID: 9

Start Time (YY DDD HH:MM:SS) : 1999 360 19:29:18

Start Time (YY DDD HH:MM:SS) : 1999 360 19:34:57

Datatype: LAC

Tip Source: Embedded TIP

Scene start/end samples numbers : 1/2048 2048

Scene start/end scan line numbers : 1/2035 2035

Sample/scan line reduction factors: 1/1

```

Start loop.
Processing routine:  brown version 1.0
    Brown, et al, JGR Vol 98, c10:18257-68
    noaa-14 coefficients
New Channel 1 slope/int 0.108100 -3.864800
New Channel 2 slope/int 0.109000 -3.674900

Information about Ouput Files Created
-----
Percent albedo image for channel 1: avhrr1.spk
Percent albedo image for channel 2: avhrr2.spk
Brightness energy temperature (deg C) image for channel 3: not requested.
Brightness energy temperature (deg C) image for channel 4: not requested.
Brightness energy temperature (deg C) image for channel 5: not requested.

Input slope(s)/intercept(s) used in: DATA = (GRAY/SLOPE) + INTCP
channel 1 (A1): Data output as reals.
channel 2 (A2): Data output as reals.

Minimum and Maximum values for each channel processed:
channel 1: -3.864800 105.316197
channel 2: -3.674900 99.657097

Number of data gap scan lines encountered: 0
Output image sizes (pixels/lines): 2048/2035
Start/end pixels of data in image: 0/2047
Start/end lines of data in image: 0/2034

Navigation Information for Output Images
-----
Control point file name: avhrr.ct1
Corner latitudes/longitudes:
Top    : 30.440039/-45.958789    25.621875/-76.079688
Bottom: 25.822852/-75.333203    44.229687/-86.860938
(Satellite descending.)
Minimum/maximum latitudes: 25.621875/50.522852
Minimum/maximum longitudes: -86.860938/-45.958789

```

**CAVEATS**

Presently this software can only handle NOAA satellites 14 and 12.

**REFERENCES**

- Kidwell, K, 1999. *KLM User's Guide*. NCDC/NESDIS, National Climatic Data Center, Washington, D.C.
- Kidwell, K, 1991. *NOAA Polar Orbiter User's Guide*. NCDC/NESDIS, National Climatic Data Center, Washington, D.C.
- Brown, J. W., O. B. Brown, and R. H. Evans, 1993. *Calibration of AVHRR infrared channels: a new approach to non-linear correction*. J. Geophys. Res. 98 (NC10), 18257-18268.

**SEE ALSO**

**avhInfo(1)**, **avhScan(1)**, **avhImage(1)**

**NAME**

avhSST – produces sea surface temperature images from brightness temperature images

**SYNOPSIS**

**avhSST** btemp\_ch3.spk btemp\_ch4.spk btemp\_ch5.spk sst.spk

**DESCRIPTION**

This program will read brightness temperature data using channels 3, 4, and 5 produced by the program **avhIngest** and generate an image file of sea surface temperature estimates. The algorithm is automatically selected based on satellite and time. The algorithms are from NOAA.

SATELLITE	DAY ALGORITHM	NIGHT ALGORITHM
NOAA-16	MCSST Split (4/5)	MCSST Split (4/5)
NOAA-15	NLSST Split (4/5)	NLSST Split (4/5)
NOAA-14	NLSST Split (4/5)	NLSST Triple (3/4/5)
NOAA-12	NLSST Split (4/5)	NLSST Split (4/5)
NOAA-11	NLSST Split (4/5)	NLSST Triple (3/4/5)

The day or night algorithm is based on whether the elevation of the sun is greater than 20 degrees at the center of the image.

**OPTIONS**

- C      Sets the output temperature scale to Celsius.
  
- d      Force use of day algorithm.
  
- F      Sets the output temperature scale to Fahrenheit.
  
- n      Force use of night algorithm.
  
- s slope,intp  
         Defines the slope and intercept used for the output SST image. By default the slope is set to 0.001 and the intercept to 20.0. Using the default type of 2 (short) this will give a data range value of -12.767 degrees C to 52.767 degrees C.
  
- t otype Sets the output type for the SST image. The default is 2 which will produce 16-bit signed integers. Options are: 1 (8-bit unsigned), 2 (16-bit signed), 3 (32-bit signed), 4 (32-bit floating point), 5 (64-bit floating point). If type 4 or 5 are selected the slope and intercept are automatically changed to 1.0 and 0.0, respectively. To force another slope and intercept value, use the -s option *after* the -t option.
  
- v      Verbose output.
  
- help    Display program help.
  
- version  
         Display program name version and time of compilation.

**REFERENCES**

The coefficients for the various SST algorithms were obtained by NOAASIS. <http://noaa-sis.noaa.gov/NOAASIS/ml/sst.html>

**SEE ALSO**

**avhIngest(1)**

**NAME**

avhScan – dumps scan line information from a NESDIS Level-1B file

**SYNOPSIS**

avhScan [-f type] [-n num] file target

**DESCRIPTION**

This program reads a NESDIS Level-1B file and writes to stdout information for each scan line based on the user specified target. The target can be one of the following: *vis-cal*, *latlon*, *prt*, *solar*, *telem*, *time*.

If the user selects *vis-cal*, then the calibration values (slope/intercept) for all five channels will be dumped.

If target *latlon* is selected, then **avhScan** will dump the scan line number, ascending/descending flag, and the number of valid lat/lon pairs followed by the 1st, 26th, and 51st lat/lon pair.

If *prt* is selected, then **avhScan** will dump the three prt counts stored in each scan line. Generally, these are duplicates of each other and contain the multiplexed PRT counts for each of the four PRTs. This dump, however, does not de-multiplex them.

If target *solar* is selected, then **avhScan** will dump the scan line number, and the number of meaningful solar zenith angles followed by the 1st, 26th, and 51st solar zenith angle.

Target *telem* will cause **avhScan** to dump the the five ramp calibration counts, the three PRT counts, the ten internal target view counts for all three channels, and the ten space view counts for all five channels.

If the user selects *time*, then **avhScan** will dump the time embedded in each scan line.

**OPTIONS**

-f type set input file type

1=NESDIS Level-1B file  
2=NESDIS KLM Level-1B file  
3=NESDIS Level-0 file

-n n skip every nth record

**EXAMPLES**

Each of the examples below are from the file noaa-14.970205200124.lac. They do not represent the full output of the program, but an excerpt to show the format used.

```
$ avhScan -n20 noaa-14.970205200124.lac cal | more
  1      1      0.1081 -3.8648   0.1090 -3.6749   -0.0017   1.6917
-0.1673 159.7771 -0.1834 178.0051
 21     21     0.1081 -3.8648   0.1090 -3.6749   -0.0017   1.6917
-0.1673 159.7771 -0.1834 178.0051
 41     41     0.1081 -3.8648   0.1090 -3.6749   -0.0017   1.6917
-0.1673 159.7771 -0.1834 178.0051
 61     61     0.1081 -3.8648   0.1090 -3.6749   -0.0017   1.6603
-0.1664 159.3572 -0.1824 177.6636
 81     81     0.1081 -3.8648   0.1090 -3.6749   -0.0017   1.6603
-0.1664 159.3572 -0.1824 177.6636
101    101    0.1081 -3.8648   0.1090 -3.6749   -0.0017   1.6603
-0.1664 159.3572 -0.1824 177.6636
```

```

121 121 0.1081 -3.8648 0.1090 -3.6749 -0.0017 1.6408
-0.1654 159.1567 -0.1827 177.7323
141 141 0.1081 -3.8648 0.1090 -3.6749 -0.0017 1.6408
-0.1654 159.1567 -0.1827 177.7323
161 161 0.1081 -3.8648 0.1090 -3.6749 -0.0016 1.6267
-0.1638 158.3915 -0.1818 177.5354
181 181 0.1081 -3.8648 0.1090 -3.6749 -0.0016 1.6267
-0.1638 158.3915 -0.1818 177.5354
201 201 0.1081 -3.8648 0.1090 -3.6749 -0.0016 1.6267
-0.1638 158.3915 -0.1818 177.5354
221 221 0.1081 -3.8648 0.1090 -3.6749 -0.0016 1.6266
-0.1639 158.4392 -0.1818 177.5393

```

```

$ avhScan -n 50 noaa-14.970205200124.lac latlon
1 1 051 0 8.56/ -77.41 6.76/ -89.91 4.64/-102.32
51 51 000 0 8.56/ -77.41 6.76/ -89.91 4.64/-102.32
101 101 051 0 9.79/ -77.66 8.01/ -90.20 5.85/-102.63
151 151 051 0 10.30/ -77.76 8.54/ -90.32 6.37/-102.77
201 201 051 0 10.79/ -77.85 9.03/ -90.43 6.85/-102.89

```

```

$ avhScan noaa-14.970205200124.lac prt | more
1 1 917 939 3
2 2 917 683 3
3 3 917 939 3
4 4 917 811 3
5 5 917 939 3
6 6 917 811 3

```

```

$ avhScan -n20 noaa-14.970205200124.lac solar | more
1 1 51 46.50 35.50 25.50
21 21 51 47.00 36.00 26.00
41 41 0 47.00 36.00 26.00
61 61 51 47.00 36.00 26.50
81 81 51 47.00 36.00 26.50
101 101 51 47.00 36.50 26.50
121 121 51 47.50 36.50 26.50
141 141 51 47.50 36.50 27.00
161 161 51 47.50 36.50 27.00
181 181 51 47.50 36.50 27.00
201 201 51 47.50 37.00 27.50
221 221 51 47.50 37.00 27.50
241 241 51 48.00 37.00 27.50

```

```

$ avhScan -n20 noaa-14.970205200124.lac telm | more Record/Line 1/
1 Ramp 644 367 860 413 527 TrgTemp 917
939 3 Black Body
72 72 617 617 738 740 736 740 733
737
708 708 119 119 391 391 391 423 423
391
862 862 120 120 371 370 370 371 371
371 Space
734 371 736 987 982 987 977 989 41
41

```

```

391      0      391      480      488      993      992      0      985
988
371      738      372      1021      989      989      733      864      992
993
737      399      41      41      169      41      41      989      990
990
391      371      169      41      41      169      169      45      41
169 ----- Record/Line      21/      21 Ramp      644      367
860      413      527 TrgTemp      917      939      3 Black Body
72      72      266      266      226      738      733      737      724
735
708      708      161      161      391      391      391      391      391
391
869      869      161      161      371      355      370      370      371
371 Space
733      371      736      988      984      989      986      991      41
41
391      0      391      992      992      992      992      0      982
990
371      739      371      477      989      990      989      992      736
868
736      391      41      41      41      41      41      989      989
989
391      371      41      41      41      41      169      41      57
41 ----- Record/Line      41/      41

```

```

$ avhScan -n20 noaa-14.970205200124.lac time | more
  1  00001  1997 036 72186473 02/05/1997 20:03:06.473
 21  00021  1997 036 72193290 02/05/1997 20:03:13.290
 41  00041  1997 036 72197306 02/05/1997 20:03:17.306
 61  00061  1997 036 72201140 02/05/1997 20:03:21.140
 81  00081  1997 036 72204473 02/05/1997 20:03:24.473
101  00101  1997 036 72207806 02/05/1997 20:03:27.806

```

**SEE ALSO****avhInfo(1), avhImage(1)**

**NAME**

avhScripts – standard AVHRR processing scripts

**DESCRIPTION**

avhScripts provides Bourne shell scripting functions to process AVHRR data from a NESDIS Level-1B file. The avhProcess is the main user interface and provides for the standard processing steps. These steps are:

- 1 Verify input file is NESDIS Level-1B file using filefmt program.
- 2 Determine if file covers user defined map using avhArea program.
- 3 Get time information from file using avhInfo program.
- 4 Parse \$L3ProdList to determine which intermediate products are required.
- 5 Ingest AVHRR data and calibrate it using avhIngest program.
- 6 If user selected "sst" as a product, generate it from brightness temperatures using avhSST program.
- 7 Generate the cloud masks and add to "l2\_flags" using avhClouds program.
- 8 If user selected any of the visible data products and input is day scene, generate them using the avhTurbid program.
- 9 Warp all the products generated above to the defined map projection.
- 10 For a large product array, tile and compress the data.
- 11 If user has defined \$AvhPreBrowse, it is now called. (Normally, not defined.)
- 12 If user had defined \$L3BrowseList, generate browse images for selected products.
- 13 If user has defined \$AvhPostProcess, it is now called. (Normally, not defined.)
- 14 Store HDF product file in the \$DATA\_BASE and the browse image in the \$IMAG\_BASE.
- 15 If user set \$L4ProdList, then run daily, 8-day, monthly, and yearly composites.

**USAGE**

A minimalist executable script for processing AVHRR data must source both the apsScripts and avhScripts located in the APS bin directory, define two required variables (see REQUIRED VARIABLES), and call the script function avhProcess passing it the name of the NESDIS Level-1B file. Many other variables can be optionally set to modify the "normal" mode of operations. These must be set prior to the call to avhProcess.

```
C#!/bin/ksh
. $AUTO_DIR/bin/apsScripts.sh
. $AUTO_DIR/bin/avhScripts.sh
MapName=GulfOfMexico
MapExt=GOM
avhProcess $1 $0
```

This script must be placed in the \$AREAS\_PROC directory which is normally the directory areas located in the main APS directory. The script must have execution permissions.

**REQUIRED VARIABLES**

These variables are required to process an area. They provide the script an area to process. Most of the remaining variables have defaults which can be overridden. They are described in the next section.

MapName

This is the name of the image map stored in the file \$MapFile.

**OPTIONAL VARIABLES**

The following sections are variables that have defaults which the user can override to change the behaviour of the default processing. They are grouped together by subject.

## Product Selection

### L3ProdList

This is a space delimited list of products to be written to the output data base. The following products are available:

albedo_ch1	Percent albedo from channel 1
albedo_ch2	Percent albedo from channel 2
albedo_ch3	Percent albedo from channel 3A
btemp_ch3	Brightness temperature from channel 3
btemp_ch4	Brightness temperature from channel 4
btemp_ch5	Brightness temperature from channel 5
sst	Sea surface temperature
c_660	Beam attenuation at 660 nm.
K_PAR	Diffuse Attenuation for photosynthetically active radiation
suspend	Total suspended solids concentration
Ray_ch1	Rayleigh reflectance for channel 1
ref_dif	Reflectance difference between channel 1 and channel 2

If not defined, the default value is "sst clouds c\_660". For example, the user might put the line

```
L3ProdList="btemp_ch3 btemp_ch4 btemp_ch5 sst"
```

to retain only the thermal channel information.

### \${prod}\_Scaling

This option is currently only defined for albedo\_ch[1,2,3], btemp\_ch[3-5] and sst products. It defines the user's desired output scaling. For example, to produce an SST byte image with a scale of 2.0 degrees per gray shade, add the line:

```
sst_Scaling="-t 1 -s 5 0"
```

in the areas script. This will cover all sea surface temperature values from 0 degrees to 51 degrees. See **avhIngest(1)** and **avhSST(1)** for more information.

By default, the following scaling is defined by avhScripts.

channel	type	slope	int	min	max
albedo_ch1	int16	500	0	-65.534	65.534
albedo_ch2	int16	500	0	-65.534	65.534
albedo_ch3	int16	500	0	-65.534	65.534
btemp_ch3	int16	200	200	36.165	363.835
btemp_ch4	int16	200	200	36.165	363.835
btemp_ch5	int16	200	200	36.165	363.835

## Data Base

These variables control information about where the data base of Level-3 and Level-4 will reside and the structure of that data base.

### Region

This variable will be used to create the default data base directories. By default it set to \$Map-Name.

#### AvhDataBase

This variable is used to indicate the location of the image data base for the generated product file. By default, it is set to:

```
$DataBase/$Level/$Sensor/$Version/$Region/$Year/$Month
```

where,

`$DataBase` is defined in the `aps.conf` file and represents the top directory of the data base.

`$Level` is set to the string "lv13" by `avhInit`.

`$Sensor` is set to the string "avhrr" by `avhInit`.

`$Version` is set to "3.0" by `avhInit`.

`$Region` is set to "`$MapName`" by `avhInit`.

`$Year` and `$Month` are set by `avhProcess` based on the input file.

The user can override `$AvhDataBase` since it is evaluated by the shell prior to use. For example, if the line:

```
AvhDataBase="\$DataBase/avhrr/\$Year"
```

is set in the areas script and we assume that `$DataBase` is set to `/data` and that for a particular file `$Year` has been set to 1999, then the product file will be moved to `/data/avhrr/1999`. Note that to use the variables, the user must "escape" the '\$' by inserting a "\".

#### AvhDAYDataBase

This variable is used to indicate the location of the Level-4 daily composites data base for the generated product file. By default, it is set to:

```
$DataBase/$CompLevel/$Sensor/$Version/$Region/daily/$Year/$Month.
```

where,

`$DataBase` is defined in the `aps.conf` file and represents the top directory of the data base.

`$CompLevel` is set to the string "lv14" by `AvhInit`.

`$Sensor` is set to the string "avhrr" by `AvhInit`.

`$Version` is set to "3.0" by `AvhInit`.

`$Region` is set to `$MapName` by `avhInit`.

`$Year` and `$Month` are set by `AvhProcess` based on the input file.

The user may override `$AvhDAYDataBase` since it is evaluated by the shell prior to use.

#### AvhNDDatabase

This variable is used to indicate the location of the Level-4 weekly (8-day) composites data base for the generated product file. By default, it is set to:

```
$DataBase/$CompLevel/$Sensor/$Version/$Region/weekly/$Year.
```

where,

`$DataBase` is defined in the `aps.conf` file and represents the top directory of the data base.

`$CompLevel` is set to the string "lv14" by `AvhInit`.

`$Sensor` is set to the string "avhrr" by `AvhInit`.

`$Version` is set to "3.0" by `AvhInit`.

`$Region` is set to `$MapName` by `avhInit`.

`$Year` is set by `AvhProcess` based on the input file.

The user may override `$AvhNDDataBase` since it is evaluated by the shell prior to use.

#### AvhMODataBase

This variable is used to indicate the location of the Level-4 monthly composites data base for the generated product file. By default, it is set to:

```
$DataBase/$CompLevel/$Sensor/$Version/$Region/monthly/$Year.
```

where,

`$DataBase` is defined in the `aps.conf` file and represents the top directory of the data base.

`$CompLevel` is set to the string "lv14" by `AvhInit`.

`$Sensor` is set to the string "avhrr" by `AvhInit`.

`$Version` is set to "3.0" by `AvhInit`.

`$Region` is set to `$MapName` by `avhInit`.

`$Year` is set by `AvhProcess` based on the input file.

The user may override `$AvhMODataBase` since it is evaluated by the shell prior to use.

#### AvhYRDataBase

This variable is used to indicate the location of the Level-4 yearly composites data base for the generated product file. By default, it is set to:

```
$DataBase/$CompLevel/$Sensor/$Version/$Region/yearly.
```

where,

`$DataBase` is defined in the `aps.conf` file and represents the top directory of the data base.

`$CompLevel` is set to the string "lv14" by `AvhInit`.

`$Sensor` is set to the string "avhrr" by `AvhInit`.

`$Version` is set to "3.0" by `AvhInit`.

`$Region` is set to `$MapName` by `avhInit`.

The user may override `$AvhYRDataBase` since it is evaluated by the shell prior to use.

#### CmpOpt

This can be defined by the user to select the type of compression program to call for the output product file before it is moved to `$AvhDataBase`. This option can be set to: "gzip", "compress", "bzip2" or "none". Only set `CmpOpt` to a compression type that is available on user's machine.

*Note:* If the user has defined `XNumChunks` or that variable is defined by `avhInit`, then the HDF file will automatically be internally compressed and this variable will have no effect.

#### XNumChunks, YNumChunks

These are used to define the number of "chunks" in each direction which will be created by `imgReformat` program for the product files. By default, a large level-3 file will be rewritten in chunks (or "tiles") with each chunk being compressed. A chunk will be no larger than 640 by 640. So, if the map image is 2430 by 1810, then `imgReformat` will create a total of 12 chunks (4 across and 3 down). A 1500 by 1500 map image will be divided into 9 chunks (3 across and 3 down). A 600 by 300 map image will NOT be chunked.

#### avhVerbose

If defined this variable will cause the script functions to call `'set -x'` within each script function. This will have the effect of printing out each step as it is executed.

**MapFile**

Name of file containing image map file. Defaults to `$AutoData/maps.hdf`

**MapExt**

This is a string that is appended to the Level-3 file which is written to the database. Usually it is a three character extension all uppercase.

**MinPixels, MinLines**

Used to set the minimum pixels/lines that must be extracted from the AVHRR file by `avhExtract` to continue processing. These are used to insure that enough of the input file covers the area of interest. By default these are not defined and, therefore, no check is performed.

**AvhAreaOpts**

This allows the use of options to the `avhArea(1)` program to be added. However, this string should not contain the `-p`, `-l`, or `-M` options.

**Browse Image Variables****L3BrowseList**

This is a list of whitespace delimited products which are converted to browse images. The products in this list must also be present in the `$L3ProdList` variable. By default, no browse images are created. That is, `BrowseList` is not defined.

**L3BrowseDir**

This variable is used to indicate the location of for the browse images. By default, it is set to:

```
$ImagBase/$Level/$Sensor/$Version/$Region/$Year/$Month
```

where,

`$ImagBase` is set in the `aps.conf` file and represents the top directory of the browse data base.

`$Level` is set to the string "l3" by `avhInit`.

`$Sensor` is set to the string "avhrr" by `avhInit`.

`$Version` is set to "3.0" by `avhInit`.

`$Region` is set to `$MapName` by `avhInit`.

`$Year` and `$Month` are set by `avhProcess` based on the input file.

The user can override `$BrowseDataBase` since it is evaluated by the shell prior to use. For example, if the line:

```
L3BrowseDir="\$ImagBase/browse/\$Year"
```

is set in the areas script and we assume that `$ImagBase` is set to `/data` and that for a particular file `$Year` has been set to 1999, then the browse image will be moved to `/data/browse/1999`.

**`\${prod}\_BrowseScaling**

This option is currently only defined for `albedo_ch[1,2]`, `btemp_ch[3-5]`, `sst`, and `c_660` products. It defines the user's desired output scaling for the browse images. For example, to produce an SST browse image that is 300 pixels by 400 lines and has a data range from 0.0 to 30.5 degrees, add the line:

```
sst_BrowseScaling="-r 0.0,30.5 -R 20,199 -s 300,400"
```

in the areas script. See `imgBrowse(1)` for more information. *Note:* The output range is set to 20 and 199 because the `avhMakeBrowse` file script uses NSIPS files for overlays and colortables by default.

By default, the following scaling is defined by avhScripts.

channel	function	min	max
albedo_ch1	linear	0.0	50.0
albedo_ch2	linear	0.0	50.0
albedo_ch3	linear	0.0	50.0
btemp_ch3	linear	260.0	310.0
btemp_ch4	linear	260.0	310.0
btemp_ch5	linear	260.0	310.0
sst	linear	2.0	35.0
c_660	log10	1.49	50.0

*Note:* For the other products ("K\_PAR", "suspend", "ref\_dif", "Ray\_ch1"), the user must provide the corresponding variables (K\_PAR\_BrowseScaling, suspend\_BrowseScaling, refdif\_BrowseScaling, Ray\_ch1\_BrowseScaling) as there are no defaults.

### Program Variables

These variables define the programs used by avhScripts. The user can override these to test a new version of a program. They are defined by avhInit.

#### ApsInfo

Set to the name of the satellite specific program used to obtain information from input Level-1 file. Defaults to \$AutoBin/avhInfo.

#### AvhArea

Set to the name of the program used to determine if a AVHRR file covers a map. Defaults to \$AutoBin/avhArea.

#### AvhInfo

Set to the name of the program used to obtain information from AVHRR file. Defaults to \$AutoBin/avhInfo.

#### AvhClouds

Set to the name of the program used to produce clouds masks from input calibrated AVHRR images. Defaults to \$AutoBin/avhClouds.

#### AvhIngest

Set to the name of the program used to ingest and calibrate the AVHRR data. Defaults to \$AutoBin/avhIngest.

#### AvhSST

Set to the name of the program used to produce sea surface temperature estimates from input brightness temperature images. Defaults to \$AutoBin/avhSST.

#### AvhTurbid

Set to the name of the program used to produce visible data products from input percent albedo images. Defaults to \$AutoBin/avhTurbid.

## SCRIPT FUNCTIONS

This section describes each script function available.

#### avhClouds

This script function run the program avhClouds:

```
$ avhClouds levelsL2.spk levels3.spk levels4.spk levels5.spk levels1.spk levels2.spk
```

If avhClouds returns successfully, the program spkToSDS is called to append the levelsL2.spk file to the Level-2 HDF file. The term 'l2\_flags' is appended to an internal variable used to maintain a list of products which need to be warped.

#### avhComposites

##### avhDataBase

This script function checks to see if the variable \$BrowseList is defined. If so, \$BrowseDir is checked for existence and set to a default value if not. For each product in BrowseList, apsAddToWWW is called to move the browse image to the BrowseDir.

Next the script function checks to see if \$L3ProdList is defined. If so, the AvhDataBase variable is checked for existence and set to a default if not. If the user "XNumChunks" was set either by the user or avhInit, then the compression command is set to none since these products are already compressed. Otherwise, the compression is set to \$L3CmpOpt, which may have been previously defined by user using CmpOpt.

##### avhDefaultBrowseParameters

This script function sets up the default scaling parameters for the browse images. To begin the function checks to see if \$ImgSamples is defined. If not, then the default image sizes are calculated using imgCalculateSize. For each product in the BrowseList, if the variable \${prod}\_BrowseScaling is not defined it is defined. See BrowseList above for more information and default variables.

##### avhExtract

This script function calls avhArea to determine if the input AVHRR file covers the map. It also uses the \$MinPixels and \$MinLines to determine coverage if they have been set. It returns a 0 if file covers area; a 1 for a processing error; and a 2 if file does not cover area.

##### avhInit

This script function defines programs used for AVHRR processing, environmental variables needed by those programs, default browse image parameters, and other miscellaneous variables. It must be called before any other scripts defined here.

##### avhMakeBrowse

This script function builds the default browse images. For each product in the L3BrowseList variable, the program imgBrowse will generate a quick-look image as a JPEG formatted file.

##### avhMap

This script function will call the imgMap program to warp each image and then call imgReformat to "chunkify" it if XNumChunks has been defined. Finally, all the attributes from the input HDF Level-2 file are copied over to the HDF Level-3 file.

##### avhIngest

This script function will call the avhIngest program to produce the required calibrated data that covers the area of interest. For each product in \$L3ProdList, the data is appended to a HDF Level-2 file.

#### avhIngestArgs

This script function builds up the argument string needed by avhIngest to output the desired products. It sets up the default scaling.

#### avhProcess

This script function is the default processing flow which is described at the top of this man page. It generally is just a series of calls to other script functions defined in this section.

#### avhProducts

This script function uses the `$L3ProdList` to determine what channels must be output by avhIngest. Some of these outputs may be temporary.

#### avhProductIngest

This script function is used by avhProducts. It removes duplicates on a list and handles non-calibrated vs calibrated output.

#### avhSetNames

This script function defines the standard naming scheme. By default, the HDF Level-2 file will be name: "\$SatCode\$Year\$DoY\$Time.L2\_HNAV\_\$.MapExt". *Note:* MapExt is defined by the user and the others are defined by the input file and the avhInfo program.

#### avhSST

This script function calls the program avhSST, if "sst" is defined in the `$L3ProdList`. If successful, the product is added to the HDF Level-2 file.

#### avhTurbid

This script function calls the program avhTurbid, if any of the visible data products ("Ray\_ch1", "ref\_dif", "K\_PAR", "suspend", or "c\_660") is defined in `$L3ProdList`. If successful, the products are added to the HDF Level-2 file.

#### avhWarp

This script function calls the program imgMap to warp the data.

## CONFORMANCE

These script function attempt to conform to the IEEE 1003.2 POSIX Shell Standard. They were, however, developed and tested using the Bourne Shell and Korn Shells under IRIX 5.3 and IRIX 6.5 respectively.

## SEE ALSO

**apsScripts(1), avhArea(1), avhClouds(1), avhInfo(1), avhIngest(1), avhSST(1), avhTurbid(1), daylight(1), filefmt(1), hdf(1), maps(1), imgMap(1), imgReformat(1), imgBrowse(1), imgSDStoImg(1), spkToSDS(1)**

**NAME**

avhSwapL0 – converts a Terascan Level–0 formatted file to NESDIS Level–0 formatted file

**SYNOPSIS**

**avhSwapL0** input.l0 output.l0

**DESCRIPTION**

This program will take an AVHRR "hrpt" Terascan Level–0 formatted file and convert it to a NESDIS Level–0 format required for input to avhL1bgen. The Terascan Level–0 format can be created by the **archive** command of the Terascan system. The input file consists of 44 512-byte sectors in which the 10-bit AVHRR Level–0 stream is merged into 16-bits. This program will swap the bytes so that they resemble the HPRT Minor Frame format described in NOAA Technical Memorandum NESS 107 – Rev 1.

**SEE ALSO**

**avhL1agen(1)**

**NAME**

avhTurbid – produces beam attenuation, diffuse attenuation, suspended solids images from percent albedo images.

**SYNOPSIS**

**avhTurbid** [-g angle] [-n angle] input.spk output.spk

**DESCRIPTION**

This program is used to estimate the beam attenuation coefficient ( $c_{660}$ ), the diffuse attenuation coefficient for photosynthetically active radiation for 400-700 nm (KPAR), and total suspended solids (or seston) in coastal waters using the percent albedo for channels 1 and 2 of the AVHRR data. The percent albedo channels are products of the program avhIngest.

The input file are defined by inserting a '1' or a '2', for percent albedo images for channels 1 and 2, respectively, before the '.' in the filename. For example, if the first argument is "albedo\_ch.spk", then the file "albedo\_ch1.spk" will hold the percent albedo image for channel 1 and the file "albedo\_ch2.spk" will hold the percent albedo image for channel 2.

Likewise, for the output file name. In this case, the output numbers follow the following table:

Num	Slope	Intercept	Type	Product
1	0.1	0.0	byte	Rayleigh reflectance for channel 1
2	0.05	-0.05	byte	Water reflectance for channel 1
3	1.0	0.0	byte	Total Suspended Solids
4	0.1	0.0	byte	Diffuse Attenuation Coefficient for PAR
5	0.005	0.0	short	Beam Attenuation Coefficient at 660 nm

Therefore, if the second argument is "turbid.spk", then "turbid5.spk" will be an image of the beam attenuation coefficient.

The slope and intercept in the tables will take the integer "gray" value and convert them into the geophysical values. The first four products are stored as 8-bit unsigned integers and the Beam Attenuation product is stored as 16-bit signed integers. Currently, there is no option to change the data conversion.

**OPTIONS**

-g angle

This sets the glint angle used to determine if an area is susceptible to glint. If an area is found to be glint contaminated, it is masked out. By default, this angle is set to -1.0 to indicate that no glint masking is being performed.

-n angle

This is the angle used to determine if for the selected control point, the sun is too low or down. If the solar zenith angle is greater than this angle the control is not processed. By default the angle is set to -1.0.

--help Display program help.

--version

Display program name version and time of compilation.

**FILES**

nmcoeff.YYYY.d

There are a series of files for each year. The files contain the tau for rayleigh and ozone for satellite NOAA-9, NOAA-10, NOAA-11, NOAA-12, NOAA-14, and NOAA-15, NOAA-16.

**ENVIRONMENT VARIABLES**

COEFFDIR

This environmental variable points to the location of the coefficients files (nmcoeff.YYYY.d).

**EXAMPLES**

This example will produce floating point outputs of albedo channels 1 and 2 for input to the turbidity program **avhTurbid(1)**.

```
$ avhIngest -1 R -2 R NJ1999360192854.L1B_HNAV avhrr.spk
$ avhTurbid avhrr.spk turbid.spk
Turbid 0
$ ls -l turbid*.spk
-rw-r----- 1 aps      aps      4168192 Jan  6 12:37 turbid1.spk
-rw-r----- 1 aps      aps      4168192 Jan  6 12:37 turbid2.spk
-rw-r----- 1 aps      aps      4168192 Jan  6 12:37 turbid3.spk
-rw-r----- 1 aps      aps      4168192 Jan  6 12:37 turbid4.spk
-rw-r----- 1 aps      aps      8335872 Jan  6 12:37 turbid5.spk
```

Notice that by default, turbid5.spk, is written out as signed 16-bit integers. This product is the beam attenuation at 660 nm (c\_660).

**REFERENCES**

Stumpf, R. P. (1992) *Remote Sensing of Water Clarity and Suspended Sediments in Coastal Waters*. In Proceedings of the First Thematic Conference on Remote Sensing for Marine and Coastal Environments. SPIE 1930, 293-305. Environmental Research Institute of Michigan, Ann Arbor, MI.

Gould, R. W. and Arnone, R. A. (1997) *Estimating the beam attenuation coefficient in coastal waters from AVHRR imagery*. Continental Shelf Research, Vol 17, No 11, p 1375-1387.

**SEE ALSO**

**avhIngest(1)**