

Off-line Retrieval Notes

Chris Barnet
NOAA/NESDIS/ORA

Mailing Address

World Weather Building
E/RA2
5200 Auth Road
Camp Springs, MD. 20746



Office Address

Airmen Memorial Building
Suite 204
5211 Auth Road
Camp Springs, MD. 20746



chris.barnet@noaa.gov
(301)-316-5011

August 30, 2006

Contents

| | | |
|----------|---|-----------|
| 1 | Introduction | 1 |
| 1.1 | Revision History & Cross-training using this document | 4 |
| 1.2 | Current Users of This Code | 4 |
| 1.3 | Location of Program and Coefficient Files | 4 |
| 1.4 | Location of documentation | 5 |
| 1.4.1 | Fetching AVN files | 6 |
| 1.4.2 | GOES Images | 6 |
| 1.4.3 | JPL Granule Maps | 6 |
| 1.5 | FORTRAN directory structure | 6 |
| 1.6 | IDL directory structure | 7 |
| 1.7 | Short summary of MAIN program files in src_util | 9 |
| 1.8 | Short description of selected subroutines & functions | 9 |
| 1.9 | Summary of top-level IDL Programs | 10 |
| 2 | How to set-up and compile the FORTRAN code | 11 |
| 2.1 | Set-up & Environment Variables | 11 |
| 2.1.1 | Example bash login script for external users (not NOAA or UMBC) | 12 |
| 2.1.2 | Example bash login script for NOAA LINUX machines | 13 |
| 2.1.3 | Example of tsch login script for asl machines | 14 |
| 2.2 | How to compile or re-compile airs_b.F | 15 |
| 2.2.1 | Example of rebuild script for orbit machines | 15 |
| 2.3 | How to build the coefficient database | 17 |
| 2.3.1 | Coefficient Files for the Forward Computation of Radiances | 18 |
| 2.3.2 | Coefficient Files REQUIRED by the Retrieval Code | 19 |
| 2.4 | Unpacking and Running the Test Case | 19 |
| 2.5 | Using the PC version of the code with the COMPAC compiler | 21 |
| 2.6 | what are G401 and G422 files | 22 |
| 3 | preproc: Processing Real AIRS Data | 24 |
| 3.1 | Why do we have a preprocessor at all? | 24 |
| 3.2 | What does the preprocessor do? | 24 |
| 3.3 | How do I actually *use* this fabulous preprocessor? | 26 |
| 3.4 | autoproc: a preprocessor for the preprocessor | 29 |
| 3.5 | How to run the whole enchilada | 29 |
| 3.6 | How to order from the GSFC AIRS DAAC | 30 |
| 3.7 | Selection of AVN files | 32 |
| 3.8 | Selection of ECMWF files | 33 |
| 3.9 | Top level functional map of the preproc.pro program | 33 |
| 3.10 | NOAA Error codes for L1b files | 34 |
| 3.10.1 | AMSU L1 QA | 35 |
| 3.10.2 | AIRS L1 QA | 39 |

| | | |
|----------|---|-----------|
| 4 | match_proc: Processing JPL Matchup Files | 43 |
| 5 | cnv_gbgrib: Processing NOAA Gridded (Grads) Files | 44 |
| 6 | simrad: simulation of radiance scan sets | 45 |
| 7 | airsb: retrieval of geophysical parameters | 46 |
| 7.1 | command line control | 46 |
| 7.1.1 | airsb direct mode | 46 |
| 7.1.2 | airsb indirect mode | 46 |
| 7.2 | required namelist files | 47 |
| 7.3 | Top level flow diagram for airsb.F | 48 |
| 7.4 | I/O units for airsb code | 49 |
| 7.5 | Selection of Channels (io namelist: tauble) | 51 |
| 7.6 | airsb error code word | 52 |
| 7.7 | Making the code run faster | 52 |
| 7.8 | Quality indicators stored in ispare() and r spare() | 53 |
| 7.9 | AIRS science versions | 58 |
| 7.9.1 | differences between v3.0 and v4.0 | 58 |
| 7.9.2 | differences between v4.0 and v4.2 | 58 |
| 7.9.3 | differences between v4.2 and v4.3 (July 2005) | 59 |
| 7.9.4 | differences between v4.3 and v4.5 (Sep. 2005) | 59 |
| 7.9.5 | v3.x rejection methodology | 59 |
| 7.9.6 | v4.0 rejection methodology | 60 |
| 7.9.7 | Example output for quality indicators in the .out file | 64 |
| 8 | simstat: off-line statistics of ensembles of retrievals | 66 |
| 8.1 | Using the SIMSTAT program | 70 |
| 8.2 | Subroutines used by simstat | 74 |
| 9 | Diagnostic Tools | 75 |
| 9.1 | My color table | 76 |
| 9.2 | pl_amsul1b: plotting AMSU granules | 77 |
| 9.3 | pl_sim: plotting retrieval statistics | 78 |
| 9.4 | stat1 & stat2 & stat3: IDL driver for simstat.F | 81 |
| 9.5 | pl_ftn92: plotting radiance statistics | 85 |
| 9.6 | pl_ret: plotting retrieval statistics at all steps | 87 |
| 9.7 | pl_ampl: plotting amplification factor | 89 |
| 9.8 | pl_lev2 & pl_l2d: utility to plot individual retrieval profiles | 90 |
| 9.9 | profsumm: plotting individual retrieval summaries | 92 |
| 9.10 | pl_eigen: plot eigenvector plots for selected profiles | 96 |
| 9.11 | pl_smat: plot the S-matrix for selected profiles | 101 |
| 9.12 | pl_retccr: plotting individual retrieval diagnostics | 104 |
| 9.13 | pl_realgran: plotting maps of retrieval products | 107 |
| 9.14 | pl_trace: plotting trace gas profiles | 114 |
| 9.15 | pl_cmdl: comparing retrievals to matchup datasets | 115 |
| 9.16 | pl_trscan: plotting trace results versus scan position | 120 |
| 9.17 | pl_f65: plotting results from co2_per | 121 |

| | |
|--|------------|
| 10 Utilities for Processing Large number of granules | 122 |
| 10.1 calling IDL from a script | 122 |
| 10.2 Retrievals near validation sites | 123 |
| 10.2.1 search_all: find retrievals within a given distance of a ground station | 124 |
| 10.2.2 build_cnf: concatenate selected cases & days into one run | 126 |
| 10.2.3 reform_ret: find retrievals within a given distance of a ground station using concatenated retrievals | 127 |
| 11 Simulated Datasets | 129 |
| 11.1 openl2.b.F & readl2.b.F: Documentation for reading L2 files | 129 |
| 11.2 Conversion of LAYER Column density to Column Density | 132 |
| 11.3 Conversion of LAYER Column density to mixing ratio | 133 |
| 12 Rapid Transmittance Algorithms (RTA's) | 135 |
| 12.1 comp_rad.F & airs_rad.F: radiance generator utilities | 135 |
| 12.2 Utility Programs for RTA and RT computations | 136 |
| 12.3 The MIT RTA | 137 |
| 12.4 The UMBC RTA | 144 |
| 12.4.1 Program to convert UMBC files to the off-line format | 144 |
| 12.4.2 UMBC library, code/umbc9803 | 146 |
| 12.4.3 State variable definitions and units | 148 |
| 12.4.4 Computational efficiency issues | 149 |
| 12.4.5 UMBC predictor definitions | 150 |
| 13 Infrared and Microwave Noise Models | 158 |
| 13.1 Microwave models | 158 |
| 13.2 Infrared models | 158 |
| 13.2.1 converting $NE\Delta N(n)$ to $NE\Delta T(n)$ | 158 |
| 13.2.2 converting $NE\Delta T(n)$ to $NE\Delta N(n)$ | 159 |
| 13.2.3 Noise models | 159 |
| 13.2.4 Interferometer Noise Modeling | 163 |
| A Documentation for UMBC RTA Versions | 1 |
| A.1 v9 AIRS RTA: January 2004 | 1 |
| A.2 v8 AIRS RTA: January 2004 | 1 |
| A.3 v7 AIRS, IASI, CrIS RTA's: January 2003 | 2 |
| A.4 v6a AIRS RTA : August 2002 | 2 |
| A.5 v5 AIRS RTA: August 2000 | 3 |
| A.6 v4 AIRS RTA: Sep. 1999 | 4 |
| A.7 v3 RTA's for AIRS (5/98), CrIS (5/98 & 10/98), NAST (10/98) & IASI (10/98) | 5 |
| B Description of the Phillip's dataset | 1 |
| C Description of η model: NA,NB,NC,DB,DC,DD,DP | 1 |
| D Description of the Nov. 1996 dataset (16 scan lines) | 1 |
| D.1 16 track training ensemble, ~/level2/TR.trn | 3 |
| D.2 Selected 16 scan lines | 6 |
| D.3 Selected 8track datasets | 8 |

| | | |
|----------|---|----------|
| E | NOAA 1988 and 1989 radiosonde dataset | 1 |
| E.1 | The 66 level NOAA 1988 and 1989 dataset | 1 |
| E.1.1 | Statistics of the 1988 radiosonde dataset | 1 |
| E.1.2 | Statistics of the 1989 radiosonde dataset | 2 |
| E.2 | The 100 level NOAA 1988 and 1989 dataset. | 2 |
| E.2.1 | Statistics of the 1988 radiosonde dataset | 3 |
| E.2.2 | Statistics of the 1989 radiosonde dataset | 4 |
| E.3 | How to read the NOAA88 & 89 files | 8 |
| E.3.1 | openl2_b.F & readl2_b.F: Documentation for reading L2 files | 8 |
| F | Sept. 13, 1998 AIRS Science Team Dataset | 1 |
| G | Dec. 15, 2000 AIRS Science Team Dataset | 1 |

List of Figures

| | | |
|------|---|-----|
| 1.1 | Example of how we use a simulation system. | 2 |
| 1.2 | The NOAA plan for the off-line code. | 3 |
| 9.1 | Index scheme for my color table | 76 |
| 9.2 | Example output from the pl_amsul1, 2003,1,14, [51], 2 | 77 |
| 9.3 | Example output from pl_sim, 'test', menu=1 | 80 |
| 9.4 | Example output from pl_sim, 'test', /bias, menu=1 | 80 |
| 9.5 | Example: stat2, 'run1', 'd65bin', 'test' followed by pl_sim, 'test', menu=2 (see test.cnf listing) | 83 |
| 9.6 | Example: stat2, 'run1', 'd65bin', 'test' followed by pl_sim, 'test', menu=2, /bias (see test.cnf listing) | 83 |
| 9.7 | Example: stat2, 'run1', 'd65bin', 'test', /ocean followed by pl_sim, 'test', menu=3 (see test.cnf listing) | 84 |
| 9.8 | Example: stat2, 'run1', 'd65bin', 'test', /ocean followed by pl_sim, 'test', menu=3, /bias (see test.cnf listing) | 84 |
| 9.9 | Example output from the pl_ftn92.pro program | 86 |
| 9.10 | Example output from the pl_ftn92.pro program with the /bias option | 86 |
| 9.11 | Example $T(p)$ statistic output from the pl_ret, 'd65bin_test', xmax=[10,200,200], menu=1 | 87 |
| 9.12 | Example $q(p)$ statistic output from the pl_ret, 'd65bin_test', xmax=[10,200,200], menu=2 | 88 |
| 9.13 | Example output from the pl_ampl.pro program | 89 |
| 9.14 | Example output from the profsumm, 'd65bin_test', pick=34.3 | 94 |
| 9.15 | Example output from the profsumm, 'd65bin_test', /nocldfrc, pick=36.1 | 94 |
| 9.16 | Example output from the profsumm, 'd65bin_test', pick=28.3 | 95 |
| 9.17 | Example output from the pl_eigen, 'd65bin_test', /eps, The file d65_amsu1_1_eig.eps is shown | 97 |
| 9.18 | Example output from the pl_eigen, 'd65bin_test', /eps, The file d65_wat1_1_eig.eps is shown | 97 |
| 9.19 | Example output from the pl_eigen, 'd65bin_test', /eps. The file d65_tmp2_1_eig.eps is shown | 98 |
| 9.20 | Example output from the pl_eigen, 'd65bin_test', /eps. The file d65_tmp2_555_eig.eps is shown | 98 |
| 9.21 | Example output from the pl_eigen, 'd65bin_test', /eps. The file d65_ozo1_1_eig.eps is shown | 99 |
| 9.22 | Example output from the pl_eigen, 'd65bin_test', /eps. The file d65_ozo1_395_eig.eps is shown | 99 |
| 9.23 | Example output from the pl_eigen, 'd65bin_test', /eps. The file d65_CO1_1_eig.eps is shown | 100 |
| 9.24 | Example output from the pl_eigen, 'd65bin_test', /eps. The file d65_CH41_1_eig.eps is shown | 100 |
| 9.25 | Example output from the pl_smat, called via temp_fig with the /LW option | 102 |
| 9.26 | Example output from the pl_smat, called via temp_fig with the /SW option | 102 |
| 9.27 | Example output from the pl_smat, called via ch4_fig | 103 |
| 9.28 | Example output from the pl_smat, called via co_fig | 103 |
| 9.29 | Example output from the pl_retccr, 'd65bin_b', 'test', 51 | 106 |
| 9.30 | Example output from the pl_realgran, 2003,1,14,'d65bin', 'test'. Selecting menu item 2 | 110 |
| 9.31 | Example output from the pl_realgran, 2003,1,14,'d65bin', 'test'. Selecting menu item 15 | 111 |
| 9.32 | Example output from the pl_realgran, 2003,1,14,'d65bin', 'test'. Selecting menu item 6 | 111 |
| 9.33 | Example output from the pl_realgran, 2003,1,14,'d65bin', 'test'. Selecting 'ALL' and then menu item 6 | 112 |
| 9.34 | Example output from the pl_realgran, 2003,1,14,'d65bin', 'test'. Selecting 'ALL' and then menu item 7 | 112 |

| | | |
|------|--|-----|
| 9.35 | Example output from the pl_realgran, 2003,1,14,'d65bin', 'test'. Selecting 'ALL' and then menu item 9 | 113 |
| 9.36 | Example output from pl_trace, 'd65bin','test', plist=[1.0001,1.0135] | 114 |
| 9.37 | Example CO matchup using pl_cmdl: IDL> pl_cmdl,'d65.lst', /noco2, /noch4, /all, /ps | 118 |
| 9.38 | Example CH ₄ matchup using pl_cmdl: IDL> pl_cmdl,'d65.lst', /noco1, /noco2, /all, /ps | 118 |
| 9.39 | Example CO ₂ matchup using pl_cmdl: IDL> pl_cmdl,'d65.lst', /noco1, /noch4, /all, /ps | 119 |
| 12.1 | RMS RTA fitting errors for all 7 models. The files plotted here are included with this delivery in the file rmserr.zip. Note the error scale on AIRS is smaller (0.3 versus 0.5 on all others) . . | 147 |
| D.1 | Distribution of profiles in the training dataset. | 4 |
| D.2 | Location of 16 profiles in original simulation set | 6 |
| E.1 | Location of the original 66 level 1988 profiles | 2 |
| E.2 | Location of profiles in the NOAA 1988b dataset | 3 |

List of Tables

| | | |
|------|--|-----|
| 1.1 | Paths specified by HOST Environment Variable | 5 |
| 2.1 | Environment Variables used by the off-line code | 12 |
| 2.2 | AIRS G401 file set | 22 |
| 2.3 | AIRS granules available on CD | 22 |
| 3.1 | Size of files for an AIRS granule | 28 |
| 3.2 | AVN files required for an observation at time = UT | 32 |
| 3.3 | AVN truth table | 32 |
| 3.4 | Recommended ECMWF file for an observation at time = UT | 33 |
| 3.5 | ECMWF truth table | 33 |
| 3.6 | CALBUF CODES for AMSU & HSB Level 1 files | 36 |
| 3.7 | AMSU/HSB QA Test Matrix | 37 |
| 3.8 | qa_channel | 37 |
| 3.9 | qa_receiver* | 37 |
| 3.10 | state1val .or. state2val for AMSU, state for HSB | 38 |
| 3.11 | CALBUF CODES for AMSU & HSB Level 1 files | 39 |
| 3.12 | AIRS L1 QA Test Matrix | 40 |
| 3.13 | Summary of Channel Property Files | 41 |
| 3.14 | AIRS L1b state variable | 41 |
| 3.15 | A/B State (ExcludedChans) | 41 |
| 3.16 | CalChanSummary (granule) | 42 |
| 3.17 | L1b Calflag tests | 42 |
| 7.1 | ispare definitions | 54 |
| 7.2 | ispare(2) rejection definitions | 55 |
| 7.3 | rspace QA definitions | 56 |
| 7.4 | rspace data formatting | 57 |
| 7.5 | Delivery Dates for AIRS Science Versions | 58 |
| 7.6 | rejection thresholds for v3.0,3.18,v4.0 | 61 |
| 9.1 | pl.ftn92 command line systax | 85 |
| 9.2 | pl.ftn92: symbols used in plots | 85 |
| 10.1 | Validation Sites in resolve_site.pro | 125 |
| 11.1 | summary of geophysical datasets in #/airsb/level2 | 129 |
| 12.1 | MIT RTA file format for oxygen channels | 139 |
| 12.2 | MIT RTA file format for oxygen channels with magnetic correction | 140 |
| 12.3 | MIT RTA file format for water channels | 141 |
| 12.4 | MIT RTA file format for window channels | 142 |
| 12.5 | MIT RTA file format for water channels (old format) | 143 |

| | | |
|-------|---|-----|
| 12.6 | Number of Predictors for the Seven UMBC Fitting Models | 152 |
| 12.7 | UMBC AIRS RTA: Fixed Gas Predictors | 154 |
| 12.8 | UMBC AIRS RTA: Water Continuum Predictors | 154 |
| 12.9 | UMBC AIRS RTA: Carbon Dioxide Predictors | 154 |
| 12.10 | UMBC AIRS RTA: Water Vapor Predictors | 155 |
| 12.11 | UMBC AIRS RTA: Ozone Predictors | 156 |
| 12.12 | UMBC AIRS RTA: Carbon Monoxide Predictors | 156 |
| 12.13 | UMBC AIRS RTA: Methane Predictors | 157 |
| | | |
| A.1 | Available UMBC RTA's for AIRS | 2 |
| A.2 | Available IASI & CrIS RTA files | 3 |
| A.3 | Thermal Down-welling version in UMBC RTA's | 3 |
| A.4 | 66 level RTA's developed at GSFC (Amita Mehta) | 3 |
| | | |
| D.1 | Distribution of Training Profiles for the AST Sixteen Scan Line Set | 5 |
| D.2 | Location of the AST Sixteen Scan Lines | 7 |
| D.3 | Cloud and Surface Pressure Properties of the AST Sixteen Scan Lines | 7 |
| | | |
| E.1 | NOAA88 66 level set: distribution of profiles over OCEAN | 5 |
| E.2 | NOAA88 66 level set: distribution of profiles over LAND | 5 |
| E.3 | NOAA89 66 level set: distribution of profiles over OCEAN | 6 |
| E.4 | NOAA89 66 level set: distribution of profiles over LAND | 6 |
| E.5 | NOAA88b (100 level set), distribution of profiles over OCEAN | 7 |
| E.6 | NOAA88b (100 level set), distribution of profiles over LAND | 7 |

Chapter 1

Introduction

This document is intended to be a resource guide for the user of the AIRS off-line retrieval code. The code changes too rapidly to include specific documentation for detailed documentation of the code or namelists. The user must refer to the code to determine proper calling sequences and limitations of individual routines.

The off-line code was designed to:

1. Emulate the JPL Product Generator Executable (PGE), the retrieval executable for AIRS processing at the DAAQ.
2. Run simulation experiments for AIRS, CrIS, and IASI (using simrad.F to generate radiances).
3. Provide a plethora of diagnostic output for
 - developing and evaluating algorithms
 - studying difficult cases

There is a single code that has been successfully compiled and run, without modification, on HP Unix/f77, SGI Irix/f90, Linux/Portland Group, Linux/ABSOFTE, PC/Compac (see Section 2.5). The namelist input and namelist echo has been written to be machine independent and to allow inter-comparison of statistics and diagnostic files from different machines. This was done via

- Using special character symbols to pre-pend environment variables
 - #
in filenames will substitute a default path to the coefficient tables for known machines and use \$AIRS_coef for machines that are not recognized.
 - #xxx will look for an environment variable named xxx and pre-pend that to the filename.
- The namelist echo is formatted so that the namelist field can be easily diff'ed between different machines.

The original code was developed to support pre-launch simulation experiments for AIRS/AMSU/HSB, IASI/AMSU/MHS, and CrIS/ATMS. The components of the entire system are shown in Fig. 1.1.

The merging of models (upper left of Fig. 1.1) was accomplished with routines such as make_co2.F. Radiance generation (upper right of Fig. 1.1) was done with simrad.F and the retrieval algorithm (lower portion of Fig. 1.1) is driven by airsb.F. Statistics of each and every retrieval step is performed by airsb.F if a reference (or “truth”) profile set is given. Otherwise, statistics of output files (final retrieval and options for NOAA regression, and MIT microwave algorithm) can be performed after the retrieval by simstat.F.

With the launch of AIRS on May 4, 2002, the system was quickly revamped to handle real data. A preprocessor was written in IDL (preproc.pro) to handle the AIRS HDF formats. Local angle correction, a necessary step for cloud clearing (see rs_notes.pdf), was added to the preprocessor. AIRS channels can go bad, either intermittently or after a warm-up event (*e.g.*, Oct. 19-22, 2002 and Oct. 28-Nov. 6, 2003). The preprocessor allowed quick adaptation to the post-launch issues.

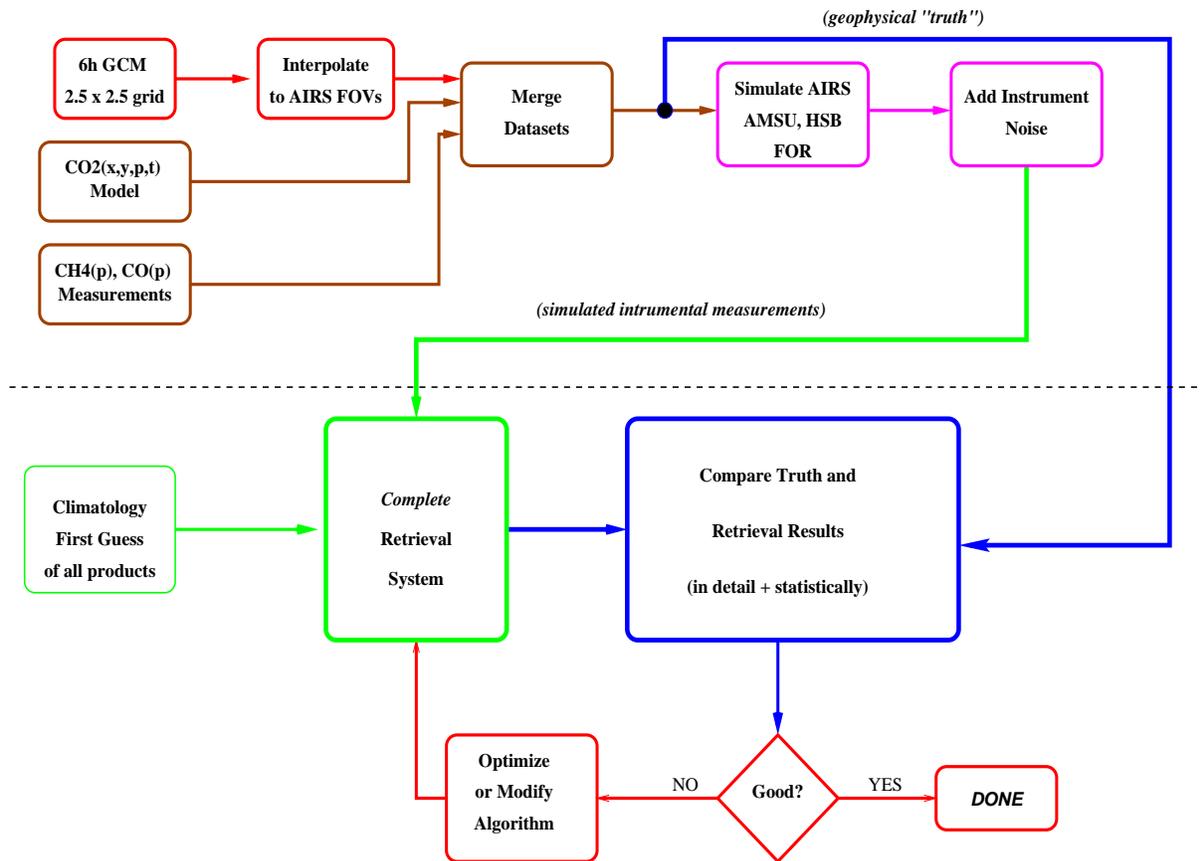


Figure 1.1: Example of how we use a simulation system.

The decision was made, by myself, to have all real-time issues be handled by the IDL processor and to minimize modifications to the simulation version of the code. This initial decision was made because it simply allowed rapid development of a code without any programming support. The IDL preprocessor allowed having the operational code process the AIRS first light granules on Jun. 13, 2002 and be able to compare retrieval results to AVN and ECMWF forecasts immediately.

The decision to have all file formatting handled outside of the retrieval code has; however, proved to be generally useful. Special files developed by UMBC (the “RTP file), JPL matchup files, and NOAA’s gridded files could all be handled, without modification to the retrieval code. Each file type has it’s own preprocessor.

The `airsb.F` code, illustrated in Fig. 1.1, already had the ability to compare retrievals to a reference profile. In simulation, this reference is the “truth”, the atmospheric state used to generate the radiances. Even in simulation this is complicated by the fact that there are nine truths per AMSU field of regard and spatial variability makes comparison’s difficult. This is the “what is truth” question raised many a time in the AIRS science team meetings. Since the surface pressure, derived from the AVN forecast, is a required component of the retrieval system I chose to have AVN as the default reference profile. The preprocessor interpolates the AVN forecast in time and space and produces a full atmospheric state for comparison to the retrievals – at every iteration and every retrieval step. This plethora of diagnostic output is unique to the off-line code.

Later the ECMWF forecast fields, as processed by UMBC into RTP files was added as an optional reference state. Currently, NOAA/NESDIS is adding the capability to have ECMWF, as processed by this package, as a reference state along with radiosondes, NOAA-16 ATOVS retrievals, and the NOAA-GDAS model. The projected use of this code is shown in Fig. 1.2. In order to facilitate efficient use of human resources and to provide added benefit of inter-comparison of instruments and methodology a system has

been envisioned in which development time can be decreased and utility and synergy can be maximized.

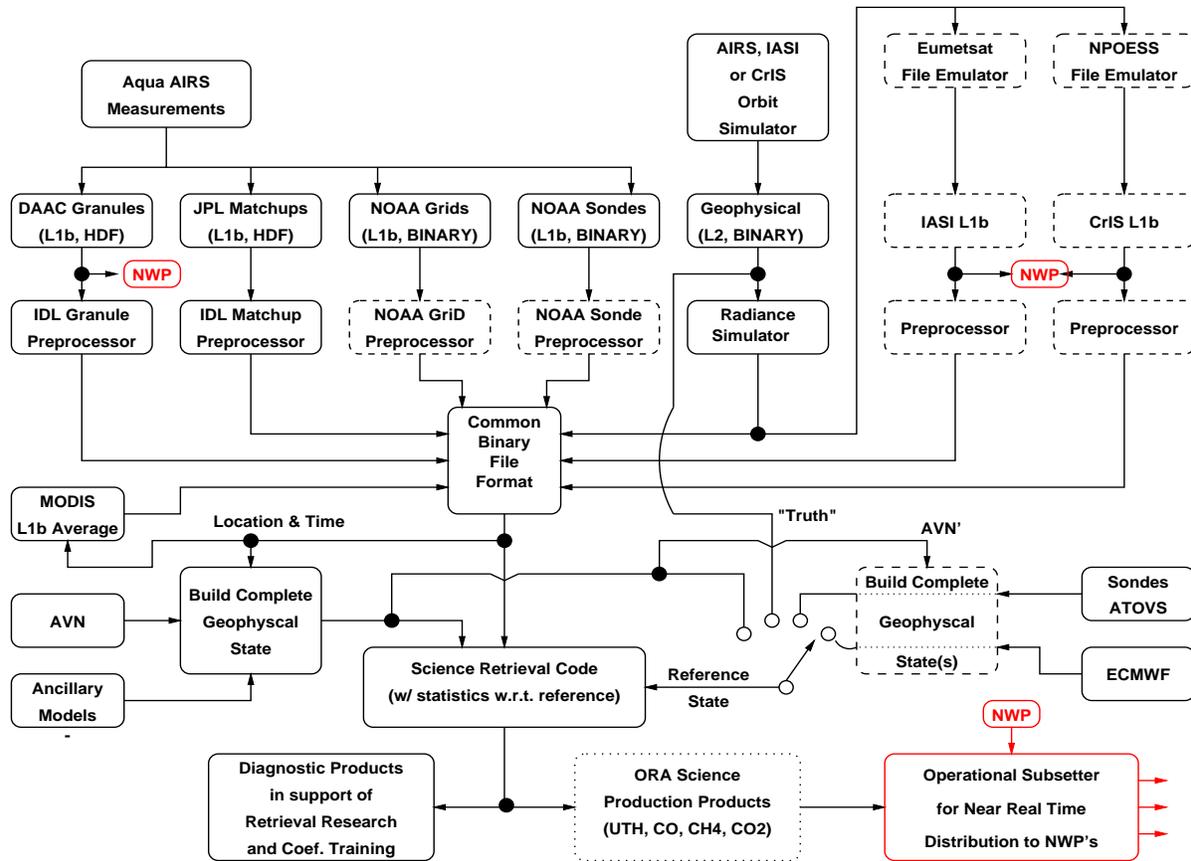


Figure 1.2: The NOAA plan for the off-line code.

The advanced sounders (*e.g.*, AIRS, IASI, CrIS, GOES_R/HES) have many file formats and methodologies of data acquisition. We currently have 4 forms of AIRS data: individual granules (DAAC granules on upper left); matchup files from the AIRS science team (JPL Matchups in HDF format); gridded sub-sets (NOAA/NESDIS/ORA in GRADS format); and NOAA sonde matchups in a binary format. In Figure 1.2 we show along the top that each of these have their own preprocessor to eliminate specific file format issues and data organization issues, such as converting scan ordered data into “golfballs” of co-located infrared & microwave data. We send the retrieval code (middle of figure) a common binary format so that

- (a) the retrieval code is simplified and
- (b) the retrieval code executes within memory and speed allocations.

Future instruments can be simulated and treated in exactly the same manner, as shown in the box labeled “AIRS, IASI, CrIS orbit simulator” and those simulations can be reformatted into the native format for those instruments (boxes in upper right) to test the preprocessors on native file formats prior to launch. For example, IASI launches in late 2005; however, we plan on simulating 100% of the IASI FOV’s and testing the system from IASI-formatted files of radiances to L2 products prior to the launch of IASI. This will extend all the way to delivery of products (subsetted radiances, principle components, and retrieval products) to the weather modelling centers to allow them to configure and test *their* systems prior to launch.

In the lower part of Fig. 1.2, on the left hand side, is the new preprocessor for the NOAA AVN forecast model, which is currently used to derive the surface pressure; the only ancillary product needed for the retrieval system. We set the entire atmospheric state ($T(p)$, $q(p)$, $O_3(p)$, $CO(p)$, $CH_4(p)$, $CO_2(p)$, SST/LST, surface wind speed, spectral emissivity, and spectral reflectivity) from the AVN model supplemented by emissivity

models (e.g., AVN wind speed is used to drive the English and Wu/Masuda models for spectral emissivity) and that state can be used for diagnostic comparison for each step/iteration within the retrieval system or used as first guess or a-priori information (e.g., wind).

In addition, it is a useful option to compare the retrieval steps to other models or measurements (NCEP-GDAS, ECMWF, ATOVS retrievals, or radiosondes). The "AVN" state can be used as a starting point (for parameters that are not within these datasets, e.g., emissivity). The retrieval system keeps statistics w.r.t. these models in order to (a) spot check for errant retrievals and (b) to identify situations where the retrieval adds information content to the models.

1.1 Revision History & Cross-training using this document

| | |
|---------------|---|
| Aug. 27, 2001 | built initial document from other doc's |
| July 9, 2004 | add overview, NOAA-88, preproc |
| Oct. 22, 2004 | discussed detailed .txt printout for test run |
| Dec. 10, 2004 | IDL diagnostic display routines |
| Jan. 21, 2005 | IDL validation site routines |

1.2 Current Users of This Code

| | | |
|-------------------|-------------|--------------|
| Murty Divakarta | NOAA/STG | 301-316-5004 |
| Lihang Zhou | NOAA/QSS | 301-316-5002 |
| Eric Maddy | NOAA/QSS | 301-316-5012 |
| Xiaozhen Xiong | NOAA/QSS | 301-316-5020 |
| Jennifer Wei | NOAA/NESDIS | 301-316-5014 |
| Fengying Sun | NOAA/NESDIS | 301-316-5023 |
| Michele McCourt | UMBC | 410-455-1929 |
| Juying Warner | UMBC | 410-455-3320 |
| Fricky Keita | GSFC | 301-286-1965 |
| John Blaisdell | GSFC | |
| Lou Kouvaris | GSFC | |
| Guyla Molnar | GSFC | |
| Steve Broberg | JPL | 818-354-1554 |
| Hai-Tien Lee | UMCP | |
| Joan Alexander | UCAR | |
| Vladimir Zavyalov | SDL | 435-797-4116 |
| Cassi Going | SDL | |
| Markus Smith | SDL | |

1.3 Location of Program and Coefficient Files

The code is usually placed off a root directory, which we will refer to as \sim in this text. Two code directories exist: $\sim/code$ is the FORTRAN code and $\sim/prog/idl$ is the IDL code. The coefficient files (e.g., RTA files, regression coef's, etc.), needed for the retrieval and simulator code, are located in a single directory referred to as $\#$ in this document. The " $\#$ " symbol can also be used in the namelist files to implicitly specify the local path to the coefficient files. In this way, namelists do not need to be altered to run on different machines. All files are in subdirectories under this path. The path locations vary from machine to machine and you may choose to build your own local copy. The locations I have used are given in Table 1.1 and are defined in FORTRAN using `src_gsfc/getdefpath.F` and in IDL using `general/cdbpath.pro`

Table 1.1: Paths specified by HOST Environment Variable

| HOST | ~ | # |
|----------|-----------------------------|-----------------------------------|
| orbit91 | /net/orbit0091/home/cbarnet | /home/cbarnet/airsb |
| orbit091 | /net/orbit0091/home/cbarnet | /home/cbarnet/airsb |
| orbit09 | /home/cbarnet | /home/cbarnet/airsb |
| orbit009 | /home/cbarnet | /home/cbarnet/airsb |
| orbit119 | /home/cbarnet | /home/cbarnet/airsb |
| orbit120 | /home/cbarnet | /home/cbarnet/airsb |
| orbit121 | /home/cbarnet | /home/cbarnet/airsb |
| orbit122 | /home/cbarnet | /home/cbarnet/airsb |
| orbit123 | /home/cbarnet | /home/cbarnet/airsb |
| orbit* | /net/orbit0091/home/cbarnet | /net/orbit0091/home/cbarnet/airsb |
| asl | /home/cbarnet | /yam/s1/cbarnet/ |
| dos | c:/airsb | c:/airsb |
| srt8 | /develop/airsb | /develop/airsb |
| | @ \$AIRS_CODE | @ \$AIRS_COEF |

1.4 Location of documentation

The code revision and namelist document files are edited as code upgrades are installed. These files are delivered with the code.

| file | topic (root=~/code/doc/) |
|----------------------|--|
| namelist.ema | airsb & simrad namelist variable summary |
| rspare.ema | documentation and pseudo code for rspare() |
| code.ema | documentation of changes for this year |
| code_old/code_xx.ema | documentation of changes for year=xx |

General documentation files are in PDF and are available upon request. Some are located on the ftp site. The “living” directory is located on /net/orbit0911/home/cbarnet/document.

| files on ftp site | |
|-------------------|---|
| file | topic |
| airsb_code.pdf | this document |
| rs_notes.pdf | General notes on remote sounding theory/definition of levels and layers and conversion of mixing ratio to/from column density UMBC transmittance model Microwave line-by-line theory Cloud clearing theory Regression theory Retrieval theory AIRS Footprint definition |
| airs_tables.xls | EXCEL spreadsheet of AIRS channel properties |
| apod_fit.pdf | Apodization functions |
| kawa3_co2.pdf | Summary of CO ₂ model used in simulations |
| phys640_new.pdf | Numerical methods |

| files on orbit0911:/home/cbarnet/document | |
|---|---------------------------------------|
| living/eos.ema | documentation of AIRS & EOS facts |
| living/gw.ema | documentation of global warming facts |
| living/idl.ema | notes about IDL |
| living/jpl_sys.ema | notes about TLSCF |
| living/pl_sim.ema | notes on use of pl_sim.pro |
| living/umbc_setup.ema | notes about running on UMBC systems |

1.4.1 Fetching AVN files

```
ftp disc2.nascom.nasa.gov
login: anonymous
passwd: your email address
ftp> cd private/data/avn/avn_YYMM/avn_YYMMDD
See Section 3.7 for discussion of the AVN file names.
```

1.4.2 GOES Images

GOES images: <http://airs3.ssec.wisc.edu/airs>

- click on GEO
- click on date

at UMBC I put the on /yam/s2/noaa/goes/

```
ftp cedar.ssec.wisc.edu
login: anonymous
ftp> cd tobin/global
ftp> dir
ftp> cd year_mm_(ddd-ddd)
```

1.4.3 JPL Granule Maps

Granule maps are useful to select granules for processing. You need to find someone with a JPL account and they can fetch them. I pulled down the following into /sep_maps/ on asl

```
scp -r nu.jpl.nasa.gov:/hosts/derecho/export/raid/dom/ files/ops/airs/tlscf/2002/09/airxamap/.pdf .
```

For co-location to UT time you can basically, take the granule # divided by 10 as an approximation to UT hour. Periodic orbital maneuvers correct the Aqua orbit such that it repeats every 16 days, so 15 maps can be used to map out future observations with reasonable precision.

If you have run retrievals for a full day, the table at the top of the .out file has lat,long etc. for each granule. These tables can be used for orbits $N \cdot 16$ days.

1.5 FORTRAN directory structure

The FORTRAN code for the simulator, retrieval, and many diagnostic codes are built from weekly backup zip files (aiYYMMDD.zip) that are unzipped into a directory called ~/code/. See Chapter 2 for details how to build the libraries and code. Many routines are shared. The libraries are arranged in a hierarchy such that routines of a higher level call the routines of lower level, but not vice versa. The RTA's are of level=1, therefore, a module that requires the use of the RTA cannot be stored in the library of level=0.

| directory | level | function (root=~ /code/) | library |
|-----------|-------|--|--------------|
| src_jpl/ | 0 | foundational routines & definitions | libairs.a |
| english | 0 | Microwave emissivity model from S.English | libenglish.a |
| rta66/ | 1 | old 66 level RTA from GSFC | librta66.a |
| rta100/ | 1 | old 100 level RTA from UMBC | librta100.a |
| oss | 1 | AER OSS RTA for CrIS | liboss.a |
| hffp0104 | 1 | Andrew Weir's HFFP RTA for HIRS | libhffp.a |
| mwrta_v4/ | 1 | MIT RTA for AMSU,ATMS,HSB | libmw_v4.a |
| umbc9803/ | 1 | all UMBC RTA's for AIRS, IASI, CrIS | libumbc3.a |
| src_gsfc | 2 | general library, high level <i>e.g.</i> , the Masuda emissivity model | libgsfc. |
| mw_ret | 4 | MIT Max. Likelihood retrieval code | libmitret.a |
| rt_noaa | 4 | NOAA regression code, post-launch version | libnoaa.a |
| sim_noaa | 4 | NOAA regression code, pre-launch version | libsimnoaa.a |
| simrad | 4 | routines in support of simrad.F | libsim.a |
| src | 4 | routines in support of airsb.F | libret.a |
| src_util | 5 | main programs | |

| FORTRAN Code on August 29, 2006 | | | |
|---------------------------------|---------|---------|-------------------------------------|
| directory | # lines | # files | use |
| /src_jpl | 3,153 | 29 | general routines (Planck, etc.) |
| /src_gsfc | 15,943 | 87 | I/O and basic sounding routines |
| /mwrta_v4 | 2,025 | 16 | MIT microwave RTA |
| /english | 738 | 8 | English microwave emissivity model |
| /umbc9803 | 14,845 | 45 | UMBC IR RTA & conversion programs |
| /mw_ret | 4,393 | 27 | MIT microwave retrieval |
| /rt_noaa | 3,518 | 24 | NOAA regression code |
| /src | 53,404 | 132 | GSFC/NOAA Physical SVD code |
| airsb.F | 2,430 | 1 | driver for retrieval code |
| total: | 100,440 | 369 | |
| /simrad | 2,600 | 18 | subroutines to simulation radiances |
| simrad.F | 1,702 | 1 | driver for simulation of radiances |
| total: | 4,302 | 19 | |
| /src_util | 24,935 | 46 | other main programs (utilities) |
| /umbc_cnv | 20,933 | 21 | Routines to convert UMBC RTA files |

1.6 IDL directory structure

The IDL code are built from backup zip files (idYYMMDD.zip) that are unzipped into a directory called ~/prog/idl/. Many routines are shared.

| directory | function |
|-----------|--|
| airs | functions unique to AIRS instrument |
| airsb | functions designed to interface with airsb retrieval code for reading files and plotting |
| amsu | functions unique to AMSU instrument |
| chris | personal crap |
| co2reg | CO2 regression code (student, Nicole Gray) |
| cris | functions unique to CrIS instrument |
| exercise | programs to support various runs w/ real AIRS data. |
| general | general functions and utilities |
| matchup | preprocessor for JPL matchups |
| plot_pro | plotting programs for real data |
| preproc | preprocessor for AIRS granules |
| test_pgm | programs to test various ideas or data files |
| trace | programs for trace gas work |

| IDL Code on March 28, 2005 | | | |
|----------------------------|---------|---------|---|
| directory | # lines | # files | use |
| preproc | 11,680 | 65 | drivers and HDF & WGRIB readers |
| avn_match | 1,124 | 5 | preprocess AVN files |
| ecmwf_match | 2,030 | 15 | preprocess ECMWF files |
| matchup | 3,585 | 12 | preprocess AIRS Matchup Files from JPL |
| total: | 18,419 | 97 | for preprocessing of AIRS |
| general | 6,950 | 77 | general functions (Planck etc.) |
| airsb | 49,100 | 220 | routines to plot FORTRAN output |
| trace | 11,850 | 48 | plots and models for trace gas work |
| cris | 10,825 | 45 | interferometer noise models & IPO IGS stuff |
| airs | 3,570 | 25 | AIRS instrument specific routines |
| amsu | 4,450 | 28 | AMSU instrument specific routines |
| exercise | 4,140 | 28 | Routines for processing specific granules |
| test_pgm | 6,175 | 52 | test programs |
| plot_pro | 11,550 | 38 | plotting programs for granules |

1.7 Short summary of MAIN program files in src_util

| file | topic (root= \sim /code/src_util) |
|--------------|---|
| airs_rad.F | AIRS RTA interface documentation & test program |
| comp_rad.F | μ W & IR RTA interface documentation & test program |
| airsb.F | retrieval code for all instruments |
| simrad.F | simulator code |
| calc_ctbl.F | build channel sensitivity to clouds(p) table |
| ncvtable.F | build a table of NCV functions (NOT USED) |
| idprolst.F | build list of commonly rejected profiles |
| simstat.F | off line retrieval statistics program |
| scan_truth.F | build a table of truth information |
| mergel1.F | merge L1 file(s) |
| mergel2.F | merge L2 file(s) (e.g., used to make EDGSC) |
| cd2mr.F | converts L2 file to mixing ratio format (f/ IPO) |
| jacobian.F | compute Jacobians for trace ggas species |
| 66to100.F | convert 66 level files to 100 level |

1.8 Short description of selected subroutines & functions

And selected useful FORTRAN subroutine locations

| direct | file | topic (root= \sim /code/) |
|----------|--------------|--|
| src_jpl | paramet.com | constants and dimensions |
| src_jpl | bi_test.F | determine if file is ASCII or BINARY |
| src_jpl | openl1.F | open L1 files and read header |
| src_jpl | readl1.F | read L1 record |
| src_jpl | writel1.F | write L1 record |
| src_jpl | brtemp.F | convert radiance to BT |
| src_jpl | planck.F | convert BT to radiance |
| src_jpl | meantemp.F | convert T(p) from levels to layers |
| src_jpl | softexit.F | exit retrieval w/ exit code |
| src_gsfc | dbdt.F | derivative of planck function w.r.t. T |
| src_gsfc | openl2_b.F | open L2 file and read pressure grid |
| src_gsfc | readl2_b.F | read L2 record |
| src_gsfc | writel2_b.F | write L2 record |
| src_gsfc | openl1_b.F | open L1 file and read pressure grid |
| src_gsfc | readl1_b.F | read L1 record |
| src_gsfc | writel1_b.F | write L1 record |
| src_gsfc | ir_rad.F | compute IR radiance |
| src_gsfc | load_noise.F | load GSFC noise files |
| umbc9803 | init_rta.F | initialize UMBC predictors |

Many FORTRAN routines have analogous IDL routines, such as

| direct | file | topic (root=~ /prog/) |
|--------|---------------|--------------------------------------|
| airsb | bi_test.pro | determine if file is ASCII or BINARY |
| airsb | openl1.pro | open L1 files and read header |
| airsb | readl1.pro | read L1 record |
| airsb | writel1.pro | write L1 record |
| airsb | openl2_b.pro | open L2 file and read pressure grid |
| airsb | readl2_b.pro | read L2 record |
| airsb | writel2_b.pro | write L2 record |

1.9 Summary of top-level IDL Programs

And some of the retrieval diagnostic plotting files are:

| file | topic (root=~ /prog/idl/) |
|----------------|--|
| pl_sim.pro | plot the retrieval statistical file $T(p), q(p), O_3(p), CO(p), CH_4(p), CO_2(p)$ |
| pl_ret.pro | plot all the retrieval statistical steps for $T(p), q(p), O_3(p)$ |
| pl_ftn92.pro | plot radiance statistics |
| pl_ampl.pro | plot histogram of amplification factor |
| pl_eigen.pro | plot eigen functions of retrieval steps |
| pl_smat.pro | plot S-matrix functions of retrieval steps |
| pl_lev2.pro | plot $T(p), q(p), O_3(p)$ profiles |
| profsumm.pro | plot rejection criteria w.r.t. errors |
| pl_alpha.pro | plot cloud fraction correlation |
| search_all.pro | Driver to select retrievals within radius of a site |
| build_cnf.pro | Program to build setup for search_all list |
| reform_ret.pro | Program to select retrievals from subset run |
| pl_cmdl.pro | Program to plot trace gas retrievals to matchups |

Chapter 2

How to set-up and compile the FORTRAN code

The steps that are required to be done depends whether you are on a machine that has already had the coefficient files and/or executable files loaded into a common area (*i.e.*, orbit machines at NOAA or asl machines at UMBC). My distribution comes with the following files (NOTE: YYMMDD is the year/month/day the file was generated) See

| | zip | | disk |
|---------------|--------|------------------------|--------|
| file name | size | contents | size |
| aiYYMMDD.zip | 4 MB | FORTRAN code & doc's | 33 M |
| idYYMMDD.zip | 2 MB | IDL code | 7 M |
| airs_coef.zip | 70 MB | AIRS Coefficient files | 94 M |
| airs_test.zip | 400 MB | Example AIRS Granule | 450 MB |

See Section 2.2 for unpacking my FORTRAN and IDL zip files and compiling the code.

The coefficient files (regression, climatology, emissivity, RTA files, etc.) needed for a particular instrument (*e.g.*, AIRS/AMSU/HSB/MODIS, IASI/AMSU/MHS, CrIS/ATMS/VIIRS) all exist on orbit009 at NOAA and can take quite a bit of memory. A zip file for AIRS/AMSU/HSB exists that requires about 100 MB of disk space. Other sets of files can be requested. See Section 2.3 for setting up the coefficient area.

I have one test granule I use to confirm that the code is compiled properly and the coefficient files are stored in an accessible location. This requires a minimum of about 1 GB of disk space. Each run of the retrieval takes about 25 MB. This large size reflects that the namelists I have provided have a lot of diagnostic output. The user can test the IDL preprocessor and/or the FORTRAN retrieval code on previously pre-processed temporary files and/or on the user's preprocessed temporary files. Since I have provided by output from the preprocessor in this zip file there is about a factor of 2 in size of the raw files need for processing a granule. When you run the test copy of the preprocessor you will add about 250 GB to the size on disk since you will now have the original data, my temporary copy, and your temporary copy. Also, you can run my diagnostic routines on the test datafiles. See Section 2.4 for running the test code.

2.1 Set-up & Environment Variables

A number of path variables are **required** for the IDL programs and are **optional** in the FORTRAN files and unix level scripts. These environment variables can be used from the LINUX level. Mostly, there are used by the IDL programs. For example, "HOST" is parsed to determine where to find data files. On machines other than NOAA orbit and ASL machines the AIRS_CODE and AIRS_COEF are used to find coefficient files and code files. The AIRS_L1b, AIRS_ECM, AIRS_AVN, and AIRS_nl are the locations to getch HDF files,

ECMWF RTP or WGRIB files, AVN WGRIB files, and default namelists. All of these files are read-only. TMP is a location to WRITE temporary files, *i.e.*, unzipping gzip files, WGRIB output files and log files from preprocessing programs. AIRS_tmp is a location to store L1b and L2 temporary files. AIRS_ret is the location for retrieval files. AIRS_tmp and AIRS_ret are both assumed to have a YYYY/MM/DD directory tree underneath and they can be equal, if desired.

Here is a summary of the environment variables used by IDL:

Table 2.1: Environment Variables used by the off-line code

| | |
|-----------|--|
| HOST | machine name - used to locate coefficient files |
| USER | User Name (echoed in .out file) |
| AIRS_CODE | location of the IDL and FORTRAN code |
| AIRS_COEF | location of the coefficient files |
| AIRS_L1b | directory tree for L1b HDF files |
| AIRS_ECM | directory tree for ECMWF files (UMBC RTP or WGRIB) |
| AIRS_AVN | directory tree for AVN wgrib files |
| AIRS_nl | directory for default script and namelists |
| TMP | temporary directory for output summaries from autoproc and preproc.pro |
| AIRS_tmp | directory tree for BINARY (output) L1 & L2 files |
| AIRS_ret | directory tree for retrieval output |

2.1.1 Example bash login script for external users (not NOAA or UMBC)

```
#!/bin/bash
#
  umask 022

USER=unknown
HOST=unknown

# location where you unpacked aiYYMMDD.zip & idYYMMDD.zip
AIRS_CODE=/disk3/pub/example
# location where you unpacked airs_coef.zip
AIRS_COEF=/disk3/pub/example/airsb
# location where you unpacked airs_test.zip
AIRS_TEST=/disk3/pub/example/airs_test

# location for temporary files
TMP=${AIRS_TEST}/temp

# location of data files
AIRS_nl=${AIRS_TEST}/nl
AIRS_L1b=${AIRS_TEST}/l1b
AIRS_AVN=${AIRS_TEST}/avn
AIRS_ECM=not.set

# location to store output
```

```

AIRS_tmp=${AIRS_TEST}
AIRS_ret=${AIRS_TEST}

echo $AIRS_tmp

export USER HOST AIRS_CODE AIRS_COEF
export AIRS_nl AIRS_L1b AIRS_AVN AIRS_ECM
export AIRS_tmp AIRS_ret

IDL_DIR=/usr/local/rsi/idl
# NOTE: /rsi might be /bin on your machine
IDL_DEVICE="X"
IDL_STARTUP="${AIRS_CODE}/prog/idl/startup.pro"
IDL_PATH="${IDL_DIR}/lib:./:${AIRS_CODE}/prog/idl:
  ${AIRS_CODE}/prog/idl/general:${AIRS_CODE}/prog/idl/airs:
  ${AIRS_CODE}/prog/idl/airsb:${AIRS_CODE}/prog/idl/preproc:
  ${AIRS_CODE}/prog/idl/matchup:${AIRS_CODE}/prog/idl/avn_match:
  ${AIRS_CODE}/prog/idl/ecmwf_match:${AIRS_CODE}/prog/idl/exercise:
  ${AIRS_CODE}/prog/idl/cris:${AIRS_CODE}/prog/idl/amsu:
  ${AIRS_CODE}/prog/idl/chris:${AIRS_CODE}/prog/idl/plot_pro:
  ${AIRS_CODE}/prog/idl/trace:${AIRS_CODE}/prog/idl/preproc/eff:
  ${AIRS_CODE}/prog/idl/test_pgm"

export IDL_DIR IDL_DEVICE IDL_STARTUP IDLPATH

```

2.1.2 Example bash login script for NOAA LINUX machines

You can look in my home space on orbit0911 /net/orbit0911/home/cbarnet/.bash_profile to see my setup for using the system that is resident on orbit0091. The bash profile sets up the Portland Group Compiler, IDL, and the preprocessor environment variables. This is a general bash shell script to run using my coefficient files located on orbit0091

```

# items for your files
EXPORT HOST="orbitXX"
EXPORT USER="yourname"
EXPORT TMP=~ /temp
EXPORT AIRS_ret=/diskN/pub/ret_dir
EXPORT AIRS_tmp=/diskN/pub/local

# location of items in my data areas
EXPORT AIRS_AVN=/net/orbit0091/disk2/avn
EXPORT AIRS_ECM=/net/orbit0091/disk2/ecmwf
EXPORT AIRS_L1b=/net/orbit0091/disk2/tlscf
EXPORT AIRS_nl=/net/orbit0091/home/cbarnet/airsb/trace_nl

# location of code and coefficients
EXPORT AIRS_CODE=/net/orbit0091/home/cbarnet
EXPORT AIRS_COEF=/net/orbit0091/home/cbarnet/airsb

```

```

# setup for Portland Group FORTRAN
EXPORT PATH=$PATH:/usr/local/pgi:~/code/exe
EXPORT LD_LIBRARY_PATH=/usr/local/pgi

# setup for IDL
EXPORT IDL_DIR=/usr/local/rsi/idl
EXPORT IDL_DEVICE="X"
EXPORT IDL_STARTUP="/net/orbit0091/home/cbarnet/prog/idl/startup.pro"
EXPORT IDL_PATH="${IDL_DIR}/lib:./:/home/yourname/myidl:
/net/orbit0091/home/cbarnet/prog/idl:
/net/orbit0091/home/cbarnet/prog/idl/amsu:
/net/orbit0091/home/cbarnet/prog/idl/airs:
/net/orbit0091/home/cbarnet/prog/idl/airsb:
/net/orbit0091/home/cbarnet/prog/idl/cris:
/net/orbit0091/home/cbarnet/prog/idl/exercise:
/net/orbit0091/home/cbarnet/prog/idl/general:
/net/orbit0091/home/cbarnet/prog/idl/matchup:
/net/orbit0091/home/cbarnet/prog/idl/plot_pro:
/net/orbit0091/home/cbarnet/prog/idl/preproc:
/net/orbit0091/home/cbarnet/prog/idl/preproc/eff:
/net/orbit0091/home/cbarnet/prog/idl/trace:
/net/orbit0091/home/cbarnet/prog/idl/test_pgm"

```

2.1.3 Example of tsch login script for asl machines

You can look in my home space on asl:/home/cbarnet/.login to see my setup for the ABSOFT Compiler, IDL, and the preprocessor environment variables. This is a tsch shell script.

```

# items for your files
# NOTE: USER and HOST should be already set

setenv TMP "/home/cbarnet/temp"
setenv AIRS_tmp "/yam/s1/cbarnet/local"
setenv AIRS_ret "/yam/s1/cbarnet/ret_dir"

# location of items in my data areas
setenv AIRS_AVN "/yam/s1/cbarnet/avn"
setenv AIRS_ECM "/yam/s1/cbarnet/ecmwf"
setenv AIRS_L1b "/yam/s1/cbarnet/tlscf"
setenv AIRS_nl "/home/cbarnet/trace_nl"

# setup for ABSOFT FORTRAN
setenv ABSOFT /usr/local/absoft
setenv PATH ./home/cbarnet/bin:/usr/local/absoft/bin: ...
setenv MANPATH /usr/local/absoft/man: ...

# setup for IDL
alias idl "/usr/local/rsi/idl/bin/idl"

```

```
setenv IDL_PATH "+/usr/local/rsi/idl/lib"
setenv IDL_PATH "${IDL_PATH}:/home/cbarnet/prog/idl"

setenv IDL_DIR "/usr/local/rsi/idl"
setenv IDL_DEVICE "X"
setenv IDL_STARTUP "/home/cbarnet/prog/idl/startup.pro"
```

2.2 How to compile or re-compile airsbf

The FORTRAN code is distributed via my weekly backup zip file called aiYYMMDD.zip, where YY is the year, MM is the month, and DD is the day that the zip file was created. There are makefiles for Portland Group Compilers (PG) for the NOAA orbit machines (Makefile.orb), ABSOFT compiler for the ASL machines (Makefile.asl), and SGI FORTRAN compilers, (Makefile.sgi). At this time, I have not been successful at using the GNU g77 compiler. G77 options to use static variables (pre-initialized to zero) and BIG-ENDIAN file formats must be used and I haven't found those yet (haven't tried very hard, either).

I have a script to rebuild all the FORTRAN and IDL in ~/bin/rebuild.orb on orbit009l which is distributed within the aiYYMMDD.zip file. The script deletes all the exe's and libraries, then rebuilds each library. I would recommend you read the contents of this file before running it so that you understand exactly what the script is doing. I have placed a copy of the script in Section 2.2.1.

For most of the machines used (asl, orbit) the commands to unpack the IDL and FORTRAN and compile all the libraries and main programs are built into scripts in the code directory. You can re-build the code with the following commands (xxx=asl or xxx=orb).

1. copy the zip files into the directory you want for the FORTRAN code and IDL code to reside.
2. mkdir code
3. mkdir prog
4. cd code
5. unzip ../aiYYMMDD.zip rebuild.xxx
6. rebuild.xxx > rebuild.yymmdd

NOTE: ignore warnings about .o files not found. This occurs the first time you rebuild because your directories are empty.

On other machines, use one of the rebuild scripts as a starting point to build the FORTRAN code on your machine.

| | |
|------|-----------------------------------|
| .orb | LINUX w/ Portland Group compilers |
| .asl | LINUX w/ ABSOFT compilers |
| .sgi | IRIX w/ SGI compilers |
| .vf | DOS w/ Compac Visual Fortran |

2.2.1 Example of rebuild script for orbit machines

```
#!/bin/sh
set +vx
```

```
umask 022

# this rebuilds from zip files (weekly backup) onto orbit machines
# -----
# **** requires PORTLAND GROUP COMPILER on host machine ****
# copy .zip files into your root area
# mkdir code (if they don't exist already)
# mkdir prog (if they don't exist already)
# cd code
# rebuild.orb YYYYMMDD > rebuild.yymmdd

usage="date YYYYMMDD needs to be specified"
dateid=${1:?$usage}

cd ../prog
unzip -o -u ../id$dateid.zip <-- unzip entire IDL library

cd ../code
/bin/rm -rf lib <-- remove old libraries
/bin/rm -rf exe <-- remove old executables
unzip -o -u ../ai$dateid.zip <-- unzip entire FORTRAN library

mkdir lib
mkdir exe

cd src_jpl
cp softexit.srt softexit.F <-- this 1 routine is compiler dependent
cp Makefile.orb Makefile <-- get correct Makefile
/bin/rm *.o <-- get rid of old objects
make <-- build src_jpl library

cd ../src_gsfc
cp Makefile.orb Makefile <-- and so on.....
/bin/rm *.o
make

cd ../mwrta_v4
cp Makefile.orb Makefile
/bin/rm *.o
make

cd ../english
cp Makefile.orb Makefile
/bin/rm *.o
make

cd ../umbc9803
cp Makefile.orb Makefile
/bin/rm *.o
make

cd ../mw_ret
cp Makefile.orb Makefile
```

```

/bin/rm *.o
make

cd ../rt_noaa
cp Makefile.orb Makefile
/bin/rm *.o
make

cd ../sim_noaa
cp Makefile.orb Makefile
/bin/rm *.o
make

cd ../src
cp Makefile.orb Makefile
/bin/rm *.o
make

cd ../simrad
cp Makefile.orb Makefile
/bin/rm *.o
make

cd ../src_util                                <-- all libraries are now built
cp Makefile.orb Makefile
/bin/rm *.o
make airs_b103                                <-- build ret code
make simrad103                                <-- build simulator code
make comp_rad                                 <-- build radiance test program
make calc_ctbl                                <-- and build utilities
make simstat
make idprolst
make append_nobs
make jacobian
make avgcld
make cnv_gbgrid

exit 0

```

2.3 How to build the coefficient database

To build and test a working environment the following steps need to be done once.

1. (if not on orbit or asl machine) Locate a disk with more than 300 MB of free space. Make a directory to build the library. (NOTE: the top level directory can be the same directory that has code and prog directories in it.) I usually call this “airsb”.
2. Change into that directory.
3. Copy airs_coef.zip into the directory.

4. unzip airs_coef.zip -o
 - o overwrites existing files, if needed
 - u only (if desired) only pulls items that have changed
 - some version of zip require an option to create directories
5. delete airs_coef.zip, it is no longer needed.
6. Check the directory tree to see if it agrees with Section 2.3.1 & Section 2.3.2.
7. Make sure the environment variable HOST, USER, and AIRS_COEF is set (if you are not on asl or orbit machines) to the directory in step #1 (top of tree).
8. Make sure the environment variable AIRS_CODE is set to the location of the ~/code directory.
9. You should now be able to compute radiances for AIRS for my three test cases (polar, mid-latitude, and tropical case defined in #/level2/sim3.asc) by typing the following command from any directory.


```
$AIRS_CODE/code/exe/comp_rad.exe sim3 test.asc -v8b
```

NOTE: -v8b is the default at this time, so it does nothing. If the RTA were to be updates, -v8b would still use the V8 RTA for AIRS.
10. You can compare your results with my results in \$AIRS_coef/comp_rad_out. These are ASCII files so they can be easily compared with diff or can be read using vi or emacs. The files can be read by openl1.b.pro in IDL or openl1.b.F in FORTRAN.
 - IR_test.asc are the 2378 AIRS radiances for the 3 cases in sim3
 - AM_test.asc are the 15 AMSU radiances for the 3 cases in sim3
 - MH_test.asc are the 5 HSB radiances for the 3 cases in sim3.asc

2.3.1 Coefficient Files for the Forward Computation of Radiances

The location of the files necessary to compute radiances for all instruments (AIRS, CrIS, IASI, NAST) are located in the following directories.

| | |
|-----------------|-------------------------------------|
| #/RTA/ | RTA files |
| #/solar/ | solar radiance files |
| #/noise/ | noise files |
| #/surface/ | Masuda emissivity model ' |
| #/comp_rad_out/ | Output file for a comp_rad test run |

where “#” is defined by the ENVIRONMENT variable “HOST”. See Table 1.1 in Section 1.3.

Many programs use these files via the radiative and transmittance algorithm libraries (*e.g.*, code/umbc9803 & code/mwrta.v4). Some of the programs using the forward model are

| | |
|-------------|--|
| comp_rad.F | RTA interface documentation & test program |
| airsb.F | retrieval code for all instruments |
| simrad.F | simulator code for all instruments |
| calc_ctbl.F | build channel sensitivity to clouds(p) table |
| ncvtable.F | build a table of NCV functions (NOT USED) |
| jacobian.F | compute Jacobians for trace gas species |

2.3.2 Coefficient Files REQUIRED by the Retrieval Code

The location of the files required to run retrievals are located in

| | |
|-------------------------|---------------------------------------|
| <code>#/angcorr</code> | local angle correction coefficients |
| <code>#/avgcld/</code> | cloud averaging tables |
| <code>#/errest/</code> | ensemble error estimates |
| <code>#/level2/</code> | US STD atmosphere (see Chapter 11 |
| <code>#/mit_ret</code> | files for mit maximum likelihood code |
| <code>#/ncvtbl/</code> | noise co-variance tables (NOT USED) |
| <code>#/noise/</code> | noise files |
| <code>#/regress/</code> | regression coefficients |
| <code>#/RTA/</code> | RTA files |
| <code>#/solar/</code> | solar radiance files |
| <code>#/surface/</code> | Masuda emissivity model ' |
| <code>#/rtaerr/</code> | model error estimates for real data |
| <code>#/tuning/</code> | tuning coefficients for real data |

where “#” is defined by the ENVIRONMENT variable “HOST”. See Table 1.1 in Section 1.3.

2.4 Unpacking and Running the Test Case

In this section we will go through unpacking and running the test granule. In this zip file I have put

- A single Granule of AIRS, AMSU, and HSB Level-1 files.
- Temporary files generated by my code for the granules.
- A run of the retrieval.
- Configuration files for plotting the retrieval results.

Begin by unzipping the complete test directory tree

1. Find a location with about 300 MB of free space.
2. make a directory called `airs_test`
3. `cd airs_test`
4. copy the `airs_test.zip` file into the directory
5. `unzip airs_test.zip -o`
 - o overwrites existing files, if needed
 - u only (if desired) only pulls items that have changed
 - some version of zip require an option to create directories
6. delete `airs_test.zip`, it is no longer needed.

I have not provided ECMWF, per our agreement with the European community. Many of the pre-process steps will require that you specify that ECMWF has not been included. In the example here I have used commands that do not assume ECMWF is available. To build your own temporary files

1. make sure your IDL environment is set-up properly. See Section 2.1 to set it up.
2. Set-up all the airs_b environment variables. This can be done in your login script (see Section 2.1) or you can run a script from the LINUX command line.
3. To run the retrieval (this will take about 20 minutes) from files I pre-processed and provided in the test directory type

```
cd $AIRS_ret/2003/01/14/ret
procret.com a01bin halftest v40
```

4. Enter IDL and type

```
IDL> pl_sim, 'test'
```

Select menu item #1 or #2. You will see statistics of your run versus a run I did on my machine. For a good run you will see a solid red line in all 3 panels and a dashed black line in all 3 panels. The black lines are plotted first and the red solid line over-writes the black line (if they are identical). If the regression fit is identical the red line is not printed. If you see black solid lines on any of the panels, then there are minor differences, possibly due to machine floating point differences. I would not expect this. If the dashed line is red or you see both a red and black dashed line then the runs were not identical. I might expect to see the red dashed line over-writing the black line due to machine floating point differences. See Section 9.3 for a discussion of the pl_sim.pro program.

The file test.plt in the retrieval directory is directing the plot that you are making.

5. (optional, To do this you will need to copy my avn file to the temporary directory, since we haven't created it yet). You can create statistics in common (in case rejection thresholds is the reason for differences in instruction #4) or to look at ocean cases.

```
cp $AIRS_tmp/2003/01/14/tmp0/avn_051.bin $AIRS_tmp/2003/01/14/tmp
```

```
IDL> stat2, 'a00bin', 'a01bin', 'halftest', /avn
```

```
IDL> pl_sim, 'test', menu=4
```

```
IDL> stat2, 'a00bin', 'a01bin', 'halftest', /ocean, /avn
```

```
IDL> pl_sim, 'test', menu=4
```

6. make sure you are in the airs_test directory. Edit the file run_preproc.pro to point to your environment variables ... or ... you can delete the IDL SETENV commands if you have setup and EXPORTED the environment variables at the LINUX level. Note that I have an option, /build, for when I preprocessed the data files. You will NOT use that option.

7. Go into IDL and type the following (this command takes less than two minutes and will print what it is doing to the screen)

```
IDL> run_preproc
```

8. exit IDL

9. See Chapter 3 for details on what just happened. Your pre-processed temporary files are in the directory \$AIRS_tmp/2003/01/14/tmp. The files in this directory are

| | |
|----------------|---------------------------------------|
| airs_051.bin | AIRS level-1 binary file |
| airs_051.nen | AIRS NEΔN ASCII file |
| amsu_051.bin | AMSU level-1 binary file |
| amsu_051.nen | AIRS NEΔN ASCII file |
| hsb_051.bin | HSB level-1 binary file |
| hsb_051.nen | AIRS NEΔN ASCII file |
| calbuf_051.bin | Level-1 QA binary file |
| avn_051.bin | Level-2 AVN forecast binary file |
| avn_051b.bin | Copy of avn_051.bin (reference state) |

10. `cd $AIRS_ret/2003/01/14/ret`
11. To run the retrieval (this will take about 20 minutes) from files you just preprocessed type
`procret.com a02bin fulltest v40`
12. Enter IDL and type
`pl.sim, 'test'`
Select menu item #3. See instruction #4 above for what you should expect to see.
13. Many of the diagnostic programs can be played with at this point. Routines like `pl.ampl`, `pl.realgran`, `pl.eigen` will work. I have made a IDL driver script to illustrate some of the programs used to make the figures in the Sections on those programs Note that a slightly older version of the retrieval system was used to make those figures and I compared to ECMWF, which is unavailable in this distribution.
`IDL> run_eps`

2.5 Using the PC version of the code with the COMPAC compiler

You can run this code on laptops and desktops and easily update the code from the zip files. NOTE: in this document “/” will refer to a subdirectory even though in DOS it is a backslash. When using the `airsb` code and associated namelists the “/” is automatically converted to a backslash and in most gnu software (emacs, etc.) you can use either.

A CD is available from C. Barnet that has the complete package for running AIRS granules that have been already preprocessed (a CD with temporary files). The CD contents should be copied to `c:/airsb`.

- All coefficient files for running most recent AIRS namelists. This includes RTA file, regression coefficients, solar radiance, cloud tables, MIT covariance files, and example level2 files.
- The zip file used to extract code
- The expanded zip contents and compac workspace documents.
- The object and executable code generated by the compac compiler
- A single granule set (Jan. 14, 2003, granule 051) of temporary L1 and L2 forecast files.
- Examples of run-time scripts and namelists.
- A batch file, `runairsb.bat`, that emulates the LINUX script `procret.com`.
- Output from recent runs with the test case.

Since the code is already compile you do not need to compile it to use it. To update the code from my zip files and re-compile you can follow the following instructions.

- copy the new `aiYYMMDD.zip` file into the directory `c:/airsb/code` and use a my documents windows to select the zip file.
- unzip the file
 - click on file in upper left corner
 - click on extract all
 - dialog window will pop-up, unzip contents into `c:/airsb/code` (default will be `c:/airsb/code/aiYYMMDD`)

- fix rta_stat.com: This file on LINUX allows any UMBC RTA. IASI or NAST require 256K to load the RTA, therefore, to run on PC's with less memory you need to edit this file. You can run AIRS or CrIS RTA's on machines with 128 MB by editing `c:/airsb/code/umbc9803/rta_stat.com`. One problem is that windows level (*e.g.*, my documents) treats a .com file as a DOS executable program and does not open it in any word processor. I use emacs under a MSDOS window to accomplish this.
- then in my documents window, click on airsb.dsw (this is the compaq workspace environment).
- You can select the airsb103 project and then click on rebuild all.

The test case is in `c:/airsb/test/ret/2003/01/14/ret`. The runtime scripts is `runairsb.bat`, that emulates the LINUX script `procret.com`. Goto that directory and type

```
runairsb runID g050 v45
```

to run granule 50 configuration file and v4.5 namelists.

2.6 what are G401 and G422 files

G401's are a concatenation of all the 1st scan lines of a full day of data AIRS. G422's are the same idea, but the 22nd scan line is used. Since there are 45 scan lines in a granule and 240 granules a day this data set constitutes 1/45 of a day of data or

$$240 \text{ granules} / 45 = 5.3 \text{ granules in size.}$$

Therefore, a run will take 5.3 times longer than running a single granule, but there is global coverage, day/night, ocean/land, polar/mid-lat/tropical cases.

There are also G401's available for AIRS, CrIS, and IASI that are simulated from the real G401 datasets. The ECMWF forecast and AIRS retrieval products were merged to form a reasonable simulation truth set. Then AIRS (v8b RTA), IASI (v7 RTA) and CrIS (a number of v7 RTA's for different OPD's) radiances were generated. These were used to support product comparison and instrument trade studies in late 2004.

To run these on the PC version you can use a CD with the temporary files without copying them to your harddisk. The run time is approximately the same on most laptops and desktops for temporary files stored on CD's and hard-disks because the code is dominated by CPU cycles.

The g401 and g422's that exist for real data (real) and simulated data (sim) at this time (10/7/05) are given in Table 2.2.

Table 2.2: AIRS G401 file set

| date | G401 real | G422 real | G401 sim | G422 sim |
|----------------|--------------|--------------|-------------|-------------|
| Sep. 6, 2002 | yes | yes | yes | no |
| Sep. 29, 2002 | yes | yes | yes | no |
| Jan. 25, 2003 | yes | yes | yes | no |
| Sept. 29, 2004 | yes | no | no | no |
| Mar. 6, 2005 | yes | no | no | no |
| Sep. 6, 2005 | yes | no | no | no |

And there are some CD's available with specific granules on them.

Table 2.3: AIRS granules available on CD

| date | granules | use |
|---------------|-------------------|-----------------------|
| Sep. 6, 2002 | 16,82,129,193,208 | daytime, land & glint |
| Sep. 6, 2002 | 17,58,126,127,167 | Africa and Australia |
| Nov. 16, 2002 | 1, 83, 111, 194 | SGP & Egypt-1 D/N |

The program to build the G401's is located on `~/prog/idl/exercise/build_g401.pro`.

1. build the file Pre-process an entire day of granules, e.g., `autoproc, 2002,9,6`
2. Copy the `cnf` file, `020906_ecm.cnf`, (with all ≈ 240 granules specified) into a new directory
3. Run `build_g401, '020906_ecm'`

My original notes on how I did it (very sparse) is on `orbit0911:/home/cbarnet/document/living/howto.ema`
`build_g401.pro` calls FORTRAN programs to merge multiple files into a single file

- `mergel1.F` merges L1 files (3 scan lines from AIRS & HSB, 1 scan line from AMSU) into 1 file for AIRS, AMSU, and HSB.
- `mergel2.F` merges the AVN and ECMWF files into 1 file for each.
- `mergeqa.F` merges the QA information into 1 file

Chapter 3

preproc: Processing Real AIRS Data

This documentation written by Ryan Caveney on 31 March 2003, based on notes from a talk given by Chris Barnet on 12 March 2003.

3.1 Why do we have a preprocessor at all?

JPL sends the Level One-B (level 1B) input data in HDF format, which requires adding Fortran libraries. It is very easy to read in IDL (version 4 or greater). The code to do this is in `~prog/idl/preproc/rd_XXXXhdf.pro`, where XXXX is the name of the instrument: `airs`, `amsu` or `hsb`. At first, `preproc.pro`'s job was just to read in the JPL HDFs and output our local level 1B format, in binary or ascii. In the past, our retrieval code (`~/code/exe/airsb103.exe`) read our native format using `~/code/src_jpl/openl1.F`, `readl1.F`, and `writel1.F`, but now we use `~/code/src_gsfc/openl1_b.F`, `readl1_b.F`, and `writel1_b.F`.

The following preprocessors are envisioned:

| | |
|-------------------------|--------------------------------|
| DAAC HDF Granule Files | <code>prog/idl/preproc</code> |
| JPL HDF Matchup Files | <code>prog/idl/matchups</code> |
| NOAA BINARY Sonde Files | ask Murty Divarkarla |
| NOAA BINARY Grid Files | <code>prog/idl/noaagrid</code> |
| UMBC CLEAR RTP Files | <code>prog/idl/umbcrtp</code> |

3.2 What does the preprocessor do?

1. Re-order AIRS footprints. In the JPL HDF files, the fields of view are stored as they are obtained, 90 per AIRS or HSB scan line but just 30 per AMSU scan line; geographically. The retrieval code expects three AIRS and three HSB scan lines (270 FOV's) for every AMSU scan line (30 FOV's). Therefore, we need to reorder the AIRS and HSB footprints so that each group of nine which corresponds to a single AMSU spot would be consecutive in the input; `preproc.pro` does this.
2. The one required ancillary input for the AIRS PGE is the aviation forecast files (AVN) surface pressure for use in our forward radiance calculations. The surface pressure value actually used in the retrieval is an interpolation of AVN `P_surf` as a function of latitude, longitude, temperature, and local topographical altitude. This process is performed by the IDL programs `fetch_avnfg.pro`, `fill_avnfg.pro` and `wgrib_avn.pro`, which spawn an executable called `wgrib`. Since we're reading the whole file anyway, and already processing a fair number of fields, we may as well take the others, too: we construct surface emissivity (via a Masuda model, based on surface wind speed) and full profiles of temperature, humidity, ozone, CO, CO2 and CH4, which gives us enough to compute a radiance. This full state is written

to `avn.GGG.bin` in the `$AIRS_TMP` directory (more on this below), and may be used as truth or as a first guess.

A summary of the steps in building the AVN forecast file

- compute %land via JPL recipe
 - interpolate all parameters to latitude,longitude, and time of AIRS FOV
 - compute `Psurf` at the observation topography (altitude above reference geode (topog)).
 - compute English model, `e(uW)`, for Ocean cases
 - compute Masuda model, `e(IR)`, for Ocean cases
 - fix problems with AVN forecast (zero or negative `q(p)`, `T(p)`, etc)
 - convert mixing ratio to 100 levels of layer column density
 - add `CO2,CO,CH4` profiles
3. Local angle correction. The algorithm for doing this comes from Larry McMillan at NOAA, but since it really is a change to the input radiances, and JPL now packages the coefficients for the huge matrix multiplications into another HDF file, it is most convenient to handle it in our system as part of the IDL preprocessor. What is this correction all about? Well, our cloud-clearing algorithm assumes that the only significant difference in the brightness temperature seen between the nine AIRS spots associated with a single AMSU spot is due to clouds. However, since AIRS steps in different angular increments than the AMSU does, 6 out of the 9 AIRS spots are offset from the AMSU angle by 1.1 degrees; this causes a difference in the observed radiances due to differing optical path lengths, so it is necessary before cloud clearing to correct the AIRS observations to what they would have been if they had all been measured at the AMSU angle. The procedure for doing this involves principle component (eigenvector) decomposition of mean differences of the radiance in each channel from some ensemble average at each AMSU spot angle, and then multiplication of the result by some large correlation matrix. The coefficients for doing this are read by `rd_lachdf.pro` (`rd_lacbin.pro` and `rd_lacnoaa.pro` handle earlier, superseded formats of the same data) and the computations are done by `airs_lac.pro`. Note: `airs_lac.pro` operates on entire scan lines; if for some reason you are working with less than a full scan line, you will need to use `sngl_lac.pro` instead.
4. Construction of the reference or “truth” files for making comparisons to forecasts. Scott Hannon at UMBC turns the ECMWF forecast into “RTP” files, which are a type of HDF but reordered to match the distribution of AMSU footprints used in retrievals. We then change the UMBC emissivity model and trace gas profiles to match the way JPL operates, we fix “bad” or missing data (like zeroes) to prevent errors and crashes in the downstream computations, insert surface pressure from the AVN file since UMBC does not interpolate ECMWF `P_surf` by location or correct for topographic height. We then use the new surface pressure to fix the bottom of the atmosphere, which often requires extrapolation rather than interpolation. In order to make fuller comparisons, the (unused) ECMWF surface pressure goes into one of the `rspace` fields in the retrieval output. Note that other than changing the surface pressure, we do not interpolate ECMWF to the AMSU footprint; therefore, since ECMWF data is presented on a 1x1 degree grid and every three hours, it may be offset from the satellite observations by up to 90 minutes and a whole AMSU footprint. Also, the alteration we make near the surface means we have no good truth for doing comparisons in the lowest two kilometers: extrapolation is always dangerous, and interpolation of the AVN surface pressure may go seriously wrong, especially over land with rapid topographic variation as AVN pressures are calculated on sigma levels. All of this work is actually also done by the `fill_avnfg` program also, since so much of the data is reused in constructing a complete state for a first guess.
- NOTE: UMBC picks closest 0.5 degree grid cell to center FOV and converts to 100 level. They add `CO,CH4,CO2`.
 - add `e(uW)`, `e(IR)` from English and Masuda model using ECMWF `Psurf`.

- fix problems (zero $q(p)$, $T(p)$, etc)
- substitute AVN Psurf
- add -0.4 K to Tskin for Ocean (pland=0) cases.

3.3 How do I actually *use* this fabulous preprocessor?

The airsb code has the following limitations

- 99,000 individual cases
- 480 files

Since there are 1350 retrievals in a granule, there is an upper limit of 72 granules that can be processed in one run.

The first step in running it is to set a number of environment variables in your UNIX shell (see Section 2.1) , perhaps in your .cshrc/.kshrc/etc. profile script. Set AIRS.L1B to a particular directory above that in which the level 1B data files in HDF from JPL are to be found; when preproc runs it will add several extra directories onto the end of this path, corresponding to the year, month, day, and instrument to which the data belong. For a date expressed as year YYYY, month MM and day DD, AIRS data will be searched for in \$AIRS.L1B/YYYY/MM/DD/airibrad/. The last directory name is the ESDT (Earth Science Data Type) name assigned by the Goddard DAAC; the analogous names for AMSU and HSB are airabrad and airhbrad, respectively. An option exists in preproc.pro, /asl, to set the directory tree equal to \$AIRS.L1b/AIRXBRAD.002/YYYY.MM.DD/ where X=A,I,or H. The size of these files is given below.

| | | |
|-------------|-------------|------------------------------|
| AMSU | 540K/G | 0.130 GB/day |
| HSB | 1.76MB/G | 0.425 GB/day |
| AIRS | 121.3MB/G | 29.2 GB/day |
| WGRIB AVN | 26.5MB/file | 318 MB/day |
| RTP ECMWF | 33MB/file | 7.92 GB/day |
| WGRIB ECMWF | 194 MB | 1.55 GB/day (59% if gzipped) |

Similar path variables indicate where to find AVN and ECMWF forecast data; AVN data should be in \$AIRS_AVN/YYYY/MM/ and ECMWF data (in UMBC's .rtp format) should be in \$AIRS_ECM/YYYY/MM/. Also, set AIRS.TMP to point to an area the preprocessor can use for scratch work and temporary files which will be frequently overwritten; into this directory will go dump files made by the wgrib utility for reading AVN files, and several progress report outputs by the preprocessor. These will be ASCII text files with names of the form preproc_XXX.ZZZ, where XXX is the hostname of the machine on which the preprocessor was run, and ZZZ is either summ, log or err. The .summ file contains a one-line summary of each granule of data which was processed, the .log file contains a detailed description of what the preprocessor actually did, and the .err file contains descriptions of anything the preprocessor thought was "bad" and therefore "fixed". Finally, set AIRS_RET to the location the output of the final retrieval should go; again, this will have /YYYY/MM/DD/ tacked onto the end.

For a given date, mm/dd/yyyy, the preprocessor will look for the following directory tree substructures:

| | |
|----------------|---|
| AIRS HDF | \$AIRS_L1b/yyyy/mm/dd/airibrad/ |
| AMSU HDF | \$AIRS_L1b/yyyy/mm/dd/airabrad/ |
| HSB HDF | \$AIRS_L1b/yyyy/mm/dd/airhbrad/ |
| AVN WGRIB | \$AIRS_AVN/yyyy/mm/gblav.PGrbFxx.YYMMDD.yyz xx=03,06,09 forecast yy=00,06,12,18 run time, 12 files in all per day |
| RTP ECMWF | \$AIRS_ECM/yyyy/mm/ircenterof3x3.YYMMDDgxxx.op.rtp for xxx = granule # 001 to 240 |
| preproc output | \$AIRS_tmp/yyyy/mm/dd/tmp |
| ret products | \$AIRS_ret/yyyy/mm/dd/ret (default) \$AIRS_ret/yyyy/mm/dd/retN (ret=N option) |

For more information about the AVN file names see Section 3.7 and Section 1.4.1.

If the /asl option is used then the HDF directory tree is assumed to be

| | |
|----------|-------------------------------------|
| AIRS HDF | \$AIRS_L1b/AIRIBRAD.002/yyyy.mm.dd/ |
| AMSU HDF | \$AIRS_L1b/AIRABRAD.002/yyyy.mm.dd/ |
| HSB HDF | \$AIRS_L1b/AIRHBRAD.002/yyyy.mm.dd/ |

To run the preprocessor, you must start IDL. Once in IDL, and having ensured that the proper top directory (such as /airs/prog/idl/) is in your IDL_PATH UNIX environment variable, you could simply type

```
idl/preproc/preproc, year, month, day, granule_list,
[/rtp], [/nohsb], [root=' '], [/asl]
[/lacplot], [lacver=5], [/nolac], [ret=2],
```

where year=YYYY, month=MM and day=DD from above, and granule_list is an IDL array of long integers giving the ID numbers of the granules you wish to process; to do a whole day (240 granules) you could try lindgen(240)+1 (which will generate an array of the numbers from 1 to 240, inclusive). However, this would be a bad idea for several reasons. First, it assumes a number of things you probably do not want; it will not read ECMWF files and thus will generate no comparison to "the truth", and it uses the most recent local angle correction.

| preproc command line systax | |
|-----------------------------|--|
| required input | description |
| year | year of data to preprocess |
| month | month of data to preprocess |
| day | day of data to preprocess |
| granule.list | list of granule numbers to preprocess |
| option | description |
| root = ' ' | set root name for cnf file |
| ret = N | make retrieval path /retN/ in instead of /ret/ |
| /nohsb | ignore HSB files |
| /ascii | make output file ASCII |
| /asl | use ASL directory tree for L1b HDF's |
| /jpl | use JPL pre-launch set-up |
| /rtp | read UMBC center FOV ECMWF RTP file |
| /umbc | flag to set-up for UMBC/NOAA comparisons |
| /lacplot | plot LAC statistics for each granule |
| /nolac | DO NOT perform local angle correction |
| lacver = 0 | (v5a RTA) pre-launch (langcor2.1.4) |
| lacver = 1 | (v5a RTA) pre-launch (langcor2.15.new) |
| lacver = 2 | v6.3.1 (v6 RTA) 11/16/2002 – has problems |
| lacver = 3 | v6.3.2 (v7 RTA) 1/21/2003 (langcor_v104) |
| lacver = 4 | v7.2.1 (v7 RTA) 3/19/2003 (langcor_v104.new) |
| lacver = 5 | v8.2.1 (v8 RTA) 4/22/2004 (langcor_v105) |

To compare against ECMWF, set the keyword 'rtp', by adding either /rtp or rtp=1 to the command line. To facilitate processing a whole day in several sub-runs, you may wish to use the 'ret' option. Normally, with it unspecified, the results of the retrieval (.ret computed profiles, .ccr cloud-cleared radiances, and a great many text summary files) will be written to the directory \$AIRS_RET/YYYY/MM/DD/ret; if you append ret=N to the preproc call, then N will be appended to that path. To run a whole day, you might use an IDL loop like the following:

```
for i = 1, 4 do begin preproc, 2002,9,6, lindgen(60)+1+60*i, /rtp, ret=i endfor
```

This uses the proper local angle correction and sets up your system to place the retrievals for granules 1-60 in \$AIRS_RET/2002/09/06/ret1, the retrievals for granules 61-120 in \$AIRS_RET/2002/09/06/ret2, the retrievals for granules 121-180 in \$AIRS_RET/2002/09/06/ret3, and the retrievals for granules 181-240 in \$AIRS_RET/2002/09/06/ret4, with comparisons to ECMWF made along the way.

For each granule that is pre-processed the a number of files are produced. These are summarized in Table 3.1 Files marked with *'s are mandatory for the retrieval. Files marked with o's are options, but recommended if they exist. Other files are for diagnostic information.

Table 3.1: Size of files for an AIRS granule

| | file | size/G | size/day | description |
|---|---------------|----------|----------|-----------------------------|
| * | airs_XXX.bin | 117.3 MB | 28.2 GB | AIRS L1b binary format |
| | airs_XXX.nen | 0.173 MB | 42 GB | noise summary file for AIRS |
| * | amsu_XXX.bin | 0.222 MB | 53 MB | AMSU L1b binary format |
| | amsu_XXX.nen | 0.005 MB | 1.3 MB | noise summary file for AMSU |
| o | hsb_XXX.bin | 1.507 MB | 362 MB | HSB L1b binary format |
| | hsb_XXX.nen | 0.009 MB | 2.3 MB | noise summary file for HSB |
| * | avn_XXX.bin | 54.5 MB | 13.1 GB | AVN L2 file |
| | ecm_XXX.bin | 54.5 MB | 13.1 GB | ECMWF L2 file |
| o | calbuf_XXX.bi | n 1.3 MB | 312 MB | Calibration QA flags. |

Having run the preprocessor, you're still not actually done – you now have to run the retrieval itself. However, the preprocessor has set up for you everything you will need.

When preproc runs it creates three files in the \$TMP directory. The are

| | | |
|---------------------|--|---|
| preproc_machine.sum | | a 1 line summary for each granule processed |
| preproc_machine.log | | a log of processing for each granule |
| preproc_machine.err | | a summary of errors encountered and fixes |

3.4 autoproc: a preprocessor for the preprocessor

autoproc searches all paths and creates summary files, then launches preproc for all granules in which all files exist and are good. It assumes options for preproc (lacver=latest, /rtp, root='yymmdd.us'.

idl/preproc/autoproc, year, month, day, [/nortp], [/nortp],
[root=' '], [/checkonly], [checklist = [,,]], [/asl]

| autoproc command line systax | |
|------------------------------|--|
| required input | description |
| year | year of data to preprocess |
| month | month of data to preprocess |
| day | day of data to preprocess |
| option | description |
| root = ' ' | set root name for cnf file |
| checklist | list of granules to check (default is all) |
| /checkonly | do NOT run preproc, just check files |
| /asl | use ASL directory tree structure |
| /nohsb | ignores HSB files |
| /nortp | DOES NOT look for UMBC ECMWF files |

One feature that needs to be added is to break the granules into runnable chunks (≤ 70 granules/run). Currently, autoproc processes all granules found along the yyyy/mm/dd paths and leaves it to the user to break up the configuration file before running the retrieval.

When autoproc runs it creates two files in the \$TMP directory. The are

| | | |
|----------------------|--|--|
| autoproc_machine.sum | | # of granules tested and accepted for processing |
| autoproc_machine.log | | a truth table of accepted files for each granule |

The .log file is especially important because it is a truth table of which files were needed and found for each granule. The L1b files for AIRS, AMSU, and HSB along with the RTP file for UMBC are opened and checked. If the L1b files have status errors then that granule is NOT processed. Usually, this occurs when the instrument was in a non-science mode. In addition, the preproc summary files (.sum,.log,.err) are also written to the \$TMP directory.

3.5 How to run the whole enchilada

- set up the environment variables
- process AIRS L1b HDF files using preproc.pro.
- follow the instructions at the end of preproc.pro in order to run a retrieval. Running the retrieval takes about 30 minutes granule (full diagnostics).

- A plot of the granule locations can be done with.
~/idl/preproc/pl_reall1b.pro, year, mon, day, glist, Nchl
see 9.2 for details
- To plot retrieval statistics of T(p), q(p), O₃(p), CO(p), CH₄(p), and CO₂(p) in the .out file use pl_sim,'ret' (see Section 9.3).
- To plot radiance statistics use pl_ftn92.pro. This program is discussed in Section 9.5
- To plot retrieval statistics for T(p), q(p), O₃(p) for each step within the retrieval system use pl_ret.pro (see Section 9.6)
- To plot the statistics on a common set of accepted profiles between two runs, use ~/idl/preproc/stat2.pro (see Section 9.4)
- plot differences of retrieval products and ECMWF
~/idl/preproc/pl_realgran.pro
see Section 9.13) for details
- plot individual profile (MIT, NOAA Regression, and NOAA Physical retrievals), reference profiles (AVN or ECMWF) radiances in the ccr file use
~/idl/preproc/pl_retccr.pro
see Section 9.12) for details

3.6 How to order from the GSFC AIRS DAAC

1. Goto <http://disc.gsfc.nasa.gov/data/>
2. (first time) click new user registration and make an account - I think everyone ends up with the same thing: your full e-mail as a userid and daac as a password.
3. click on archived data sets
4. click on L1B Products
5. click on AIRS
6. I click on L1B-AIRS-IR-RAD (even if it is AMSU – see below)
7. select the year of interest (*e.g.*, 2003)
8. select date in calendar (*e.g.*, Mar. 3)
9. page down (near bottom) to "Related data Products" and select the other things you want including AIRABRAD AIRSX2RET,AIRVBRAD(visible)
10. click Start Granule Search (on the next line)
11. Page down past table and select file type and page number

AIRIBRAD pg. 5

1 = 1-49

2 = 50-99

3 = 100-149

4 = 150-199

5 = 200-240

12. Select Granule 239
13. click "Add Selection to Order" (at end of page)
14. verify that the filename is the one you expect
15. click on "Return to shopping"
16. page down and select pg 5 of AIRS Vis/Near IR Level1B Radiances
17. Select Granule 239
18. click "Add Selection to Order" (at end of page)
19. verify that the filename is the one you expect
20. click on "Return to shopping"
21. page down and select pg 5 of AMSU-A Level1B Radiances
22. Select Granule 239
23. click "Add Selection to Order" (at end of page)
24. verify that the filename is the one you expect
25. click on "Return to shopping"
26. page down and select pg 5 L2 Standard Retrieval Product
27. Select Granule 239
28. click "Add Selection to Order" (at end of page)
29. verify that the filename is the one you expect
30. click on "Return to shopping"
31. page down and select pg 5 L2 Support Product
32. Select Granule 239
33. click "Add Selection to Order" (at end of page)
34. verify that the filename is the one you expect
35. click on "Return to shopping"
36. Click "Proceed to Checkout"
37. (@ bottom) Click on "Continue Checkout"
38. (@ bottom) Click on "Accept & Submit Order"
39. you will get an e-mail confirming your order
40. usually within an hour or two you will get an e-mail with the ftp pull site address and userid information.

3.7 Selection of AVN files

AVN forecast is run 4 times a day, at analysis time equal to 0z, 6z, 12z, and 18z. For each analysis time there are 4 forecast times, F00, F03, F06, and F09 or 16 files per day. To fetch recent files:

```
ftp disc2.nascom.nasa.gov
login: anonymous
passwd: your email address
ftp> cd private/data/avn/avn_YYMM/avn_YYMMDD
```

The AIRS Science Team constructed the following sets of files to be used for spatial and temporal interpolation of surface pressure (the only ancillary variable for the retrieval). The WGRIB file name must be of the format gblav.PGrbFFF.YYMMDD.AAA where YY, MM, DD is the year, month, day, respectively and FFF is the forecast time (F00, F03, etc) and AAA is the analysis time (0z, 6z, etc). This is summarized in Table 3.2.

Table 3.2: AVN files required for an observation at time = UT

| | | |
|-------------------|--------------------------|--------------------------|
| $0 \leq UT < 3$ | gblav.PGrbF06.030113.18z | gblav.PGrbF09.030113.18z |
| $3 \leq UT < 6$ | gblav.PGrbF03.030114.00z | gblav.PGrbF06.030114.00z |
| $6 \leq UT < 9$ | gblav.PGrbF06.030114.00z | gblav.PGrbF09.030114.00z |
| $9 \leq UT < 12$ | gblav.PGrbF03.030114.06z | gblav.PGrbF06.030114.06z |
| $12 \leq UT < 15$ | gblav.PGrbF06.030114.06z | gblav.PGrbF09.030114.06z |
| $15 \leq UT < 18$ | gblav.PGrbF03.030114.12z | gblav.PGrbF06.030114.12z |
| $18 \leq UT < 21$ | gblav.PGrbF06.030114.12z | gblav.PGrbF09.030114.12z |
| $21 \leq UT < 24$ | gblav.PGrbF03.030114.18z | gblav.PGrbF06.030114.18z |
| NOT USED | gblav.PGrbF00.030114.00z | gblav.PGrbF00.030114.06z |
| NOT USED | gblav.PGrbF00.030114.12z | gblav.PGrbF00.030114.18z |

Each file is a 1° by 1° spatial gridded product in WGRIB format. That is, there are 360 longitude and 181 latitude points. The programs `idl/preproc/presetup.pro` & `idl/avn_match/avn_init.pro` set up a table, shown below, to preprocess the AVN WGRIB files using `idl/preproc/wgrib_filename.pro` by selecting 2 files from the UT time. The time is used to compute an index number as follows: $idx = \text{LONG}(UT)/3$. The AIRS Science Team selection of forecast files is then summarized as

Table 3.3: AVN truth table

| | 0z | 3z | 6z | 9z | 12z | 15z | 18z | 21z |
|---------|-----|-----|-----|-----|-----|-----|-----|-----|
| index | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| anal #1 | 18z | 00z | 00z | 06z | 06z | 12z | 12z | 18z |
| fcst #1 | F06 | F03 | F06 | F03 | F06 | F03 | F06 | F03 |
| anal #2 | 18z | 00z | 00z | 06z | 06z | 12z | 12z | 18z |
| fcst #2 | F09 | F06 | F09 | F06 | F09 | F06 | F09 | F06 |

The Aviation (AVN) forecast is a $1^\circ \times 1^\circ$ model that defines surface pressure (AVN variable name “PRES:sfc” $\equiv P_{ref}$) and air temperature (AVN variable name “TMP:30-0 mb above gnd” $\equiv T_{ref}$) at the geopotential height of the surface (AVN variable “HGT:sfc” $\equiv H_{ref}$).

The elevation above the reference geode, z , is specified from a digital elevation model (DEM) for the observed footprint. The AVN forecast values of P_{ref} , T_{ref} , and H_{ref} are interpolated to the footprint location and time and then adjusted for the topography given within the AIRS 13.5 km footprint as described in Chapter 2, Section 2.11.1 of `rs_notes.pdf`.

In addition, the wind vector is extracted from the model using “UGRD: 30-0 mb above gnd” and “VGRD: 30-0 mb above gnd”. This is used for comparisons of retrieval errors versus wind speed or velocity.

3.8 Selection of ECMWF files

This system uses ECMWF $\frac{1}{2}^\circ$ by $\frac{1}{2}^\circ$ model as a reference profile. In the case of ECMWF we currently select the closest footprint in space and time. Therefore, for a given UT time there is only 1 file loaded. No interpolation is used. The model is run at analysis time 0z, 6z, 12z, and 18z with a 0, 3, 6, and 9 hour forecast. The file WGRIB file names are UADMMDDAAAAMMDDFFF1 where MM is the month, DD is the day, AAAA is the analysis time, and FFFF is the forecast time (time at which the forecast is valid).

For a L1b observation at a given Universal Time, UT, Table 3.4 can be used to select a file. This logic is also summarized in Table 3.5.

Table 3.4: Recommended ECMWF file for an observation at time = UT

| | | |
|-----------------------|----------------------|------------------------|
| $0.0 \leq UT < 1.5$ | UAD01140000011400001 | 0z analysis |
| $1.5 \leq UT < 4.5$ | UAD01140000011403001 | 0z, 3h forecast |
| $4.5 \leq UT < 7.5$ | UAD01140600011406001 | 6z analysis |
| $7.5 \leq UT < 10.5$ | UAD01140000011409001 | 0z, 9h forecast |
| $10.5 \leq UT < 13.5$ | UAD01141200011412001 | 12z analysis |
| $13.5 \leq UT < 16.5$ | UAD01141200011415001 | 12z, 3h forecast |
| $16.5 \leq UT < 19.5$ | UAD01141800011418001 | 18z analysis |
| $19.5 \leq UT < 22.5$ | UAD01141200011421001 | 12z, 9h forecast |
| $22.5 \leq UT < 24.0$ | UAD01130000011300000 | 0z analysis (next day) |

Table 3.5: ECMWF truth table

| | $0 \rightarrow 1\frac{1}{2}$ | $3 \pm 1\frac{1}{2}$ | $6 \pm 1\frac{1}{2}$ | $9 \pm 1\frac{1}{2}$ | $12 \pm 1\frac{1}{2}$ | $15 \pm 1\frac{1}{2}$ | $18 \pm 1\frac{1}{2}$ | $21 \pm 1\frac{1}{2}$ | $22\frac{1}{2} \rightarrow 24$ |
|------|------------------------------|----------------------|----------------------|----------------------|-----------------------|-----------------------|-----------------------|-----------------------|--------------------------------|
| anal | 0z | 0z | 6z | 0z | 12z | 12z | 18z | 12z | 0z (next day) |
| fcst | A | 3h | A | 9h | A | 3h | A | 9h | A |

3.9 Top level functional map of the preproc.pro program

```

preproc.pro          main program
+- load_lac.pro      pre-load Local Angle Correction Coefficients
+- wgrib_filename.pro determine if AVN file(s) need to be loaded
  +- wgrib_avn.pro  pre-loads a single AVN forecast files
    +- eff/rh_to_mmr.pro convert rh to g/kg mmr
      +- eff/h2o_satpres
      +- eff/h2o_press_to_mmr.pro
+- general/rd_viamp.pro load noise files
+- load_uars.pro     loads Phil's UARS climatology
+- load_masuda.pro  loads Evan Fishbein's Masuda coefficients
+- load_co2.pro     loads Denning's CO2 climatology
+- read_cc_coef.pro load Mitch's CLEAR coefficients

  AMSU L1b processing
+- get_l1filename.pro create L1b path & file name from date & granule
+- rd_amsuhdf.pro   reads AMSU,HSB,AIRS HDF files into granule arrays
+- wr_airstmp.pro   writes local temporary L1b for AMSU
  +- airsb/openl1.pro open's L1b temporary file
  +- airsb/writel1.pro writes single L1b record

  HSB L1b processing

```

```

+- get_l1filename.pro create L1b path & file name from date and granule ID
+- rd_hsbhdf.pro      reads AMSU,HSB,AIRS HDF files into granule arrays
+- wr_airstmp.pro     writes local temporary L1b for AMSU,HSB,AIRS
    +- airsb/openl1.pro open's L1b temporary file
    +- airsb/writel1.pro writes single L1b record

    AIRS L1b processing
+- get_l1filename.pro create L1b path & file name from date and granule ID
+- rd_airshdf.pro     reads AMSU,HSB,AIRS HDF files into granule arrays
+- fetch_avnfg.pro    reads proper AVN file(s)
    +- general/calen.pro convert TAI to calendar date
    +- wgrib_filename.pro determines if AVN file(s) need to be loaded
        +- wgrib_avn.pro pre-loads a single AVN forecast files
            +- eff/rh_to_mmr.pro convert rh to mmr
                +- eff/h2o_satpres
                +- eff/h2o_press_to_mmr.pro
            +- mmr_setup.pro precompute interpolation coefficients
+- fov_clear_flag.pro compute Mitch's clear flag(s)
+- airs_lac.pro       correct radiances using Larry's LAC method
+- wr_airstmp.pro     writes local temporary L1b for AIRS
    +- airsb/openl1.pro open's L1b temporary file
    +- airsb/writel1.pro writes single L1b record

    prepare & write AVN reference profile
+- fill_avnfg.pro     convert,fill, and write L2
    +- airsb/def_l2d.pro create L2 structure
    +- trace/get_trace_std.pro get reference profiles for CH4(p) & CO(p0)
    +- masuda_fg.pro (load_masuda.pro) Masuda model
    +- fetch_co2.pro (load_co2.pro) CO2 for this location & time
    +- airsb/openl2_b.pro open L2 file
    +- fastem.pro      compute English model
    +- airsb/calen.pro convert TAI to calendar date
    +- lin_interp.pro  linear interpolation routine
    +- mmr_to_cd.pro   converts moisture and ozone to CD
    +- airsb/writel2_b.pro write L2 record

```

3.10 NOAA Error codes for L1b files

- Note: in calbuf bit 1 is the LSB = 2^0 whereas in L1b nomenclature this would be called "bit 0")
- BB = Black Body
- SV = Space View

- The AIRS noise specification (NEDT(T=250 K) is given by

$$\text{SPEC} = \begin{array}{ll} 0.35K & f < 740 \text{ cm}^{-1} \\ 0.20K & 740 \text{ cm}^{-1} \leq f < 2374 \text{ cm}^{-1} \\ 0.14K & 2375 \text{ cm}^{-1} \leq f < 2400 \text{ cm}^{-1} \\ 0.20K & f \geq 2400 \text{ cm}^{-1} \end{array} \quad (3.1)$$

- JPL Radiance Limits Test ("175- ≤ BT ≤ 360+")

- (main/Level_2_main.c): currently rejects GOLFBALL if 20 channels (passing CPF L2_ignore test) exceed this test: $R_{\text{low}}(n) \leq R(n, \text{ifov}, \text{iscan}) \leq R_{\text{hgh}}(n)$
- (airs_l2_initialize.F): sets the values
 - * $R_{\text{low}}(n) = B_{\nu}(f(n), T=175) - 10 * \text{freq.nen_airs}(n)$
 - * $R_{\text{hgh}}(n) = B_{\nu}(f(n), T=360) + 10 * \text{freq.nen_airs}(n) + 0.5 * \Omega * \text{Hsun0}(n)$
 - * $\text{freq.nen_airs}(n)$ is computed from NEDT in channel properties file, $\text{NEDN}(n) = \text{NEDT_250}(n) \cdot \text{dBdT}(f(n), T=250)$
 - * $\Omega = \pi * (\text{Rs}/\text{Ds})^2 = 6.79\text{E-}05$
 - * $\text{Hsun0}(n) = \text{solar irradiance (f/ ATMOS file) or } \text{Hsun0}(n) \simeq B_{\nu}(f(n), T=5600.0)$
 - * 0.5 = estimate for $\text{rho_solar} * \text{COS}(\text{solzang})$
 - * $B_{\nu}(f(n), T) = \text{Planck Function for frequency } f, \text{ temperature } T$

3.10.1 AMSU L1 QA

Table 3.6: CALBUF CODES for AMSU & HSB Level 1 files

| Bit | description | Original Data Source |
|-------|---|--|
| 1 | No good BB counts for scan line | QA_channel bit 3 \neq 0 |
| 2 | No good SV counts for this line | QA_channel bit 0 \neq 0 |
| 3 | No good PRTs for this line | QA_receiver bit 1 \neq 0 |
| 4 | Some bad BB counts for this line | QA_channel bit 4 \neq 0 |
| 5 | Some bad SV counts for this line | QA_channel bit 1 \neq 0 |
| 6 | Some bad PRT temps on this line | QA_receiver bit 5 \neq 0 |
| 7 | BB counts could not be smoothed | QA_channel bit 5 \neq 0 |
| 8 | SV counts could not be smoothed | QA_channel bit 2 \neq 0 |
| 9 | Scanline was calibrated, but the moon was in the space view | QA_receiver bit 2 \neq 0 |
| 10 x | The channel was calibrated, but there was a SV scan position error | QA_receiver bit 3 \neq 0 |
| 11 x | The channel was calibrated, but there was a BB scan position error | QA_receiver bit 4 \neq 0 |
| 12 | The scanline was calibrated, but there was a data gap | QA_receiver bit 6 \neq 0 |
| 13 | Some channels were not calibrated | QA_receiver bit 7 \neq 0 |
| 14 * | Excessive NeDT Estimate | QA_channel bit 7 \neq 0 |
| 15 | Most recent calib. coefficients used | QA_channel bit 6 \neq 0 |
| 16 | The instrument was not in science mode | state1 \neq 0 or state2 \neq 0 or QA_Receiver bit 0 \neq 0 |
| 17-23 | Reserved (bits set to zero) | |
| 24 | Missing value | |
| * | This bit is set to zero in calbuf for both AMSU and HSB. | |
| x | This bit ignored by rd_l1qafile.F for HSB. | |

Table 3.7: AMSU/HSB QA Test Matrix

| QA Source | REG | PHYS | description |
|-----------------------|-----|------|---|
| state(*)val \neq 0 | X | X | Instrument not in science mode |
| NeDT2NomRatio < 4 | X | X | NEDT within Range (Granule) |
| rad < -999 | X | X | L1b over/underflow (FOV) |
| $50 \leq BT \leq 350$ | . | X | Radiance out of range (FOV) |
| qa_channel, b.7 | . | . | Excessive NeDT Estimate |
| qa_channel, b.6 | . | . | Most recent calibration coefficients used |
| qa_channel, b.5 | . | . | BB counts could not be smoothed |
| qa_channel, b.4 | X | X | Some bad BB view counts for this line |
| qa_channel, b.3 | X | X | No good BB counts for scan line |
| qa_channel, b.2 | . | . | SV counts could not be smoothed |
| qa_channel, b.1 | X | X | Some bad SV counts for this line |
| qa_channel, b.0 | X | X | No good SV counts for this line |
| qa_receiver*, b.7 | X | X | Some channels were not calibrated |
| qa_receiver*, b.6 | X | X | calibrated, but there was a data gap |
| qa_receiver*, b.5 | X | X | Some bad PRT temps on this line |
| qa_receiver*, b.4 | X | X | calibrated, but BB position error |
| qa_receiver*, b.3 | X | X | calibrated, but SV position error |
| qa_receiver*, b.2 | X | X | calibrated, but moon in SV |
| qa_receiver*, b.1 | X | X | No good PRTs for this line |
| qa_receiver*, b.0 | X | X | The instrument was not in science mode |

* = 1 or 2 for AMSU 1 or 2, nada for HSB

Table 3.8: qa_channel

| L1b bit | NOAA bit | description |
|---------|----------|---------------------------------|
| 7 | 14 | excessive noise |
| 6 | 15 | most recent cal coef's used |
| 5 | 7 | BB counts could not be smoothed |
| 4 | 4 | BB counts were marginal |
| 3 | 1 | All BB counts were bad |
| 2 | 8 | SV counts not be smoothed |
| 1 | 5 | SV counts were marginal |
| 0 | 2 | All SV counts were bad |

Table 3.9: qa_receiver*

| L1b bit | NOAA bit | description |
|---------|----------|--|
| 7 | 13 | some channels were not calibrated |
| 6 | 12 | calibrated, but data gap |
| 5 | 6 | calibrated, but some PRT values were bad or marginal |
| 4 | 11 | calibrated, but BB scan position error |
| 3 | 10 | calibrated, but SV scan position error |
| 2 | 9 | calibrated, but moon was in SV |
| 1 | 3 | calibration was not derived due to missing or bad PRT values |
| 0 | 16 | calibration was not derived due to instrument mode |

Table 3.10: state1val .or. state2val for AMSU, state for HSB

| bit L1b | bit NOAA | description |
|------------|-------------|--------------|
| 0 | . | Science Mode |
| 1 | 16 | Special |
| 2 | 16 | Erroneous |
| 3 | 16 | Missing |

3.10.2 AIRS L1 QA

Table 3.11: CALBUF CODES for AMSU & HSB Level 1 files

| Bit | description | Original Data Source |
|-----|------------------------------------|-------------------------------|
| 1 | Reserved (bits set to zero) | |
| ... | Reserved (bits set to zero) | |
| 9 | Moon in SV | calSCANsummary bit 2 \neq 0 |
| 15 | Reserved (bits set to zero) | |
| 16 | Instrument was NOT in science mode | state \neq 0 |
| 17 | High Noise in Channel | calCHANsummary bit 3 \neq 0 |
| 18 | Bad Telemetry | calflag bit 1 \neq 0 |
| 19 | Pop Detected in Channel | calflag bit 4 \neq 0 |
| 20 | Bad Gain for Channel | calflag bit 5 \neq 0 |
| 21 | Bad Offset for Channel | calflag bit 6 \neq 0 |
| 22 | Undesirable Channel Properties | excludedchans \geq 2 |
| 23 | Reserved (bit set to zero) | |
| 24 | Missing value | |

Table 3.12: AIRS L1 QA Test Matrix

| LAC = NOAA local angle correction step REG = NOAA regression step PHYS = NOAA physical algorithm step | | | | | |
|---|-----|-----|------|---------------------------------------|----------------|
| QA Source | LAC | REG | PHYS | description | when |
| state $\neq 0$ | X | X | X | AIRS not in data mode | FOV |
| CalChanSummary b.3 $\neq 0$ | X | X | X | High Noise in Channel | Gran |
| CalChanSummary b.2 $\neq 0$ | . | . | . | Spectral Calibration | Gran |
| Calflag b.6 $\neq 0$ | X | X | X | Bad Offset for Channel | Scan |
| Calflag b.5 $\neq 0$ | X | X | X | Bad Gain for Channel | Scan |
| Calflag b.4 $\neq 0$ | X | X | X | Pop Detected in Channel | Scan |
| Calflag b.2 $\neq 0$ | X | X | X | Moon in View | Scan |
| Calflag b.1 $\neq 0$ | X | X | X | Bad Telemetry | Scan |
| ExcludedChans > 2 | X | X | X | Non-optimal Properties | Gran |
| rad < -999 | X | X | X | L1b over/underflow | FOV |
| 175+ \leq BT \leq 360+ | X | X | X | Radiance out of range | FOV |
| pristine list | X | X | . | Accum. CPF Evaluation | multi-epoch |
| Chan.Prop.File | . | . | X | Dead | CP epoch |
| Chan.Prop.File | . | . | X | Noise | CP epoch |
| Chan.Prop.File | . | . | X | Popping | CP epoch |
| Chan.Prop.File | . | . | X | Poor SRF | CP epoch |
| Chan.Prop.File | . | . | X | Bad SRF | CP epoch |
| Chan.Prop.File | . | . | . | Spatial | Cij) (CP epoch |
| Chan.Prop.File | . | . | . | Non-Gauss Noise | CP epoch |
| Chan.Prop.File | . | . | X | A/B State | CP epoch |
| Chan.Prop.File | . | . | X | NEDT $> 5*$ SPEC | CP epoch |
| Chan.Prop.File | . | . | . | Cij < 0.85 | CP epoch |
| Chan.Prop.File | . | . | X | RTA ERR $>$ SPEC | CP epoch |
| Rdiff_swindow | . | . | ? | 2260 cm^{-1} C _{ij} | FOV |
| Rdiff_lwindow | . | . | ? | 850 cm^{-1} C _{ij} | FOV |
| Glint_distance ≤ 200 | . | . | ? | kick SW channels if(pland=0) | GOLFBALL |

Table 3.13: Summary of Channel Property Files

| version 6.0 | | | | | | | |
|-----------------|----------|----------|----------|----------|----------|----------|----------|
| | 05/04/02 | 07/09/02 | 08/30/02 | 09/17/02 | 10/22/02 | 01/10/03 | 11/19/03 |
| Good | 2108 | 2108 | 2103 | 2102 | 2091 | 2109 | 2097 |
| Dead | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Noise | 97 | 97 | 104 | 106 | 100 | 37 | 104 |
| Popping | 14 | 14 | 8 | 8 | 18 | 13 | 13 |
| Poor SRF | 142 | 142 | 140 | 139 | 143 | 153 | 141 |
| Bad SRF | 10 | 10 | 12 | 12 | 15 | 53 | 11 |
| Spatial | 6 | 6 | 6 | 6 | 6 | 7 | 7 |
| NEDT > 5·SPEC | 97 | 97 | 105 | 107 | 102 | 40 | 108 |
| Cij < 0.5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| RTAERR > 5·SPEC | 1 | 1 | 6 | 6 | 6 | 6 | 6 |
| version 8.1 | | | | | | | |
| | 05/04/02 | 07/09/02 | 08/30/02 | 09/17/02 | 10/22/02 | 01/10/03 | 11/19/03 |
| Good | 2110 | 2110 | 2116 | 2113 | 2097 | 2115 | 2111 |
| Dead | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Noise | 61 | 61 | 56 | 60 | 61 | 43 | 46 |
| Popping | 14 | 14 | 8 | 8 | 18 | 13 | 13 |
| Poor SRF | 144 | 144 | 143 | 143 | 145 | 150 | 150 |
| Bad SRF | 46 | 46 | 51 | 50 | 53 | 53 | 54 |
| Spatial | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| NEDT > 5·SPEC | 63 | 63 | 57 | 61 | 64 | 46 | 50 |
| Cij < 0.5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| RTAERR > 5·SPEC | 1 | 1 | 2 | 2 | 2 | 2 | 2 |

Table 3.14: AIRS L1b state variable

| L1b bit | NOAA bit | description |
|------------|-------------|--------------|
| 0 | . | Science Mode |
| 1 | 16 | Special |
| 2 | 16 | Erroneous |
| 3 | 16 | Missing |

Table 3.15: A/B State (ExcludedChans)

| L1b val | NOAA bit | description |
|------------|-------------|--|
| 0 | | A and B are both used |
| 1 | | A is good and is used |
| 2 | | B is good and is used |
| 3 | 22 | A and B are both used; but both pop or are noisy |
| 4 | 22 | A is used, but pops or is noisy |
| 5 | 22 | B is used, but pops or is noisy |
| 6 | 22 | both detectors are dead |

Table 3.16: CalChanSummary (granule)

| L1b val | NOAA bit | description |
|------------|-------------|----------------------------------|
| 7 | | .or. of Calflag(bit.7) scene |
| 6 | | .or. of Calflag(bit.6) offset |
| 5 | | .or. of Calflag(bit.5) gain |
| 4 | | .or. of Calflag(bit.4) pop |
| 3 | 17 | excessive noise in granule |
| 2 | 13 | spectral quality |
| 1 | | .or. of Calflag(bit.1) telemetry |
| 0 | | reserved |

Table 3.17: L1b Calflag tests

| L1b val | NOAA bit | description |
|------------|-------------|--------------|
| 7 | | Scene |
| 6 | 21 | Offset |
| 5 | 20 | gain |
| 4 | 19 | pop detected |
| 3 | | DCR Occurred |
| 2 | 9 | Moon in view |
| 1 | 18 | telemetry |
| 0 | | Reserved |

Chapter 4

match_proc: Processing JPL Matchup Files

JPL matchup files contain AIRS, AMSU, HSB, and validation data all in one file. In granule HDF files the variables are organized in swath format. For example latitude is given as an array of (135,90) for AIRS or (45,30) for AMSU and radiance is given as an array (2378,135,90) for AIRS.

For matchup files there are multiple swath structures for a set of N golfballs, where N can be any value. For radiances the arrays are (N,2378,3,3) for AIRS and (N,15) for AMSU. While this preprocessor is very similar to preproc, internally there are significant differences.

Chapter 5

cnv_gbgrid: Processing NOAA Gridded (Grads) Files

NOAA builds gridded files in which the golfball closest to the grid center is saved. The grid files contain the AIRS 9 footprints, AMSU, and HSB for each of the grid cells.

The $3^\circ \times 3^\circ$ grid has $61 \times 120 = 7320$ cells. We have a separate file for ascending and descending orbits so in a given day there are 2 files with approximately 6000 cells filled with data. The rest of the cells are in AIRS orbital gaps.

The program `cnv_gbgrid.F` converts the files from the gridded format to the internal format for the retrieval code. The gridded format has multiple planes of 61×120 maps whereas the retrieval wants a sequential set of complete radiances. The program is completely written in FORTRAN and does the following

- read the entire grid file into memory (needs about 840 MB)
- determine which cells are valid
- perform local angle correction for AIRS off-axis radiances
- write sequential files, in radiance units, for AIRS, AMSU, and HSB for valid cells.

AVN and ECMWF files can be created directly from the AMSU and AIRS L1b files using `avn_match.pro` and `ecmwf_match.pro`.

Chapter 6

simrad: simulation of radiance scan sets

The simrad.F program will simulate level.1 radiances using two namelist files (sim & io). To simulate radiances the namelist filenames are derived from the geophysical ensemble name, [ens], (*e.g.*, blank, EDGSC_fm, ERGST, jan01.s01, etc.) and an instrument name, [inst], (*e.g.*, airs, iasi, cris, atms, hirs).

The namelist file sim[_ens]_[inst].nl designates the input "truth" filename and a number of options for processing the level.1 file.

| | [ens] = ensemble (optional), [inst] = instrument |
|---------------------|--|
| sim[_ens]_[inst].nl | simulation setup |
| io[_ens]_[inst].nl | I/O specifications (simulator & retrieval) |

The I/O namelist file io[_ens]_[inst].nl is shared between the simulator (simrad.F) and the retrieval (airsb.F) and designates the output level.1 and level.2 files. It also specifies the RTA files, solar radiance file, and instrument noise files. This file is shared with the retrieval (airsb.F) and specifies all the I/O specific information for the retrieval (first guess files, regression files, and I/O unit #'s).

Chapter 7

air**sb**: retrieval of geophysical parameters

7.1 command line control

The retrieval code, `airsb.F`, requires both command line instructions and 7 namelist files to operate. There are 2 modes of operation

7.1.1 `airsb direct mode`

The command line has a direct mode

```
airsb103.exe rootname
```

In this file there is one, `Ngroup=1`, groups of input files. The input file names are those defined by the names given in `io[_ens]_[inst].nl`. All the output files have same root name, given by the name, `rootname`, in the command line. `rootname` has a maximum of 64 characters. In this context, `graname`, defined below, and `rootname` are identical.

7.1.2 `airsb indirect mode`

The indirect mode allows `Ngroup` sets of files to be processed sequentially. A maximum of `MAXGROUP=1024` groups are allowed and the retrieval system allows a total of $2^{18} = 262,144$ profiles to be processed within a single run. The statistics (`.out` file) will be generated for the accepted profiles for all the cases in the `Ngroup` file(s). The calling sequence looks like

```
airsb103.exe cnfname runID
```

In this mode a file named `cnfname.cnf` must be in the directory that `airsb` is invoked from. This file contains a header record with the number of file groups (`Ngroup`) and then commands and filenames that will overwrite the filenames in the `io[_ens]_[inst].nl` file. Each group is assigned its own *groupname*, given within the configuration file, such that output files are organized identically to the same structure as the input files defined in the group definition. Here are the valid configuration file commands:

| command | action |
|---------|---|
| TRUE | overwrites trufile.in (simrad.F) |
| L2_I | overwrites L2file.in (or not.set) |
| L1_I | overwrites L1FILE.IN |
| L1_A | overwrites L1AMFILE.IN |
| L1_M | overwrites L1MHFILE.IN |
| L1_C | overwrites L1QAFfile.in |
| MIT | overwrites MWFGFILE.IN |
| NOAA | overwrites l2fgfile.in |
| TUNE | overwrites tuningcoefffile |
| NEDN | overwrites IRnoisefile |
| COMM | comment - 24 char string echoed in namelist |
| UPRT | allow user print levels |
| ZPRT | set print levels to zero |

There are two kinds of output files in this mode. Files that are generated for the entire set of groups will have the name

runname = runID_cnfname.xxx.

Examples, of these files are .txt, .out and there are options to concatenate certain files into one file, such as .ccr.

File that are mapped to the input files are named

graname = runID_graname.xxx.

Example of these files are .ret, .fg, .mit, .ccr, etc. This mode requires using defaults within the namelist file, io[_ens]_[inst].nl. These are given in the table below.

| | |
|-------------|---|
| graname.ret | L2file_out = 'ret.asc' or 'ret.bin' |
| graname.fg | L2FGFILE.OUT 'fg.asc' or 'fg.bin' |
| graname.err | L2ERROR.IN = 'err.asc' or 'err.bin' |
| graname.tru | l2tru_out = 'tru.asc' or 'tru.bin' |
| graname.ccr | L1FILE_CCR = 'ccr.asc' or 'ccr.bin' |
| graname.mit | L2MITFILE.OUT = 'mit.asc' or 'mit.bin' |
| runname.ccr | L1FILE_CCR = 'allccr.asc' or 'allccr.bin' |

7.2 required namelist files

The retrieval requires seven namelist files. They are attached by the retrieval script to fort.40 through fort.46. Each contains specific information for the retrieval algorithm. They can have any name, but to allow multiple instruments to be saved in the same directory I usually have the instrument have (airs, iasi, cris, hirs, etc) included in the namelist filename. The namelists required are

| | |
|------------------------|--|
| pro[_ens]_[inst].nl | directs the retrieval methodology |
| io[_ens]_[inst].nl | I/O (shared with simrad.F) |
| microw[_ens]_[inst].nl | microwave channels, functions, and parameters |
| temp[_ens]_[inst].nl | $T(p)$ and surface channels, functions, and parameters |
| water[_ens]_[inst].nl | $q(p)$ channels, functions, and parameters |
| ozone[_ens]_[inst].nl | $O_3(p)$ channels, functions, and parameters |
| clouds[_ens]_[inst].nl | cloud channels, functions, and parameters |

In old scripts (run.airsb.com) the ensemble variable is used to select the simulation datasets such that many datasets can be setup in the same directory (*e.g.*, EDGST, ERGST, EDG, etc.). In real-time scripts

(procret.com) airtsb is run in the indirect mode and a configuration file merges many granules of data. In this case the ensemble name is not used because there is usually only 1 set of namelists per instrument. Many configuration files can be specified.

7.3 Top level flow diagram for airtsb.F

```

.. read_nl
  .. default
  <-- namelist read in
  <-- namelist mod's
  .. setsteps (sets STEPNAME() if namelists are old style)
.. writ_nl to unit = 6 addendum
do 1000 igrup = 1, Ngroup
  .. groupinit load initial L1
  if(igrup.eq.1)
    .. trfinit load RTA's
    .. CLDSTAT1 initialize cloud parameter statistic arrays
    .. stattwoeta initialize cloud clearing statistic arrays
    .. profdist initialize profile statistic arrays
    .. termzero initialize termination statistic arrays
    .. SIMSTAT_INI initialize profile (.out) statistic arrays
    .. BTRESIDALL initialize radiance (.f92) statistic arrays
    <-- open run level files
  else
    .. test for new noise file f/ CNF file
    .. load_tuning load tuning file
  endif
  <-- open granule level files
do 2010 npro = 1, numobs_L1
  <-- determine print level for this case (on/off)
  .. read_tru read "truth" file (simulation)
    or reference file (real)
  .. readsim1 read radiance files (IR,AMSU,HSB)
  .. getfg read first guess file(s) (CLIM,MIT,NOAA,etc)
  <-- if this profile is not to be processed, goto 2010
  .. changlev adjust function levels to Psurf
  .. prtprof print a summary for this profile
  <-- convert emissivities from hinge points to channels
  .. ret_driver perform a retrieval for this case based on steps
    given in STEPNAME(). ret_driver runs retrieval
    modules (like ret_tmpe, ret_watc, ...) and
    opens (on record=1) and writes record to
    .mit, .fg, .ret, .ccr, etc.
    see namelist.ema (search for "stepname():") for
    list of step names and associated subroutine calls.
  .. union_f perform a union of fg emissivity hinge pts.
  .. union_v compute emissivity on union of hinge points
  .. profdist update statistics on lat,long,season for this profile
2010 continue
  <-- close all granule level files B
1000 continue
  <-- close run level files

```

```

.. profterm      close binary profsumm file
.. stattwoeta    write eta stat's into .out file
.. CLDSTAT1      write cloud stat's into .out file
.. BTRESIDALL    write radiance stat's into .out file & .f92 file
.. SIMSTAT_PRNT  write profile (T(p),q(p),...) into .out file
.. termprnt      write termination counts into .out file
.. prntsumm      read .bin and write to .tbl and summary to .out
.. profdist      write profile location stat's into .out file
done

```

7.4 I/O units for airsb code

Hardwired Units:

| | | |
|----|-------------------------------------|-------------|
| 6 | text file | runname.txt |
| 10 | single load input files | |
| 18 | configuration file | |
| 19 | BINARY profile summary (profsumm.F) | |
| 20 | statistics output & run summary | runname.out |
| 70 | error summary file | runname.f70 |

Optional hardwired units. Flags in the I/O namelist file must be set to `.TRUE.` to invoke these output files.

| | | | |
|----|------|--------|--|
| 66 | .f66 | f66flg | eta_calc.F, cloud clearing eta's |
| 68 | .f68 | f68flg | eigenvalue summary |
| 71 | .f71 | f71flg | rejection summary |
| 85 | .f85 | f85flg | truth summary for cloud height |
| 88 | .f88 | f88flg | cloud height diagnostics |
| 89 | .f89 | tautbl | channel selection tables (see Section 7.5) |
| 93 | .f93 | f93flg | trace gas retrieval weighting functions |
| 95 | .f95 | | product error output file summary |
| 99 | .f99 | | propagated error estimate summary |

Selectable units (but beware of conflict with other hardwired units):

| | | | |
|----|--------------|---------------|-------------------------------|
| 25 | logl2_in | L2FILE.IN | truth input [optional] |
| 26 | logl2fg_in | L2FGFILE.IN | NOAA first guess file |
| 27 | logmwfg_in | MWFGFILE.IN | MIT first guess file |
| 30 | logl1cl_in | L1CLFILE.IN | NOT USED anymore |
| 31 | logl1_in | L1FILE.IN | AIRS radiance input |
| 32 | logl1am_in | L1AMFILE.IN | AMSU radiance input |
| 33 | logl1mh_in | L1MHFILE.IN | MHS radiance input |
| 34 | logl2_out | L2FILE.out | final product outout |
| 35 | logl2fg_out | L2FGFILE.OUT | step=SET_FG state |
| 36 | l2error_unit | L2ERROR.IN | input ensemble error estimate |
| 36 | l2error_unit | L2ERROR.out | error estimate output |
| 36 | l2error_unit | L2ERROR.RMS | ensemble error output |
| 37 | logl2tru_out | l2tru.out | echo of truth file |
| 38 | logl1_ccr | L1FILE.CCR | radiance output |
| 39 | logl2mit_out | L2MITFILE.OUT | step=SET_MIT state |

Finally, some output files have a default setting to allow aairsb to build a name from the rootname (specified for each block in the configuration file). In the table below the default unit number, the file extension, and namelist control parameters are defined. For example, if the first guess output file is to be named automatically, then l2fgfile_out = 'fg.asc' or 'fg.bin'. The ascii/binary is determined from the rootname. If the rootname has a 'bi' then it is binary, otherwise it is ascii. This feature minimizes how many variables have to be set within the configuration file. The cloud cleared radiance file (ccr) has an additional mode to merge all blocks within a configuration file into one large file with name runname.ccr instead of many files with granname.ccr. This is invoked with l1file_ccr = 'allccr.asc' or l1file_ccr = 'allccr.bin'

| default unit | file exten | unit variable | filename variable |
|-----------------|---------------|------------------|----------------------|
| 34 | .ret | logl2_out | L2FILE.out |
| 35 | .fg | logl2fg_out | l2fgfile_out |
| 36 | .err | l2error_unit | L2ERROR.out |
| 37 | .tru | logl2tru_out | l2tru.out |
| 38 | .ccr | logl1_ccr | l1file.ccr |
| 39 | .mit | logl2mit_out | l2mitfile_out |

In a typical run the following files will probably be in the retrieval directory (default namelists).

| Files Generated per granule | | |
|--------------------------------|----------------|---|
| # bytes | filename | what's in it |
| 130146624 | runID_G051.ccr | CCR file, contents controlled by Iformat_ccr(), Ichan_ccr() |
| 4721529 | runID_G051.err | error estimates, L2 format |
| 120328 | runID_G051.f61 | 1 line summary of TRUTH |
| 83822 | runID_G051.f69 | kernel function statistics |
| 315470 | runID_G051.f89 | 1 line summary for each channel (used for picking channels) |
| 431772 | runID_G051.f91 | eigen-function output (see pl_eigen.pro) |
| 1160780 | runID_G051.f93 | trace gas ret. functions (f93flg=T) |
| 11676 | runID_G051.f99 | propagated error table |
| 6003637 | runID_G051.fg | L2 state at step = 'SET FG' |
| 5696185 | runID_G051.mit | L2 state at step = 'SET MIT' |
| 6155165 | runID_G051.ret | L2 state at end |
| Files Generated for entire run | | |
| # bytes | filename | what's in it |
| 79261 | runID_test.out | namelist echo, statistics, exec.time see pl_sim.pro, pl_ret.pro |
| 3371186 | runID_test.txt | unit=6 text printout |
| 1253365 | runID_test.f92 | level: pro_airs/iprt(), io_airs/idprotest() radiance statistics, if writeradstat=T see pl_ftn92.pro |
| 1134560 | runID_test.bin | BINARY summary for each ret see pl_profsumm.pro, pr_tbl.F |
| 68271 | runID_test.f64 | summary of CO ₂ , CH ₄ , and CO |
| 225791 | runID_test.f70 | completion summary |
| 621465 | runID_test.f95 | radiance error estimate summary see pl_errest.pro |
| 33247 | runID_test.kic | summary of kicked channels (i.e., bad channels that were requested) |
| 455628 | runID_test.tbl | 1 line rejection summary table must run pr_tbl.F to get this |

7.5 Selection of Channels (io namelist: tauble)

The tautbl variable in the I/O namelist is used to build tables suitable for selection of channels. The EXCEL spreadsheet, airs_chls.xls, is built from *.f89 tables. Up to 10 cases (golfballs) can be selected with tautbl. It is important that IDPRO (default NUMPRO=-1 runs all cases) has the selected cases with the runtime list, otherwise pr_tautbl.F is never called. My default cases (those in sim_3.asc) are run from the g401 on Sep. 6, 2002. In that case “tautbl = 925, 1473, 5207”.

| 3 test cases from 9/6/02 G401 | | | | | |
|-------------------------------|--------------|----------|---------|---------|--------|
| index | region | latitude | viewang | solzang | cm-h20 |
| 925 | polar | 85.25 | 31.5 | 85.3 | 0.53 |
| 5207 | mid-latitude | 41.29 | 5.1 | 39.1 | 1.44 |
| 1473 | tropical | 9.76 | -41.2 | set | 4.46 |

The f89 file contains the following information for each of the profiles selected by tautbl.

- echo of the frequency lists from the namelist files (only on first profile)
- echo of the temperature, moisture, ozone and trace gas profiles
- a single line for each channel containing
 - index and frequency
 - a truth table of how that channel is used
 - NEDN and QA information from channel properties file and L1b file.
 - tuning and error estimate values
 - Brightness temperature computed for this profile
 - transmittance for fixed, water, ozone, CO, and methane as well as the pressure that the transmittance crosses 50%.

7.6 airsb error code word

The airsb code has a feature to trap and abort the run if a software error occurs. These are errors in I/O, dimensioning, matrix inversion, etc. This is handled by the routine `~/code/src_jpl/softexit()`. The first error trapped by `softexit` is also passed to `rspace(12)` in the retrieval file and can be parsed by the routine `soft.decode()` within `softexit.F`.

The module the error occurs in is recorded along with the type of error that occurred. The calling routine is responsible for printing information to `unit=6` prior to calling `softexit`. This information is echoed by the calling script and will be the last lines in the `.txt` file. A five digit error code is build as follows

typeidx specified as `XXYZZ`

Y = step ID number (*e.g.*, if `RETSURF` is called 3 times, then `Y=1,2,3` for the 3 occurrences)

| XX | retrieval module | ZZ | type of error |
|----|------------------|----|---|
| 00 | not specified | 00 | normal exit (airsb) |
| 01 | RETAMSU | 01 | spare |
| 02 | RETMHS | 02 | used a bad channel |
| 03 | RETSURF | 03 | option does not exist |
| 04 | RETMPC | 04 | namelist error |
| 05 | RETWATC | 05 | I/O file error: open or read |
| 06 | RETOZON | 06 | decoding input file error |
| 07 | ETACHECK | 07 | dimension exceeded error |
| 08 | CLOUDHGT | 08 | illogical value obtained, code error |
| 09 | CLOUDMAT | 09 | highly singular matrix |
| 10 | RET_CO2 | 10 | tqli – too many iterations |
| 11 | RET_CH4 | 11 | invalid iphys |
| 12 | RET_CO | 12 | file already exists (write only modes) |
| 13 | NOAA REG | 13 | inverse of ncv matrix (in <code>sqmatinv.F</code>) |
| | | 14 | inverse of $[F*F^T]$ (in <code>err2slab.F</code>) |

For example, if `TQLI` exceeded the maximum number of iterations on the 3rd call to `RETSURF` would be given as `typeidx = 03310`.

7.7 Making the code run faster

The retrieval code generates statistics w.r.t. the reference profiles (set by `L2.I` in the `cnf` file) and computes radiances from the reference state, `.mit` state, `.fg` state, and `.ret` state for all channels. In addition, the `ccr`

file can be quite large. To save the computation of radiances and I/O time the following flags can be turned off

| set to .false. | namelist | function |
|----------------|-------------|---|
| writeradstat | pro_airs.nl | turns off radiance computation (all chl's |
| readlev2 | io_airs.nl | turns off statistics to the .out file |
| writeradused | pro_airs.nl | turns off the computation of radiances (chl's used) |

Also, the makefiles for the FORTRAN code is set to compile quickly, that is, minimum optimization. Turn up the optimization and recompile..... This can be 2x to 3x faster.

7.8 Quality indicators stored in `ispare()` and `rspace()`

Table 7.1: ispare definitions

| ispare index | description |
|---|---|
| 1 | 0=IR retrieval, 1=AMSU retrieval |
| 2 | reject flag (see table below) |
| 3 | numFOV_CLD = # of FOV's for clouds in r spare |
| 4 | numCLD = # of cld layers in r spare |
| 5 | NumQual = # of quality indicators |
| 6 | ieta_rej = cloud clearing iteration used for rej. |
| 7 | CLEARFLAG, 0 = clear |
| 8 | starting index of 1st MIT parameter in r spare() |
| 9 | nMIT = # of MIT parameters (NOTE: can be zero) 1 = emismw_R0 2 = emismw_T0 3 = emismw_Tinf 4 = sea ice concentration 5 = secant ratio 6 = iret (microwave rejection flag) |
| if ispare(10) \geq 0 | |
| 10 | Nkick = # of kicked channels |
| 10 + i | kicked channel list, i = 1, Nkick Nfovtbl = 0 # of items/FOV in L1 table numFOV_rad = 0 # of FOV's in L1 table |
| if ispare(10) = -1 (version 1) | |
| 11 | irad = starting index of L1 information block in r spare() |
| 12 | numFOV_rad = # FOV in radiance block Nfovtbl = 2 |
| 13 | # of kicked channels = Nkick |
| 13 + i | kicked channel list, i = 1, Nkick |
| if ispare(10) = -2 (version 2) | |
| 11 | irad = starting index of L1 information block in r spare() |
| 12 | numFOV_rad = # FOV in radiance block |
| 13 | Nfovtbl = # of items in radiance block |
| 14 | Nkick = # of kicked channels |
| 14 + i | kicked channel list, i = 1, Nkick |
| if ispare(10) = -3 (version 3) | |
| 11 | irad = starting index of L1 information block in r spare() |
| 12 | numFOV_rad = # FOV in radiance block |
| 13 | Nfovtbl = # of items in radiance block |
| 14 | K = N_pge_qual_flags = \$ of PGE quality flags |
| 14 + k | PGE quality flag(k), k=1, K |
| 15 + K | Nkick = # of kicked channels |
| 15 + K + i | kicked channel list, i = 1, Nkick |
| MAXSPARE determines size of the ispare() array Therefore, only MAXSPARE - 15 kicked channels can be written; however, usually Nkick \leq 10 | |

Table 7.2: ispare(2) rejection definitions

| ispare(2) | theyrejected | rejected by |
|-----------|--------------|--|
| 0 | 0 | none |
| 1 | 0 | NOAA physical (see rspares()) |
| 2 | 1 | MIT external file |
| 3 | 1 | MIT external file & NOAA physical |
| 4 | 2 | NOAA external file |
| 5 | 2 | NOAA external file & NOAA physical |
| 7 | 3 | both external files & NOAA physical failed |
| 8 | 4 | internal MIT only |
| 9 | 4 | internal MIT & NOAA physical |
| 16 | 8 | internal NOAA regression |
| 17 | 8 | internal NOAA regression & NOAA physical |
| 25 | 12 | all 3 failed |

Table 7.3: r spare QA definitions

| r spare index | rejthresh index | description |
|---------------|-----------------|--|
| 1 | n/a | OLRret |
| 2 | n/a | COLRret |
| 3 | 1 | etarej(ieta_rej) |
| 4 | 10,4 | cldfrctot |
| 5 | ,8 | cldfrc(Pcld>500) |
| 6 | 16,5 | A(end) |
| 7 | 2 | IRx = RMS(rad(IR.ret)-radobs) for AMSU chl's in IVRADREJ() |
| 8 | 3 | RMS(T(p) f/IR.ret - T(p) f/AMSU.ret) (in bottom 2-kms) |
| 9 | 6 | qualsurf |
| 10 | 7 | qualtemp |
| 11 | 9 | qualwatr |
| 12 | ≠ 0 | code rejection (see Section 7.6 |
| 13 | 12 | Ts(NOAA)-Ts(MIT) |
| 14 | 24 | A_eff(1) |
| 15 | 17,22 | A_eff(end) |
| 16 | | intercept of $\alpha(1) = f(\alpha(2))$ fitting |
| 17 | | χ^2 of $\alpha(1) = f(\alpha(2))$ fitting |
| 18 | 19 | fzeta (first zeta has significant cloud) |
| 19 | 20 | rzeta (significant multiple cloud effect) |
| 20 | CLRFLG | bias_ccdiff(1) = BIAS(ccr-< R >) over 800-900 cm^{-1} |
| 21 | CLRFLG | lambda(2,iter_eta) = $U^T S^T N^{-1} S U$ |
| 22 | 21, | totliqwat_ret, gm/cm^2 |
| 23 | 28 | NOAA_score |
| 24 | 26 | etarej(1) |
| 25 | 27 | A(1) |
| 26 | 25 | cldfrctot(1) (after 3/1/05) |
| 27 | | cldfrc500(1) (after 3/1/05) |
| 28 | | etatype(1) (after 3/1/05) |
| 29 | 31 | gtest_a(end) $gtest_a = (\text{BT}(2395) - \text{BT}(2392)) * \text{COS}(\text{zenang})^{0.4}$ |
| 30 | 30 | Ts(NOAA) |
| 31 | 32 | NOAA diff test = $\text{BT}(2390.5) - [a_0 + a_1 * \text{BT}(A_4) + a_2 * \text{BT}(A_5) + a_3 * \text{BT}(A_6)] - [a_4 * \text{COS}(\text{zenang}) + a_5 * (1 - \text{COS}(\text{scanang}))]$ |
| 32 | | etatype(end) (after 3/1/05) |
| 33 | | qual_o3 (after 3/10/05) |
| 34 | | lammax_o3 |
| 35 | | qual_co |
| 36 | | lammax_co |
| 37 | | qual_ch4 |
| 38 | | lammax_ch4 |
| 39 | | qual_co2 |
| 40 | | lammax_co2 |
| 41 | | spare |
| ... | | spare |
| 64 | | spare |

Table 7.4: r spare data formatting

| rs pare index | description |
|---|--|
| ISPARE(5) + i | cldfrc(iFOV,iCLD), $i=iFOV + numFOV_CLD*(iCLD-1)$ $iFOV = 1, numFOV_CLD = ISPARE(3) (1 \leq numFOV_CLD \leq 9)$ $iCLD = 1, numCLD = ISPARE(4), (1 \leq numCLD \leq 2)$ ISPARE(5) = 64, last QA location (old files = 40) |
| MIT parameters, I = ISPARE(8)=64, ISPARE(9) \leq 6 | |
| I | emismw_R0 |
| I + 1 | emismw_T0 |
| I + 2 | emismw_R0 |
| I + 3 | emismw_Tinf |
| I + 4 | sea ice conc |
| I + 5 | secant ratio |
| I + 6 | MIT return code |
| L1 parameters, I = ISPARE(11) = start index N = ISPARE(12) = # L1 FOV, ISPARE(13) \leq 8 items | |
| I + 0*N + j | alat(j+1), j = 0, N-1 |
| I + 1*N + j | alon(j+1), j = 0, N-1 |
| I + 2*N + j | spike_flg(j+1), j = 0, N-1 |
| I + 3*N + j | pland(j+1), j = 0, N-1 |
| I + 4*N + j | glint(j+1), j = 0, N-1 |
| I + 5*N + j | $C_{ij}(j+1)$, j = 0, N-1 |
| I + 6*N + j | $C_{ij}(j+1)$, j = 0, N-1 |

7.9 AIRS science versions

Table 7.5: Delivery Dates for AIRS Science Versions

| | | | |
|------|---------|--------|---------|
| v3.0 | 4/9/03 | 5/8/03 | |
| v3.7 | | | |
| v4.0 | 2/15/05 | 5/05 | 7/13/05 |

7.9.1 differences between v3.0 and v4.0

- rejection thresholds (see Table 7.6)
- IVRADREJ = 8(v3), 1(v4)
- v7 RTA used in v3.0, v3.1.8
- modified to use L1 QA flags per NOAA prescription
- upgrade tuning: 040625_v8b.tuning.asc
- upgrade error term: 040625_v8b.rtaerr.asc
- wind = 15 m/s (v3) and 5 m/s (v4)
- # cloud clearing channels: 75(v3), 45(v4)
- cloud height optimized (HWGT,75iter,deptapclmax,cloudts2)
- # surface channels: 59(v3), 31(v4)
- surface Bmax 0.2(v3), 0.5(v4)
- # surface emissivities = 9(v3), 2(v4)
- # surface reflectivities = 3(v3), 1(v4)
- # temperature channels: 153(v3), 69(v4)
- # water channels: 73(v3), 42(v4)

7.9.2 differences between v4.0 and v4.2

- Added water residual (qualwatr) rejection test
- perform at least 3 iteration before terminating iterations with 75% convergence test in all retrieval steps.
- perform at least 3 iteration before terminating iterations with 75% convergence test in water retrieval.
- remove STEPS ETA #2, SURFACE #1, AMSU #2
- RTA v8c (with new down-welling term)
- Use new NOAA surface regression (050415_noaa.reg)
- use new ensemble error estimate file (jpl_100c) (NOTE: not in GSFC v4.2, but is in GSFC v4.3)
- added CO2 (type=4) and CH4 first guess fields after NOAA regression step (NOT in GSFC system)
- use nchtemp=68 (was 58)

- remove 937.91 from freqwatr()
- use 9 ozone functions (was 7)
- 36 CO channel, w/ COwgt=1.5 (was 20 chl's, COwgt=1.75)
- 99 CH4 chl's w/ CH4wgt = 0.2
- 50 CO2 chl's w/ CO2wgt=0.3
- only use AMSU.5 for IVRADREJ
- Use CO2 fast transmittance code

7.9.3 differences between v4.2 and v4.3 (July 2005)

- Fixed non-physical transmittance problem with MIT retrieval (not installed as a namelist option).
- Move CO2fg and CH4fg to beginning of steps (These are NOT in GSFC v4.3)
- remove microwave channels from surface retrieval & do not solve for microwave emissivity
- reduce stratospheric T(p) channels from 23 to 11
- remove AMSU chls 3,4,5,6 from coupled T(p) retrieval

7.9.4 differences between v4.3 and v4.5 (Sep. 2005)

- Use RTA v8d (new Ozone adjustments to coefficients)
- Remove IR RTA error term (rtaerrfile = v45_rtaerr.asc)
- Set Twgt=0.25, Twgt2=0.25 (was 0.5)
- Set Owgt=0.5 (was 0.75)
- Turn on diff_co2() in T(p) retrieval (2%) and CCR (2%)
- Make CH4 function 2% (was 10%) and adjust CH4wgt=0.9 (was 0.18)
- add diff.CH4 (2% CH4 error) in RETWATR retrieval (NOT in GSFC v4.5)

7.9.5 v3.x rejection methodology

The airtsb program tucks a number of quality indicators into the ispare() and r spare() fields of the retrieval file. All output files (.mit, .fg, and .ret) have the same QA indicators as determined at the end of the processing. Some of these flags are compared to thresholds.

The value of ispare(2) is a bit-wise flag that has a summary of rejection criteria from the NOAA physical retrieval, the MIT microwave retrieval, and the NOAA regression retrieval. A value of zero means all retrievals completed successfully.

If the physical retrieval failed (ispare(2), bit.1 is non-zero) then the r spare() fields can be tested to see why the retrieval was marked bad. The tests are defined by rejthresh(). See Table 7.6 for definitions for various v3.x versions. For v4.0 the rejection thresholds are set to a mid-tropospheric test.

7.9.6 v4.0 rejection methodology

In v4.0 rejection there are 11 flags. Each flag can be set to 0, 1, or 2. The meaning of these QA levels is:

0 = "highest quality" means include in statistics and gridding

1 = "good quality" means include in gridding, not statistics

2 = "bad" means do not grid or include in statistics

There are a number of tests. A flow chart is provided on the next couple of pages. The meaning of the variables in the flow chart is given in this table:

```

NFO = non-frozen ocean (MIT surface class 2)

liqwat = Total Liquid Water (gm/cm^2, from MIT ret) = r spare(22)
etarej_1 = Cloud clearing residual (K) from 1st CC = r spare(24)
Aeff_1 = Effective amplification factor from 1st CC = r spare(14)
NOAA_PC_score = Principal Component Score = r spare(23)

code_problems = flag if codes fail to complete = r spare(12)
DR_AMSU5 = obs-calc(final state) for AMSU Ch1.5 = r spare( 7)
DT_b2 = RMS of T(p) difference between final T(p)
        and T(p) from previous AMSU in bottom 2-km's = r spare( 8)
amplif_4 = amplification factor from last CC = r spare( 6)
Aeff_4 = Effective amplification factor from last CC = r spare(15)
qualsurf = residual from surface retrieval = r spare( 7)
qualtemp = residual from temperature retrieval = r spare(10)
qualwatr = residual from water retrieval = r spare(11)
DTs_init_final = ABS(Tsurf(NOAA)-Tsurfret(FINAL)) uses r spare(30)
final_cloud_fraction = total FOR cloud fraction = r spare( 4)

```

There are many problems with this approach. For this reason, the code only prints v4.0 acceptance statistics at this time. The program simstat.F (next chapter) can apply these rejection tests. Here is my quick list of the problems:

- When a retrieval fails (usually due to cloud clearing) the errors tend to propagate vertically. Therefore, accepting one vertical region (*e.g.*, mid-troposphere) when the surface and/or lower troposphere is bad doesn't make sense.
- water, CO, etc. depend on the temperature state.
- Many of the flags do not use QA from that retrieval (it *e.g.*, the water flag, Qual_H2O, doesn't test if the retrieval convergence (qualwatr) was good).
- The tests and thresholds are empirically determined using 1 day (Sep. 6, 2002).

In the v3.x rejection (a.k.a., one size fits all) the surface and temperature retrieval convergence was tested (qualsurf & qualtemp) and it was deemed unnecessary to test all the individual convergence criteria). In v4.0 the logic has changed and as a result the user can be misled. For example, the flag Qual_H2O has proven to be useless. Instead the water retrieval is now QA'd with the Qual_Temp_Profile flags.

Table 7.6: rejection thresholds for v3.0,3.18,v4.0

| rejthresh index | rspare index | short description | v3.0 | v3.1.8 | *v4.0 |
|---|-----------------|-----------------------------|------|--------|-------|
| | | ieta_rej | 1 | 1 | 2 |
| | | rejlandfac | 1.0 | 1.0 | 1.0 |
| | | rejlwtempfac | - | 1.5 | 1.0 |
| | | rejlwtemp | - | 265 | - |
| 1 | 3 | etarej(ieta_rej) | 4.0 | 4.0 | 3.0 |
| 2 | 7 | IRx | 5.0 | 1.0 | 1.0 |
| 3 | 8 | RB2 | 1.5 | 500 | off |
| 4* | 4 | cldfrc lower | 0.6 | 0.6 | 0.6 |
| 5* | 6 | A(end) | 3.0 | 3.0 | 3.0 |
| 6 | 9 | qualsurf | 4.0 | 4.0 | 4.0 |
| 7 | 10 | qualtemp | 4.0 | 4.0 | 2.0 |
| 8* | 5 | cldfrc(P>500) | 0.1 | 0.1 | 0.1 |
| 9 | 11 | qualwatr | | | |
| 10 | 4 | clfrfc upper | 0.8 | 0.8 | 0.9 |
| 11 | | qual cldhgt | | | |
| 12 | 13 | Ts(NOAA)-Ts(MIT) | | | |
| 13 | | a0 patho test | | | |
| 14 | | chi ² patho test | | | |
| 15 | | clearflag | | | |
| 16 | 6 | A(end) | 5.0 | 5.0 | 5.0 |
| 17 | 15 | A_eff(end) | | | 30.0 |
| 18 | | | | | |
| 19 | 18 | fzeta | | | |
| 20 | 19 | rzeta | | | |
| 21 | 22 | totliqwat | 0.03 | 0.04 | 0.03 |
| 22 | 15 | A_eff(oc) | | 10.0 | 30.0 |
| 23 | | spare | | | |
| 24 | 14 | A_eff(1) | | | |
| 25 | 26 | cldfrc(1) | 0.8 | 0.8 | off |
| 26 | 24 | etarej(1) | 5.0 | 5.0 | 5.0 |
| 27 | 25 | A(1) | 5.0 | 5.0 | 5.0 |
| 28 | 23 | NOAA_score | | | 2.0 |
| 29 | | | | | |
| 30 | 30 | Ts(NOAA) | | | |
| 31 | 29 | gtest_a(end) | | | |
| 32 | 31 | NOAA Diff Test | | | |
| rejthresh(4),(5),(8) are logically AND'ed | | | | | |
| * v4.0 rejection thresholds emulate mid-tropospheric test | | | | | |

```

Qual_Cloud_OLR = 2, cloud fractions & top pressure, OLR
Qual_H2O = 2,          water retrieval
Qual_CO = 2,          carbon monoxide retrieval
Qual_CO2 = 2,         carbon dioxide retrieval
Qual_O3 = 2,          ozone retrieval
Qual_CH4 = 2,         methane retrieval
Qual_Temp_Profile_Top = 2, T(p<200 mb)
Qual_Temp_Profile_Mid = 2, T(p>200 mb, z>3 km)
Qual_Temp_Profile_Bot = 2, T(z<=3km)
Qual_Surf = 2,        Tskin, emissivity, reflectivity
Qual_CC_Rad = 2,      cloud cleared radiance
|
+-----+-----+
|Did MW/Strat IR and cloud retrieval complete?| ---> No -> no output -> stop
+-----+-----+
| yes
|
Qual_Cloud_OLR = 1
|
+-----+-----+
Up      |Did MIT product pass previously?|-> Fail any test -> goto (A)
Front   |aeff_1 < 200?                    |
Check   +-----+-----+
        | pass all
        |
Coarse  +-----+-----+
Check   |final_cloud_fraction < 0.9?      |
        |code_problems = 0                |-> Fail any test -> goto (A)
        |NOAA_PC_score < 10              |
        +-----+-----+
        | pass all
        |
Qual_Temp_Profile_Top = 0
Qual_Cloud_OLR = 0
|
Water & +-----+-----+
Ozone   |etarej_1 < 8 | bypass
Check   |amplif_4 < 8 | -----+
        |liqwat < 0.03 |
        +-----+-----+
        | pass all
        |
Qual_H2O = 0
Qual_O3 = 0
Qual_CO = 0
Qual_CH4 = 0 *
Qual_CO2 = 0 *
        |
        +-----+-----+
|
(continue on next page)

```

* off-line code only

```

      (from previous page)
      |
      +-----+
      | etarej_1 < 6 |
      | DT_b2 < 2 |
      | DR_AMSU5 < 2 |
      | qualtemp < 0.75 |
      | amplif_4 < 2 |
      | liqwat < 0.03 |
      | NOAA_PC_score < 4 |
      | if(NFO) then | --> Fail -> goto (B)
      |   etarej_1 < 2 |
      |   aeff_1 < 30 |
      |   qualsurf < 0.75 |
      +-----+
      | pass all |
      |
      Qual_Temp_Profile_Mid = 0
      Qual_Temp_Profile_Bot = 1
      Qual_CC_Rad = 0
      If (.NOT.NFO) then Qual_surf = 1
      |
      +-----+
      | etarej_1 < 1.5 |
      | DTs_init_final < 1.5 |
      | Aeff_1 < 30 | --> Fail -> goto (B)
      | if (NFO) then Aeff_4 < 15 |
      +-----+
      | pass all |
      |
      Qual_Temp_Profile_Bot = 0
      |
      +-----+
      | Aeff_1 < 8 |
      | Aeff_4 < 8 |
      | NOAA_PC_score < 1.2 | --> Fail -> goto (B)
      | liqwat < 0.01 |
      +-----+
      | pass all |
      |
      Qual_surf = 1
      |
      +-----+
      | aeff_1 < 5 | --> Fail -> goto (B)
      +-----+
      |
      Qual_surf = 0
      |
      goto (B)

```

(A) output MW/SIR solution

(B) Output IR Solution

7.9.7 Example output for quality indicators in the .out file

File Edit Options Buffers Tools Help

v3_reject: temp_low = 265.0 temp factor, f = 1.00 land factor, L = 1.00

```

flagged  unique  flag name & rejection threshold
  190      6 MIT,NOAA REJ MIT or NOAA step(s) were rejected
  235      3 LIQ water |rspare(22)| > rejthresh(21)= 0.030
1897     20 NOAA Diff |rspare(31)| > rejthresh(32)= 9.000
1356     61 cldfrc(1) |rspare(26)| > rejthresh(25)= 0.980
  186      0 etarej(1) |rspare(24)| > rejthresh(26)= 6.000*f*L
   65      0 A(1) |rspare(25)| > rejthresh(27)= 9.000*f
2680    207 Aeff(1) |rspare(14)| > rejthresh(24)= 30.000
  321      0 noaaPCscore |rspare(23)| > rejthresh(28)= 4.000
1006     39 Ts(R)-Rs(M) |rspare(13)| > rejthresh(12)= 10.000*f
1004     15 cldfrc(end) |rspare( 4)| > rejthresh(10)= 0.900
  442      0 etarej(2) |rspare( 3)| > rejthresh( 1)= 2.500*f
1216      6 A(end) |rspare( 6)| > rejthresh(16)= 2.000*f
1839     52 Aeff(end) |rspare(15)| > rejthresh(17)= 16.000
2117    131 first zeta |rspare(18)| > rejthresh(19)= 8.000
  217      1 rest-o-zeta |rspare(19)| > rejthresh(20)= 5.000
   63      0 gtest_a |rspare(29)| > rejthresh(31)= 9.000
  823     23 qualsurf |rspare( 9)| > rejthresh( 6)= 1.500*f
  740      6 qualtemp |rspare(10)| > rejthresh( 7)= 1.000*f
1877    263 qualwatr |rspare(11)| > rejthresh( 9)= 1.500*f
  116      0 BTamsu |rspare( 7)| > rejthresh( 2)= 2.000*f
  932     50 dT(bt2) |rspare( 8)| > rejthresh( 3)= 2.000*f
1209     49 Ts(F)-Ts(N) |Ts(FINAL)-rspare(30)| > rejthresh(30)= 4.000*L
   23      0 Low Cloud Logical AND of:
          cldfrc = |rspare( 4)| > rejthresh( 4)= 0.900
          A(end) = |rspare( 6)| > rejthresh( 5)= 8.000*f
          cldlow = |rspare( 5)| > rejthresh( 8)= 0.100
1657    284 Aeff(NFO) |rspare(15)| > rejthresh(22)= 8.000
  251      0 Patho-cloud Logical AND of:
          alpha0 = rspare(16) > rejthresh(13)= 0.990
          chi^2 = rspare(17) < rejthresh(14)= 5.000
          Aeff = |rspare(15)| >= rejthresh(18)= 18.000
  215      0 Software Errors in code (see softexit.F)
BTAMSU uses 1 channels:      5

```

```

number of cases with 1 rejection flags set = 1216
number of cases with 2 rejection flags set = 864
number of cases with 3 rejection flags set = 717
number of cases with 4 rejection flags set = 582
number of cases with 5 rejection flags set = 400
number of cases with 6 rejection flags set = 321
number of cases with > 7 rejection flags set = 1118

```

```

          CLEAR  2 FOV  0 CLD  1 CLD  2 CLD  3+CLD
ALL eta(1)      0      0    11  1722  3334  2043

```

| | | | | | | |
|---------------------|------|------|------|------|------|------|
| ACC eta(1) | 0 | 0 | 7 | 712 | 1000 | 173 |
| ALL eta(end) | 0 | 0 | 12 | 2130 | 2669 | 2299 |
| ACC eta(end) | 0 | 0 | 8 | 890 | 841 | 153 |
| PGE rejection codes | =0 | =1 | =2 | | | |
| Qual_Cloud_OLR | 5996 | 899 | 215 | | | |
| Qual_h2o | 5885 | 0 | 1225 | | | |
| Qual_CO | 5885 | 0 | 1225 | | | |
| Qual_CO2 | 5885 | 0 | 1225 | | | |
| Qual_O3 | 5885 | 0 | 1225 | | | |
| Qual_CH4 | 5885 | 0 | 1225 | | | |
| Qual_Temp_Top | 5996 | 0 | 1114 | | | |
| Qual_Temp_Mid | 3955 | 0 | 3155 | | | |
| Qual_Temp_Bot | 2099 | 1856 | 3155 | | | |
| Qual_Surf | 428 | 2111 | 4571 | | | |
| Qual_CC_Rad | 3955 | 0 | 3155 | | | |

Chapter 8

simstat: off-line statistics of ensembles of retrievals

The values shown in the left panel of the retrieval error (using plsim) are layer mean temperature errors in roughly 1 km layers from the surface to 300 mb, ≈ 3 km layers from 300 mb to 30 mb, and ≈ 5 km layers between 30 mb and 1 mb. The layer boundaries are shown as light dotted lines. The values shown in the center panel (of pmult=3 plot) are water vapor profile results, with results plotted corresponding to percent error of column integrated water vapor in ≈ 2 km layers from the surface to 100 mb, and ≈ 4 km layers from 100 mb to 1 mb.

$$\hat{q}_{ret}(i) = \sum_{L=LEV(i-1)}^{LEV(i)} q_{ret}(L) \quad (8.1)$$

$$\hat{q}_{tru}(i) = \sum_{L=LEV(i-1)}^{LEV(i)} q_{tru}(L) \quad (8.2)$$

The RMS percent error in coarse layer, $e(i)$, over an ensemble of n profiles is weighted by the quantity of water (ozone % errors are done this way), $q(p)$, as follows:

$$BIAS(e(i)) = \frac{1}{N} \sum_{n=1}^N (\hat{q}_{ret}(i, n) - \hat{q}_{tru}(i, n)) \quad (8.3)$$

$$RMS(e(i)) = \sqrt{\frac{1}{N} \sum_{n=1}^N (\hat{q}_{ret}(i, n) - \hat{q}_{tru}(i, n))^2} \quad (8.4)$$

$$\%BIAS(e(i)) = 100 \cdot \frac{\frac{1}{N} \sum_{n=1}^N (\hat{q}_{ret}(i, n) - \hat{q}_{tru}(i, n))}{\frac{1}{N} \sum_{n=1}^N (\hat{q}_{tru}(i, n))} \quad (8.5)$$

$$\%RMS(e(i)) = 100 \cdot \frac{\sqrt{\frac{1}{N} \sum_{n=1}^N \left[\hat{q}_{tru}(i, n) \cdot \left(\frac{\hat{q}_{ret}(i, n) - \hat{q}_{tru}(i, n)}{\hat{q}_{tru}(i, n)} \right) \right]^2}}{\sqrt{\frac{1}{N} \sum_{n=1}^N [\hat{q}_{tru}(i, n)]^2}}$$

$$= \frac{\sqrt{\frac{1}{N} \sum_{n=1}^N [\hat{q}_{ret}(i, n) - \hat{q}_{tru}(i, n)]^2}}{\sqrt{\frac{1}{N} \sum_{n=1}^N [\hat{q}_{tru}(i, n)]^2}} \quad (8.6)$$

For T(p)

NLEVPHIL = 31,

LEVPHIL = 1, 12, 14, 16, 17, 18, 20, 21, 24, 26, 30, 33, 37, 41, 44,
47, 49, 51, 54, 57, 61, 64, 66, 70, 74, 77, 81, 84, 88, 92,
100,

For q(p)

NLEVWAT1 = 21,

levwat1 = 1, 16, 20, 23, 29, 37, 44, 49, 52, 55,
58, 63, 67, 70, 72, 76, 80, 85, 89, 93, 100,

The simstat program can be used for many applications; however, strict compliance to file formats and rejection formats must be taken to ensure proper operation of this software.

There is a known bug that occurs when the retrieval surface pressure is significantly different than the truth surface pressure. It is possible the program will compare the retrieval to non-sensical truth fields if $P_{surfret} > P_{surftru}$. This has been repaired for the moisture statistics; however, the temperature statistics have not been repaired. If the vendor retrieval algorithm retrieves surface pressure or uses a first guess surface pressure that is derived from a forecast model then a rejection based on positive values of $(P_{surfret} - P_{surftru})$ should be imposed. The quantity $(P_{surfret} - P_{surftru})$ is printed in the statistics output file for each profile.

This section describes the methodology and provides an example use of the simstat program which can be used to compute statistics of retrievals.

The specification of 1-km layers is rather vague and, therefore, there are many possible intergrations. The intent was to keep it very simple; however, there are two physical realities that make it complex

- Given that we use a fixed pressure grid, the height is a function of the geophysical state, due to variability of water, gravity, etc.
- Topography makes the thickness of the bottom fixed layer variable. If we build the layers upward from the real surface, then inter-comparison and thick statistics becomes extremely difficult.

To keep the meaning of the statistics as simple and reasonable as possible we convert the 100 fixed pressure levels to fixed coarse layers that are reasonable close to the thickness intervals specified in the EDR's. If topography makes the bottom coarse layer less than 750 meters then it is ignored in the calculation, therefore, the number of profiles within the lowest few coarse statistics intervals can be less than the total number of profiles.

For temperature statistics (NOTE the -wptb98 option in simstat) the coarse layers are shown in the table below. The coarse layer i is bounded by $p(L(i-1))$ on the top and $p(L(i))$ on the bottom. The top level $L(0)$ is defined as zero with pressure value of 0.005. This level is implied and is never directly indexed. The subroutine fin2slb.F will convert the truth and retrieval temperature profiles from the fine grid levels, $T(l)$, to averages on the coarse grid, $T_{coarse}(i)$

$$T_{coarse}(i) = \frac{\sum_{l=1}^{L(1)} T(l)}{L(1)} \quad \text{for } i = 1$$

$$T_{coarse}(i) = \frac{\sum_{l=L(i-1)}^{L(i)} T(l)}{1 + L(i) - L(i-1)} \quad \text{for } i > 1 \quad (8.7)$$

The statistics are then computed (bias, RMS, and standard deviation) for the coarse grid values. The coarse layer thicknesses are computed from the truth (using height.F) and statistics are also kept on those values for validation of the correctness of these coarse layers.

| coarse layer definition for $T(p)$ | | |
|------------------------------------|-----------|-----------------------------|
| $L(i)$ | $p(L(i))$ | approximate layer thickness |
| 5 | 0.224 | 17.5 |
| 10 | 1.297 | 13 |
| 15 | 4.077 | 8.2 |
| 19 | 8.165 | 4.7 |
| 24 | 16.43 | 4.6 |
| 30 | 32.27 | 4.3 |
| 35 | 51.53 | 2.9 |
| 41 | 83.23 | 3.0 |
| 48 | 133.85 | 3.0 |
| 56 | 212.03 | 3.0 |
| 64 | 314.14 | 2.5 |
| 67 | 358.97 | 0.9 |
| 70 | 407.47 | 0.9 |
| 74 | 477.96 | 1.1 |
| 77 | 535.23 | 0.8 |
| 81 | 617.51 | 1.1 |
| 85 | 706.57 | 1.0 |
| 89 | 802.37 | 1.0 |
| 93 | 904.87 | 1.0 |
| 100 | 1100.0 | 1.0 |

In addition, statistics are kept for vertical regions and are the averages of the statistics within the intervals given above. They have the following properties

| p_{upper} | p_{lower} | average thickness |
|-------------|-------------|-------------------|
| 1 | 32 | 5.43 |
| 32 | 314 | 2.85 |
| 314 | 707 | 1.01 |
| 707 | 1100 | 0.986 |

For moisture, the statistics are for layer values. The moisture and ozone in the noaa88b.asc file was specified in units of volumetric mixing ratio. Upon reading the file, (readl2.b.F) the moisture and ozone mixing ratio's, $r(p)$, units are converted into column densities, $q(p)$, (molecules per cm^2 per fine layer) using the subroutine mw2cd.F. Appendix-A describes the methodology of this subroutine.

The column densities are added for the fixed coarse statistics layer, analogous to the way temperature levels were added, using the subroutine precip.F. In order to avoid having low moisture regions dominate the statistic, the true water amount is used as a weight. For coarse layer i is bounded by $p(L(i-1))$ on the top and $p(L(i))$ on the bottom. As before, the top level $L(0)$ is defined as zero with pressure value of 0.005. Therefore, $q(l=1)$ is the column density from 0.005 to 0.016 mb and the indexing value of $l = 1 + L(i-1)$ reduces to $l = 1$ for the top layer. The error in moisture for coarse layer i is then given by

$$e(i) = \frac{\sqrt{\left(\sum_{l=1+L(i-1)}^{L(i)} q_{tru}(l) \cdot \left(\frac{q_{ret}(l)-q_{tru}(l)}{q_{tru}(l)}\right)\right)^2}}{\sqrt{\left(\sum_{l=1+L(i-1)}^{L(i)} q_{tru}(l)\right)^2}} = \frac{\text{RMS}\left(\sum_{l=1+L(i-1)}^{L(i)} q_{ret}(l) - q_{tru}(l)\right)}{\text{RMS}\left(\sum_{l=1+L(i-1)}^{L(i)} q_{tru}(l)\right)} \quad (8.8)$$

| "1-km" water coarse layer specification | | |
|---|-----------|-----------------------------|
| $L(i)$ | $p(L(i))$ | approximate layer thickness |
| 5 | 0.224 | 17.5 |
| 10 | 1.297 | 13.0 |
| 17 | 5.878 | 10.7 |
| 20 | 9.512 | 3.2 |
| 24 | 16.43 | 3.6 |
| 30 | 32.27 | 4.3 |
| 37 | 60.99 | 4.0 |
| 44 | 103.02 | 3.2 |
| 49 | 142.39 | 2.0 |
| 52 | 170.08 | 1.1 |
| 55 | 200.99 | 1.1 |
| 58 | 235.23 | 1.0 |
| 62 | 286.26 | 1.3 |
| 67 | 358.97 | 1.5 |
| 70 | 407.47 | 0.9 |
| 74 | 477.96 | 1.1 |
| 77 | 535.23 | 0.8 |
| 81 | 617.51 | 1.1 |
| 85 | 706.57 | 1.0 |
| 89 | 802.37 | 1.0 |
| 93 | 904.87 | 1.0 |
| 100 | 1100.0 | 1.0 |

In addition, statistics are kept for vertical regions and are the averages of the statistics within the intervals given above. They have the following properties

| p_{upper} | p_{lower} | average thickness |
|-------------|-------------|-------------------|
| 201 | 407 | 1.2 |
| 407 | 618 | 1.0 |
| 618 | 1100 | 1.0 |

Also, statistics are kept for 2-km layers specified as follows

| "2-km" water coarse layer specification | | |
|---|-----------|-----------------------------|
| $L(i)$ | $p(L(i))$ | approximate layer thickness |
| 5 | 0.224 | 17.5 |
| 10 | 1.297 | 13.0 |
| 17 | 5.878 | 10.7 |
| 20 | 9.512 | 3.2 |
| 24 | 16.43 | 3.6 |
| 30 | 32.27 | 4.3 |
| 37 | 60.99 | 4.0 |
| 44 | 103.02 | 3.2 |
| 49 | 142.39 | 2.0 |
| 54 | 190.32 | 2.0 |
| 61 | 272.92 | 2.3 |
| 68 | 374.72 | 2.1 |
| 74 | 477.96 | 1.7 |
| 82 | 639.14 | 2.2 |
| 89 | 802.37 | 1.8 |
| 100 | 1100.00 | 1.5 |

In addition, statistics are kept for vertical regions and are the averages of the statistics within the intervals given above. They have the following properties

| p_{upper} | p_{lower} | average thickness |
|-------------|-------------|-------------------|
| 190 | 273 | 2.3 |
| 273 | 639 | 2.0 |
| 639 | 1100 | 1.7 |

8.1 Using the SIMSTAT program

The simstat program reads all uses selections from the command line. The first 3 items are the name of the truth (or reference) file, retrieval file, and the output file. After that options can be entered in any order. For IPO work the -wptb98 option should be used to conform to the standard levels described above.

The simstat program will make an attempt to reconcile differences between the number of records in these two files and interpret these as number of fields-of-view per truth record (this program was written for CrIS/AMSU where there were 9 CrIS fields-of-view per retrieval field-of-regard). Therefore, it is imperative that the truth file and retrieval file have the appropriate number of records.

If the vendors have performed 1 retrieval per truth record (this will be my assumption) then the retrieval file must have exactly the same number of records as the truth file. Any retrieval sub-sets of the noaa88b.asc must contain a record, even if it is rejected or not used in the statistics (due to options specified on the command line).

So the simstat command to compute statistics for all profiles in the datasets would be
 exe/simstat.exe truthfile refile statfile -wptb98

where,

truthfile = truth path + file name

refile = retrieval path + file name

statfile = user name for simstat output file with statistics tables

There are many options for the simstat program. In the table below all the options are listed with an asterisk if they might be useful for AIRS.

| options for the SIMSTAT program | |
|---|---|
| option | short description |
| -notable -wptb98 -jpl | DOES NOT print summary line for each case will use IPO/CrIS coarse statistics levels will use JPL ALL coarse statistics levels |
| -wgt -wgtout -multiscan | is an option to read in a weighting file is an option to write out a weighting file makes the truth, ret, and weight files indirect |
| -rej1 -rej2 -rej3 -pgev30 -pgev40 -pgelev1 -rejsbrs -rejitt | uses ispare(1) as rejection flag (see below) uses ispare(1),ispare(2) as rejection flag (see below) uses ispare(1),ispare(2) as rejection flag (see below) computes v3.x rejection from rspares() reads rejection thresholds from ./rejthresh.dat computes v4.0 rejection flags = 0 (uses v4_reject.F) allows v4.0 rejection flags =0 or =1 uses ispare(1) as rejection flag (see below) uses ispare(1) as rejection flag (see below) |
| -rej50 -rej80 -coslat -tropics -polar -midlat -lat40 -lat60 -ocean -land -coast -day -ngt -clear | only includes cases where cloudiness is $\leq 50\%$ only includes cases where cloudiness is $\leq 80\%$ adds a weighting factor of $\text{COS}(\text{latitude})$ only includes cases where latitude is within $\pm 23.5^\circ$ only includes cases where latitude is <i>ge</i> 50° only includes cases where $23.5 < \text{latitude} < 50^\circ$ limit latitude to with $\pm 40^\circ$ limit latitude to with $\pm 60^\circ$ only includes cases where % land = 0 only includes cases where % land is = 0 only includes cases where $0 < \% \text{ land} < 1$ use only daytime (solzang <i>i</i> 90) use only nighttime (solzang <i>i</i> 90) use only cases declared clear, ispare(7)=1 also creates list of cases in ./clear_lst.txt |
| -showir -shownmw -showrej -cldhi -cldlo | only uses IR accepted cases only uses microwave accepted cases shows all rejected cases only includes cases where cloudiness is $>80\%$ cloudy only includes cases where cloudiness is $\leq 80\%$ |
| -avg -fov1 -set9 | averages the truth FOV's within a FOR forces the use of truth FOV # 1 within a FOR used primarily with ERGS dataset will force 9 truth FOV's per FOR |
| options is TRUTH is RAOB file | |
| -rbsum -disttf -dfxxx.y -timedf -tmdfx.y -rbinst file -rblev | creates summary of RAOB's in ./raob_summ.txt limit RAOB to 100 km radius limit RAOB to xxx.y km radius limit RAOB to 3 hour difference limit RAOB to x.y hour difference Use only NOAA selected RAOB types, specified in file Use only the RAOB valid levels for comparison |

For RAOB files (must be used as reference profile in call to simstat) the rspare fields are used for co-location and pressure range information as follows

- ispare(3:5) is the STATION ID (2 ASCII characters per ispare)
- ispare(6) is the instrument ID
- rspare(1) is the RAOB-AIRS time difference,
- rspare(2) is the RAOB-AIRS distance,
- rspare(6) is the upper valid pressure level,
- rspare(7) is the lower valid pressure level,

If -rbinit file option is used then a file defines the valid RAOB instrument types (ispare(6)) and RAOB stations (sites) which is an 6-character ASCII string located in ispare(3:5). The file is a field formatted file (*i.e.*, spaces matter) with a format of

```
(a10,i3)
(10(i6,1x))    <- multiple lines of these if INSTTYPE > 10
(a10,i3)
(10(a6,1x))    <- multiple lines of these if SITETYPE > 10
```

If the # of instrument types or # of station ID's is less than or equal to zero then NO ID's are read in. example of RAOB selection file when both instrument types and station ID's are selected.

```
INSTTYPE =      5
  27  51  52  61  71
SITETYPE =      3  !
  02963  02365  LDWR
```

An example of RAOB selection file for selecting only instrument types

```
INSTTYPE =      24  ! (a10,i6) followed by (10i4) lines
  9  12  19  27  28  29  37  45  47  49
  50  51  52  55  60  61  62  63  66  67
  71  74  78  90
SITETYPE =      -1  (allow all sites)
```

example of RAOB selection fil for selecting only sites

```
INSTTYPE =      -1
SITETYPE =      3  !
  02963  02365  LDWR
```

The simstat.F program also allows selecting cases using a weighting file. For example, a weighting file could be created by an external program based on any criteria the user wanted to prescribe. This is typically used when comparing a number of different retrievals on a commonly accepted set of profiles. A program is written to search through all the retrievals and make a list of profiles that were accepted by all retrievals. Then simstat is called for each retrieval but using the weight determined by all retrievals. The weight file is machine readable. The first line is the number of profiles and then each line that follows has the integer profile index (that index which is determined from the subroutine rd_jdpro() using the idprof character string from the truth record) and the weight (a real value between zero and one) on each line.

```
exe/simstat.exe truthfile refile statfile -wptb98 -wgt weightfile
```

The simstat program will also use the `ispare()` field within the retrieval file to determine if the record should be used within the statistics (`-rej1`, `-rej2`, `-rej3`, `-rejsbrs`, `-rejitt`, `-pgev30`, `-pgev40`, `-pgelev1`). Many of these are specific to vendor formats (*e.g.*, `-rejsbrs`, `-rejitt`) or old formats of output (`-rej1`, `-rej2`). Internally, there are only 3 types specified by a variable called `irejval`: 0=rejected, 1=accepted infrared/microwave, 3=microwave only. The directives `-showir`, `-showmw`, `-showrej` will then determine if rejection type 0, 1, or 3 are displayed. The default is `irejval=0`. The retrieval file can specify these in the following ways

`-rej1` `ispare(1)` is set equal to the internal value (`irejval=0,1,or 3`)

`-rej2` an old GSFC format where `ispare(1)=1` if microwave only, `ispare(2)=1` if rejected. If either of these are set then it is treated as microwave only (`irejval=3`), and if both `ispare(1)` and `ispare(2)` are zero then `irejval=1`

`-rej3` `ispare(1)=1` if microwave only (`irejval=3`), `ispare(2)=1` if rejected (`irejval=0`). If both are zero then `irejval=1`.

`-pgev30` uses `ispare()` and `rspace()` QA indicators as defined in Section 7.9.5 and calls `src_gsfc3_reject.F` to compute the rejection in the same manner as the `airsb` code did. The rejection thresholds are read in from a file in the current directory called `rejthresh.dat`. See the `io_xxx.nl` file for values used in your retrieval. The format of the file is:

```
(i6)      # of rejection indicators (should be >= 32)
(3f6.2)  rejlwtemp, rejlwtempfac, rejlandfac
(i4,f9.2) i, rejthresh(i)
```

`-pgev40` performs the v4.0 rejection methodology (see Section 7.9.6 for details). This uses `ispare()` and `rspace()` from the retrieval file and computes rejection. If the `-wgtout` option is used the flags are written out such that other files can be compared with the same rejection using the `-wgt` option. The v4.0 QA indicators must be set to zero to be accepted.

`-pgelev1` same as `-pgev40` except that the QA indicators can be zero or one.

`-rejsbrs` SBRS definition of rejection for CrIS/AMSU retrievals. `ispare(1)` has the following interpretation

- 0 first guess (`irejval=0`)
- 1 accepted infrared and microwave (`irejval=1`)
- 2 rejected infrared, microwave returned (`irejval=3`)
- 3 accepted, microwave (`irejval=3`)
- 4 rejected, microwave without infrared channels (`irejval=3`)
- 5 accepted, microwave with infrared channels (`irejval=3`)

`-rejitt` ITT/AER definition for rejection for CrIS/AMSU. `ispare(1)` has the following interpretation

- 0 Background (first guess) (`irejval=0`)
- 1 infrared/microwave retrieval (`irejval=1`)
- 2 microwave only retrieval (`irejval=3`)

It is possible to use these options to build rather complex interactions. For example, a given retrieval file could be used to build a weight file, as follows

```
exe/simstat.exe truthfile refile1 weightfile -wptb98 -rej3 -ocean -tropics -wgtout yourfile
```

then another retrieval could be forced to use the same exact weighting

exe/simstat.exe truthfile refile1 weightfile -wptb98 -wgt yourfile

We have provided a subroutine (`~/src_util/mr2cd.F`) to convert the mixing ratio values to layer column density (molecules/cm²), which is the required input for the government version of the 100 layer UMBC RTA. A description and derivation of the methodology for the conversion is discussed in Appendix A, at the end of this document.

8.2 Subroutines used by simstat

The routines necessary to compile simstat are

```

root/src_util/      directory of main program
  simstat.F         main program
root/src_gsfc/     general subroutines
  Makefile         make file for SGI F90 compiler
  fin2slb.F        convert fine levels to coarse (slab) layers
  getemis2.F       convert emissivity hinge points to value at channels
  grav.F           gravity as a function of latitude and height
  height.F         convert T(p),q(p) to thicknesses
  julian.F         convert calendar date to julian time
  mr2cd.F          convert mixing ratios to column densities (see appendix A)
  openl2_b.F       open L2 file and read header
  precip.F         compute coarse layer column densities
  profdist.F       statistics on profile distribution
  rd_jdpro.F       convert 8 character L2 idprof string to integer
  readl2_b.F       read L2 file record
  sunang2.F        compute solar zenith angle
  tai_to_utc.F     convert TAItime to UTC
  wcd2mr.F         convert column density to mixing ratio (approximate)

root/src_jpl/     I/O and primitive functions
  Makefile         make file for SGI F90 compiler
  paramet.com      parameter block w/ constants and dimension sizes
  lsurface.f       subroutine to find the pressure index for surface
  tai_to_date.c    convert TAI time to date (needed for new JPL formats)
  bi_test.F        routine to determine if file is binary (from name)
  softexit.F       error trap exit routine
  uppercase.F      convert string to upper case

```

Chapter 9

Diagnostic Tools

To set up IDL (in BASH) when you have a local copy of the system programs:

Edit your `.bashrc` or `.bash_profile` login file:

```
IDL_DIR=/usr/local/rsi/idl
IDL_DEVICE="X"
IDL_STARTUP=~ /prog/idl/startup.pro"
IDL_PATH=${IDL_DIR}/lib:./:~/prog/idl:~/prog/idl/general:
~/prog/idl/airs:~/prog/idl/airsb:~/prog/idl/preproc:
~/prog/idl/matchup:~/prog/idl/exercise:~/prog/idl/cris:
~/prog/idl/amsu:~/prog/idl/chris:~/prog/idl/plot_pro:
~/prog/idl/trace:~/prog/idl/preproc/eff:~/prog/idl/test_pgm"
```

copy and edit the following programs in `~/prog/idl/`

```
startup.pro
claser.pro
laser.pro
logo.pro
```

To set up IDL (in BASH) when you are on a orbit maching and you want to use the MASTER copy of the system programs. On ASL remove `"/net/orbit009l"`.

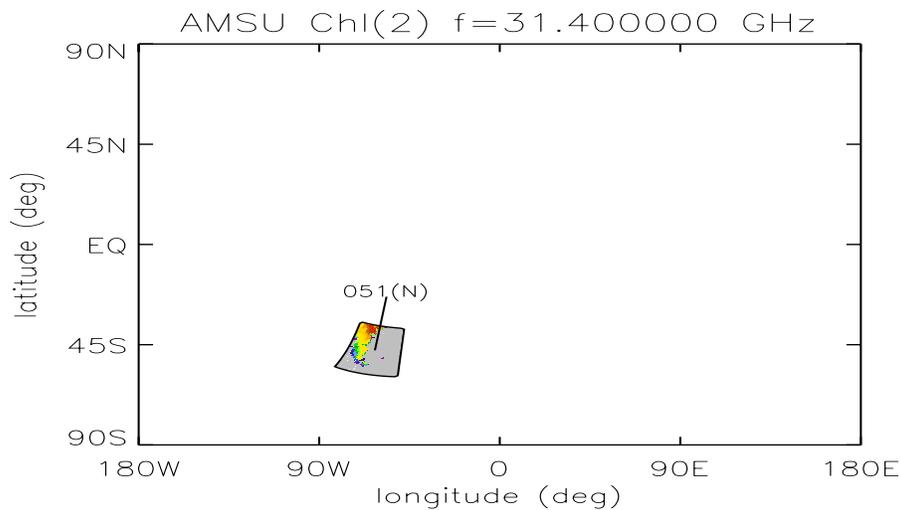
```
IDL_DIR=/usr/local/rsi/idl
IDL_DEVICE="X"
IDL_STARTUP="/net/orbit009l/home/cbarnet/prog/idl/startup.pro"
IDL_PATH=${IDL_DIR}/lib:./:/net/orbit009l/home/cbarnet/prog/idl:
/net/orbit009l/home/cbarnet/prog/idl/general:
/net/orbit009l/home/cbarnet/prog/idl/airs:
/net/orbit009l/home/cbarnet/prog/idl/airsb:
/net/orbit009l/home/cbarnet/prog/idl/preproc:
/net/orbit009l/home/cbarnet/prog/idl/avn_match:
/net/orbit009l/home/cbarnet/prog/idl/matchup:
/net/orbit009l/home/cbarnet/prog/idl/exercise:
/net/orbit009l/home/cbarnet/prog/idl/cris:
/net/orbit009l/home/cbarnet/prog/idl/amsu:
/net/orbit009l/home/cbarnet/prog/idl/chris:
/net/orbit009l/home/cbarnet/prog/idl/plot_pro:
/net/orbit009l/home/cbarnet/prog/idl/trace:
/net/orbit009l/home/cbarnet/prog/idl/preproc/eff:
/net/orbit009l/home/cbarnet/prog/idl/test_pgm"
```


9.2 pl_amsul1b: plotting AMSU granules

```
plot_pro/pl_amsul1b.pro, year, mon, day, glist, Nchl, $
[annang = [0,90,180,270]]
```

| pl_amsul1 command line systax | |
|-------------------------------|--|
| required input | description |
| year | year of observation |
| month | month of observation |
| day | day of observation |
| glist | LONARR() list of granules to plot |
| Nchl | AMSU channel number to plot |
| option | description |
| annang | select annotation angle for granule(s) |

This program expects to plot many granules, so the example below looks a bit silly. If you are looking for features within granules you can plot many granules and use a AMSU channel that sounds the region of interest. Channel 2, shown below, is the 31 GHz channel - which is a window channel. Since ocean emissivity is very low in this frequency interval it is useful to identify land types and temperature.



C.Barnet: Tue Jan 18 12:53:08 EST 2005

Figure 9.2: Example output from the pl_amsul1, 2003,1,14, [51], 2

9.3 pl_sim: plotting retrieval statistics

Plots of results (.out file) for T(p), q(p), O₃(p), CO(p) CH₄(p) and CO₂(p) RMS, BIAS, or SDV statistics.

see ~/pl_sim.ema for additional documentation. It is best to get a recent example of a .plt file (one exists in the test granule).

```
pl_sim, plotname, [ipo=0], [multi=3], $
[xmin=0.0,0.0,0.0], [xmax=[3.5,40.0,50.0]], $
[tmin=[0.0,0.0,0.0], [tmax=[75.0,8.0,8.0]], $
[/DU], [amsu=1], [menu=1], [plot='T'], [/publ], $
[csizscl=0.6], [/bias], [/sdv], [/rms], [pass=2]
```

The program will write a menu defined by the .plt file. The user can then select which plot to make. The .plt is a good way to document runs and keep track of plots for presentations.

| pl_sim command line systax | |
|-----------------------------|--|
| required input | description |
| plotname | root name of .plt file |
| option | description |
| plotname | root name of .plt file |
| ipo | format for annotation |
| multi | number of panels per plot, e.g., 2= T(p),q(p) |
| xmin | vector of minimum for T(p) ,q(p), and O ₃ (p) plots |
| xmax | vector of maximum for T(p) ,q(p), and O ₃ (p) plots |
| tmin | vector of minimum for CO(p) , CH ₄ (p), and CO ₂ (p) plots |
| tmax | vector of mximum for CO(p) , CH ₄ (p), and CO ₂ (p) plots |
| /DU | plot ozone as Dobson units |
| amsu | select which AMSU is plotted |
| csizscl | Character size scale |
| /bias | plot BIAS statistics |
| /sdv | plot standard deviation statistics |
| /rms | plot RMS statistics |
| pass | select which pass is plotted (1 or 2) |
| /publ | publication quality |
| External control parameters | |
| menu=1 | select which menu item to plot |
| plot | select 'T', 'Q', 'O' |

The program control file, test.plt, for the example below contains:

```
Nmap : 6
rem  :NOAA RET NOAA  RET NOAA  RET
umap :  0, 10,  2, 12,  1, 11,
cmap : 22, 10, 132, 120, 332, 320
bmap :  0, 40, 20, 11, 41, 21,
rem  : -----
rem  : internal mode
pmult: 3
xmax :  5.0, 75.0, 40.0
xmin :  0.0,  0.0,  0.0
tmax : 60.0,  6.0,  4.0,
tmin :  0.0,  0.0,  0.0,
amsu : 10
rejec: 1
IPO  : 2
annot: 3
qpmin: 110.0
path1: ./
rem  : =====
rem  : =====
menu : d60 run
titl1: Jan. 14, 2003. G# 51
titl2: v4.0 Emulation vs ECMWF
plot1: 10 0 d65bin_test
comm  : new ECMWF
plot1: 12 2 d60bin_test
comm  : old ECMWF
end   :
```

See Fig. 9.1 for a description of the allowed colors.

In the previous delivery of the code the wrong ECMWF reference file was provided (a 0z analysis was used instead of the 6h analysis). The plots below illustrate the improvement by using the correct ECMWF file.

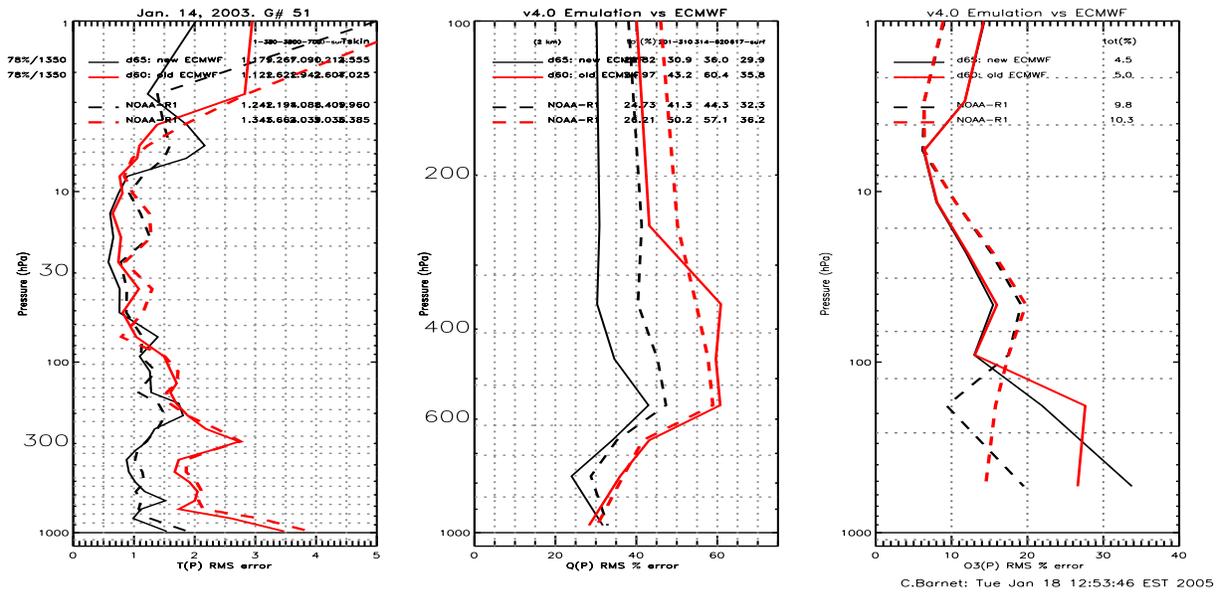


Figure 9.3: Example output from pl.sim, 'test', menu=1

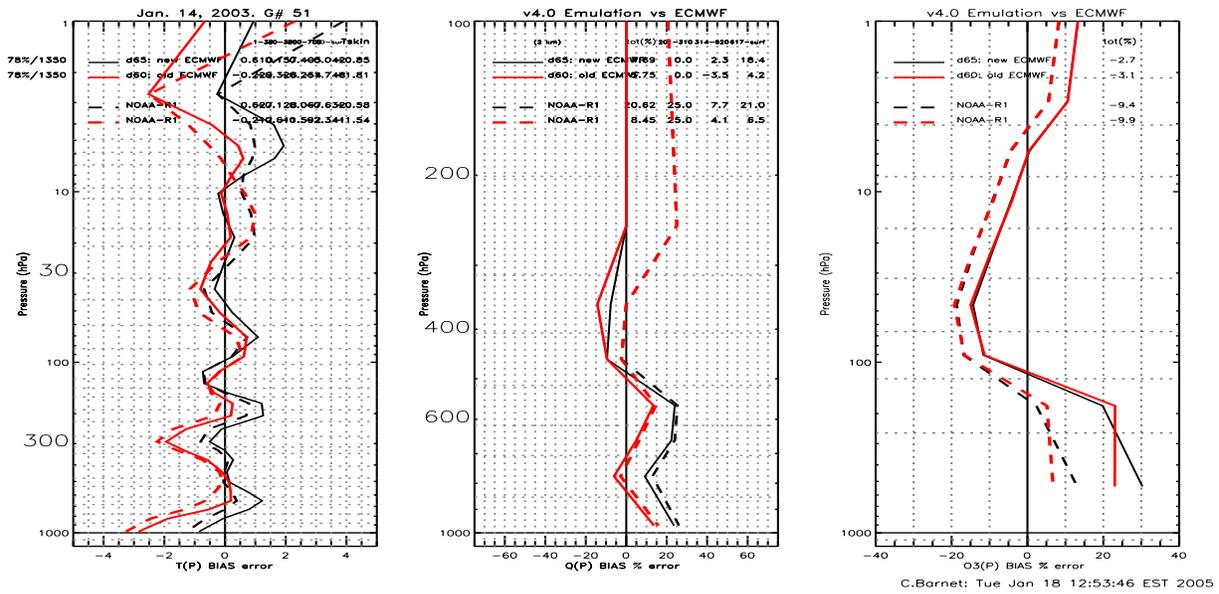


Figure 9.4: Example output from pl.sim, 'test', /bias, menu=1

9.4 stat1 & stat2 & stat3: IDL driver for simstat.F

One problem with pl_sim plotting of .out files is that a small change can seem significant because rejection of cases is different. In addition, it is sometimes useful to recompute statistics using a different rejection criteria. A valid comparison is one in which the same ensemble of cases is considered. Statistics can be recomputed using the simstat.F program using a common set of profiles determined by IPROLST.F or and/or new rejection thresholds. Also, the /bias, /sdv option in pl_sim plots is only an approximation when using .out files (a warning is printed). Simstat output can plot accurate /bias and /sdv. The 2 runs to be compared must have the same cnf file

```
preproc/stat1, runID#1, runID#2, cnfname, nowgt=nowgtx, $
[/ocean], [/night], [/day], [/reonly], [/pgev40], [/mit]
```

```
preproc/stat2, runID#1, runID#2, cnfname, nowgt=nowgtx, $
[/ocean], [/night], [/day], [/reonly], [/pgev40], [/mit]
```

or if 3 retrieval files are to be compared, then use

```
preproc/stat3, runID#1, runID#2, runID#3, cnfname, nowgt=nowgtx, $
[/ocean], [/night], [/day], [/reonly], [/pgev40], [/mit]
```

| stat2 and stat3 command line syntax | |
|-------------------------------------|---|
| required input | description |
| runID#1 | runID for the 1st retrieval |
| runID#2 | runID for the 2nd retrieval |
| runID#3 | runID for the 3rd retrieval |
| cnfname | rootname of the configuration (.cnf) file |
| option | description |
| /ocean | select only ocean cases |
| /land | select only land cases |
| /night | select only night cases |
| /day | select only day cases |
| /reonly | only build the RET file (default is RET & FG) |
| /mit | also build statistics for MIT file |
| /pgev40 | Use v4.0 QA methodology in JPL PGE |
| /nowgt | debugging option for simstat, skips IDPROLST step |

Since the format of runID and cnfname is used this means that this program is geared towards files preprocessed with preproc.pro. The you can use pl_sim.pro to plot the results. In your plotting configuration file you need a plot item like

The program control file, test.plt, contains the following items (after the section printed in pl_sim.pro:)

```
rem : =====
menu : d60 run - common rejection
rem : MUST first run stat2, 'd65bin', 'd60bin', 'test'
stat1: 10 d65bin_d60bin_test.RET.CM2
comm : new ECMWF
stat1: 12 d60bin_d65bin_test.RET.CM2
comm : old ECMWF
stat1: 0 d65bin_d60bin_test.FG.CM2
```

```
comm : new fg
stat1: 2 d60bin_d65bin_test.FG.CM2
comm : old fg
rem  : =====
rem  : MUST first run stat2,'d65bin','d65bin','test',/ocean
menu : d65 run - common rejection, ocean only
stat1: 10 d65bin_d60bin_test_0.RET.CM2
comm : new ECMWF
stat1: 12 d60bin_d65bin_test_0.RET.CM2
comm : old ECMWF
stat1: 0 d65bin_d60bin_test_0.FG.CM2
comm : new fg
stat1: 2 d60bin_d65bin_test_0.FG.CM2
comm : old fg
end  :
```

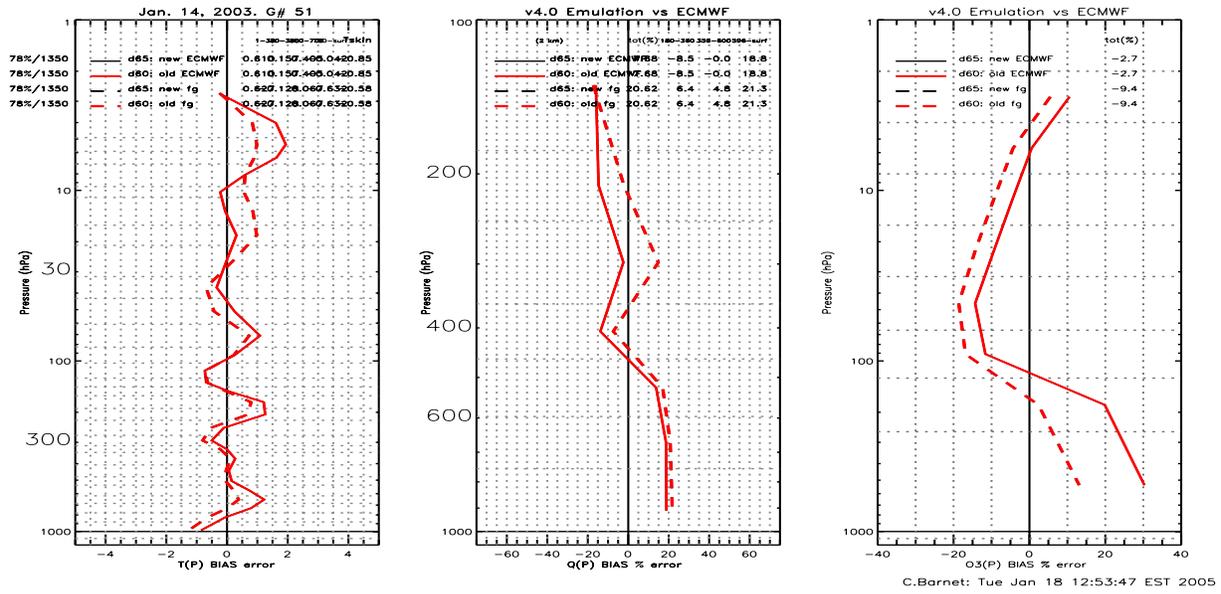


Figure 9.5: Example: stat2, 'run1', 'd65bin', 'test' followed by pl.sim, 'test', menu=2 (see test.cnf listing)

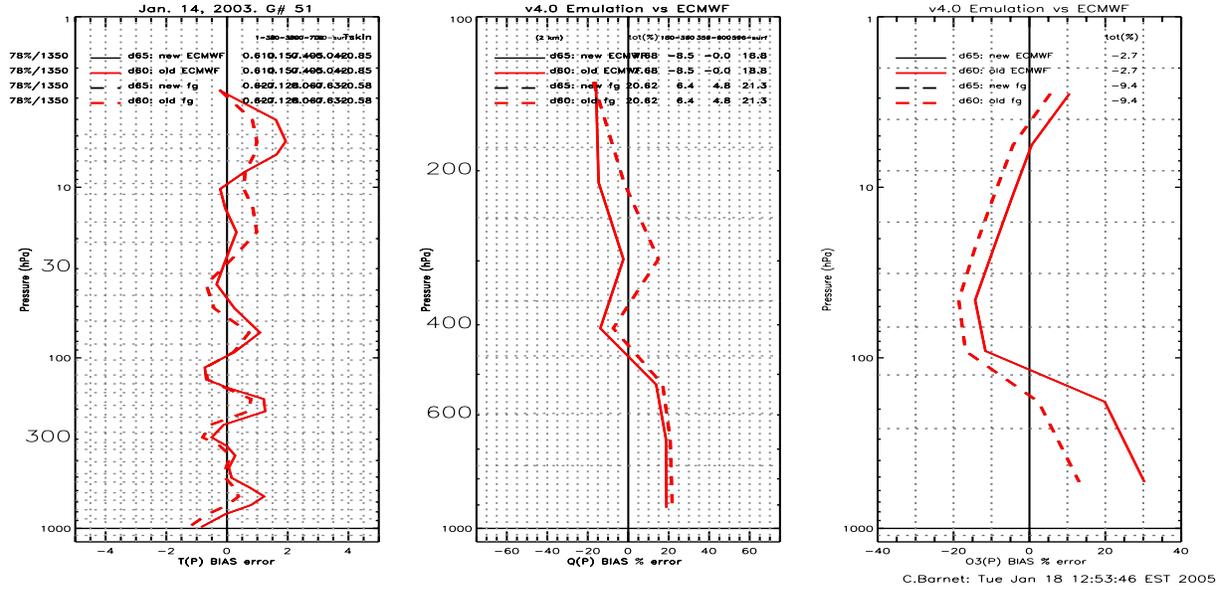


Figure 9.6: Example: stat2, 'run1', 'd65bin', 'test' followed by pl.sim, 'test', menu=2, /bias (see test.cnf listing)

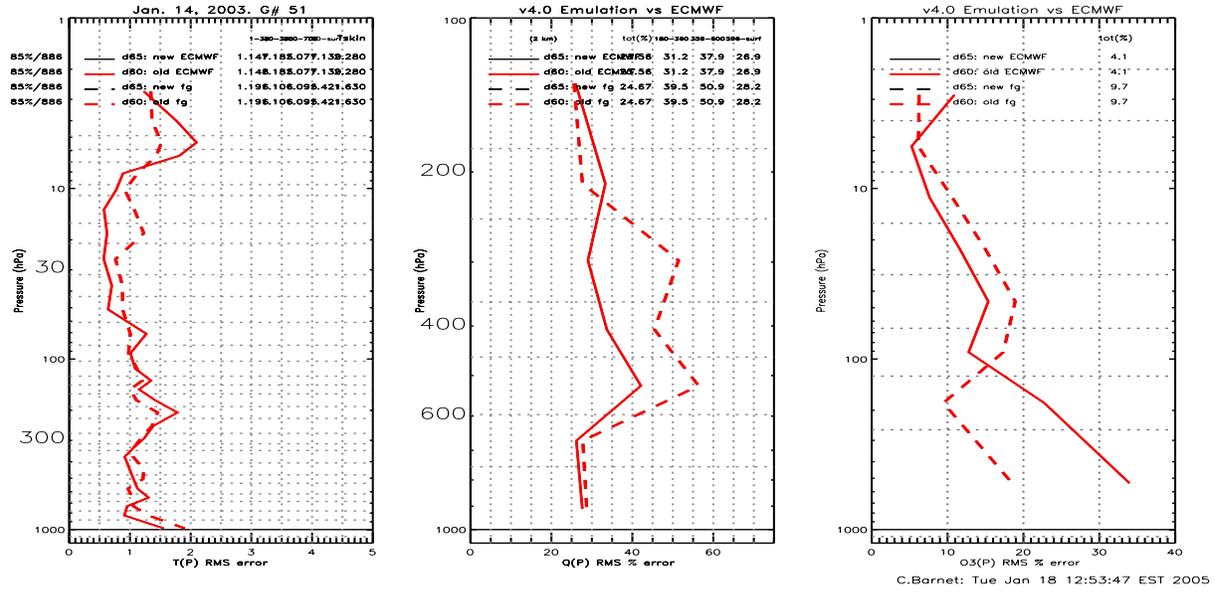


Figure 9.7: Example: stat2, 'run1', 'd65bin', 'test', /ocean followed by pl_sim, 'test', menu=3 (see test.cnf listing)

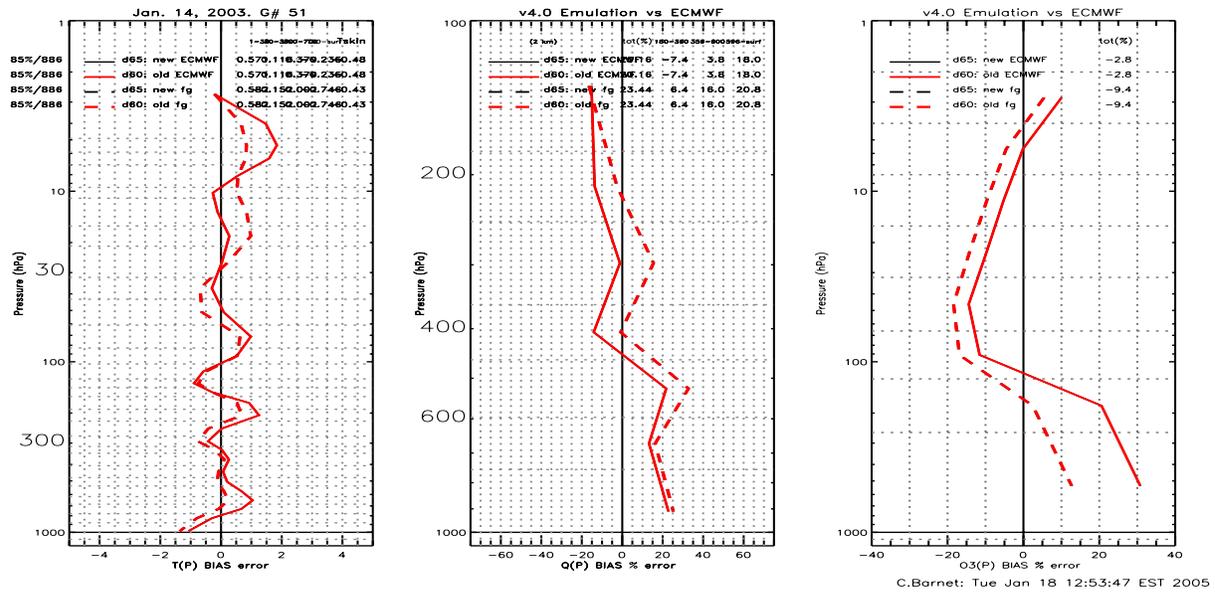


Figure 9.8: Example: stat2, 'run1', 'd65bin', 'test', /ocean followed by pl_sim, 'test', menu=3, /bias (see test.cnf listing)

9.5 pl_ftn92: plotting radiance statistics

Radiance residual statistics are written in the file graname.f92. pl_ftn92.pro is a driver for the program radrms.pro. radrms.pro reads and plots the contents of .f92 file(s). There are many options and formats for pl_ftn92.pro.

```
idl/airsb/pl_ftn92, rootname, [/bias], [/sdv], [type='eta'], $
[ymax = 5.0], [xmax = [600.0, 2700.0]], [bigtitle = ' ']
```

Table 9.1: pl_ftn92 command line systax

| required input | description |
|-------------------------|---|
| rootname | root name of the .f92 (same as .out) file |
| option | description |
| bigtitle = ' ' | makes a large title along the top |
| yymax=4.0 | sets maximum for all plots to 4.0 |
| yymax=[4.0,4.0,3.0,2.0] | sets maximum for individual plots |
| xymax=[800.0,900.0] | sets frequency axis |
| /bias | will plot biases instead of RMS, ymin=-ymax |
| /sdv | will plot standard deviation instead of RMS |
| type='eta' | default plot: rad's vs truth |
| type='obs-cal' | obs-calc's for all eta's |
| type='fl' | AIRS first light plots of obs-calc's (1 st eta only) |
| type='ccdif' | plot for Joel's clear flag |
| type='average' | |
| type='eta-bias' | |
| type='water' | plots detail water region |

Table 9.2: pl_ftn92: symbols used in plots

| | |
|----------------------------------|--|
| η_i | is the cloud cleared radiances at step = i |
| R(RET(2)) | are radiances computed from the first guess retrieval state (.fg file) |
| R(RET(2)) | are radiances computed from the final retrieval state (.ret file) |
| R(TRU) | are radiances computed from the truth state |
| $\delta R_{\text{PRED}}(\eta_i)$ | are predicted cloud cleared radiance errors at step = i |

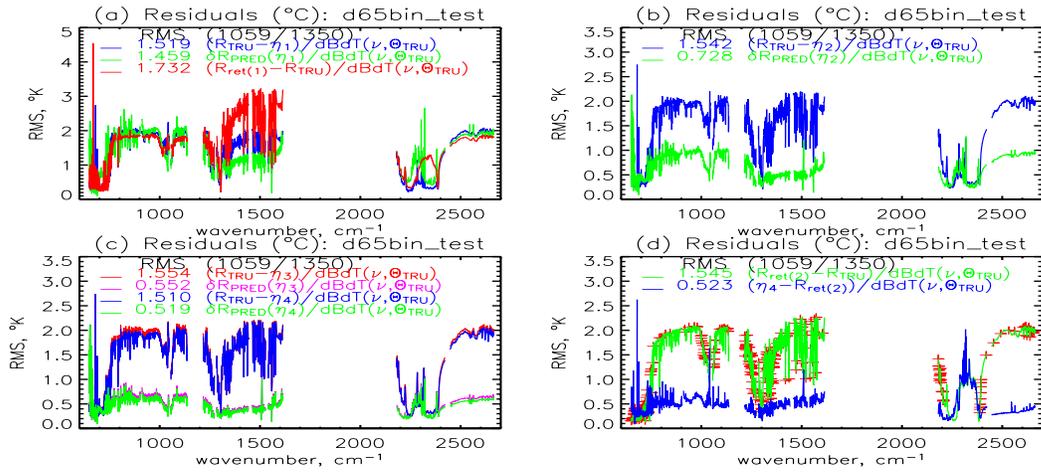


Figure 9.9: Example output from the pl.ftn92.pro program

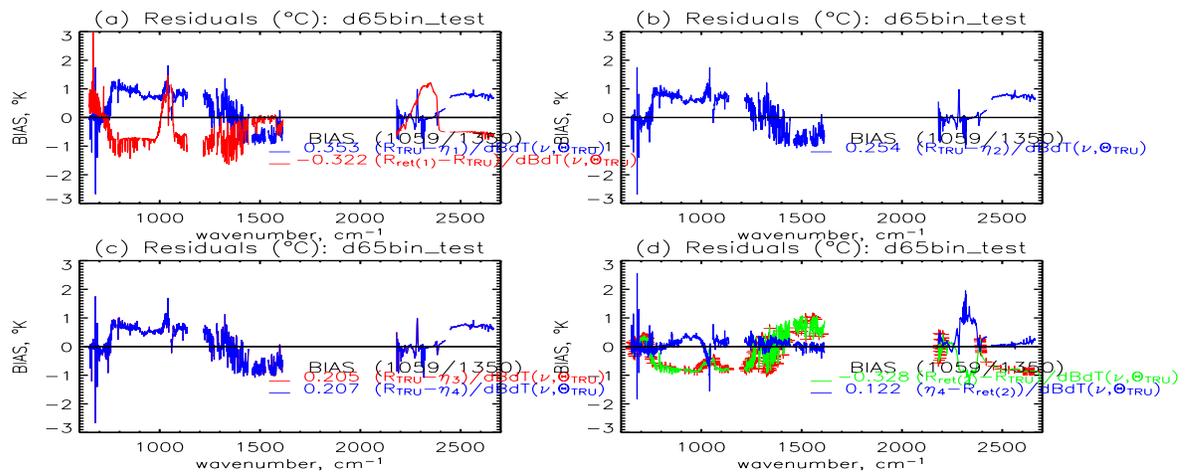


Figure 9.10: Example output from the pl.ftn92.pro program with the /bias option

9.6 pl_ret: plotting retrieval statistics at all steps

The program `pl_ret` plots the $T(p)$, $q(p)$, and $O_3(p)$ statistics (rms, bias, sdv) for all retrieval steps in a run. Uses the `.out` file. You must be in the retrieval directory to run this program.

```
pl_ret,'rootname', [xmin=[0.0,0.0,0.0]], [xmax=[8.0,75,50]], $
[menu=1], [/bias], [/sdv], $
[path=GETENV('AIRS_ret') + '/2003/01/14/ret/']
```

| pl_ret command line systax | |
|----------------------------|---|
| required input | description |
| rootname | root name of .out file |
| option | description |
| path | Path to the retrieval directory |
| xmin | sets minimum for $T(p)$, $q(p)$, and $O_3(p)$ plots |
| xmax | sets maximum for $T(p)$, $q(p)$, and $O_3(p)$ plots |
| menu | select $T(p)$ (menu=1), $q(p)$ (menu=2), or $O_3(p)$ (menu=3) |
| /bias | Switch to plot BIAS instead of RMS |
| /sdv | Switch to plot standard deviation instead of RMS |

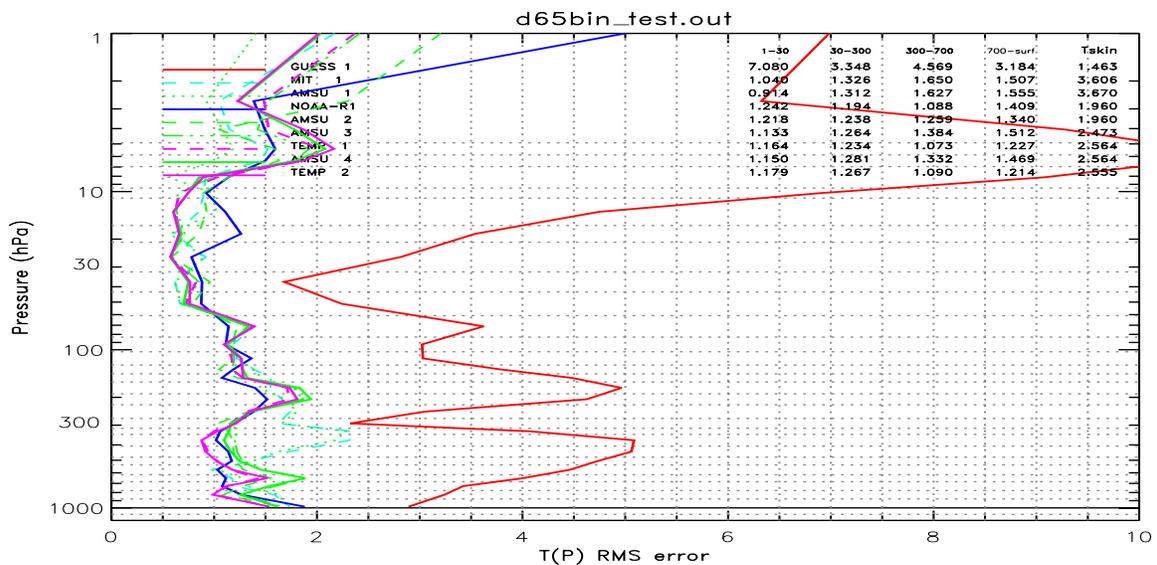


Figure 9.11: Example $T(p)$ statistic output from the `pl_ret`, 'd65bin_test', `xmax=[10,200,200]`, `menu=1`

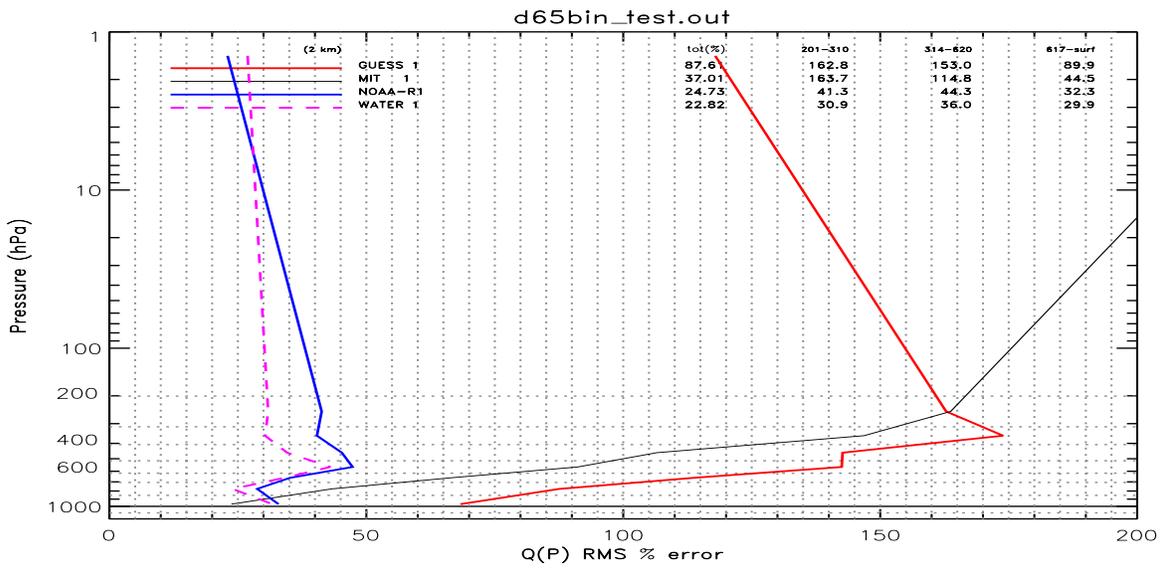


Figure 9.12: Example $q(p)$ statistic output from the pl_ret, 'd65bin_test', xmax=[10,200,200], menu=2

9.7 pl_ampl: plotting amplification factor

This program plots a histogram of the cloud clearing amplification factor for both accepted and rejected cases. You must be in the retrieval directory to use this program. The .out and .tbl files are used.

The IDL syntax is

```
pl_ampl, rootname, [desc='Plot Title'], [num=1350], [ymax=100], $
[path=GETENV('AIRS_ret') + '/2003/01/14/ret/']
```

| pl_ampl command line systax | |
|-----------------------------|--|
| required input | description |
| rootname | root name of .out file |
| option | |
| option | description |
| rootname | root name of .out, .tbl file |
| desc | Optional Title for Plot |
| path | Path to the retrieval directory |
| num | Optional parameter for number of cases |
| ymax | Optional range for y-axis (useful f/ multiple plots) |

This program is stupid in that it doesn't know how many profiles you ran (yes, I can fix this). Therefore, it currently assumes 1350 (a granule) and you must tell it if it is not 1350.

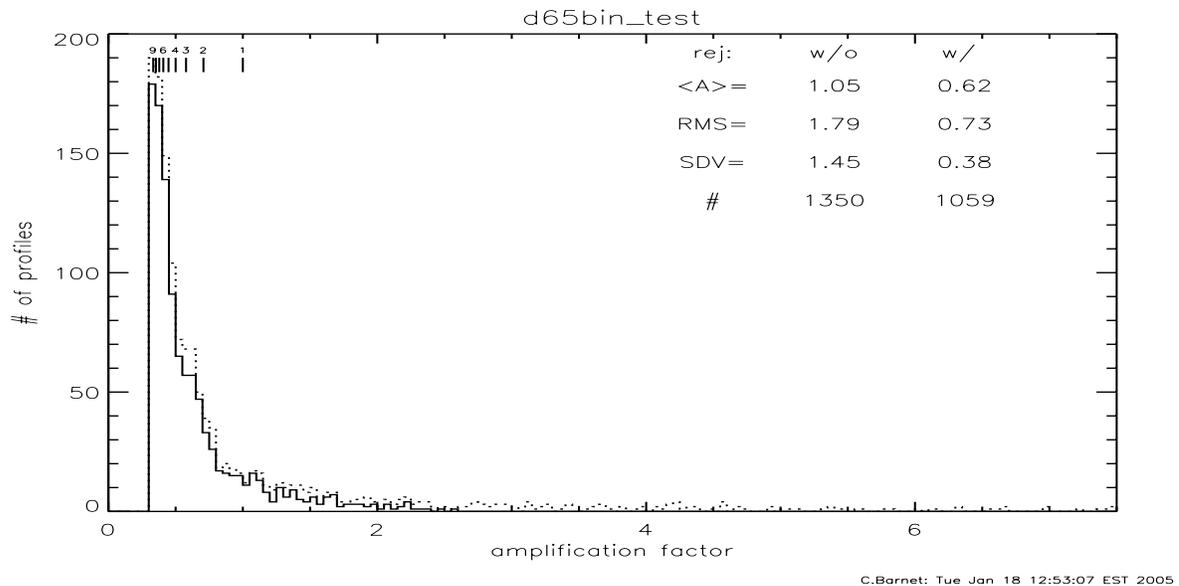


Figure 9.13: Example output from the pl_ampl.pro program

If the amplification factor is $\frac{1}{3}$ then the cloud clearing effectively average all 9 FOV's. In the upper left corner of the plot there is a set of lines indicating the effective number of FOV's that were averaged. In the figure, approximately 180 cases had all 9 FOV's determined clear and approximately 140 cases had 4 FOV's determined clear. The number of cases with only 1 FOV determined clear is about 15.

9.8 pl_lev2 & pl_l2d: utility to plot individual retrieval profiles

The program `pl_lev2.pro` plots $T(p)$, $q(p)$, and $O_3(p)$ profiles and requires a driver program to select the file(s) to be plotted. The program has been replaced, for the most part, by `pl_l2d`, however there are still applications in which the relative simplicity of use is warranted. The program opens and reads L2 files and prints the selected profiles. Sequential reads are allowed and there are many options for the plotting. Example driver programs are:

| | |
|-------------------------------|--|
| <code>idl/pl_truth.pro</code> | Various TRUTH datasets from days of old |
| <code>idl/pl_jan01.pro</code> | The January 2001 JPL exercise, G401 files |
| <code>idl/pl_nov01.pro</code> | The November 2001 JPL exercise, G401 files |
| <code>idl/pl_jun02.pro</code> | June 2002 first light AIRS data |

A more robust version is `pl_l2d.pro` which passes structures for 3 or 4 profiles (read via `openl2_b.pro`) into the subroutine. This program was written later so there are options for plotting differences and more annotation. This program is called by `pl_retccr`. See the profile plots in Fig. 9.12 for an example. This is the program I would recommend using. This means it is up to you to read in the structures, but that means you can plot any of the files within our system.

```
airsb/pl_l2d, l2d_tru, l2d_mit, l2d_fg, l2d_ret, $
Qtype, Qscl, Otype, [/debug], [title=' '], desc=descx, qmax=qmaxx, $
line=linex, color=colorx, qlin=qlinx, csiz=csizx, diff=diffx
```

The `l2d_mit` structure must exist, but it will not be plotted unless there are 4 elements in the optional `desc` parameter. Yes, this is awkward, but it has to do with a kludge I did when the retrieval does not output the MIT step. Sorry.

| pl_l2d command line syntax | |
|----------------------------|---|
| required input | description |
| l2d_tru | The first profile to plot, reference for /diff |
| l2d_mit | The second profile to plot (must use desc=[] option) |
| l2d_fg | The second or third profile to plot |
| l2d_ret | The third or fourth profile to plot |
| Qtype | 0 = Volumetric mixing ratio 1 = Column Density 2 = mass mixing ratio (g/Kg) (default) 5 = relative humidity |
| Qscl | number of decades to plot w.r.t. maximum |
| Otype | Ozone plot type 0 = Volumetric mixing ratio (ppm) (default) 1 = Column Density, (DU/layer) 2 = mass mixing ratio, (g/kg) 3 = Number Density, (molecules/cm ³) 4 = partial pressure, (mBar) |
| option | description |
| diff N | Plot differences w.r.t. N'th l2d structure |
| Qmax | force maximum value of x-axis for water plot |
| Qlin | makes the water plot linear, instead of logarithmic |
| desc = [] | description for the 4 profiles, must have 3 or 4 elements |
| title = [] | This is the title for the T,q,and O ₃ plots it can have 1, 2, or 3 elements |
| color = [] | set color for each profile, must have 3 or 4 elements |
| line = [] | line type for the 4 l2d structures |

9.9 profsumm: plotting individual retrieval summaries

The retrieval program stores a summary of each retrieval in the .bin file. The program profsumm plots the contents of this binary profile as scatter diagrams for a number of pre-defined quantities.

```
profsumm, rootname, [threshlim=3.0], [path=' '], [/rej], $
[pick=5.1], [/land], [/badlist], [/co2], [/nocldfrc], $
[xmax=], [ymax=], [/publ], [title=' ', [csiz=1.2]
```

There are 12 pages of 9 plots/page. These are organized into 3 blocks of 36 items. The 36 items are

- retrieved items (e.g., cloud fraction),
- rejection criteria (e.g., etarej, qualtemp)
- clear criteria (e.g., amplification criteria)

The 3 blocks (36 plots per block) are

1. The 36 items versus TRUE CLOUD FRACTION
2. The 36 items versus RMS of bottom 2 km's of $T(\text{ret})-T(\text{tru})$, T_{b2}
3. The 36 items versus $T_s(\text{ret})-T_s(\text{tru})$

The default color scheme is white are accepted profiles, blue are rejected profiles, and red are profiles that are accepted but have large T_{skin} and $T(p)$ errors. The current rejection threshold, if appropriate, is plotted as a horizontal yellow line.

| profsumm command line systax | |
|------------------------------|--|
| required input | description |
| rootname | root name of .out file |
| option | description |
| /land | red=land, blue=ocean, white=coast, lakes, rivers |
| /rej | does not plot rejected cases |
| /nocldfrc | will assume NO truth cloud fraction – uses ret cldfrc |
| /publ | will increase charsiz of labels, etc. |
| /badlst | print a table of the bad cases |
| path | set the retrieval path |
| co2 | will plot stuff vs CO2-CO2_true (i.e., simulation) |
| csiz | will scale the character size |
| threshlim | set plot maximum = rejthreshold*threshlim Plots that do not have rejthesh() are not affected. |
| title | adds a title to the center top plot |
| External control parameters | |
| pick | pick item and plot (see table below) |

| Pick option definition <i>e.g.</i> , pick=11.1 is A(eta.2) vs cloud fraction | |
|---|--|
| X-axis definition | |
| .1 | versus cloud fraction |
| .2 | versus error in bottom two km's of T(p) |
| .3 | versus error in Tskin |
| .4 | versus wind |
| Y-axis definition | |
| 1 | cloud fraction |
| 2 | cloud fraction for Pj500 |
| 3 | RMS(AMSU-BT(FINAL)) |
| 4 | RETTMP residual, QUALTEMP |
| 5 | RETSURF residual, QUALSURF |
| 6 | RETWAT residual, QUALWATR |
| 7 | CLOUDHGT residual, QUALCLOUD |
| 8 | BIAS(T(AMSU)-T(AIRS)) in bottom 2-km's |
| 9 | RMS(T(AMSU)-T(AIRS)) in bottom 2-km's |
| 10 | A(eta.1) |
| 11 | noaa score |
| 12 | A(eta.2) |
| 13 | ETAREJ(eta.1) |
| 14 | ETAREJ(rej) or ETAREF(FINAL) if ieta_rej = 1 |
| 15 | Aeff(eta.1) |
| 16 | Aeff(final) |
| 17 | SDV(AVHRR.4) |
| 18 | Ts(NOAA)-Ts(AMSU) |
| 19 | BIAS(AMSU-BT(NOAA)) |
| 20 | RMS(AMSU-BT(NOAA)) |
| 21 | lambda of observed radiances |
| 22 | lambda of observed-computed radiances |
| 23 | BIAS(CCR-jR _i ,800-900 cm-1) |
| 24 | SDV(CCR-jR _i ,800-900 cm-1) |
| 25 | LIQ(truth) if(/nocldfr) then BIAS[R(z)-R(z1)], amount of multiple cloud |
| 26 | BIAS[R(z1)-jR _i], effect of 1-layer cloud' |
| 27 | LIQ(MIT)-LIQ(true), LIQ(true)=0 if(/nocldfr) |
| 28 | BIAS(T(NOAA)-T(MIT)) in bottom 2-km's |
| 29 | RMS(T(NOAA)-T(MIT)) in bottom 2-km's |
| 30 | %land error (TRU-RET) |
| 31 | BLMULT |
| 32 | numlevret-90 |
| 33 | Lapse rate test |
| 34 | Ts(FIN)-Ts(NOAA) |
| 35 | Tb2 |
| 36 | Delta Ts |

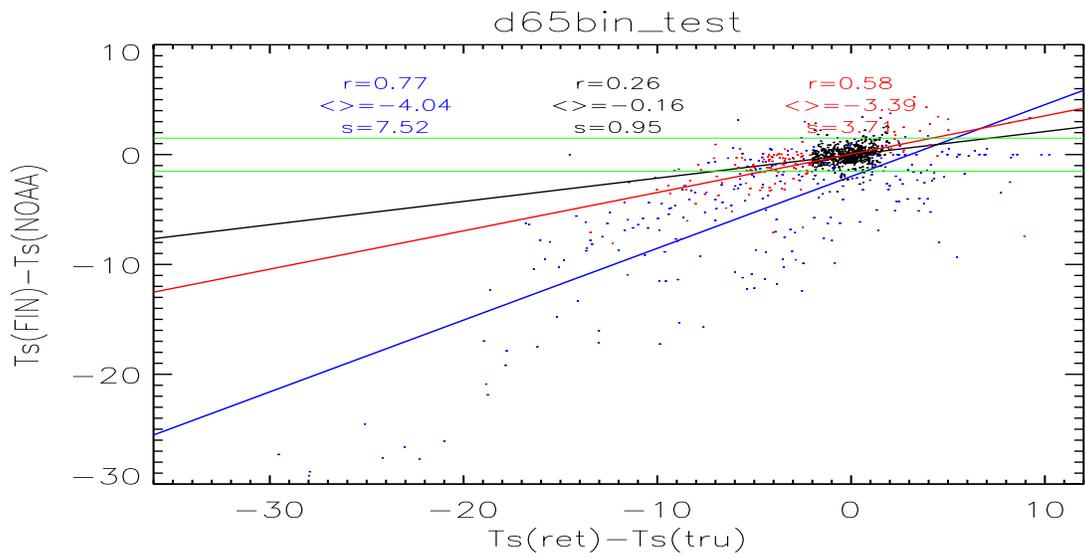


Figure 9.14: Example output from the profsumm, 'd65bin_test', pick=34.3

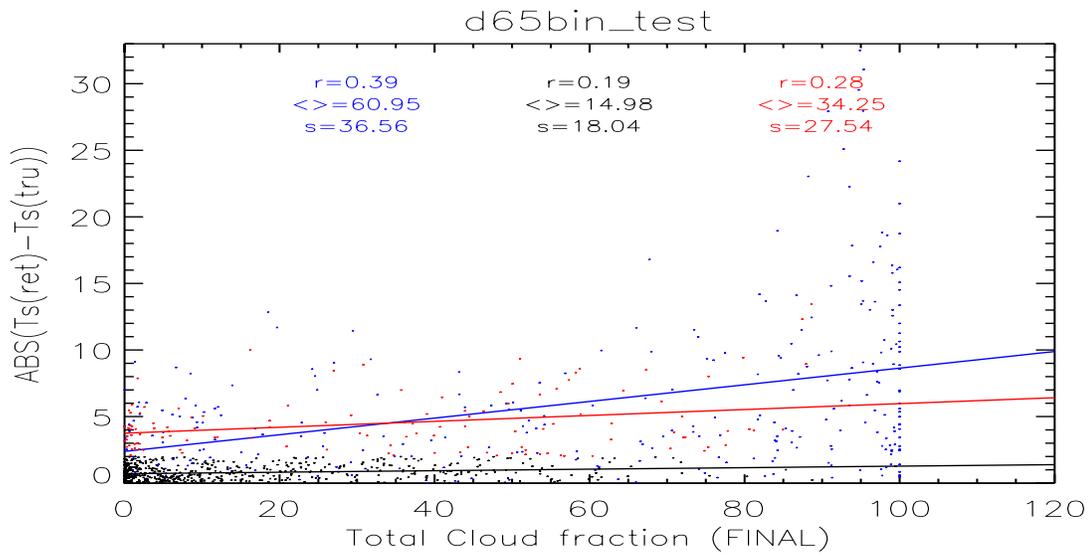


Figure 9.15: Example output from the profsumm, 'd65bin_test', /nooldfrc, pick=36.1

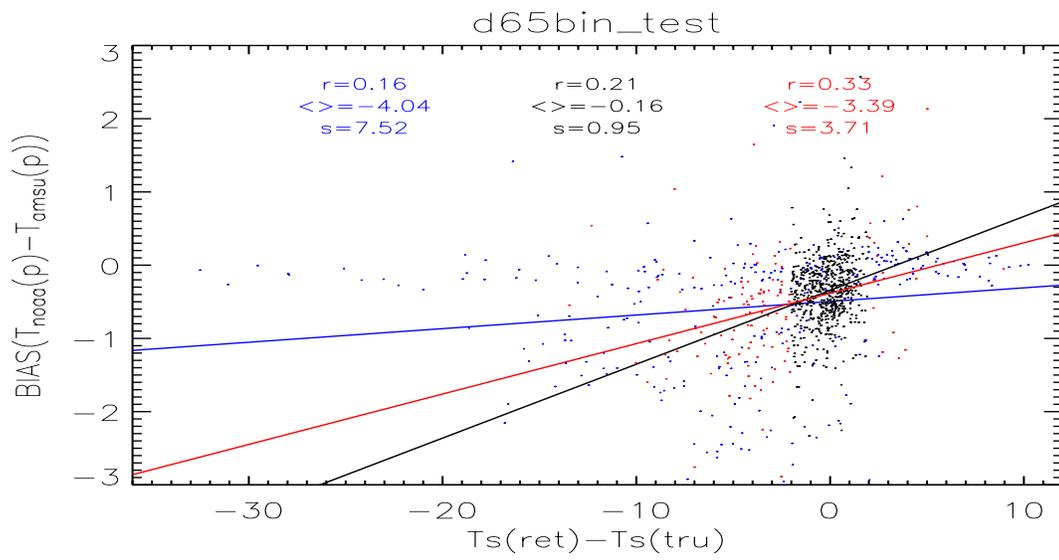


Figure 9.16: Example output from the profsumm, 'd65bin_test', pick=28.3

9.10 pl_eigen: plot eigenvector plots for selected profiles

AIRSB makes a file (*.f91) which has eigenvalue and eigenvector information for the profiles selected for detailed printout (IDPROTEST list). These can be plotted with

```
pl_eigen,'rootname', [/eps], [/ps], [pick='xx'], [prefix=], $
[path=' '], [desc=' '], $
```

| pl_eigen command line systax | |
|------------------------------|-------------------------------------|
| required input | description |
| rootname | root name of .out file |
| option | description |
| /eps | makes encapsulated postscript files |
| /ps | makes printable postscript files |
| path | set retrieval path |
| desc | modify the title of the plots |
| prefix | A prefix for the file names |

Since the number of levels are a namelist option for every retrieval type, a good experiment to try is to set the number of functions to a large value to see the output of the information content analysis. For good retrieval performance the empirical damping parameter, called B_{max} in the theoretical notes and called XXXwgt in the namelists, where XXX is the retrieval type (TMP, WAT, OZO, CO, CH4, AMSU, etc.). Here is the 34 function set I typically use:

```
NLEVXXX = 35,
LEVXXX = 1, 14, 19, 24, 29, 34, 39, 44, 47, 49,
         51, 53, 55, 57, 59, 61, 63, 65, 67, 69,
         71, 73, 75, 77, 79, 81, 83, 85, 87, 89,
         91, 93, 95, 97, 100
```

The file names for /ps and /eps will have the 1st 3 characters of the runID, the retrieval type and number, the profile ID and end with '_eig.eps'. Since there are many steps (amsu1, tmp1, tmp2, wat1, ozo1, CO1, CH41, CO21) and there can be many profiles selected by IDPROTEST, the number of output files can be VERY large. For example, a file name for the 1st profile of the d65bin run for the final temperature retrieval will be "d65_tmp2_1_eig.eps". This file is shown in the following figure.

In the "AMSU" retrieval 12 AMSU channels and 17 AIRS stratospheric channels are used to solve for 15 functions. In this system there are more functions than vertical pieces of information and in Fig. 9.10 we can see the 11 significant ($\lambda \geq 5\%$) functions that were determined by the information content.

In the "TEMP" retrieval about 45 AIRS channels are used to solve for 23 functions. The AIRS has more information content; however, in v4.0 many of the useful channels have been removed. Therefore, we still have about 11 significant functions. The new functions for Two profiles are shown in Fig. 9.10 and Fig. 9.10.

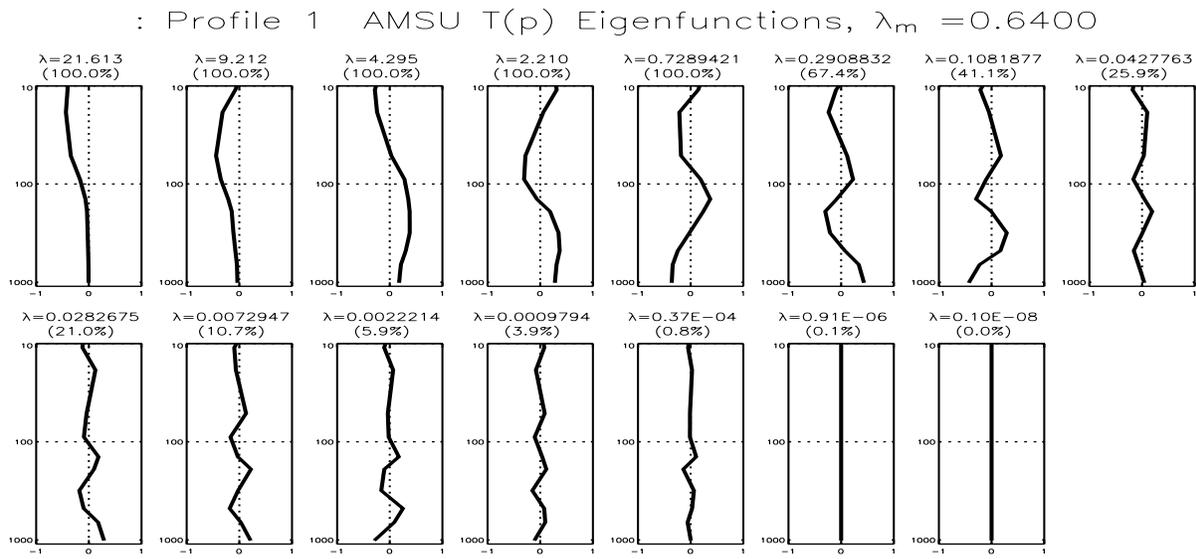


Figure 9.17: Example output from the pl_eigen, 'd65bin_test',/eps, The file d65_amsu1.1.eig.eps is shown

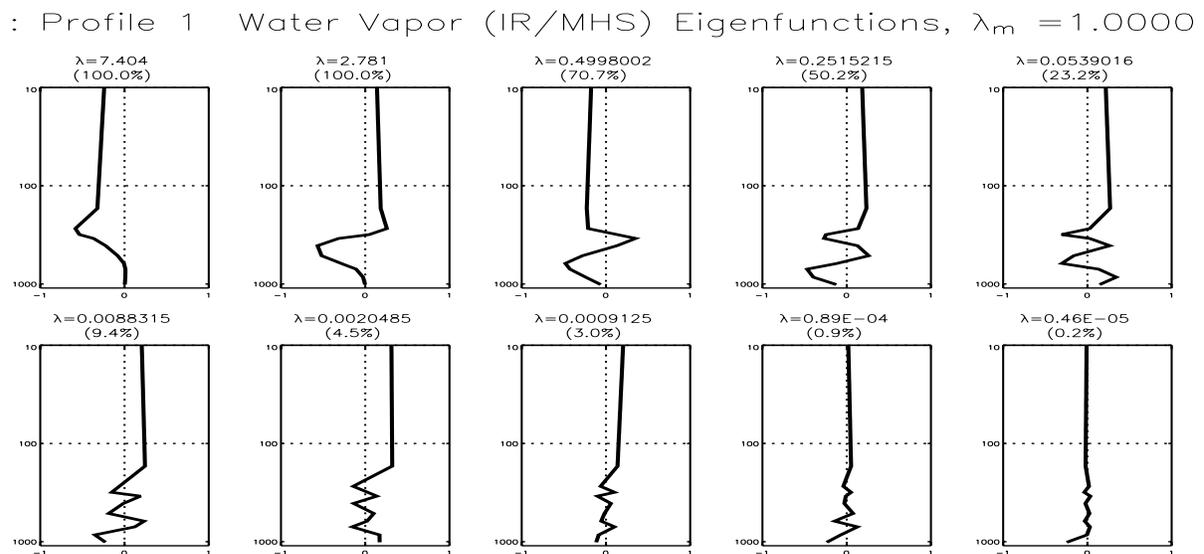


Figure 9.18: Example output from the pl_eigen, 'd65bin_test',/eps, The file d65_wat1.1.eig.eps is shown

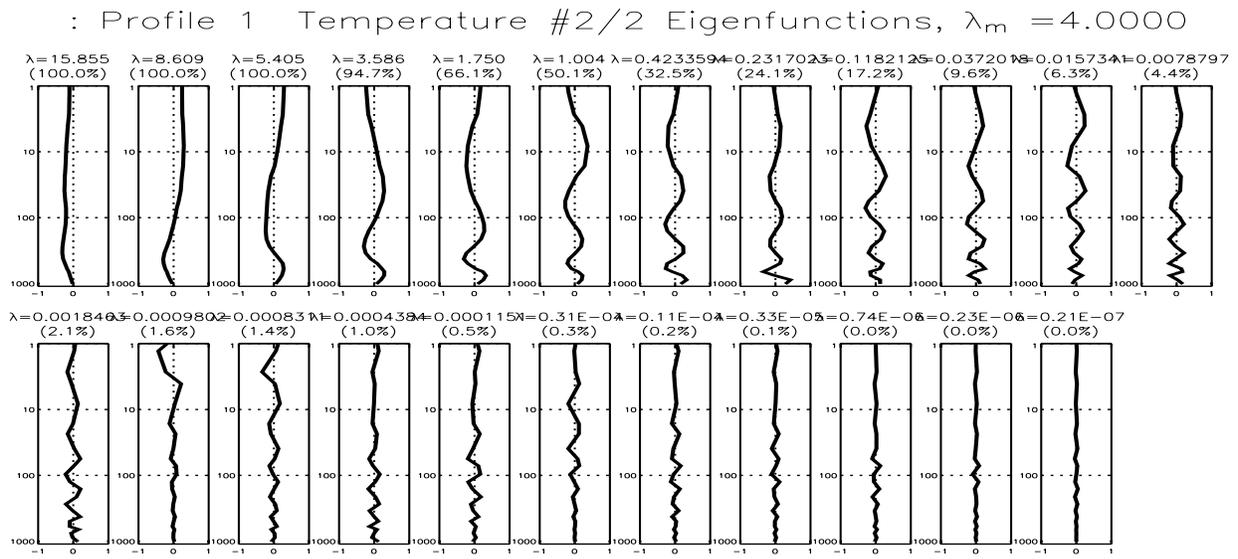


Figure 9.19: Example output from the pl_eigen, 'd65bin.test', /eps. The file d65_tmp2.1_eig.eps is shown

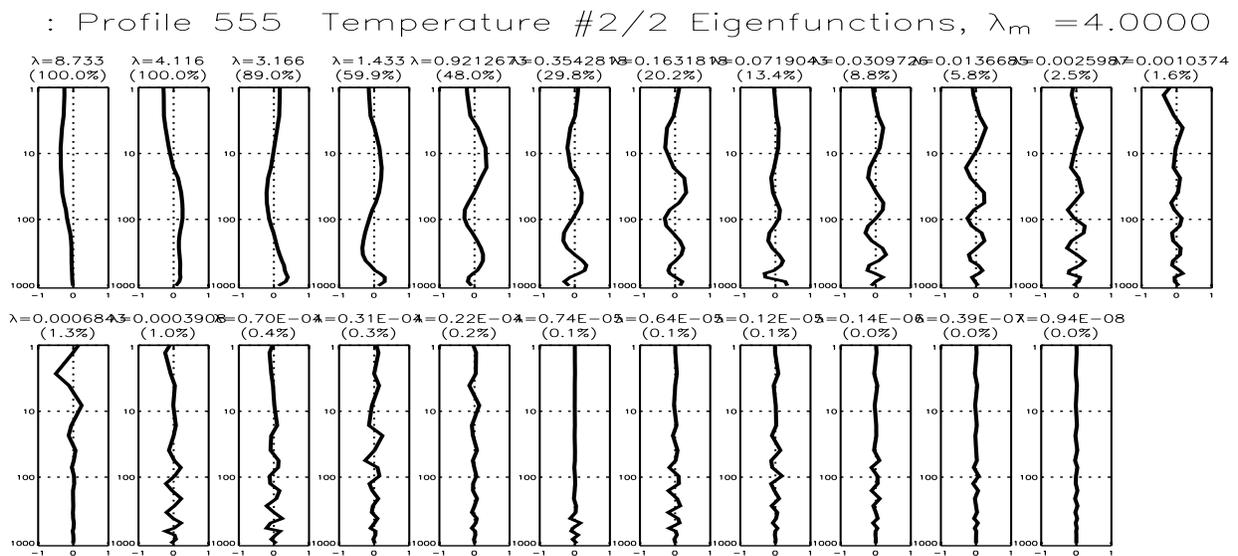


Figure 9.20: Example output from the pl_eigen, 'd65bin.test', /eps. The file d65_tmp2.555_eig.eps is shown

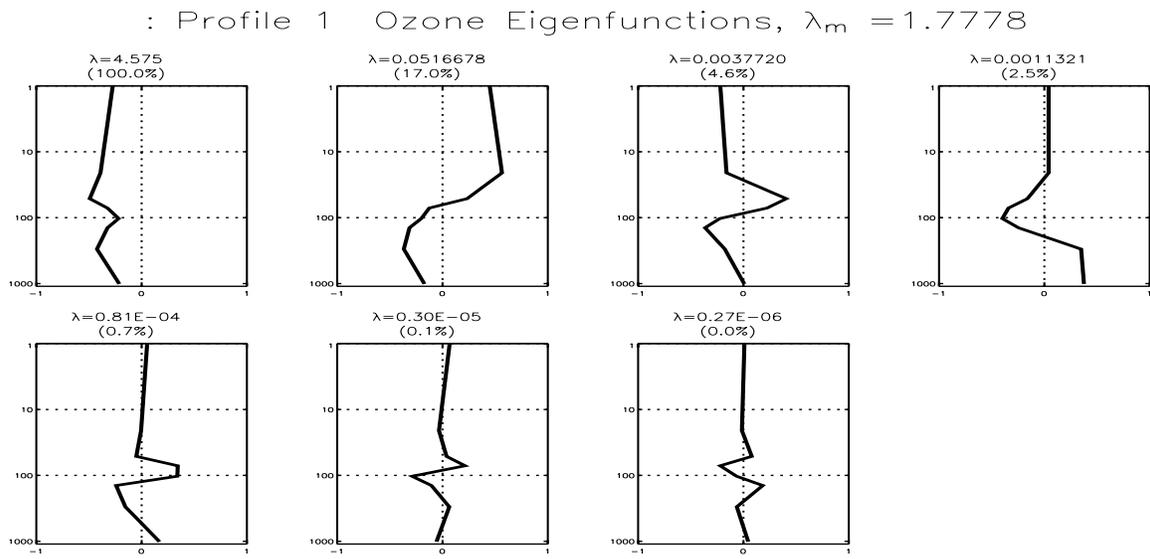


Figure 9.21: Example output from the `pl_eigen`, 'd65bin_test', /eps. The file `d65_ozo1_1.eig.eps` is shown

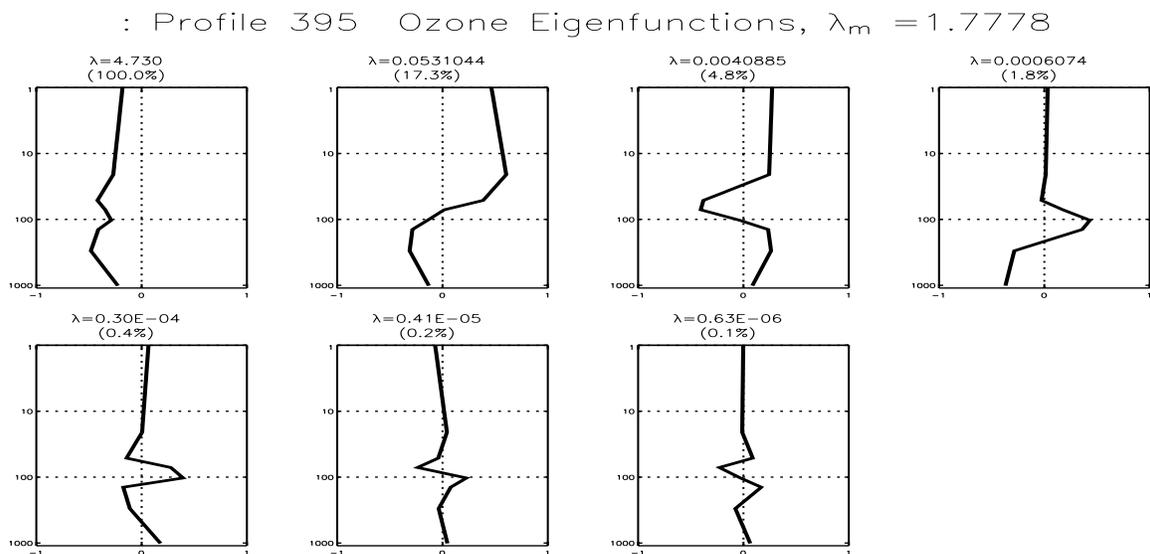


Figure 9.22: Example output from the `pl_eigen`, 'd65bin_test', /eps. The file `d65_ozo1_395.eig.eps` is shown

: Profile 1 Carbon Monoxide (CO) #1/2 Eigenfunctions, $\lambda_m = 0.4444$

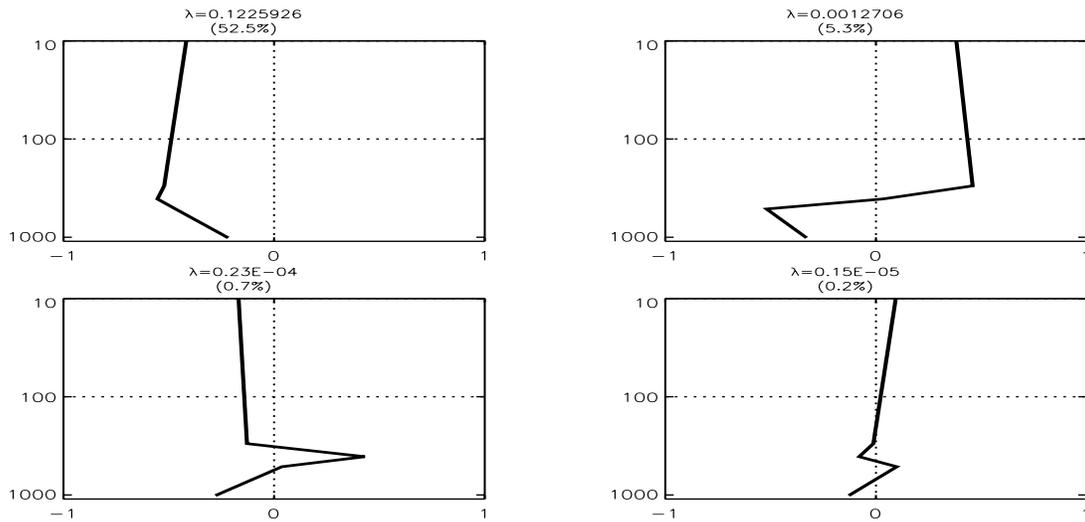


Figure 9.23: Example output from the pl_eigen, 'd65bin_test', /eps. The file d65_CO1.1_eig.eps is shown

: Profile 1 Methane (CH4) #1/2 Eigenfunctions, $\lambda_m = 25.0000$

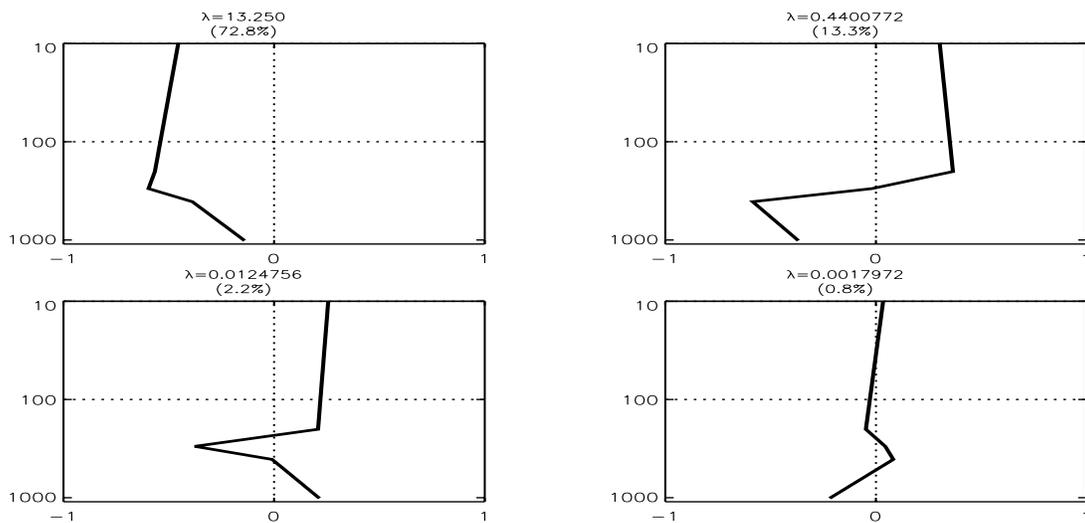


Figure 9.24: Example output from the pl_eigen, 'd65bin_test', /eps. The file d65_CH41.1_eig.eps is shown

9.11 pl_smat: plot the S-matrix for selected profiles

AIRSB prints the S-matrix in the output file (*.txt). This table can be plotted for the channels used in the retrieval using pl_smat. The S-matrix table must exist in the retrieval file for this program to work which means that

- The iprt() level must be set ≥ 3 for the desired retrieval in pro_airs.nl.
- The profiles are selected with the NPROTEST & IDPROTEST() variables in io_airs.nl
- I also strongly suggest using 34 functions when making these plots with the optional namelist parameter for functions used in the retrieval you want to plot (XXX=CO2,CO,CH4,TEMP,WATR, etc).

```
NLEVXXX = 35,
LEVXXX = 1,14,19,24,29,34,39,44,47,49,
          51,53,55,57,59,61,63,65,67,69,
          71,73,75,77,79,81,83,85,87,89,
          91,93,95,97,100
```

```
trace/pl_smat, rootname, [path=' '], [/eps], [flist=[]], [smax=0.0], $
[/co], [/ch4], [/o3], [/temp], [/watr], $
[/sw], [/lw], [/publ]
```

The default is to plot ALL the CO₂(p) channels. Other retrieval types are supported with optional switches (see below). Typically there are too many channels to plot. The program will break the channels in groups of 28 channels (still too many). The optional parameter freq=[] allows select channels for plotting. Therefore, this is an interactive program that is usually used to make plots for presentations.

Examples of figures I have made are shown below. The programs used to generate these figures are in the top portion of the file pl_smat.pro in the idl/trace directory (e.g., pro ch4_fig, pro co2_fig, pro co_fig, pro o3_fig, pro temp_fig, pro watr_fig). The retrieval test files are also hardwired and located on orbit009l. You can copy these driver programs (only a few lines long) to create your own customized plots.

| pl_smat command line syntax | |
|-----------------------------|---|
| required input | description |
| rootname | root name of .out file |
| option | description |
| flist | list of frequencies to plot |
| /eps | makes encapsulated postscript files |
| path | set retrieval path |
| /co | plot the CO(p) channels |
| /ch4 | plot the CH ₄ (p) channels |
| /temp | plot the T(p) channels |
| /watr | plot the q(p) channels |
| /o3 | plot the O ₃ (p) channels |
| /sw | plot the SW only |
| /lw | plot the LW only |
| /publ | removes rootname from the title of the plot |
| pmin | minimum pressure (maximum y-axis) for plot |

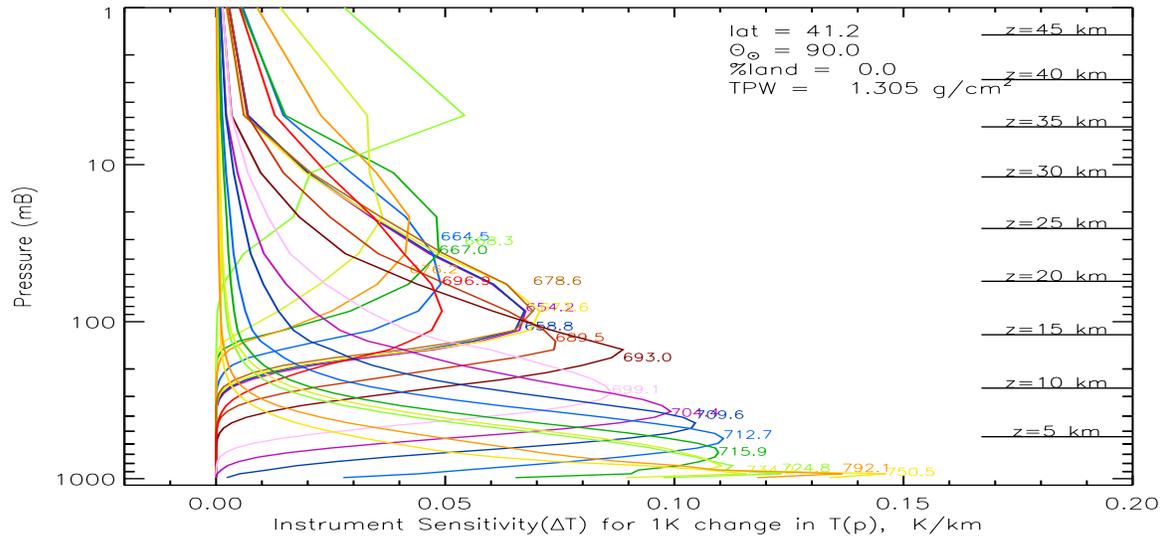


Figure 9.25: Example output from the pl_smat, called via temp_fig with the /LW option

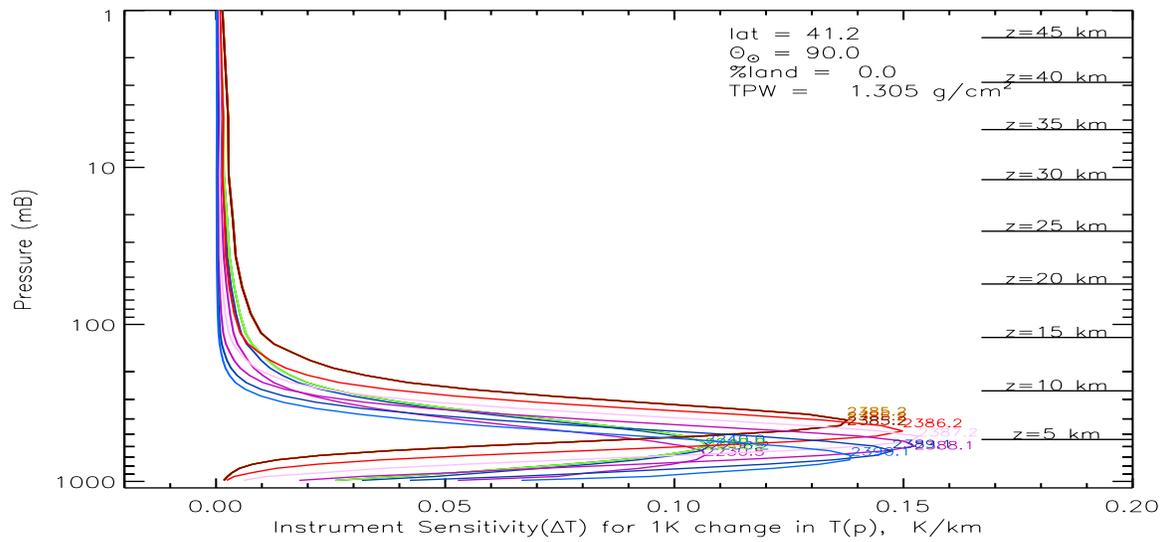


Figure 9.26: Example output from the pl_smat, called via temp_fig with the /SW option

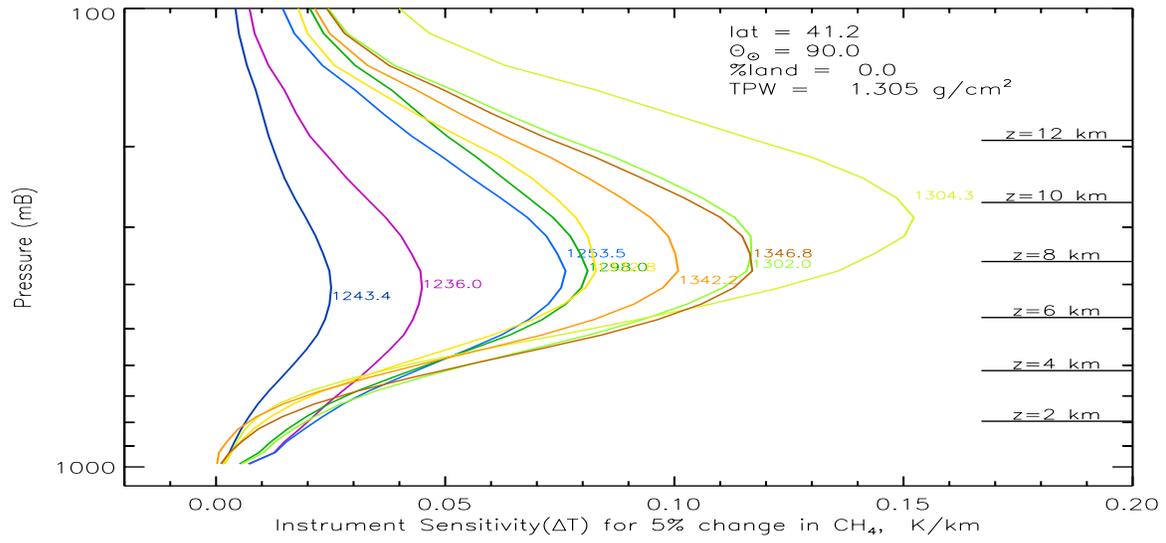


Figure 9.27: Example output from the pl_smat, called via ch4_fig

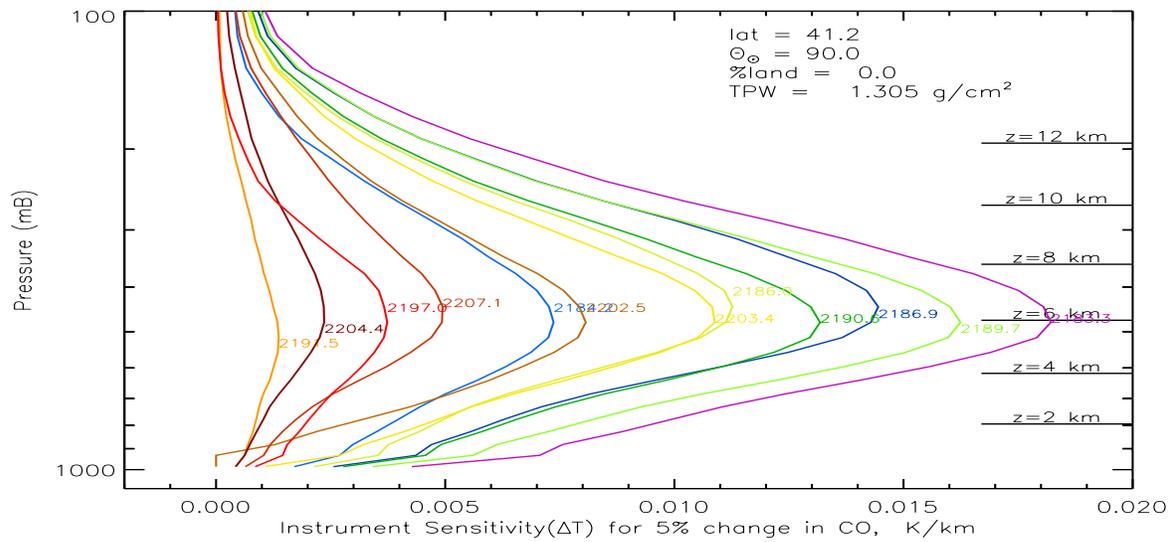


Figure 9.28: Example output from the pl_smat, called via co_fig

9.12 pl_retccr: plotting individual retrieval diagnostics

This program is useful for detailed analysis of retrieval performance. I expect that only 1 granule will be analysed at a time; however, this granule can be composed of merged scan lines (*e.g.*, G401, G422 files), match files, etc. It plots single retrievals and there is a plethora of information shown on the screen. There is a crude menu to aid in analysis of the retrievals and there is a lot of annotation on the plots. It requires proper configuration of the namelists when running the retrieval.

```
pro_airs.nl
```

```
-----
```

```
writeradused = T
stepname = $\dots$, 'SET MIT', $\dots$, 'SET FG', $\dots$
```

```
io_airs.nl:
```

```
-----
```

```
L2MITFILE_OUT = 'mit.asc'

Nchan_ccr = -1
Nformat_ccr=10
Iformat_ccr=0,3,1,11,2,19,29,7,8,40
l1file_ccr = 'ccr.bin'
```

The program is invoked and the entire ccr file must be read in (this is a very large file and takes a while to read). If you do not need to see the granule spatial map you can use the /nomap option to speed up entry to the program. The callign sequence and options are:

```
plot_pro/pl_retccr, runID, cnfname, granule, [retpath=retpathx], $
  [/avn, [fg='NOAA'], [freq=2616.0], $
  [/nomap], , [/sim], [/match], [/subset]
```

| pl_retccr command line systax | |
|-------------------------------|--|
| required input | description |
| runID | Run ID |
| cnfname | root name of the configuration file |
| glist | LONARR() list of granules to plot |
| retpath = | specify the path to the retrievals |
| option | description |
| freq | changes the channel displayed in the spatial map |
| /sim | tells the code there are 9 FOV's/FOR in the truth file also changes label on TRUTH plots |
| /match | tells the code that these are not SCANSET's (30 FOR's/SCAN) |
| /avn | changes the truth label to 'AVN', default = 'ECM' |
| /fg | changes the label on the "fg" file, default = 'NOAA' |
| /nomap | does not display the map |
| /noccr | DOES NOT read the CCR file or plot radiances or map |
| /subset | tells the code that the full granule was not run, in this case io_[inst].nl/IDPRO was used to select FOR's and l2tru_out = 'tru.bin' |

In the lower right the entire granule is shown and the location of the current field of regard is shown. This plot is observed radiance in 2616, although any frequency can be selected with the optional parameter freq.

In the upper left the output of pl_lev2.pro is shown. Profiles of $T(p)$, $q(p)$, and $O_3(p)$ are shown for ECMWF (black), the MIT microwave solution (yellow), the NOAA regression state (green), and the the NOAA physical algorithm (red). There are also column values of moisture, liquid water (profile is also shown, if there is liquid water determined from the microwave, ozone, carbon monoxide, methane, and carbon dioxide. View geometry (latitude, longitude, view angle, solar zenith angle, etc.) are also shown. The panel below shows spectral emissivity in the microwave and infrared and the infrared reflectivity solution on the SW.

The top right panels the brightness temperature computed from those state are shown along with the μR_i of the 9 AIRS FOV's. In the second panel the observations (cloud cleared radiances) minus the computed solution (obs-cal's) are shown. The lower panel shows the computed radiances from the solutions versus the reference profile - in this case ECMWF.

| Menu Options within pl_retccr command line systax | |
|---|---|
| option | description |
| exit | exits program |
| redo | replots the current profile (if window size changed) |
| browse | steps through cases, grab a High Life and enjoy |
| time x | sets timer for pause in browse mode |
| diff | plots difference of ret's and reference in $T(p), q(p),$ and $O_3(p)$ plots |
| qa | prints a table of QA information |
| home | goes to profile #1 |
| + | skips 30 profiles (1 scan line) |
| skip N | skips N profiles |
| goto | goes to the profile index number |
| skiprej | skip all rejected profiles |
| plotrej | do not skip all rejected profiles |
| fmin 800 | sets left side of radiance plots to 800 |
| fmax 900 | sets left side of radiance plots to 900 |
| dtsv x | skips along until a profile with $\Delta T_{skin} > x$ |
| alat y | finds 1st profile within ± 5 degrees of y |
| alon x | finds 1st profile within ± 5 degrees of x |
| find x y | finds 1st profile within ± 5 degrees of x & y |
| xwd | does a X window dump and makes a jpg file of entire screen |
| eps | makes encapsulated postscript files for each of the panels |
| ps | makes postscript files for each of the panels |
| jpg | makes JPEG files for each of the panels |

9.13 pl_realgran: plotting maps of retrieval products

The program `pl_realgran` was originally written to plot fields of L2 products. It has many modes and options. To use the program the following namelist variables must be set.

```
pro_airs.nl
  writeradused = T

io_airs.nl:
  Nchan_ccr = 10,
  Ichan_ccr = 175,204,227,257,843,950,1199,1236,1260,2333,
  Nformat_ccr = 8,
  Iformat_ccr = 0,1,2,5,40,11,14,41,
  l1file_ccr = 'allccr.bin'
```

This program will plot retrieval fields, differences between the retrieval and ECMWF use the following from any directory of your choice (it used the environment variables) and all created files will be deposited in your current directory:

```
plot_pro/pl_realgran, year, month, day, runID, cnfname, glist, $
```

| pl_realgran command line systax | |
|---------------------------------|--|
| required input | description |
| year | year of observation |
| month | month of observation |
| day | day of observation |
| runID | Run ID |
| cnfname | root name of the configuration file |
| glist | LONARR() list of granules to plot if glist = -1, then will plot all granules in .out file |
| option | description |
| /fg | use .fg file(s) instead of ret files |
| /mit | use .mit file(s) instead of ret files |
| /avn | use AVN file instead of ECMWF file |
| ret=N | will use \$AIRS_ret/year/month/day/retN |
| /notru | NO reference profile exists |
| /noreal | will allow re-entry to pl_realgran without re-reading data |
| /trop | O3 in ppb, O3,CO,CH4 is average from tropopause to surface |
| /cloudstat | if 'ALL' mode will plot error(cloudiness) as a scatter plot |
| tvpran = [] | set lower,upper pressure for index=5 T(350) plot default = [300,400] |
| markplot = [] | place an "X" at the specified [long,lat] or [long,lat,symbol,size,color] |
| coast = color | plot coastlines and boundaries of US states |
| /ptrop | field=7 is tropopause pressure |
| /psurf | field=7 is surface pressure |
| v4qatest = 8 | uses mid-trop QA for T(mid) |
| region = | set latitude/longitude range to preset values |
| latrng = [] | set [min,max,step] for latitude scale NOTE: [0,30,10] is the same as region=4 |
| lonrng = [] | set [min,max,step] for longitude scale NOTE: [-120,-60,15] is the same as region=4 |
| autoplot=[] | will perform commands in the list this accepts most of the commands in the menu e.g., ['eps', 'reg', 'day', 'night', '1', '2', 'exit'] |

| region | latitude range | longitude range | geo-region |
|--------|-------------------------|------------------------------|------------------|
| 1 | $-60 \leq \phi \leq 60$ | $-150 \leq \lambda \leq 60$ | US,Brazil,Africa |
| 2 | $20 \leq \phi \leq 80$ | $0 \leq \lambda \leq 90$ | Moscow |
| 3 | $-60 \leq \phi \leq 30$ | $-90 \leq \lambda \leq 60$ | |
| 4 | $0 \leq \phi \leq 30$ | $-120 \leq \lambda \leq -60$ | |

It takes 5s per granule to load radiances and 45s per granule to load L2 files and then there is a menu to do it all. An example of the menu is:

```

dev) toggle device type (X-term/ps/eps): X-term
mod) toggle between ECMWF,RET,DIFF,ALL, currently: RET
max) set max for RET
min) set min for RET
dis) display granules on SINGLES, REGIONAL, WORLD map, currently: REGIONAL
D/N) day (DAY), night (NGT), or both (D/N): D/N

```

```

rej) rejection criteria used to make plots: ON
ppp) # of plots per page: 1
tit) default
all) loop through all plots
exi) EXIT program
 1)   cldfrc    0.0   90.0           2)   rejmap    0.0    1.0
 3)   ccrbias  -0.0   40.0           4)   tair     260.0  310.0
 5)   t350     220.0  250.0          6)   tskin    260.0  300.0
 7)   toth2o   0.0   10.0           8)   pcld     0.0  1100.0
 9)   toto3    100.0  500.0          10)   co       50.0  300.0
11)   ch4     1700.0 1900.0          12)   co2     355.0  385.0
13)   e9       0.8   1.0           14)   e10     0.8   1.0
15)   LIQ     0.0   0.4           16)   BT714   230.0  260.0
17)   BT937   260.0  300.0          18)   BT2616  260.0  300.0
19)   CCR714  240.0  260.0          20)   CCR937  260.0  300.0
21)   CCR2616 260.0  300.0
Enter code or command ==>

```

A very useful feature upon having IDL crash due to code or keyboard problems is to re-enter `pl_realgran` without re-loading the common block

```

IDL> retall
IDL> .run pl_realgran
IDL> pl_realgran,,2003,1,14, 'd65bin', 'test', [51]

```

Another useful feature is to call `pl_realgran` with a predetermined list of commands. The plots below are made using

```

autocmd = ['eps', '2', '15', '6', 'all', '6', '7', '9', 'exit']
pl_realgran, 2003, 1, 14, 'd65bin', 'test', [51], autoplot=autocmd

```

The colors in the rejection map indicate various types of rejection. The flag type, # of unique occurrences (in the order of rejection from top to bottom) and the total number of occurrences of this flag being evoked are given in the annotation on the right hand side. This is a cryptic code, but here is a quick summary

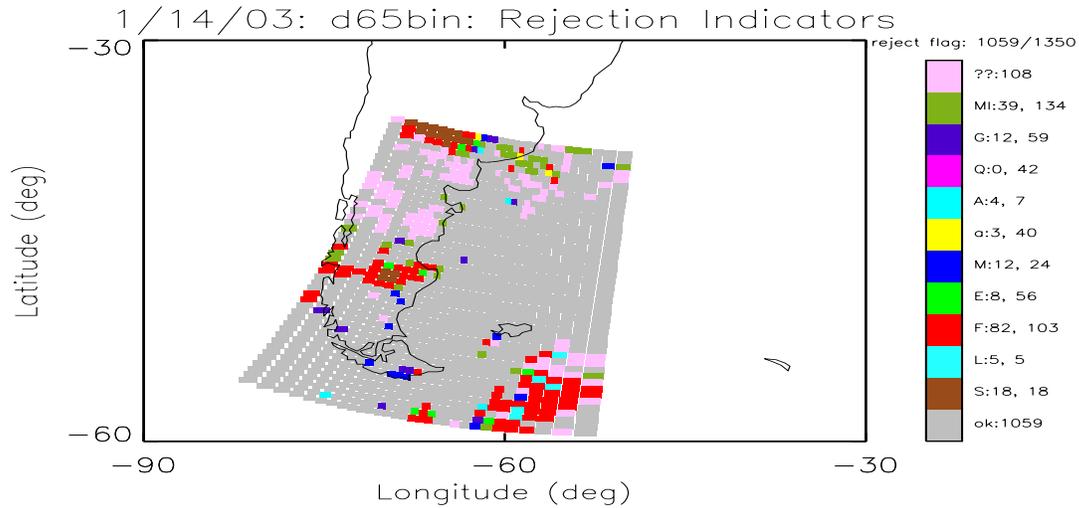


Figure 9.30: Example output from the pl_realgran, 2003,1,14,'d65bin', 'test'. Selecting menu item 2

| Rejection Codes | |
|-----------------|--|
| code | description |
| ?? | a rejection criteria not in this list (reject du jour) |
| MI: | MIT rejection criteria |
| G: | Effective amplification factor |
| q: | ret quality (convergence) indicators |
| A: | cloud clearing amplification & low cloud test |
| a: | cloud clearing amplification factor, NOAA PC Score |
| M: | microwave/infrared comparisons |
| E: | cloud clearing residual test (ETAREJ) |
| F: | Cloud fraction tests |
| L: | Liquid water test |
| S: | software errors (failed matrix inversion, eigenvector decomposition, not enough good channel, etc. these can occur when running extremely cloudy cases to the end |

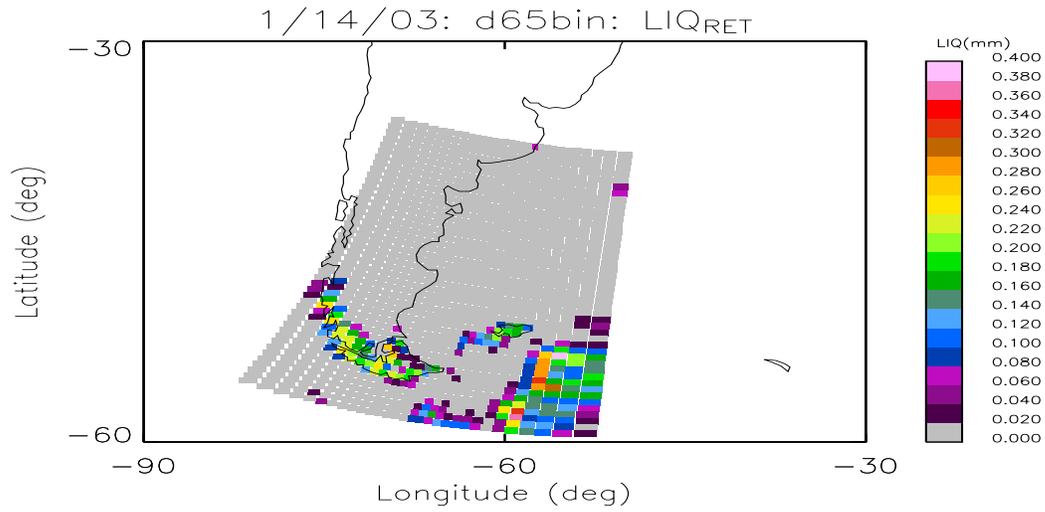


Figure 9.31: Example output from the pl_realgran, 2003,1,14,'d65bin', 'test'. Selecting menu item 15

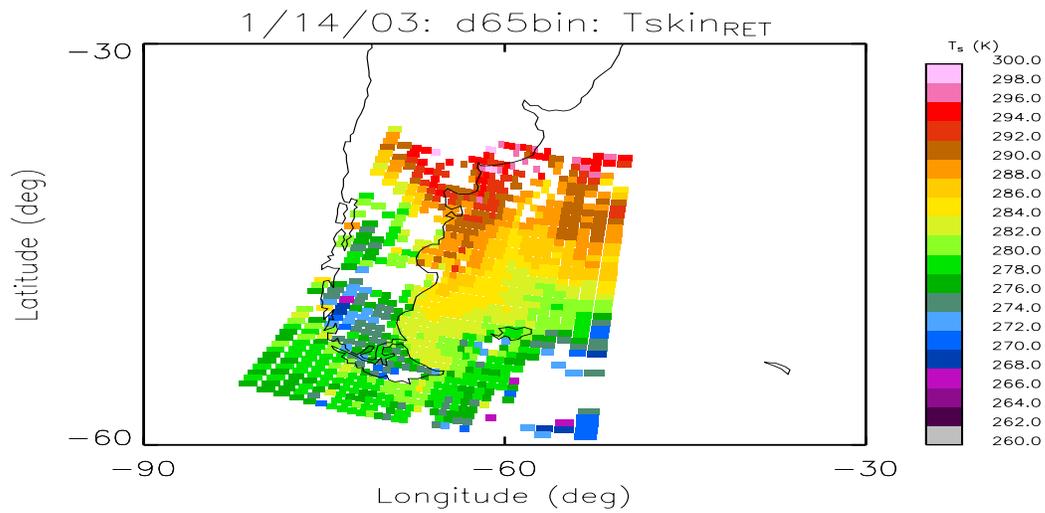


Figure 9.32: Example output from the pl_realgran, 2003,1,14,'d65bin', 'test'. Selecting menu item 6

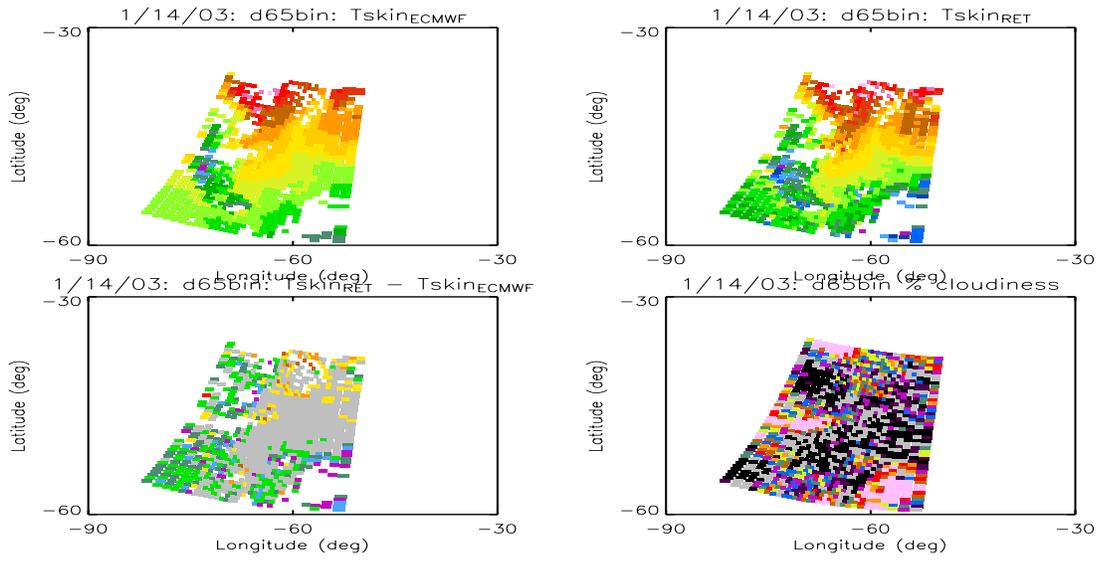


Figure 9.33: Example output from the pl_realgran, 2003,1,14,'d65bin', 'test'. Selecting 'ALL' and then menu item 6

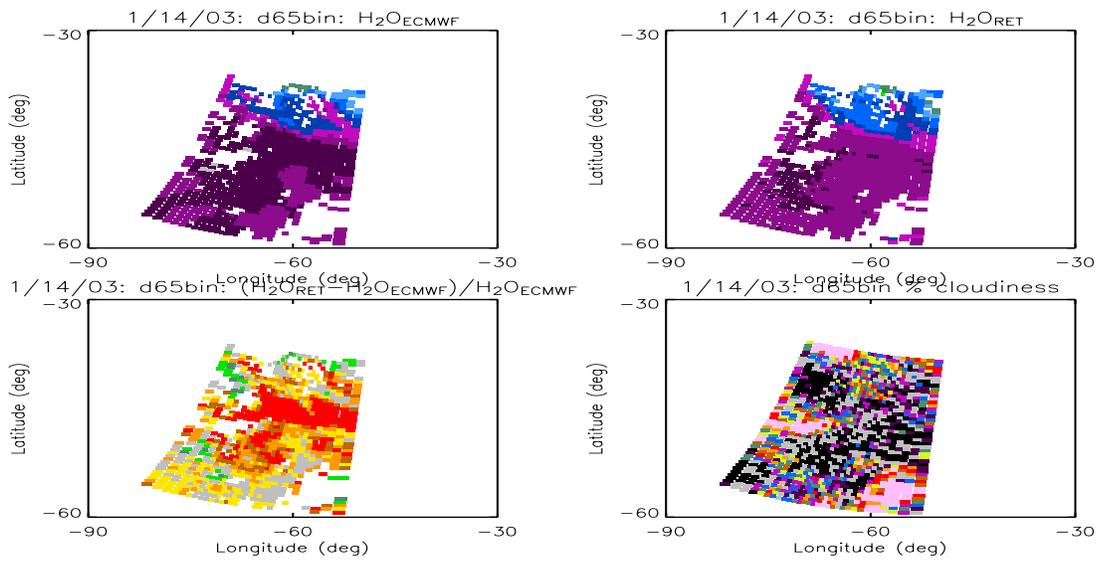


Figure 9.34: Example output from the pl_realgran, 2003,1,14,'d65bin', 'test'. Selecting 'ALL' and then menu item 7

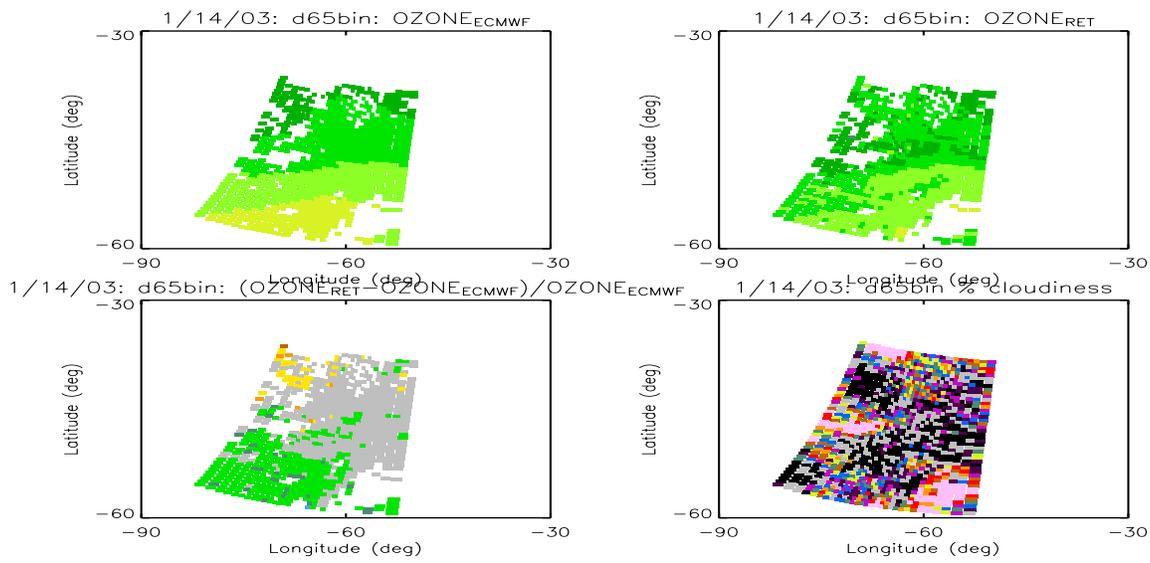


Figure 9.35: Example output from the `pl_realgran`, 2003,1,14,'d65bin', 'test'. Selecting 'ALL' and then menu item 9

9.14 pl_trace: plotting trace gas profiles

The program trace/pl_trace.pro will plot individual profiles of CO₂, CH₄, and CO from the retrieval (.ret), first guess (.mit) and truth files (ECMWF or simulation truth files). It also prints a useful summary page in the current directory named "pl_trace.runID". There are 2 modes the program can work in.

trace/pl_trace, runID, cnfname, [plist=[]], [gloop=[]]

| pl_realgran command line systax | |
|---------------------------------|--|
| required input | description |
| runID | Run ID |
| cnfname | root name of the configuration file |
| plist or gloop | one of these options must exist |
| option | description |
| plist = [] | list of granule index numbers and profiles to plot |
| gloop = [] | list of granule numbers |

The first mode plots selected retrievals. The optional parameter selected which granule (index into the .cnf file) and a profile number. Since granules have 1350 profiles this is given as a 4-digit mantissa such that 1.0001 will plot the 1st footprint of the 1st granule in your cnf file. For the example below I used `plist=[1.0001,1.0135]`

The second mode is intended to plot a quick look at a group of granules. Here the granule list is given (not applicable to the single test granule) and the program will plot the 1st, 15th, and 30th FOV (all within the first scan line). In this way a quick global view of the dataset can be made.

trace/pl_trace, runID, cnfname, glist, [gloop=[]]

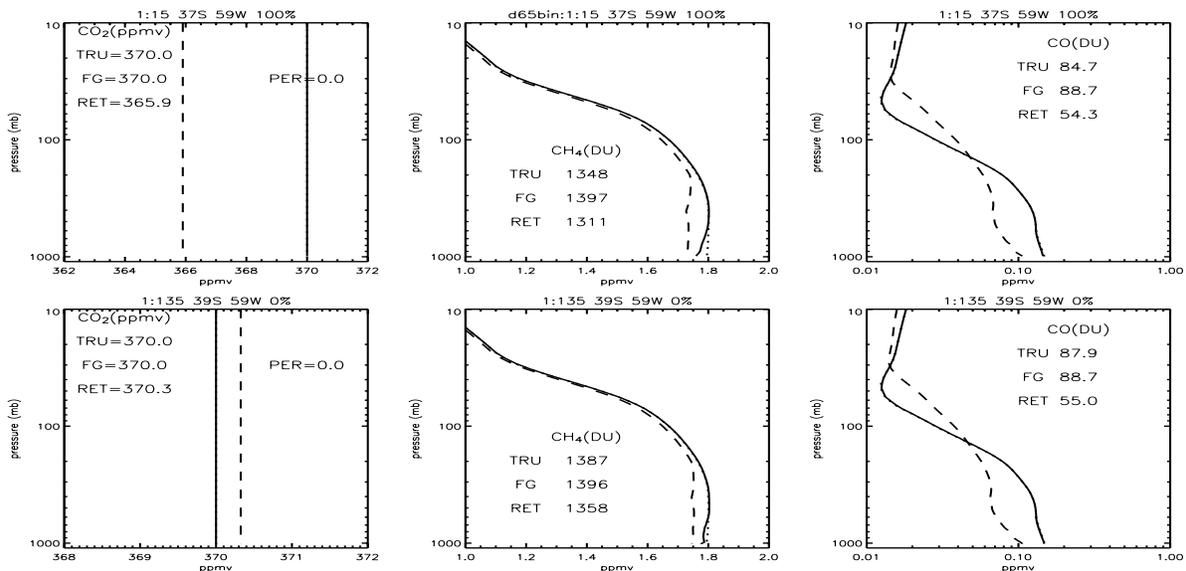


Figure 9.36: Example output from pl_trace, 'd65bin', 'test', `plist=[1.0001,1.0135]`

9.15 pl.cmdl: comparing retrievals to matchup datasets

This program that may be useful, at least as an example, of how I have used my system to compare retrievals to CMDL files. This was the program I used to make some quick plots for the proposal.

My approach is built this way because I want to self-document what I am doing. As a proof of this concept I was able to reproduce exactly where I left off in April of 2003 (when the proposal was written) and make comparison plots of CMDL to the current retrieval system within an hour.

There are many ways to approach this. Remember the goal is to compare our retrievals with CMDL quickly so that we can improve the retrieval. This means you need to make it as simple as possible to re-run and re-analyse the data. I am not saying this is best way – you tell me a better way. But, I think it is important, as a first step, to look at each and every aircraft profile in detail. The second step is to build statistics of the retrieval versus CMDL so that we have a metric for retrieval skill. If I were to do it, which I know I am not, I would start with pl.cmdl and build on that to compute statistics.

I offer what I have done in the past in case it is useful. Use or don't use as you see fit.

The CMDL data is on orbit009 in /home/cbarnet/airsb/data/cmdl

The re-run is in document/c60 & document/d65

There are 6 steps I did to make the figures below

1. run retrievals for all granules in database
2. run the search_loc program to build a concatenated retrieval file from the complete run in step #1.
3. Use build_cnf to rerun the retrieval on the subset of cases
4. Use reform_ret to build new retrieval files.
5. select profiles for comparison and put them into a list file.
6. run a program like pl.cmdl.pro to plot CMDL profiles versus selected retrievals.

Step #1 run retrievals

When you process the L1b's and run the retrieval I suggest using a single output location for your experiment by setting the environment variables for temporary and retrieval output to the same place.

```
AIRS_tmp=/yourpath
```

```
AIRS_ret=/yourpath
```

As you preprocess and run retrievals you will be storing each days temporary files in

```
$AIRS_tmp/yyyy/mm/dd/tmp
```

and retrievals in

```
$AIRS_ret/yyyy/mm/dd/ret
```

In this case the entire output will be in the same directory path

```
/yourpath/yyyy/mm/dd/tmp
```

```
/yourpath/yyyy/mm/dd/ret
```

Each day's granules should be run as a group. See /net/orbit009l/home/cbarnet/airsb/scripts/run_teide.com for a recent example of a scripts to do this. If you need to re-process or re-run retrievals you can simply re-execute the script - so I find this a useful way to organize things.

Step #2 build concatenated ret files

The next step is to find the retrievals that are within reasonable range of the date, time, and site location. I used `search_all.pro` because I have previously processed a large number of granules for a variety of experiments. The run I did was over a year ago, but I can still use the retrieval files from that time frame to build new retrievals from. If granules have been added (which they have for new dates) I simply need to add these days to the search program.

In your case, you should be searching a limited set of granules in `/yourpath/` so your version of `search_all` should be short and quick.

Look at `id/exercise/search_teide.pro` for an example of how I use the `search_loc` program to build `.log`, and `.ret.` files for a limited subset.

Step #3 & #4 re-running the retrievals

If the retrieval system has changed you can re-run on the subset in the `.log` file. See Section 10.2 for an explanation of `build_cnf.pro` and `reform_ret.pro`.

Step #5 select profiles to compare to

For each day and site (a CMDL file we have in-hand) I look through the log file until I find the date. Then I look for the clearest profile (low values of `%cld`) and also the closest profile (low values of `r(km)`).

For example, for Carr CO we have a file for Sep. 6, 2002 (`car 2002-09-06.1408.mrg`) and if I look in a log file I find that

G=193, SC=68, FOV=2 is

a) accepted

b) 0.1% clouds

G=193, SC=77, FOV=2 is

a) accepted

b) the closest (48.1 km) to the CARR CO site

So from the log file and the list of profiles we have in house I select one of each and build a `.lst` file. The `.lst` file has a header with the number of cases, `Ncmdlmax`, and the maximum number of retrievals, `Nretmax`, to be plotted per CMDL (so I can dimension arrays).

Then there `Ncmdlmax` number of blocks. Each block has a header line with the number of retrievals, the `runID`, the site, the `cmdl` file name. For each retrieval there is a line with the year, month, day, granule number, scan line number, and `fov` number for the retrieval you want to compare to. For the example plot I have shown, this is what the list file looked like

```

20  2
2 c60 car 2002-09-06.1408.mrg
2002 9 6 193 68 2
2002 9 6 193 77 2
2 c60 car 2002-09-14.1404.mrg
2002 9 14 201 86 59
2002 9 14 201 89 53
.....

```

Step #6 running pl.cmdl.pro

pl.cmdl simply needs the name of the list file to do its job.

Example on orbit0091

So here is an example of how to do something useful. Back in April of 2003 I did a run (b67) and made plots for my proposal using b67.lst. I want to run v4.0 emulation on just those cases and make new plots. The plots I made were produced as follows

1. In Jan. 2004 I used autoproc and procret.com via scripts/run1.com → run6.com to process 3500 granules in a run I called c60bin in /disk3/pub/ret_dir/yyyy/mm/dd/ret.

This took about 2 weeks.

2. I ran search_all.pro for the 'carr', 'raro', 'harv', 'wlef' sites. I did this in the directory
airsb/data/cmdl/document/c60

search_all, 'carr' search_all, 'wlef' search_all, 'raro' search_all, 'harv'

These took about 15 minutes each because they were searching 3000 granules.

3. I then copied renamed it c60.lst changing the runID from 'b67' to 'c60' inside the file. I was then able to use

pl.cmdl, 'c60.lst'

to make plots of the c60 run.

4. I used the 4 .log files from document/c60 to build a new log file called
cmdl.eml

in a new directory: document/d65. To do this I pulled the lines out of the 4 log files that corresponded to the ones I selected in b67.lst.

This takes about 1/2 hour to set-up but it is worth it since the retrieval will run in minutes instead of hours. No need to reprocess more than your going to use, right.

IDL> build.cnf, 'cmdl'

5. I re-ran the retrieval using the *v40.nl from /home/cbarnet/airsb/trace_nl/*v40.nl

I edited cmdl.cnf's header to 18 granules and put the 442 profiles I selected to re-run (for all 4 sites) into io.v40.nl.

LINUX> procret.com d65bin cmdl v40

This ran in 7 minutes, 45 seconds

NOTE: that this retrieval is running all the retrievals for the multiple sites in one run.

6. I built new concatenation retrieval files for each site with

IDL> reform_ret, 'carr', 'd65bin', 'cmdl'

IDL> reform_ret, 'wlef', 'd65bin', 'cmdl'

IDL> reform_ret, 'raro', 'd65bin', 'cmdl'

```
IDL> reform_ret, 'harv', 'd65bin', 'cmdl'
copied ../c60/c60.lst to d65.lst
search and replace 'c60 ' with 'd65bin'
See the attached plots for example output.
```

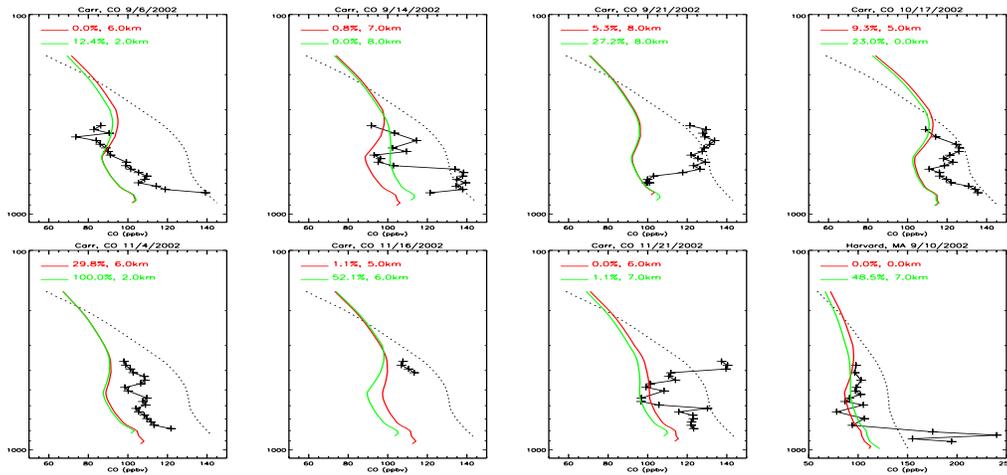


Figure 9.37: Example CO matchup using pl_cmdl: IDL> pl_cmdl,'d65.lst', /noco2, /noch4, /all, /ps

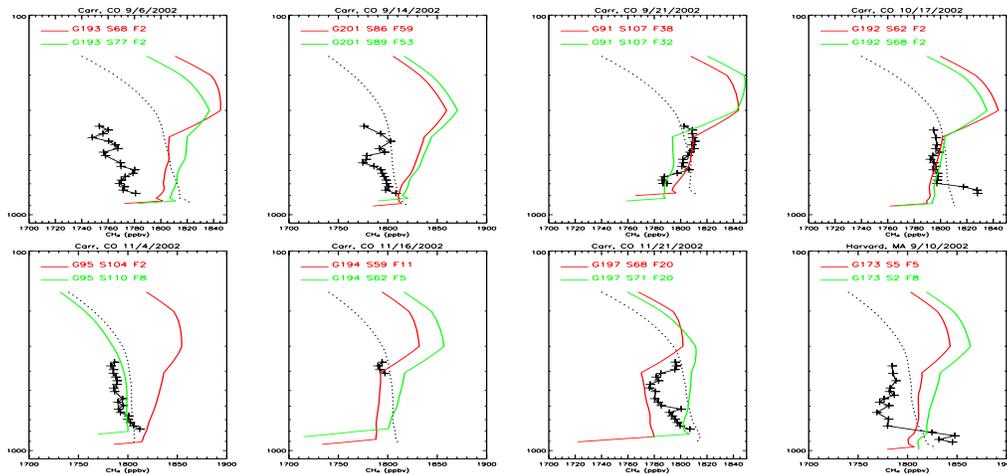


Figure 9.38: Example CH₄ matchup using pl_cmdl: IDL> pl_cmdl,'d65.lst', /noco1, /noco2, /all, /ps

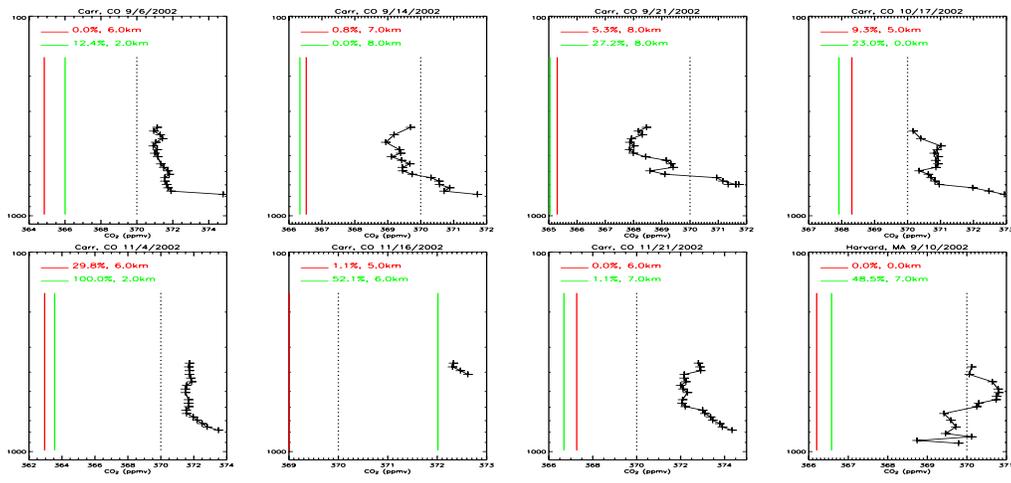


Figure 9.39: Example CO₂ matchup using pl_cmdl: IDL> pl_cmdl,'d65.lst', /noco1, /noch4, /all, /ps

9.16 pl_trscan: plotting trace results versus scan position

Another useful program is trace/pl_trscan. This plots the retrieval as a function of scan line and is called similar to the granule mode of pl_trace, specifically,

```
pl_trscan, 'b18g_tret2', gloop=[1,151,10]
```

This program will become useful if we ever have skill with CO2. In that event, looking for matching of patterns in CO2 will be important because that is what will convince the CO2 community that our product can be used in process studies and campaigns. Right now it shows we have little skill.

9.17 pl_f65: plotting results from co2_per

The output from `co2_per.F` can be analysed with `idl/pl_f65.pro`. You should familiarize yourself with this program. I wrote some sparse documentation on how to use the program at the beginning of that file. For example, if I do a run with a rootname of “b18g_tret1” then the IDL program is called with

```
pl_f65,'b18_tret2'
```

It will take care of adding the configuration name (in simulation this is “s01”) and then open the `.out` and `.f65` files. This program is intended to be a diagnostic program, but it might be useful in operation. For this reason, you can set “`useCO2_p = .TRUE.` or `.false.`” to use the output from this program. It keeps statistics either way, but the first guess for `ret_co2.F` will be set to the output of `co2_per` only if `useCO2_p` is `.TRUE.` (default).

Chapter 10

Utilities for Processing Large number of granules

10.1 calling IDL from a script

Whenever processing a large volume of data it is rational to keep on eye on the total disk spaced used. There are many reasons to use LINUX or UNIX scripts to drive the preprocessor

1. The preprocess essentially doubles the size of the L1b files, therefore a script can prepare the data and clean house (*i.e.*, delete temporary files) after the job is done.
2. The script can be a record of processing such that re-running the entire job can be accomplished easily. This can be important if the most current version of the PGE needs to be emulated.
3. Many jobs can be stacked to keep a CPU busy for many hours or days.
4. The script can also delete the diagnostic output file from the retrieval. Sometimes the script can rename retrieval files and move them to a new location. As a minimum be sure to save
 - the .out file (record of namelists and stat's)
 - the .ret file(s)
 - the .f61 file(s)

In my scripts I set all the environment variables and direct the graphics output from IDL to a file. I usually use T-shell, so the beginning of my scripts look like

```
#!/bin/tcsh
#
  umask 022

  setenv IDL_DEVICE "PS"
  setenv IDL_STARTUP "/home/cbarnet/prog/idl/unixstart.pro"

  setenv HOST "orbit09"
  setenv USER "cbarnet"
  setenv TMP "/home/cbarnet/temp"

  setenv NLDIR "/disk3/pub/example/nl"
```

```
setenv AIRS_ret "/disk3/pub/example"
setenv AIRS_tmp "/disk3/pub/example"
setenv home "/home/cbarnet/airsb/scripts"
```

The for each day, I can preprocess and run the retrieval with a block of commands that look like:

```
cd $home
echo "autoproc, 2002, 10, 01, checklist=[65,66,174,175],/nortp" | idl
cd $AIRS_ret/2002/10/01/ret
cp $NLDIR/* .
procret.com d65 021001_us v40
mv *.ret *.mit *.fg $hill/2002/10/01
$home/cleanret.com
cd $AIRS_tmp/2002/10/01/tmp
$home/cleantmp.com
```

This script does the following

1. Call the preprocessor
2. Updates the namelists
3. Runs the retrieval
4. Removes, renames, or moves retrieval output (I do this via another script cleanret.com)
5. Removes temporary files.

All my scripts are located on /net/orbit009l under the directory /home/cbarnet/airsb/scripts.

10.2 Retrievals near validation sites

If you want to extract retrievals co-located to a specific validation site there is a suite of IDL programs to assist you. The program suite is composed of

| | |
|------------------|--|
| search_all.pro | program to select FOR's from granule files |
| repreproc.pro | program to run preproc on subset of granules |
| build_cnf.pro | program to set-up to re-run retrieval |
| reform_ret.pro | program to select FOR's from new run(s) |
| resolve_site.pro | site latitude & longitude, search radius |
| search_loc.pro | subprogram used by search_all & reform_ret.pro |

In order to use this set of programs you must begin by running retrievals on individual granules for the set of days of interest to your research. The same runID must be used in all the runs and the cnf files must conform to autoproc.pro (YYMMDD_us.cnf or [root=''].cnf). As a minimum the following files must remain intact (that is, you cannot overwrite them with subsequent runs) to use this set of programs

- all cnf files
- all runID.cnfname.ret files
- all L1b, calbuf binary files in \$AIRS_tmp/yyyy/mm/dd/tmp

10.2.1 search_all: find retrievals within a given distance of a ground station

This program will search a set of retrieval runs, all with the same runID, to find and summarize all retrievals that fall within a given radius of a particular ground station (*i.e.* a site). The files must be located with the directory structure given by \$AIRS_tmp/yyyy/mm/dd/tmp and \$AIRS_ret/yyyy/mm/dd/ret and all the files must have the same runID. At a minimum, the .ret and .f61 files must exist in order to search for retrieval files.

The program, search_all.pro, is really a driver for preproc/search_loc.pro & search_head.pro, where the real work is done. The program opens a log file and calls search_loc.pro for each and every day that is to be searched. In order to use search_loc.pro you need to copy search_all.pro to your own IDL directory and edit the program to

- your data days
- your configuration file names

The existing search_all.pro searches all the files in the first 3 months of AIRS data on the orbit0091 system at NOAA. In the near future I will make this a generic program that searches the entire directory tree under \$AIRS_ret so that it is generically useful. The calling sequence is:

```
preproc/search_all, site, runID, [/fg], [/mit], [root="]
```

| search_all command line systax | |
|--------------------------------|--|
| required input | description |
| site | Name of site (argument for resolve_site.pro) |
| runID | run ID for a set of retrieval files |
| option | description |
| /fg | will build a summary file for the .fg files |
| /mit | will build a summary file for the .mit files |

The program search_all.pro builds an output filename from the site root name (see right column in Table 10.1) and the first 3 characters of the runID. For example, if we called the program with the following arguments

```
search_all, 'port', 'd65bin'
```

Then the output files would have the rootname of

```
porto_velho_d65
```

search_all.pro creates a number of files that are within the specified radius (radius is from resolve_site.pro) of the validation sites latitude and longitude. The files all have the name rootname with the following extensions:

| extension | description |
|-----------|---|
| .ecm | A concatenated L2 file of reference (L2.I in cnf file, usually the ECMWF or AVN) records. |
| .ret | A concatenated L2 file of retrieval (.ret) records. |
| .fg | If /fg is used then a L2 file of .fg records. |
| .mit | If /mit is used then a L2 file of .mit records. |
| .log | A 1 line summary of each retrieval (see below). |

The following columns exist in the log file

Table 10.1: Validation Sites in resolve_site.pro

| code | Location | latitude | Longitude | site name |
|------|--------------------------|----------|------------------|---------------|
| PHOE | Phoenix AZ | 33.453 | -112.083 phoenix | |
| DOME | Dome C, Antarctica | -75.0 | 123.0 | domec |
| ARM | ARM SGP Cart Site | 36.606 | -97.485 | arm_sgp |
| ABOV | Chesapeake Lighthouse | 36.91 | -75.71 | above |
| UMBC | UMBC Physics Dept | 39.2603 | -76.7089 | umbc |
| GSFC | GSFC Bldg. 33 | 38.9922 | -76.8392 | gsfc |
| MSFC | Huntsville | 34.7 | -86.6 | msfc |
| CARR | CMDL Carr CO Site | 40.90 | -104.8 | carr |
| WLEF | CMDL WLEF Tower Site | 45.93 | -90.27 | wlef |
| RARO | Rarotonga CMDL Site | -21.25 | -159.83 | rta |
| HARV | Harvard Forest CMDL Site | 42.54 | -72.17 | hfm |
| HAWA | Hawaii CMDL Site | 21.23 | -158.95 | haa |
| POKE | Poker Flats CMDL Site | 65.07 | -147.29 | pfa |
| SANT | Santarem LBA Site | -2.85 | -54.95 | santarem |
| FORT | Fortaleza LBA Site | -4.15 | -38.28 | fortaleza |
| VILH | Vilhena LBA Site | -12.4206 | -60.0537 | vilhena |
| PORT | Porto Velho LBA Site | -8.4256 | -63.5352 | porto_velho |
| ALTA | Alta Floresta LBA Site | -9.5200 | -56.06 | alta_floresta |
| OURO | Ouro Preto LBA Site | -10.7500 | -62.2600 | ouro_preto |
| REBI | Rebio Jaru LBA Site | -10.0502 | -61.5555 | rebio_jaru |
| CUIB | Cuiaba LBA Site | -15.6000 | -56.1000 | cuiba |
| PRUD | Prudente LBA Site | -22.1751 | -51.3727 | prudente |
| BAUR | Bauru LBA Site | -22.3578 | -49.0270 | bauru |
| MIRI | Guajara Mirim LBA Site | -10.4718 | -65.1654 | guajara_mirim |
| GMIR | GMmirim LBA Site | -10.75 | -65.30 | gmirim |

| column | description |
|------------|--|
| record | record number in concatenated file |
| mm/dd/yy | UT date of center AIRS FOV |
| hr:mn | UT time of center AIRS FOV |
| gr | granule number |
| sc | AIRS L1b scan line of center footprint |
| fv | AIRS L1b footprint number of center footprint |
| R | prints "R" if rejected |
| #kic | number of kicked channels in this retrieval |
| co2 | retrieved value of CO2 |
| co | retrieved value of CO |
| ch4 | retrieved value of CH4 |
| zenang | Satellite zenith angle (scanang in L1b) of center AIRS FOV |
| lat | Latitude of center AIRS FOV |
| lon | Longitude of center AIRS FOV |
| r(km) | Distance of AIRS center FOV from validation site |
| %lnd | percent land in retrieval FOV |
| Psurf | surface pressure (from AVN forecast) |
| Tsurf | retrieved surface temperature |
| %cld, Pcld | retrieved cloud fraction and cloud top pressure |
| Clr | clear flag |
| Spk | blue spike flag |

Statistics on the entire subset can be computed using the .ecm and .ret, .fg, and .mit files with the program simstat.F (see Chapter 8). Options could be added to simstat to perform post-process rejection to test the impact of a different rejection criteria.

Also, this is a useful file to distribute to scientists at the validation sites, since extraneous observations are removed and the data volume is substantially reduced.

But the real utility of search_all.pro is to select a subset of retrievals for quick re-processing. We can copy rootname.log to a file called rootname.eml and edit rootname.eml to select a subset of retrievals satisfying the research objectives. Then we can re-run only those cases. This is discussed in the next section.

10.2.2 build_cnf: concatenate selected cases & days into one run

This program is used to setup for reprocessing a subset of cases. The calling sequence is

```
build_cnf, filename[, root=""]
```

It is usually called after search_all.pro. The output log file is copied to a .eml file that can be e-mailed and/or edited to select a subset of cases. build_cnf uses this file to find which granules are required, builds a cnf file, and get a set of cases, IDPRO(), to put in the io_xxx.nl file. The proper use of this program requires the following steps

1. run the retrieval for all cases of all granules (only done once)
2. run search_all.pro to select cases, creates .log file
3. rename .log file as .eml and edit, if desired.
4. run build_cnf.pro to create new cnf file
5. copy NUMPRO, IDPRO() from .cnf file to io_xxx.nl
6. run airsb (via procret.com)
7. if a concatenated ret file is desired, run reform_ret.pro
8. if tmp files have been deleted you may be able to quickly rebuild them with repreproc.pro

This program sets up the necessary file to allow running retrievals near a validation site. The file naming scheme of the off-line system is redundant because the granule numbers are redundant. However, the cnf file allows naming the ret files anything you want. Since each day has granules numbered 1 to 240, preproc.pro and the airsb retrieval code named the retrieval output file (.ret file) with the runID and the granule number. Therefore, since the file name does not include the date in the .ret file names, it would not be possible to run multiple days in a single run. To solve this build_cnf.pro will build a new cnf file, constructed from your original cnf files and merge them into a single run. When running the retrieval code the .ret files can (probably will) have only a few retrievals per file and will be written with a sequential indexing.

For example, on 8/31/02 G088, line 104 (out of 135), field of view (fov) 59 (out of 90) is within 18 km of Carr CO. This is field of regard, FOR, 1040 in a granule of 1350 retrievals. This is constructed from the line and fov of the center AIRS FOV as follows

$$\text{FOR} = 1 + 30 \cdot \text{INT}[(\text{airs.line} - 2)/3] + \text{INT}[(\text{airs.fov} - 2)/3] \quad (10.1)$$

On 9/1/02 G206, line=113, fov=86 (FOR=1139) is within 7.5 km of Carr CO. These 2 retrievals could be run in a matter of seconds by selecting those 2 lines from a log file generated from runs of 8/31/02 and 9/1/02 and the configuration output file would run build .ret files from the first group with a name of "G001" by pointing to the temporary files from 8/31/02, G088. The second group, called "G002" would point to temporary files for 9/1/02, G091. If, when running this new configuration file we set NUMPRO=2 and IDPRO=1040,2489. This run will take a matter of seconds, whereas to re-run the entire set of granules

would take 12 minutes/granule. In addition, many of the files are now concatenated (*e.g.* the .f92 and .out file has statistics only the selected retrievals.)

The output file from search_all.pro, rootname.log, (written via search_head.pro) can be copied to a file rootname.eml and edited to select only those retrievals desired for re-processing. This edited list can be used to create a setup for a new retrieval run of only those granules and cases within the granules that the user wants information on.

exercise/build_cnf, rootname

build_cnf.pro is a short and simple program to read in a file of selected retrieval footprints from the file rootname.eml and set up the cnf file for a re-run of the retrieval code. I used the '.eml' extension to avoid overwriting the original log file and because usually this file originates from an email where a user took the log file and returned a subset. Take a look at the IDL code and you will see that build_cnf.pro is simply extracting the segments from your original cnf files to build the new cnf file. Two files are created with the following extensions

.cnf Is a cnf file for airsb that contains all the granules necessary to run the sub-set list of cases in rootname.eml and the list of profiles (NUMPRO, IDPRO()) to re-run only a subset.

.sum A list of all the granules needed to run the subset (see exercise/repreproc.pro). This list is the mapping from the new index scheme to the old granule index scheme.

If the temporary files have been deleted (*e.g.*, to save disk space) the program exercise/repreproc.pro will read the rootname.sum file and rebuild all the temporary files. You should check that repreproc.pro calls preproc.pro in the manner you want.

The steps to re-run the retrieval are summarized here

1. Use preproc/search_all.pro to find all the retrievals within a given radius of a validation site.
2. Edit the log file.
3. Use exercise/build_cnf.pro to build a cnf file.
4. Edit the cnf file to add the number of granules to the first line (written at the end of the file as 'num Gran =')
5. If desired, edit the I/O namelist (usually io_airs.nl) to include NUMPRO and IDPRO() from the end of the rootname.cnf file. Note that the retrieval cannot allow more than 99000 individual retrievals.
6. Run the retrieval code (use procret.com with a new runID, the rootname of the CNF file, and the desired namelists).
7. If a new .log file or new concatenated .ret, .fg, or .mit files are desired (with results from new retrieval run) then run reform_ret.pro (discussed in the next section).

10.2.3 reform_ret: find retrievals within a given distance of a ground station using concatenated retrievals

When running the off-line retrieval using the output from build_cnf.pro (rootname.cnf) it might be desirable to have the same output files generated by search_all.pro. That is, new rootname.ecm, rootname.ret, rootname.fg, rootname.mit, and rootname.log file so that statistic (using simstat.F) or the 1-line summary, based on the new retrieval, can be used for diagnostics. This can be accomplished with reform_ret.pro.

This program calls the search_loc.pro and search_head.pro in the same manner as search_all.pro except it uses the files in the concatenated retrieval directory and it maps the sequential granule ID number built by build_cnf.pro back to the original granule, line, and fov index scheme.

The calling sequence is

exercise/reform_ret.pro, site, runID, cnfname, [/ecm], [/fg], [/mit]

and it creates

| extension | description |
|-----------|---|
| .ecm | A concatenated L2 file of reference output by the retrieval (runID_cnfname.tru) |
| .ret | A concatenated L2 file of retrieval (.ret) records. |
| .fg | If /fg is used then a L2 file of .fg records. |
| .mit | If /mit is used then a L2 file of .mit records. |
| .log | A 1 line summary of each retrieval (see below). |

The site name is the same site name as the original search_all.pro call. The runID is the new runID of the concatenated retrieval and cnfname is the name of the cnf file generated by build_cnf.pro.

Since this program uses the subset files the concatenated reference file (.ecm) will only exist if the /ecm option is used. In addition, the retrieval must output the subset of reference profiles by setting l2tru_out and logl2tru_out in the I/O namelist file (usually io_airs.nl). These can be swapped within the file. That is change

```
logl2tru_out = 0,  
l2tru_out = 'not.set',
```

to

```
logl2tru_out = 37,  
l2tru_out = 'tru.asc',
```

Chapter 11

Simulated Datasets

Table 11 summarizes the datasets available.

Table 11.1: summary of geophysical datasets in #/airsb/level2

| dataset | # levels | date | features | discussion |
|--------------|----------|----------|---|------------|
| p1/us_std | 66 & 100 | | 1976 US STD in various file formats | |
| p1/TIGR | | | TIGR dataset | |
| p1/aes | 66 | | Stratospheric (SAGE) profiles | |
| p1/mcprof | 66 | | 5 test profiles (Amita) | |
| p1/prof22 | 66 | | 22 profiles used f/ RTA training (Amita) | |
| p1/umbcprof | 100 | | 48 UMBC training profiles | |
| p1/NOAA.L2 | 66 & 100 | | 1988 & 1989 radiosonde profiles | Apdx. E |
| p1/phillips | 66 | | 100 profiles (Phillips 1988) | Apdx. B |
| p1/JPL_eta | 66 | 7/19/93 | NA,NB,NC,DB,DC,DD,DP | Apdx. C |
| p1/JPL_orb | 100 | 11/05/96 | “16 scan” set | Apdx. D |
| p3/truth | 100 | 11/05/96 | “8” track datasets | Apdx. D.3 |
| p2/sep00_xxx | 100 | 9/13/98 | 7 selected granules | Apdx. F |
| p2/jan01_s01 | 100 | 12/15/00 | 1 st scan line of 240 granules | Apdx. G |

p1 \equiv #/level2

p2 \equiv #/exercise

p3 \equiv #/8track

11.1 openl2_b.F & readl2_b.F: Documentation for reading L2 files

All L2 files are read and written by a single I/O driver. The files are opened (for read or write) with openl2_b.F and then records are read by readl2_b.F and written by writl2_b.F. L2 files come in a variety of shapes and sizes. For example,

- Truth files for simulation contain a truth for each AIRS FOV. Therefore, there are 270 FOV's per AMSU Scan line.
- First Guess files (f/ MIT, NOAA) contain one record per AMSU FOR.
- Retrieval files (output from airsb.F) contain one record per AMSU FOR.

In all cases, the file contains all the geophysical information necessary to compute a radiance. For a retrieval file, this might include items that were NOT retrieved, such as cloud emissivity, but when a cloudy radiance was computed the cloud emissivity that was specified is included in the file.

level.2 header record (read with ~/src/src_gsfc/openl2.F)

- version: a 5 character version identification (a5).

| version | index | description |
|---------|-------|---|
| L2V1B | 1 | 1 frequency scale for cloud spectral properties |
| L2P1C | 2 | independent cloud spectral properties, real seconds, but L2P1C was an incorrect label. This type == 3 |
| L2V1C | 3 | independent cloud spectral properties, real seconds |
| M2V1C | 4 | same as iver=3, except water, ozone, and liq.water are given in mass mixing ratio at pressure levels. |
| L2V2B | 5 | JPL binary file from Evan (ASCII not defined) for simulation files prior to 4/1/00 |
| L2V2C | 6 | JPL binary file from Evan (ASCII not defined) |
| L2P1D | 7 | Mitch's ASCII FORMAT (binary not defined yet) |
| GPV2A | 15 | GSFC internal I/O version w/ CO2 |
| GPV2C | 16 | GSFC internal format to match JPL's L2V2B & L2V2C |
| GPV3A | 17 | GSFC internal format w/ 100 level CO2 nemis and ncemis variable |
| GPV3B | 18 | same as 17 but more binary is more IDL friendly ASCII is identical to GPV3A (# 17) |

- numpro: the number of profiles in this file (i10)
- nemis: number surface emissivity/reflectivity wavenumber hinge points (see **getemis1.f** for use of these hinge points) (i10)
- ncemis: number cloud emissivity/reflectivity wavenumber hinge points (i10)
- nl2lev: number of layers in the vertical profiles (i10)
- ndescr: number of description lines in header (i10)
- Pobs(L), L=1, nl2lev: pressure value at bottom level of layer (8f10.3). The top of the atmosphere is **implicitly** set to 0.005 mb (*i.e.*, Pobs(0)=0.005).
- descr(i), i=1,ndescr: 80 character labels, (a80)

level.2 data record (read with ~/src/src_gsfc/readl2.F)

Each record (profile or FOV) within the level.2 file has the following format. This is the specification for GPV3a (iver=17):

- idprof: 8 character profile identification, usually 1st 4 characters are descriptive and the last 4 characters are a number (a8).
- latitude (± 90) and east longitude (± 180) in degrees (2f9.3)
- year, month, day, hour, minute, second: (i6,4i3,f7.3). NOTE: the year is now 4 digits and the sec is now a real*4 value.
- pland: fraction of land (0 = all ocean, 1 = 100% land) (f10.5)
- Psurf: surface pressure in milli-bar (f10.2)
- Tsurf: skin temperature (f10.2)
- emismw: microwave emissivity (f10.5) at 50.3 GHz.
- topog: topography parameter (f10.2) . This is the altitude (km) above sea level if pland < 0.4. Otherwise, it is the ocean depth in meters.

- nclد: number of cloud layers (i3) NOTE: The format of nclد has changed from (i4) to (i3).
- Smw_surfclass: the microwave surface type

| version | index | description |
|------------------------------|-------|-------------------------------------|
| MIT surface type definitions | | |
| for FLAND _i .5 | 0 | Coastline (0 < % LAND < 50) |
| | 2 | Water (% LAND = 0) |
| | 3 | High emissivity Sea Ice |
| | 4 | Low emissivity Sea Ice |
| for FLAND _i .5 | 1 | Land |
| | 5 | Snow (higher-freq scattering) |
| | 6 | Glacier/Snow (very-freq scattering) |
| | 7 | Snow (lower-freq scattering) |

- N_Smw: number of microwave spectral points
- nemis: For versions that have this entry, this value overwrites the number of surface emissivity hinge points in the header.
- ncemis: For versions that have this entry, this value overwrites the number of cloud emissivity hinge points in the header.
- satheight: The satellite altitude from the surface in kilo-meters.
- viewang: The satellite view-angle. This angle is defined to be the IDEAL view angle and is related to the satellite zenith angle, θ , at the surface by

$$\text{viewang} \equiv \frac{180}{\pi} \cdot \sin^{-1} \left[\frac{R_e}{(R_e + \text{satheight})} \cdot \sin \left(\frac{\pi \cdot \theta}{180} \right) \right] \quad (11.1)$$

where, $R_e = 6371.004$ (see src_jpl/paramet.com, REARTH)

- solzang: The solar zenith angle in degrees.
- avgCO2: The average value of CO₂ expressed in parts-per- million. If CO2ppm(L) is not specified than CO2ppm(L)=avgCO2. Otherwise, this value is ignored.
- pclدtop(i), clدfrc(i): cloud top pressure and cloud fraction for i=1,nclد (f10.2,f10.5)
- tair(L), wcd(L), ocd(L), wlcd(L), ciw(L), cocd(L), ch4cd(L), CO2ppm(L) for L=1,numlev: (f9.3,3e15.5,i2,3e15.5)
 - Tair(L): temperature in Kelvin at Pobs(L)
 - wcd(L): water layer column density in units of molecules/cm² of water between levels at Pobs(L-1) and Pobs(L).
 - ocd(L): ozone layer column density in units of molecules/cm² of ozone between levels at Pobs(L-1) and Pobs(L).
 - wlcd(L): the liquid water layer column density in units of molecules/cm² of liq. water between levels at Pobs(L-1) and Pobs(L).
 - ciw(L): ice/liquid flag for gpg_l(L) (0=water, 1=ice)
 - cocd(L) the carbon monoxide (CO) layer column density in units of molecules/cm² of CO between levels at Pobs(L-1) and Pobs(L).
 - ch4cd(L) the methane (CH₄) water layer column density in units of molecules/cm² of water between levels at Pobs(L-1) and Pobs(L).
 - co2ppm(L) is the dry mixing ratio of Carbon dioxide (CO₂) in units of parts-per-million.
- Smw_freq(1:N_Smw), Smw_emiss(1:N_Smw): If N_Smw is greater than zero then the spectral array of microwave hinge points will be read in. The emissivity for each channel is computed according to these hinge points and the function in src_gsfc/getemis2.F.
- freqemis(i), emisir(i), rhoir(i), i=1,nemis: the IR surface emissivity and reflectivity defined at frequency hinge points (f10.3,2f10.5). The emissivity for each channel is computed according to these hinge points and the function in src_gsfc/getemis2.F.

- `cldfreq(i,k)`, `cldemis(i,k)`, `cldrho(i,k)`, `i=1,ncemis`, `k=1,ncl`: the IR cloud emissivity and reflectivity defined at frequency hinge points (f11.3,2f10.5). The emissivity for each channel is computed according to these hinge points and the function in `src.gsfc/getemis2.F`.
- `nspr`, `nspi`: number of spare real and integer values (2i3)
- `ispare(nspr)`: integer spare values (8i8)
 - The off-line retrieval uses this parameter to store rejection and clear flags as well as format of `rspare`.
 - see `doc/namelist.ema` and `src/ret_driver.F` to see details
 - `MAXSPARE = 500` is set in `src_jpl/paramet.com`
- `rspace(nspr)`: real spare values (4e15.5)
 - The off-line retrieval uses this parameter to store OLR, COLR, rejection criteria and 9-spot cloud fractions
 - see `doc/namelist.ema` and `src/ret_driver.F` to see details
 - `MAXSPARE = 500` is set in `src_jpl/paramet.com`

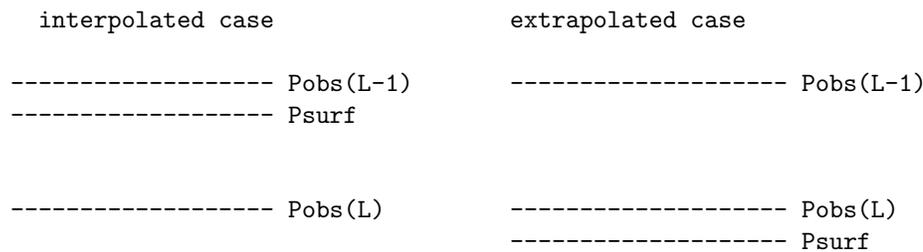
11.2 Conversion of LAYER Column density to Column Density

The retrieval output files contain layer column densities. That is, the column density in molecules/cm² within the layer defined by `Pobs(L-1)` and `Pobs(L)`. For the top layer, `L=1`, `Pobs(0)` is assumed to be 0.005 mb. CO₂ is handled differently in our system and is given in parts-per-million by volume (ppmv), A.K.A. volumetric mixing ratio of DRY air.

To compute total column you must sum the layers. The bottom layer must be interpolated/extrapolated for the real surface pressure. To find the bottom LEVEL you MUST use the function `src.gsfc/l_surface.F`. This function makes sure that a layer is not thinner than 5 mb and IS THE DEFINITION of how the radiances WERE MINIMIZED. Any other method of determining the lowest level is WRONG!

L_{bot} is the first level in which $Pobs(L_{bot}) > P_{Surf} + 5.0$.

This means that there are cases in which the bottom layer is interpolated and cases where it is extrapolated



The last layer column density, given in molecules/cm², is multiplied by

$$BLMULT = \frac{P_{Surf} - Pobs(L_{bot} - 1)}{Pobs(L_{bot}) - Pobs(L_{bot} - 1)} \quad \text{for } L_{bot} = \text{bottom LEVEL} \quad (11.2)$$

Since the bottom layers are ABOUT 25 mb's, $0.2 < BLMULT < 1.2$. It can be convenient to make an array to do this. It looks like

$$\begin{aligned} PLMULT(L) &= 1.0 \quad \text{for } L = 1, L_{bot} - 1 \\ &= BLMULT \quad \text{for } L = L_{bot} \\ &= 0.0 \quad \text{for } L > L_{bot} \end{aligned} \quad (11.3)$$

The total level to space column density is then given by

$$CD_{xxx} = \sum_{L=1}^{100} PMULT(L) \cdot LCD_{xxx}(L) \quad \text{in molecules/cm}^2 \quad (11.4)$$

where xxx is CO, CH₄, or O₃ for our retrievals. The units can be changed to traditional units, if desired. For example, for ozone

$$CD_{xxx} = 1000.0 \cdot \sum_{L=1}^{L_{bot}} \frac{PMULT(L) \cdot LCD_{xxx}(L)}{N_{loschmidt}} \quad \text{in Dobson Units} \quad (11.5)$$

where, Dobson Unit = milli-cm-atmosphere \equiv milli-cm-amagat and Nloschmidt is the NIST value of Loschmidt's number given as $2.6867775 \cdot 10^{19}$ molecules/cm³ per atm @ STP

For moisture we usually use total precipitable water, which is given as

$$gm/cm^2 = \text{cm of water} = 1000.0 \cdot \sum_{L=1}^{L_{bot}} \frac{LCD_{wat}(L)}{N_{avog} \cdot MW_w} \quad (11.6)$$

11.3 Conversion of LAYER Column density to mixing ratio

NOTE: Variables used in this section were defined in previous section.

For average mixing ratio, I usually compute the layer column densities of DRY air and sum both dry air and the specie under consideration to get total level-to-space column density and then ratio them. First, we compute the layer pressure difference as

$$\begin{aligned} dP(L) &= Pobs(L) - Pobs(L - 1) \quad \text{for } L = 2, 100 \\ &= Pobs(L) - 0.005 \quad \text{for } L = 1 \end{aligned} \quad (11.7)$$

The total number of molecules in each layer is given by

$$LCD_{tot}(L) = 1000.0 \cdot \frac{dP(L) \cdot N_{avog}}{mw_d \cdot g_{std}} \quad (11.8)$$

where, 1 mb = 1000 gm/s²/cm² = dyne/cm², $N_{avog} = 6.02214199 \cdot 10^{23}$ molecules/mole, $mw_d = 28.9644$ gm/mole, $g_{std} = 980.665$ cm/s².

The number of DRY molecules in the layer are given by

$$LCD_{dry}(L) = LCD_{tot}(L) - (mw_w/mw_d) \cdot LCD_{wat}(L) \quad (11.9)$$

where, $mw_w = 18.0151$ and $LCD_{wat}(L)$ is the layer column density of water in molecules/cm² given in the L2 file.

So that VOLUMETRIC mixing ratio is given by

$$ppm_{xxx} = 1.0 \cdot 10^6 \cdot \frac{\sum_{L=1}^{100} LCD_{xxx}(L) \cdot PMULT(L)}{\sum_{L=1}^{100} LCD_{dry}(L) \cdot PMULT(L)} \quad (11.10)$$

Other layers can be considered. For example, since we do not have much signal in the boundary layer, one could add a weighting function, $WGT(L)$, to given the TOTAL column density over some other interval. For example, a mid-tropospheric weighting function might be

$$\begin{aligned} WGT(L) &= 1.0 \quad \text{for} \quad 300 \leq Pobs(L) \leq 800 \text{ mb} \\ &= 0.0 \quad \text{for} \quad Pobs(L) < 300 \quad \text{or} \quad Pobs(L) > 800 \text{ mb} \end{aligned} \quad (11.11)$$

Chapter 12

Rapid Transmittance Algorithms (RTA's)

12.1 `comp_rad.F` & `airs_rad.F`: radiance generator utilities

This program is designed to be the documentation for the interface to all RTA's. The RTA's are in libraries. There are 2 microwave libraries: `mwrtav4` and `rta66` and 5 IR libraries: `umbc9803`, `rta66`, `rta100`, `hffp0104`, `oss`. All libraries have the same set of call lists; however, the architecture of the library underneath is radically different. The overall flow of the RTA calls is outlined below. The program "documents" what each pass parameter does and how it relates it back to the original level.2 input file or the loading of the RTA coefficient file.

Many options have been added to `comp_rad.F` over the years to produce a quick result for meetings or evaluation of a problem and, therefore, `comp_rad` has become more and more complex over time. To become familiar with how to use the UMBC RTA interface I would recommend studying/using another program, `airs_rad.F`, that is a stripped down version of `comp_rad.F`. `airs_rad.F` has the following limitations:

- only computes noise free clear radiances
- only computes the AIRS infrared radiances (most recent RTA)
- does not have any options, such as modification of the atmospheric state, adding noise to the radiance, etc.
- it does not link to the libraries but uses include statements to directly import the routines needed. This makes the dependencies more obvious and you can easily find the subroutines.
- It also only works with the most recent L1 and L2 file formats (version 5 for L1 and version 18 for L2)

Both programs are callable from IDL, which makes it a useful utility for a variety of diagnostic purposes. The default operation of `comp_rad` is to specify a L2 file as input and a rootname for the output radiances. `comp_rad` will write out 3 radiance files with the prefix "IR_", "AM_" and "MH_" for AIRS, AMSU, and MHS (or HSB, ATMS) prepended to the rootname for the L1 files. `airs_rad.F` does not prepend any identified to the filename specified on the command line.

`comp_rad` will assume that the view angle, solar zenith angles, and satellite height are set in the following priority

1. defined from the command line with the `-VIEW`, `-SOLZ`, or `-SATH`
2. defined from within the L2 file (if they exist)

3. zero

| steps required to compute a CLEAR radiance | | | |
|--|--------------|----------|--|
| step | module | library | purpose |
| Load IR RTA Files | | | |
| 1 | load_ir | IR RTA | load RTA, freq(), Plb(), Peff() |
| 2 | init_rtacom | IR RTA | initialize RTA common blocks |
| 3 | set_solar | src_gsfc | load SOLAR radiance file |
| Load microwave RTA Files | | | |
| 4 | setmwstart | MW RTA | tell MW RTA where channels start |
| 5 | OPEN_MW W | MW RTA | open MW file |
| 6 | READ_MW(n) | MW RTA | read MW file for channel freq(n) |
| Things done once per dataset | | | |
| 7 | openl2_b | src_gsfc | open geophysical parameters file, Pobs(L) |
| Things Done For Each Profile | | | |
| 8 | readl2_b | src_gsfc | read single geophysical parameter set |
| 9 | tai_to_utc | src_gsfc | convert DD/MM/YY to nUTC seconds |
| 10 | sun_distance | src_gsfc | compute distance to SUN |
| 11 | set_orbit | src_gsfc | compute solar irradiance |
| 12 | comp_alt | src_gsfc | compute altitude(L) for given $T(p), q(p), O3(p)$ |
| 13 | lsurface | src_jpl | compute # of levels from P_{surf} |
| 14 | setang_ir | IR RTA | compute SECANT angles as a f(ALT) |
| 15 | setang_mw | MW RTA | compute SECANT angles as a f(ALT) |
| 16 | init_trace | IR RTA | set trace gas profiles (CO2,CO,CH4) |
| 17 | init_rta | IR RTA | compute RTA predictors (T, q, O3) |
| Things Done For Each Channel, n | | | |
| 18 | getemis2(n) | src_gsfc | compute surface emissivity & reflectivity for microwave channel |
| 19 | MW_RTA(n) | MW RTA | compute transmittance-to-space |
| 20 | MW_BT(n) | src_gsfc | compute Brightness Temperature for infrared channel, n |
| 21 | IR_RTA(n) | IR RTA | compute transmittance-to-space |
| 22 | IR_RAD(n) | src_gsfc | compute clear radiance (w/o noise) |

In order to compute cloudy radiances step # 22 needs to be replaced and some additional functions need to be added.

| additional steps required to compute a CLOUDY radiance | | | |
|--|------------|----------|--|
| step | module | library | purpose |
| Things Done For Each Profile | | | |
| 17.1 | cldy_setup | src_gsfc | pre-compute arrays |
| 18.1 | getemis2 | src_gsfc | compute $\epsilon(n)$ and solar $\rho_{\odot}(n)$ for all clouds |
| 18.2 | getemis2 | src_gsfc | compute thermal reflectivity for all clouds |
| 22 | cldy_rad | src_gsfc | compute cloudy radiance(s) |

12.2 Utility Programs for RTA and RT computations

These programs are located in the \sim /src/src_gsfc library and used to compute level.2 variables to

- tai_to_utc: convert day/month/year to # of seconds since 1/1/1970
- sunang2: compute the solar zenith angle
- comp_alt: compute altitude as a function of pressure
- set_solar: read or initialize the solar irradiance array
- sun_distance: : compute the Earth-Sun distance
- set_orbit: use the Earth-Sun distance to convert solar irradiance to radiance
- ir_rad: radiative transfer for clear scenes
- cldy_rad2: radiative transfer for cloudy scenes

12.3 The MIT RTA

The theory and file formats are discussed in detail in rs_notes.pdf under section 5.11.1, Microwave RTA. In this section, we will discuss the origin of the available files in #/airsb/RTA.

| filename | version | description |
|-----------------------|---------|---|
| amsu.v7a | ver2a | Aqua AMSU, latest version |
| hsb.v7a | ver2a | Aqua HSB, latest version |
| tr_amsua.eos | ver1a | 12/13/00 delivery f/ Phil |
| tr_amsub.eos | ver1a | 12/13/00 delivery f/ Phil |
| mhs.v7a | ver2a | METOP-1 MHS instrument |
| amsu.v6a | ver2a | older version |
| hsb.v6a | ver2a | older version |
| amsu_jpl.v5a | ver2a | == amsu.v5a but w/ JPL Cosmic Tb(n) (convergence) |
| amsu.v5a | ver2a | = tr_amsua.eos + GSFC file mods |
| hsb.v5a | ver2a | = tr_amsub.eos + GSFC file mods |
| amsu.v4b | ver1a | = ascii.trcoef.amsu.v4- + GSFC file mods |
| hsb.v4b | ver1a | = ascii.trcoef.hsb.v4- + GSFC file mods |
| ascii.trcoef.amsu.v4- | ver1a | JPL version of MIT RTA (3/3/98) |
| ascii.trcoef.hsb.v4- | ver1a | JPL version of MIT RTA (3/3/98) |
| ascii.trcoef.amsu.v1a | ver1a | 1 st MIT delivery to JPL |
| ascii.trcoef.mhs.v1a | ver1a | 1 st MIT delivery to JPL |
| trcoef.amsu | ver01 | AMSU + MHS RTA f/ GSFC |
| msu.v4a | ver2a | == tr_msu.dat (via idl_spc/cnv_msu.pro) |
| tr_msu.dat | ver1a | 4-chl MSU file f/ Phil on 5/7/98 |
| ATMS.v4- | ver1a | 32 CHL ATMS channel file - see header |
| atms.v4b | | = ATMS.v4- + GSFC file mods |

Three corrections are made to the recent MIT RTA files for use within the off-line code. The program `~/prog/idl_spc/cnv_trcoef.pro` was used to create the modified amsu, hsb, and atms files from Phil's delivered code.

1. The downwelling factor, κ_D/κ , for each channel was also hard-wired. Row 27, column 5 was used to store this value.
used in amsutau.F
2. The effective center frequency of the channel (to be used for cosmic background conversion of Tb=2.73 K) Row 28, column 5 was used to store this value.
3. The value of the Cosmic Background Brightness Temperature (Value to be used for cosmic background conversion of Tb=2.73 K) Row 29, column 5 was used to store this value.

See Chapter 5, Section 11 for a summary of the theory for the MIT RTA. The coefficients are stored in ASCII files with the formats, given in the following tables.

Table 12.1: MIT RTA file format for oxygen channels

| ABGCFOX3: Oxygen Channel (AMSU 3-13) | | | | | | | |
|--|------------------|------------------|------------------|------------------|-------------------|-----------------|------------------|
| see text for definitions of coefficients | | | | | | | |
| L | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 1 | $\alpha_1(L=1)$ | $\alpha_2(L=1)$ | $\alpha_3(L=1)$ | $\alpha_4(L=1)$ | $\beta_f(i=1)$ | $\beta_s(i=1)$ | |
| 2 | $\alpha_1(L=2)$ | $\alpha_2(L=2)$ | $\alpha_3(L=2)$ | $\alpha_4(L=2)$ | $\beta_f(i=2)$ | $\beta_s(i=2)$ | |
| 3 | $\alpha_1(L=3)$ | $\alpha_2(L=3)$ | $\alpha_3(L=3)$ | $\alpha_4(L=3)$ | $\beta_f(i=3)$ | $\beta_s(i=3)$ | |
| 4 | $\alpha_1(L=4)$ | $\alpha_2(L=4)$ | $\alpha_3(L=4)$ | $\alpha_4(L=4)$ | $\beta_f(i=4)$ | $\beta_s(i=4)$ | |
| 5 | $\alpha_1(L=5)$ | $\alpha_2(L=5)$ | $\alpha_3(L=5)$ | $\alpha_4(L=5)$ | $\beta_f(i=5)$ | $\beta_s(i=5)$ | |
| 6 | $\alpha_1(L=6)$ | $\alpha_2(L=6)$ | $\alpha_3(L=6)$ | $\alpha_4(L=6)$ | $\beta_f(i=6)$ | $\beta_s(i=6)$ | |
| 7 | $\alpha_1(L=7)$ | $\alpha_2(L=7)$ | $\alpha_3(L=7)$ | $\alpha_4(L=7)$ | $\beta_f(i=7)$ | $\beta_s(i=7)$ | |
| 8 | $\alpha_1(L=8)$ | $\alpha_2(L=8)$ | $\alpha_3(L=8)$ | $\alpha_4(L=8)$ | $\beta_f(i=8)$ | $\beta_s(i=8)$ | |
| 9 | $\alpha_1(L=9)$ | $\alpha_2(L=9)$ | $\alpha_3(L=9)$ | $\alpha_4(L=9)$ | $\beta_f(i=9)$ | $\beta_s(i=9)$ | |
| 10 | $\alpha_1(L=10)$ | $\alpha_2(L=10)$ | $\alpha_3(L=10)$ | $\alpha_4(L=10)$ | $\beta_f(i=10)$ | $\beta_s(i=10)$ | $\gamma_0(i=10)$ |
| 11 | $\alpha_1(L=11)$ | $\alpha_2(L=11)$ | $\alpha_3(L=11)$ | $\alpha_4(L=11)$ | $\beta_f(i=11)$ | $\beta_s(i=11)$ | $\gamma_0(i=11)$ |
| 12 | $\alpha_1(L=12)$ | $\alpha_2(L=12)$ | $\alpha_3(L=12)$ | $\alpha_4(L=12)$ | $\beta_f(i=12)$ | $\beta_s(i=12)$ | $\gamma_0(i=12)$ |
| 13 | $\alpha_1(L=13)$ | $\alpha_2(L=13)$ | $\alpha_3(L=13)$ | $\alpha_4(L=13)$ | $\beta_f(i=13)$ | $\beta_s(i=13)$ | $\gamma_0(i=13)$ |
| 14 | $\alpha_1(L=14)$ | $\alpha_2(L=14)$ | $\alpha_3(L=14)$ | $\alpha_4(L=14)$ | $\beta_f(i=14)$ | $\beta_s(i=14)$ | $\gamma_0(i=14)$ |
| 15 | $\alpha_1(L=15)$ | $\alpha_2(L=15)$ | $\alpha_3(L=15)$ | $\alpha_4(L=15)$ | $\beta_f(i=15)$ | $\beta_s(i=15)$ | $\gamma_0(i=15)$ |
| 16 | $\alpha_1(L=16)$ | $\alpha_2(L=16)$ | $\alpha_3(L=16)$ | $\alpha_4(L=16)$ | $\beta_f(i=16)$ | $\beta_s(i=16)$ | $\gamma_0(i=16)$ |
| 17 | $\alpha_1(L=17)$ | $\alpha_2(L=17)$ | $\alpha_3(L=17)$ | $\alpha_4(L=17)$ | $\beta_f(i=17)$ | $\beta_s(i=17)$ | $\gamma_0(i=17)$ |
| 18 | $\alpha_1(L=18)$ | $\alpha_2(L=18)$ | $\alpha_3(L=18)$ | $\alpha_4(L=18)$ | $\beta_f(i=18)$ | $\beta_s(i=18)$ | $\gamma_0(i=18)$ |
| 19 | $\alpha_1(L=19)$ | $\alpha_2(L=19)$ | $\alpha_3(L=19)$ | $\alpha_4(L=19)$ | $\beta_f(i=19)$ | $\beta_s(i=19)$ | $\gamma_0(i=19)$ |
| 20 | $\alpha_1(L=20)$ | $\alpha_2(L=20)$ | $\alpha_3(L=20)$ | $\alpha_4(L=20)$ | $\beta_f(i=20)$ | $\beta_s(i=20)$ | $\gamma_0(i=20)$ |
| 21 | $\alpha_1(L=21)$ | $\alpha_2(L=21)$ | $\alpha_3(L=21)$ | $\alpha_4(L=21)$ | $\beta_f(i=21)$ | $\beta_s(i=21)$ | $\gamma_0(i=21)$ |
| 22 | $\alpha_1(L=22)$ | $\alpha_2(L=22)$ | $\alpha_3(L=22)$ | $\alpha_4(L=22)$ | $\beta_f(i=22)$ | $\beta_s(i=22)$ | $\gamma_0(i=22)$ |
| 23 | $\alpha_1(L=23)$ | $\alpha_2(L=23)$ | $\alpha_3(L=23)$ | $\alpha_4(L=23)$ | $\beta_f(i=23)$ | $\beta_s(i=23)$ | $\gamma_0(i=23)$ |
| 24 | $\alpha_1(L=24)$ | $\alpha_2(L=24)$ | $\alpha_3(L=24)$ | $\alpha_4(L=24)$ | $\beta_f(i=24)$ | $\beta_s(i=24)$ | $\gamma_0(i=24)$ |
| 25 | $\alpha_1(L=25)$ | $\alpha_2(L=25)$ | $\alpha_3(L=25)$ | $\alpha_4(L=25)$ | $\beta_f(i=25)$ | $\beta_s(i=25)$ | $\gamma_0(i=25)$ |
| 26 | $\alpha_1(L=26)$ | $\alpha_2(L=26)$ | $\alpha_3(L=26)$ | $\alpha_4(L=26)$ | δ | | |
| 27 | $\alpha_1(L=27)$ | $\alpha_2(L=27)$ | $\alpha_3(L=27)$ | $\alpha_4(L=27)$ | κ_D/κ | | |
| 28 | $\alpha_1(L=28)$ | $\alpha_2(L=28)$ | $\alpha_3(L=28)$ | $\alpha_4(L=28)$ | | | |
| 29 | $\alpha_1(L=29)$ | $\alpha_2(L=29)$ | $\alpha_3(L=29)$ | $\alpha_4(L=29)$ | | | |
| 30 | $\alpha_1(L=30)$ | $\alpha_2(L=30)$ | $\alpha_3(L=30)$ | $\alpha_4(L=30)$ | | | |
| 31 | $\alpha_1(L=31)$ | $\alpha_2(L=31)$ | $\alpha_3(L=31)$ | $\alpha_4(L=31)$ | | | |
| 32 | $\alpha_1(L=32)$ | $\alpha_2(L=32)$ | $\alpha_3(L=32)$ | $\alpha_4(L=32)$ | | | |
| 33 | $\alpha_1(L=33)$ | $\alpha_2(L=33)$ | $\alpha_3(L=33)$ | $\alpha_4(L=33)$ | | | |
| 34 | $\alpha_1(L=34)$ | $\alpha_2(L=34)$ | $\alpha_3(L=34)$ | $\alpha_4(L=34)$ | | | |
| ... | ... | ... | ... | ... | ... | ... | |
| 65 | $\alpha_1(L=65)$ | $\alpha_2(L=65)$ | $\alpha_3(L=65)$ | $\alpha_4(L=65)$ | | | |
| 66 | $\alpha_1(L=66)$ | $\alpha_2(L=66)$ | $\alpha_3(L=66)$ | $\alpha_4(L=66)$ | | | |

- NOTE: (1,26) $\neq 0$ is flag for O₂ channel
- α is defined as follows

$$\alpha(L) = (\alpha_1(L) + \alpha_2(L) \cdot q + \alpha_3(L) \cdot q^2 + \alpha_4(L) \cdot q^3) \cdot [1 + (\sec(\theta) - 1) \cdot \delta] \quad (12.1)$$

where $q = T_{std}(L)/T(L) - 1$

- β is defined as follows

$$\beta(L) = (\beta_s(T) \cdot p_{dry} + \beta_f(T) \cdot p_{H_2O}) \cdot \sec(\theta) \quad (12.2)$$

Table 12.2: MIT RTA file format for oxygen channels with magnetic correction

| ABGCFOX3: Oxygen Channel w/ magnetic correction (AMSU 14) | | | | | | | |
|---|------------------|------------------|------------------|------------------|-------------------|------------------|------------------|
| see text for definitions of coefficients | | | | | | | |
| L | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 1 | $\alpha_1(L=1)$ | $\alpha_2(L=1)$ | $\alpha_3(L=1)$ | $\alpha_4(L=1)$ | $\beta_f(i=1)$ | $\alpha_5(L=1)$ | $\alpha_6(L=1)$ |
| 2 | $\alpha_1(L=2)$ | $\alpha_2(L=2)$ | $\alpha_3(L=2)$ | $\alpha_4(L=2)$ | $\beta_f(i=2)$ | $\alpha_5(L=2)$ | $\alpha_6(L=2)$ |
| 3 | $\alpha_1(L=3)$ | $\alpha_2(L=3)$ | $\alpha_3(L=3)$ | $\alpha_4(L=3)$ | $\beta_f(i=3)$ | $\alpha_5(L=3)$ | $\alpha_6(L=3)$ |
| 4 | $\alpha_1(L=4)$ | $\alpha_2(L=4)$ | $\alpha_3(L=4)$ | $\alpha_4(L=4)$ | $\beta_f(i=4)$ | $\alpha_5(L=4)$ | $\alpha_6(L=4)$ |
| 5 | $\alpha_1(L=5)$ | $\alpha_2(L=5)$ | $\alpha_3(L=5)$ | $\alpha_4(L=5)$ | $\beta_f(i=5)$ | $\alpha_5(L=5)$ | $\alpha_6(L=5)$ |
| 6 | $\alpha_1(L=6)$ | $\alpha_2(L=6)$ | $\alpha_3(L=6)$ | $\alpha_4(L=6)$ | $\beta_f(i=6)$ | $\alpha_5(L=6)$ | $\alpha_6(L=6)$ |
| 7 | $\alpha_1(L=7)$ | $\alpha_2(L=7)$ | $\alpha_3(L=7)$ | $\alpha_4(L=7)$ | $\beta_f(i=7)$ | $\alpha_5(L=7)$ | $\alpha_6(L=7)$ |
| 8 | $\alpha_1(L=8)$ | $\alpha_2(L=8)$ | $\alpha_3(L=8)$ | $\alpha_4(L=8)$ | $\beta_f(i=8)$ | $\alpha_5(L=8)$ | $\alpha_6(L=8)$ |
| 9 | $\alpha_1(L=9)$ | $\alpha_2(L=9)$ | $\alpha_3(L=9)$ | $\alpha_4(L=9)$ | $\beta_f(i=9)$ | $\alpha_5(L=9)$ | $\alpha_6(L=9)$ |
| 10 | $\alpha_1(L=10)$ | $\alpha_2(L=10)$ | $\alpha_3(L=10)$ | $\alpha_4(L=10)$ | $\beta_f(i=10)$ | $\alpha_5(L=10)$ | $\alpha_6(L=10)$ |
| 11 | $\alpha_1(L=11)$ | $\alpha_2(L=11)$ | $\alpha_3(L=11)$ | $\alpha_4(L=11)$ | $\beta_f(i=11)$ | $\alpha_5(L=11)$ | $\alpha_6(L=11)$ |
| 12 | $\alpha_1(L=12)$ | $\alpha_2(L=12)$ | $\alpha_3(L=12)$ | $\alpha_4(L=12)$ | $\beta_f(i=12)$ | $\alpha_5(L=12)$ | $\alpha_6(L=12)$ |
| 13 | $\alpha_1(L=13)$ | $\alpha_2(L=13)$ | $\alpha_3(L=13)$ | $\alpha_4(L=13)$ | $\beta_f(i=13)$ | $\alpha_5(L=13)$ | $\alpha_6(L=13)$ |
| 14 | $\alpha_1(L=14)$ | $\alpha_2(L=14)$ | $\alpha_3(L=14)$ | $\alpha_4(L=14)$ | $\beta_f(i=14)$ | $\alpha_5(L=14)$ | $\alpha_6(L=14)$ |
| 15 | $\alpha_1(L=15)$ | $\alpha_2(L=15)$ | $\alpha_3(L=15)$ | $\alpha_4(L=15)$ | $\beta_f(i=15)$ | $\alpha_5(L=15)$ | $\alpha_6(L=15)$ |
| 16 | $\alpha_1(L=16)$ | $\alpha_2(L=16)$ | $\alpha_3(L=16)$ | $\alpha_4(L=16)$ | $\beta_f(i=16)$ | $\alpha_5(L=16)$ | $\alpha_6(L=16)$ |
| 17 | $\alpha_1(L=17)$ | $\alpha_2(L=17)$ | $\alpha_3(L=17)$ | $\alpha_4(L=17)$ | $\beta_f(i=17)$ | $\alpha_5(L=17)$ | $\alpha_6(L=17)$ |
| 18 | $\alpha_1(L=18)$ | $\alpha_2(L=18)$ | $\alpha_3(L=18)$ | $\alpha_4(L=18)$ | $\beta_f(i=18)$ | $\alpha_5(L=18)$ | $\alpha_6(L=18)$ |
| 19 | $\alpha_1(L=19)$ | $\alpha_2(L=19)$ | $\alpha_3(L=19)$ | $\alpha_4(L=19)$ | $\beta_f(i=19)$ | $\alpha_5(L=19)$ | $\alpha_6(L=19)$ |
| 20 | $\alpha_1(L=20)$ | $\alpha_2(L=20)$ | $\alpha_3(L=20)$ | $\alpha_4(L=20)$ | $\beta_f(i=20)$ | $\alpha_5(L=20)$ | $\alpha_6(L=20)$ |
| 21 | $\alpha_1(L=21)$ | $\alpha_2(L=21)$ | $\alpha_3(L=21)$ | $\alpha_4(L=21)$ | $\beta_f(i=21)$ | $\alpha_5(L=21)$ | $\alpha_6(L=21)$ |
| 22 | $\alpha_1(L=22)$ | $\alpha_2(L=22)$ | $\alpha_3(L=22)$ | $\alpha_4(L=22)$ | $\beta_f(i=22)$ | $\alpha_5(L=22)$ | $\alpha_6(L=22)$ |
| 23 | $\alpha_1(L=23)$ | $\alpha_2(L=23)$ | $\alpha_3(L=23)$ | $\alpha_4(L=23)$ | $\beta_f(i=23)$ | $\alpha_5(L=23)$ | $\alpha_6(L=23)$ |
| 24 | $\alpha_1(L=24)$ | $\alpha_2(L=24)$ | $\alpha_3(L=24)$ | $\alpha_4(L=24)$ | $\beta_f(i=24)$ | $\alpha_5(L=24)$ | $\alpha_6(L=24)$ |
| 25 | $\alpha_1(L=25)$ | $\alpha_2(L=25)$ | $\alpha_3(L=25)$ | $\alpha_4(L=25)$ | $\beta_f(i=25)$ | $\alpha_5(L=25)$ | $\alpha_6(L=25)$ |
| 26 | $\alpha_1(L=26)$ | $\alpha_2(L=26)$ | $\alpha_3(L=26)$ | $\alpha_4(L=26)$ | δ | $\alpha_5(L=26)$ | $\alpha_6(L=26)$ |
| 27 | $\alpha_1(L=27)$ | $\alpha_2(L=27)$ | $\alpha_3(L=27)$ | $\alpha_4(L=27)$ | κ_D/κ | $\alpha_5(L=27)$ | $\alpha_6(L=27)$ |
| 28 | $\alpha_1(L=28)$ | $\alpha_2(L=28)$ | $\alpha_3(L=28)$ | $\alpha_4(L=28)$ | | $\alpha_5(L=28)$ | $\alpha_6(L=28)$ |
| 29 | $\alpha_1(L=29)$ | $\alpha_2(L=29)$ | $\alpha_3(L=29)$ | $\alpha_4(L=29)$ | | $\alpha_5(L=29)$ | $\alpha_6(L=29)$ |
| 30 | $\alpha_1(L=30)$ | $\alpha_2(L=30)$ | $\alpha_3(L=30)$ | $\alpha_4(L=30)$ | | $\alpha_5(L=30)$ | $\alpha_6(L=30)$ |
| 31 | $\alpha_1(L=31)$ | $\alpha_2(L=31)$ | $\alpha_3(L=31)$ | $\alpha_4(L=31)$ | | $\alpha_5(L=31)$ | $\alpha_6(L=31)$ |
| 32 | $\alpha_1(L=32)$ | $\alpha_2(L=32)$ | $\alpha_3(L=32)$ | $\alpha_4(L=32)$ | | $\alpha_5(L=32)$ | $\alpha_6(L=32)$ |
| 33 | $\alpha_1(L=33)$ | $\alpha_2(L=33)$ | $\alpha_3(L=33)$ | $\alpha_4(L=33)$ | | $\alpha_5(L=33)$ | $\alpha_6(L=33)$ |
| 34 | $\alpha_1(L=34)$ | $\alpha_2(L=34)$ | $\alpha_3(L=34)$ | $\alpha_4(L=34)$ | | $\alpha_5(L=34)$ | $\alpha_6(L=34)$ |
| ... | ... | ... | ... | ... | ... | ... | ... |
| 65 | $\alpha_1(L=65)$ | $\alpha_2(L=65)$ | $\alpha_3(L=65)$ | $\alpha_4(L=65)$ | | $\alpha_5(L=65)$ | $\alpha_6(L=65)$ |
| 66 | $\alpha_1(L=66)$ | $\alpha_2(L=66)$ | $\alpha_3(L=66)$ | $\alpha_4(L=66)$ | | $\alpha_5(L=66)$ | $\alpha_6(L=66)$ |

- $(1,26) \neq 0$ is flag for O₂ channel and $(6,09) \neq 0$ is flag for magnetic channel
- α is defined as

$$\alpha(L) = (\alpha_1(L) + \alpha_2(L) \cdot q + \alpha_3(L) \cdot q^2 + \alpha_4(L) \cdot q^3) \cdot [1 + (\sec(\theta) - 1) \cdot \delta] \quad (12.3)$$

$$\cdot [1 + B^2 (\alpha_5 + \alpha_6 \cdot \cos^2(\theta)) / \alpha_1] \quad (12.4)$$

where $q = T_{std}(L)/T(L) - 1$

- β is defined as

$$\beta(L) = 0 \quad (12.5)$$

Table 12.3: MIT RTA file format for water channels
 ABGCFV8: Water Channel (AMSU 1, 18,19,20)
 see text for definitions of coefficients

| L | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|-----|------------------|-----------|------------------|-------------|-------------------|-----------------|------------------|
| 1 | $\alpha_0(i=1)$ | $S(i=1)$ | $w_D(i=1)$ | $w_S(i=1)$ | $\beta_f(i=1)$ | $\beta_s(i=1)$ | |
| 2 | $\alpha_0(i=2)$ | $S(i=2)$ | $w_D(i=2)$ | $w_S(i=2)$ | $\beta_f(i=2)$ | $\beta_s(i=2)$ | |
| 3 | $\alpha_0(i=3)$ | $S(i=3)$ | $w_D(i=3)$ | $w_S(i=3)$ | $\beta_f(i=3)$ | $\beta_s(i=3)$ | |
| 4 | $\alpha_0(i=4)$ | $S(i=4)$ | $w_D(i=4)$ | $w_S(i=4)$ | $\beta_f(i=4)$ | $\beta_s(i=4)$ | |
| 5 | $\alpha_0(i=5)$ | $S(i=5)$ | $w_D(i=5)$ | $w_S(i=5)$ | $\beta_f(i=5)$ | $\beta_s(i=5)$ | |
| 6 | $\alpha_0(i=6)$ | $S(i=6)$ | $w_D(i=6)$ | $w_S(i=6)$ | $\beta_f(i=6)$ | $\beta_s(i=6)$ | |
| 7 | $\alpha_0(i=7)$ | $S(i=7)$ | $w_D(i=7)$ | $w_S(i=7)$ | $\beta_f(i=7)$ | $\beta_s(i=7)$ | |
| 8 | $\alpha_0(i=8)$ | $S(i=8)$ | $w_D(i=8)$ | $w_S(i=8)$ | $\beta_f(i=8)$ | $\beta_s(i=8)$ | |
| 9 | $\alpha_0(i=9)$ | $S(i=9)$ | $w_D(i=9)$ | $w_S(i=9)$ | $\beta_f(i=9)$ | $\beta_s(i=9)$ | |
| 10 | $\alpha_0(i=10)$ | $S(i=10)$ | $w_D(i=10)$ | $w_S(i=10)$ | $\beta_f(i=10)$ | $\beta_s(i=10)$ | $\gamma_0(i=10)$ |
| 11 | $\alpha_0(i=11)$ | $S(i=11)$ | $w_D(i=11)$ | $w_S(i=11)$ | $\beta_f(i=11)$ | $\beta_s(i=11)$ | $\gamma_0(i=11)$ |
| 12 | $\alpha_0(i=12)$ | $S(i=12)$ | $w_D(i=12)$ | $w_S(i=12)$ | $\beta_f(i=12)$ | $\beta_s(i=12)$ | $\gamma_0(i=12)$ |
| 13 | $\alpha_0(i=13)$ | $S(i=13)$ | $w_D(i=13)$ | $w_S(i=13)$ | $\beta_f(i=13)$ | $\beta_s(i=13)$ | $\gamma_0(i=13)$ |
| 14 | $\alpha_0(i=14)$ | $S(i=14)$ | $w_D(i=14)$ | $w_S(i=14)$ | $\beta_f(i=14)$ | $\beta_s(i=14)$ | $\gamma_0(i=14)$ |
| 15 | $\alpha_0(i=15)$ | $S(i=15)$ | $w_D(i=15)$ | $w_S(i=15)$ | $\beta_f(i=15)$ | $\beta_s(i=15)$ | $\gamma_0(i=15)$ |
| 16 | $\alpha_0(i=16)$ | $S(i=16)$ | $w_D(i=16)$ | $w_S(i=16)$ | $\beta_f(i=16)$ | $\beta_s(i=16)$ | $\gamma_0(i=16)$ |
| 17 | $\alpha_0(i=17)$ | $S(i=17)$ | $w_D(i=17)$ | $w_S(i=17)$ | $\beta_f(i=17)$ | $\beta_s(i=17)$ | $\gamma_0(i=17)$ |
| 18 | $\alpha_0(i=18)$ | $S(i=18)$ | $w_D(i=18)$ | $w_S(i=18)$ | $\beta_f(i=18)$ | $\beta_s(i=18)$ | $\gamma_0(i=18)$ |
| 19 | $\alpha_0(i=19)$ | $S(i=19)$ | $w_D(i=19)$ | $w_S(i=19)$ | $\beta_f(i=19)$ | $\beta_s(i=19)$ | $\gamma_0(i=19)$ |
| 20 | $\alpha_0(i=20)$ | $S(i=20)$ | $w_D(i=20)$ | $w_S(i=20)$ | $\beta_f(i=20)$ | $\beta_s(i=20)$ | $\gamma_0(i=20)$ |
| 21 | $\alpha_0(i=21)$ | $S(i=21)$ | $w_D(i=21)$ | $w_S(i=21)$ | $\beta_f(i=21)$ | $\beta_s(i=21)$ | $\gamma_0(i=21)$ |
| 22 | $\alpha_0(i=22)$ | $S(i=22)$ | $w_D(i=22)$ | $w_S(i=22)$ | $\beta_f(i=22)$ | $\beta_s(i=22)$ | $\gamma_0(i=22)$ |
| 23 | $\alpha_0(i=23)$ | $S(i=23)$ | $w_D(i=23)$ | $w_S(i=23)$ | $\beta_f(i=23)$ | $\beta_s(i=23)$ | $\gamma_0(i=23)$ |
| 24 | $\alpha_0(i=24)$ | $S(i=24)$ | $w_D(i=24)$ | $w_S(i=24)$ | $\beta_f(i=24)$ | $\beta_s(i=24)$ | $\gamma_0(i=24)$ |
| 25 | $\alpha_0(i=25)$ | $S(i=25)$ | $w_D(i=25)$ | $w_S(i=25)$ | $\beta_f(i=25)$ | $\beta_s(i=25)$ | $\gamma_0(i=25)$ |
| 26 | | | ν_0 | | | | |
| 27 | | | $\Delta\nu(j=1)$ | | κ_D/κ | | |
| 28 | | | $\Delta\nu(j=2)$ | | | | |
| 29 | | | $\Delta\nu(j=3)$ | | | | |
| 30 | | | $\Delta\nu(j=4)$ | | | | |
| 31 | | | $\Delta\nu(j=5)$ | | | | |
| 32 | | | $\Delta\nu(j=6)$ | | | | |
| 33 | | | | | | | |
| 34 | | | | | | | |
| ... | ... | ... | ... | ... | ... | ... | |
| 65 | | | | | | | |
| 66 | | | | | | | |

- $(1,26) = 0$ and $(3,26) \neq 0$ is flag for NEW H₂O channel.

- α_0 is defined as follows

$$\alpha_0 = 0.293 \cdot T \cdot (\kappa_{O_2} + \kappa_{N_2}) / 800^2 \quad (12.6)$$

- α is defined as follows

$$\alpha(L) = \alpha_0 \cdot \bar{p} \cdot (p_2 - p_1) \cdot \sec(\theta) \quad (12.7)$$

Table 12.4: MIT RTA file format for window channels
 ABGCFW3: Window Channel (AMSU 2, 15, 16, 17)
 see text for definitions of coefficients

| L | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|-----|------------------|-----|-----|-----|-------------------|-----------------|------------------|
| 1 | $\alpha_0(i=1)$ | | | | $\beta_f(i=1)$ | $\beta_s(i=1)$ | |
| 2 | $\alpha_0(i=2)$ | | | | $\beta_f(i=2)$ | $\beta_s(i=2)$ | |
| 3 | $\alpha_0(i=3)$ | | | | $\beta_f(i=3)$ | $\beta_s(i=3)$ | |
| 4 | $\alpha_0(i=4)$ | | | | $\beta_f(i=4)$ | $\beta_s(i=4)$ | |
| 5 | $\alpha_0(i=5)$ | | | | $\beta_f(i=5)$ | $\beta_s(i=5)$ | |
| 6 | $\alpha_0(i=6)$ | | | | $\beta_f(i=6)$ | $\beta_s(i=6)$ | |
| 7 | $\alpha_0(i=7)$ | | | | $\beta_f(i=7)$ | $\beta_s(i=7)$ | |
| 8 | $\alpha_0(i=8)$ | | | | $\beta_f(i=8)$ | $\beta_s(i=8)$ | |
| 9 | $\alpha_0(i=9)$ | | | | $\beta_f(i=9)$ | $\beta_s(i=9)$ | |
| 10 | $\alpha_0(i=10)$ | | | | $\beta_f(i=10)$ | $\beta_s(i=10)$ | $\gamma_0(i=10)$ |
| 11 | $\alpha_0(i=11)$ | | | | $\beta_f(i=11)$ | $\beta_s(i=11)$ | $\gamma_0(i=11)$ |
| 12 | $\alpha_0(i=12)$ | | | | $\beta_f(i=12)$ | $\beta_s(i=12)$ | $\gamma_0(i=12)$ |
| 13 | $\alpha_0(i=13)$ | | | | $\beta_f(i=13)$ | $\beta_s(i=13)$ | $\gamma_0(i=13)$ |
| 14 | $\alpha_0(i=14)$ | | | | $\beta_f(i=14)$ | $\beta_s(i=14)$ | $\gamma_0(i=14)$ |
| 15 | $\alpha_0(i=15)$ | | | | $\beta_f(i=15)$ | $\beta_s(i=15)$ | $\gamma_0(i=15)$ |
| 16 | $\alpha_0(i=16)$ | | | | $\beta_f(i=16)$ | $\beta_s(i=16)$ | $\gamma_0(i=16)$ |
| 17 | $\alpha_0(i=17)$ | | | | $\beta_f(i=17)$ | $\beta_s(i=17)$ | $\gamma_0(i=17)$ |
| 18 | $\alpha_0(i=18)$ | | | | $\beta_f(i=18)$ | $\beta_s(i=18)$ | $\gamma_0(i=18)$ |
| 19 | $\alpha_0(i=19)$ | | | | $\beta_f(i=19)$ | $\beta_s(i=19)$ | $\gamma_0(i=19)$ |
| 20 | $\alpha_0(i=20)$ | | | | $\beta_f(i=20)$ | $\beta_s(i=20)$ | $\gamma_0(i=20)$ |
| 21 | $\alpha_0(i=21)$ | | | | $\beta_f(i=21)$ | $\beta_s(i=21)$ | $\gamma_0(i=21)$ |
| 22 | $\alpha_0(i=22)$ | | | | $\beta_f(i=22)$ | $\beta_s(i=22)$ | $\gamma_0(i=22)$ |
| 23 | $\alpha_0(i=23)$ | | | | $\beta_f(i=23)$ | $\beta_s(i=23)$ | $\gamma_0(i=23)$ |
| 24 | $\alpha_0(i=24)$ | | | | $\beta_f(i=24)$ | $\beta_s(i=24)$ | $\gamma_0(i=24)$ |
| 25 | $\alpha_0(i=25)$ | | | | $\beta_f(i=25)$ | $\beta_s(i=25)$ | $\gamma_0(i=25)$ |
| 26 | | | | | | | |
| 27 | | | | | κ_D/κ | | |
| 28 | | | | | | | |
| 29 | | | | | | | |
| 30 | | | | | | | |
| 31 | | | | | | | |
| 32 | | | | | | | |
| 33 | | | | | | | |
| 34 | | | | | | | |
| ... | ... | ... | ... | ... | ... | ... | |
| 65 | | | | | | | |
| 66 | | | | | | | |

- (1,26) = 0 AND (3,26) = 0 is flag for window channel

- α is defined as follows

$$\alpha(L) = \alpha_0 \cdot \bar{p} \cdot (p_2 - p_1) \cdot \sec(\theta) \quad (12.8)$$

- β is defined as follows

$$\beta(L) = (\beta_s(T) \cdot p_{dry} + \beta_f(T) \cdot p_{H_2O}) \cdot \sec(\theta) \quad (12.9)$$

Table 12.5: MIT RTA file format for water channels (old format)
see text for definitions of coefficients

| L | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|-----|------------------|-----------------|-----------------|-----------------|-----------------|-------------------------------|------------------|
| 1 | $\alpha_0(i=1)$ | $\beta_1(L=1)$ | $\beta_2(L=1)$ | $\beta_3(L=1)$ | $\beta_4(L=1)$ | $\partial B/\partial S(i=1)$ | |
| 2 | $\alpha_0(i=2)$ | $\beta_1(L=2)$ | $\beta_2(L=2)$ | $\beta_3(L=2)$ | $\beta_4(L=2)$ | $\partial B/\partial S(i=2)$ | |
| 3 | $\alpha_0(i=3)$ | $\beta_1(L=3)$ | $\beta_2(L=3)$ | $\beta_3(L=3)$ | $\beta_4(L=3)$ | $\partial B/\partial S(i=3)$ | |
| 4 | $\alpha_0(i=4)$ | $\beta_1(L=4)$ | $\beta_2(L=4)$ | $\beta_3(L=4)$ | $\beta_4(L=4)$ | $\partial B/\partial S(i=4)$ | |
| 5 | $\alpha_0(i=5)$ | $\beta_1(L=5)$ | $\beta_2(L=5)$ | $\beta_3(L=5)$ | $\beta_4(L=5)$ | $\partial B/\partial S(i=5)$ | |
| 6 | $\alpha_0(i=6)$ | $\beta_1(L=6)$ | $\beta_2(L=6)$ | $\beta_3(L=6)$ | $\beta_4(L=6)$ | $\partial B/\partial S(i=6)$ | |
| 7 | $\alpha_0(i=7)$ | $\beta_1(L=7)$ | $\beta_2(L=7)$ | $\beta_3(L=7)$ | $\beta_4(L=7)$ | $\partial B/\partial S(i=7)$ | |
| 8 | $\alpha_0(i=8)$ | $\beta_1(L=8)$ | $\beta_2(L=8)$ | $\beta_3(L=8)$ | $\beta_4(L=8)$ | $\partial B/\partial S(i=8)$ | |
| 9 | $\alpha_0(i=9)$ | $\beta_1(L=9)$ | $\beta_2(L=9)$ | $\beta_3(L=9)$ | $\beta_4(L=9)$ | $\partial B/\partial S(i=9)$ | |
| 10 | $\alpha_0(i=10)$ | $\beta_1(L=10)$ | $\beta_2(L=10)$ | $\beta_3(L=10)$ | $\beta_4(L=10)$ | $\partial B/\partial S(i=10)$ | $\gamma_0(i=10)$ |
| 11 | $\alpha_0(i=11)$ | $\beta_1(L=11)$ | $\beta_2(L=11)$ | $\beta_3(L=11)$ | $\beta_4(L=11)$ | $\partial B/\partial S(i=11)$ | $\gamma_0(i=11)$ |
| 12 | $\alpha_0(i=12)$ | $\beta_1(L=12)$ | $\beta_2(L=12)$ | $\beta_3(L=12)$ | $\beta_4(L=12)$ | $\partial B/\partial S(i=12)$ | $\gamma_0(i=12)$ |
| 13 | $\alpha_0(i=13)$ | $\beta_1(L=13)$ | $\beta_2(L=13)$ | $\beta_3(L=13)$ | $\beta_4(L=13)$ | $\partial B/\partial S(i=13)$ | $\gamma_0(i=13)$ |
| 14 | $\alpha_0(i=14)$ | $\beta_1(L=14)$ | $\beta_2(L=14)$ | $\beta_3(L=14)$ | $\beta_4(L=14)$ | $\partial B/\partial S(i=14)$ | $\gamma_0(i=14)$ |
| 15 | $\alpha_0(i=15)$ | $\beta_1(L=15)$ | $\beta_2(L=15)$ | $\beta_3(L=15)$ | $\beta_4(L=15)$ | $\partial B/\partial S(i=15)$ | $\gamma_0(i=15)$ |
| 16 | $\alpha_0(i=16)$ | $\beta_1(L=16)$ | $\beta_2(L=16)$ | $\beta_3(L=16)$ | $\beta_4(L=16)$ | $\partial B/\partial S(i=16)$ | $\gamma_0(i=16)$ |
| 17 | $\alpha_0(i=17)$ | $\beta_1(L=17)$ | $\beta_2(L=17)$ | $\beta_3(L=17)$ | $\beta_4(L=17)$ | $\partial B/\partial S(i=17)$ | $\gamma_0(i=17)$ |
| 18 | $\alpha_0(i=18)$ | $\beta_1(L=18)$ | $\beta_2(L=18)$ | $\beta_3(L=18)$ | $\beta_4(L=18)$ | $\partial B/\partial S(i=18)$ | $\gamma_0(i=18)$ |
| 19 | $\alpha_0(i=19)$ | $\beta_1(L=19)$ | $\beta_2(L=19)$ | $\beta_3(L=19)$ | $\beta_4(L=19)$ | $\partial B/\partial S(i=19)$ | $\gamma_0(i=19)$ |
| 20 | $\alpha_0(i=20)$ | $\beta_1(L=20)$ | $\beta_2(L=20)$ | $\beta_3(L=20)$ | $\beta_4(L=20)$ | $\partial B/\partial S(i=20)$ | $\gamma_0(i=20)$ |
| 21 | $\alpha_0(i=21)$ | $\beta_1(L=21)$ | $\beta_2(L=21)$ | $\beta_3(L=21)$ | $\beta_4(L=21)$ | $\partial B/\partial S(i=21)$ | $\gamma_0(i=21)$ |
| 22 | $\alpha_0(i=22)$ | $\beta_1(L=22)$ | $\beta_2(L=22)$ | $\beta_3(L=22)$ | $\beta_4(L=22)$ | $\partial B/\partial S(i=22)$ | $\gamma_0(i=22)$ |
| 23 | $\alpha_0(i=23)$ | $\beta_1(L=23)$ | $\beta_2(L=23)$ | $\beta_3(L=23)$ | $\beta_4(L=23)$ | $\partial B/\partial S(i=23)$ | $\gamma_0(i=23)$ |
| 24 | $\alpha_0(i=24)$ | $\beta_1(L=24)$ | $\beta_2(L=24)$ | $\beta_3(L=24)$ | $\beta_4(L=24)$ | $\partial B/\partial S(i=24)$ | $\gamma_0(i=24)$ |
| 25 | $\alpha_0(i=25)$ | $\beta_1(L=25)$ | $\beta_2(L=25)$ | $\beta_3(L=25)$ | $\beta_4(L=25)$ | $\partial B/\partial S(i=25)$ | $\gamma_0(i=25)$ |
| 26 | | $\beta_1(L=26)$ | $\beta_2(L=26)$ | $\beta_3(L=26)$ | $\beta_4(L=26)$ | | |
| 27 | | $\beta_1(L=27)$ | $\beta_2(L=27)$ | $\beta_3(L=27)$ | $\beta_4(L=27)$ | | |
| 28 | | $\beta_1(L=28)$ | $\beta_2(L=28)$ | $\beta_3(L=28)$ | $\beta_4(L=28)$ | | |
| 29 | | $\beta_1(L=29)$ | $\beta_2(L=29)$ | $\beta_3(L=29)$ | $\beta_4(L=29)$ | | |
| 30 | | $\beta_1(L=30)$ | $\beta_2(L=30)$ | $\beta_3(L=30)$ | $\beta_4(L=30)$ | | |
| 31 | | $\beta_1(L=31)$ | $\beta_2(L=31)$ | $\beta_3(L=31)$ | $\beta_4(L=31)$ | | |
| 32 | | $\beta_1(L=32)$ | $\beta_2(L=32)$ | $\beta_3(L=32)$ | $\beta_4(L=32)$ | | |
| 33 | | $\beta_1(L=33)$ | $\beta_2(L=33)$ | $\beta_3(L=33)$ | $\beta_4(L=33)$ | | |
| 34 | | $\beta_1(L=34)$ | $\beta_2(L=34)$ | $\beta_3(L=34)$ | $\beta_4(L=34)$ | | |
| ... | ... | ... | ... | ... | ... | ... | |
| 65 | | $\beta_1(L=65)$ | $\beta_2(L=65)$ | $\beta_3(L=65)$ | $\beta_4(L=65)$ | | |
| 66 | | $\beta_1(L=66)$ | $\beta_2(L=66)$ | $\beta_3(L=66)$ | $\beta_4(L=66)$ | | |

- The parameters(1,26) = 0 and (2,26) \neq 0 is a flag for OLD H₂O channel format. Also, the down-welling secant ratio factor, κ_D/κ , is not specified for this type, assume a value of 0

- α is defined as follows

$$\alpha(L) = \alpha_0 \cdot \bar{p} \cdot (p_2 - p_1) \cdot \sec(\theta) \quad (12.10)$$

- β is defined as follows

$$\beta(L) = \beta_1(L) + \beta_2(L) \cdot q + \beta_3(L) \cdot q^2 + \beta_4(L) \cdot q^3 + \frac{\partial \beta}{\partial S} \cdot PV \quad (12.11)$$

12.4 The UMBC RTA

This version of the rapid transmittance algorithm (RTA) implementation separates the state variables ($T(p)$, $q(p)$, $O_3(p)$, etc.) from RTA variables (coefficients, predictors, flags, etc.). The calling sequences are intended to be RTA independent so that the simulator code, retrieval code, and off-line diagnostics code are independent of the RTA. For example, the old 66 layer RTA has been interfaced with the RTA independent call structure. We expect that all future RTA's to be "plug-in" compatible with this delivery.

The radiance simulator and retrieval code are completely upward and downward compatible with many other RTA's. A single variable, `iphys`, is used within the code to select and/or verify that the RTA is loaded correctly and is consistent with physics or assumptions (both RTA and radiative transfer (RT)). The current levels of `iphys` currently supported within the IOSSPDT routines are

1. early version of the RTA without topography
2. version used for the AIRS Science Team track data (DA,DB,DC, NA,NB, NC, 66 level)
3. same as # 2, except solar reflection term NOT divided by pi
4. 12/96 version (v2a) of the UMBC algorithm for AIRS. Secant angle is not altitude dependent, no microwave background, solar bidirectional transmittance is a kludged value.
5. 5/98 version of the UMBC algorithm for AIRS + CrIS
6. same as 5, except transmissive and overlapping clouds are allowed (in development)

The RTA variables are broken into two classifications and stored in common blocks. The dimensions, declarations, and coefficients are defined in `rta_stat.com` and all variables that are profile dependent (flags, predictors, trace gas column densities, etc.) are stored in `rta_prof.com`. In a vectorized environment the common block `rta_prof` could be converted into a structure and passed from `load_ir` routine to the other routines via the upper level routines. This would violate the upward and downward compatibility of the code and require that the calling routines know the dimensionality of the arrays, therefore, the use of the `rta_prof` common block is preferred at this time.

Once the RTA is loaded, the user sets up the secant angles for that profile by calling the routine `setang_ir()`. The trace gas column densities (CO_2 , CH_4 , CO) are set by calling `init_trace()`. All the predictors are then initialized by calling `init_rta()`. After this is done the user can compute the RTA on a channel-by-channel basis with a call to `ir_rta()` for each channel. If the state variables change, then `init_rta` must be re-called. The routines `setang_ir()` and `init_trace()` only need to be called if those state variables have changed. There is a slight dependence of the secant angle to water vapor, which arises because the secant angle is a function of altitude. If the water profile changes substantially the altitude should be recomputed (`comp_alt.F`) and then `setang_ir` should be called.

A list of the files in the UMBC RTA are given below. Some of these files have the same name as the original UMBC files; however, these files are different from the UMBC delivery. **DO NOT their code with GSFC code**

12.4.1 Program to convert UMBC files to the off-line format

Philosophy and Code Implementation (f/ Oct. 22, 1998 AIRS Science Team Presentation by C. Barnet)

- Object oriented approach to variables allows complete separation of the RTA from the retrieval code. Upward and backward compatibility as well as support of multiple instruments is possible.
 - Coefficients, reference profiles, and flags are in static common block - accessible only to the RTA routines.

- Predictors, trace gas column densities, and angles, are in a profile dependent common block. Library routines interface the retrieval with this common block.
 - * Replacing common block with RTA independent structure is possible.
- **CALPAR, now called INIT_RTA:** At any given retrieval step, the atmospheric state (P_s , T_s , $T(L)$, $C_w(L)$, $C_o(L)$, and $C_l(L)$) is used to compute the RTA predictors, $P(k, L)$, for all RTA models. CALPAR brought out layer transmittances whereas IR_RTA brings out level-to-space transmittance.
- **CALTAU, now called IR_RTA or IR_RTA3:** The layer-to-space transmittance, $\tau(\theta, L)$, and the bi-directional solar surface-to-space transmittance, $\tau^{\uparrow\downarrow}(\theta, \theta_{\odot}, p_s)$ is computed on a per-channel basis. In addition, the GSFC/UMBC model of thermal radiance computes the downwelling thermal radiance. IR_RTA3 is equivalent to IR_RTA, but it brings out the transmittance components.
- **MW_RTA & MW_RTA3:** Microwave RTA is interfaced in the same manner; however, it is physically part of a different library.
- **IR_RAD:** Compute the radiance from the transmittances and atmospheric state on a per-channel basis.
- Retrieval program is “unaware” of the specific RTA used for a channel. The retrieval program “requests” the transmittance for a given atmospheric state and the library (linked at compile time) determines the specific details of the RTA model for that channel. There is no interaction between the RTA library and the calling programs except via subroutine calls
- RTA operates on single channels.
 - a call to the RTA prior to the frequency loop to initialize all predictors (`init_trace()` and `init_rta()`)
 - a call to the RTA for each channel to compute level-to-space transmittance (`ir_rta()`)
 - a call to the RT for each channel to compute radiance (`ir_rad`)
- ALL UMBC RTA's (AIRS, CrIS, IASI, MODIS, etc.) are supported from pre-launch (v5) to the most current version. The RTA coefficient file drives the code. An intelligent file loader makes conversion to other instruments completely transparent. 2 files are specified:
 - the RTA file
 - the Solar radiance file

Each instrument still requires a namelist with specific channel selections

- Retrieval code namelists are affected by the RTA in 2 ways
 - Frequencies requested in the retrieval steps must exist. An optional parameter in the pro namelist called `freqfix` will select the closest frequency
 - The cloud averaging table must be created for each RTA coefficient file. The `airsb` retrieval code verifies the name of the RTA in this file (`clouds.nl: cldavgfile`). It is created using `src_util/calc.tbl.F`

| | |
|----------------------|----------------------------------|
| run via namelist | req's compile w/ library link |
| future AIRS RTA's | HIRS |
| IASI | old AIRS 100 layer |
| NAST-I | old AIRS 66 layer |
| CrIS L=0.8, 1.0, 1.2 | OSS |

At this point in time there are 4 executables envisioned:

1. `airsb103.exe` is the executable that runs all the UMBC AIRS, CrIS, IASI, and NAST and all MIT microwave RTA's

2. airsboss.exe is the executable that runs all the AER OSS RTA's with MIT RTA
3. airsb100.exe runs the UMBC v2 100 level RTA w/ MIT RTA
4. airsb66.exe runs the GSFC 66 level RTA for both the microwave and infrared.

The program `cnv_xxx.F` is a stand alone program which was used to convert the UMBC delivery to the RTA independent format. It is located in the directory: `$AIRS_CODE/code/umbc_cnv` which is NOT provided in the normal delivery (it can be requested in a separate ZIP file that is backed up weekly and called `ai_oldcode.zip` or it is resident on `orbit0091:/home/cbarnet/code`). The UMBC file names and directories are hardwired within this program. This program is only provided to document the changes that were made to the UMBC coefficient files. The typical user does not need to compile or use this program.

The program has the following subroutines:

- `cnv_xxx.F` (uses `rta_stat.com`)
 - `load_tr0`: driver for loading UMBC files
 - `rdairsccoef`: loads UMBC coefficient files
 - `rdprof`: loads UMBC standard profiles
 - `writ_tr0`: writes output in IOSSPDT format
 - `softexit`: subroutine to gracefully exit on error

```
UMBC calt1/coef1(#coef,#chl's) --> coef_SW
UMBC coef2(#coef,#chl's) --> coef_MD(
```

```
calpar  FPREDn,WPREDn,OPREDn, COPREDn, CH3PREDn dimension exactly
        remapped these into 1 common set
        FPRED, WPRED, OPRED, etc.
-->  init_trace & init_rta  mapped version of all calpar's
        + allowed for partial derivative to be efficient
```

```
calokw.f   calokw.f
calowp.f   calowp.f
calt1.f    calt23.f
           mapped version of calt1
           + allowed for partial derivative to be efficient
calt7.f    calt29.f
```

See Table A.1 and A.2 for UMBC RTA's for AIRS, IASI, and CrIS. See Appendix A for more details. Some documentation of the individual RTA's is given in `umbc9803/init_rta()`.

- Original deliveries are expanded in `/ide400/cbarnet/airs/RTA/`
- convert programs and documentation is in `/home/cbarnet/code/umbc_cnv`
- final binary RTA files are in `/airs/RTA/`

12.4.2 UMBC library, code/umbc9803

- Makefile

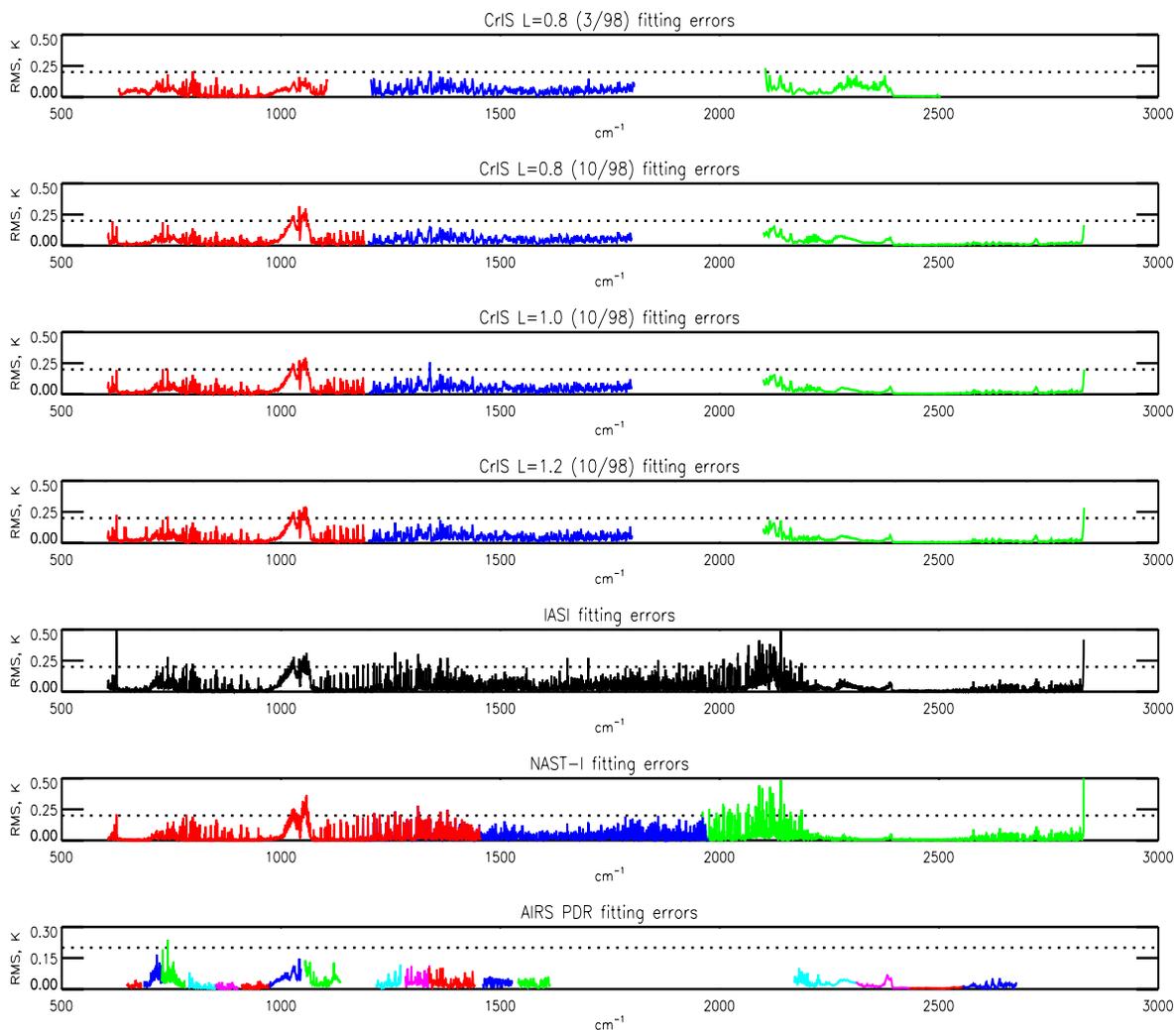


Figure 12.1: RMS RTA fitting errors for all 7 models. The files plotted here are included with this delivery in the file rmserr.zip. Note the error scale on AIRS is smaller (0.3 versus 0.5 on all others)

- rta_stat.com: common block for static variables
- rta_prof.com: common block for profile dependent variables
- load_ir.F: module to load RTA file
- init_rta.F: module to set-up all predictors
- init_trace.F module to set trace gas abundances (CO₂, CH₄, CO)
- setang_ir: module to compute secant angles
- ir_rta.F: module to compute total clear column layer-to-space transmittance and two-path SOLAR layer-to-space transmittance
 - calt1.F: v5 AIRS RTA module (8/00)
 - * calokw.F, OPTRAN water
 - calt2.F: v5 AIRS RTA module (8/00)
 - calt3.F: v5 AIRS RTA module (8/00)

- * calokw.F, OPTRAN water
- calt4.F: v5 AIRS RTA module (8/00)
- calt5.F: v5 AIRS RTA module (8/00)
- calt6.F: v5 AIRS RTA module (8/00)
- calt7.F: v5 AIRS RTA module (8/00)
- calt8.F: CrIS RTA module (in UMBC delivery: calt1.F)
- calt9.F: CrIS RTA module (in UMBC delivery: calt2.F)
- calt10.F: CrIS RTA module (in UMBC delivery: calt3.F)
- calt11.F: CrIS RTA module (in UMBC delivery: calt4.F)
- calt12.F: CrIS,IASI,NAST RTA module (in UMBC delivery: calt1.F)
- calt13.F: CrIS,IASI,NAST RTA module (in UMBC delivery: calt2.F)
- calt14.F: CrIS,IASI,NAST RTA module (in UMBC delivery: calt3.F)
- calt15.F: CrIS,IASI,NAST RTA module (in UMBC delivery: calt4.F)
- calt16.F: v3 & v4 AIRS RTA module calt1
 - * calokw.F, OPTRAN water
- calt17.F: v3 & v4 AIRS RTA module calt2
- calt18.F: v3 & v4 AIRS RTA module calt3
 - * calokw.F, OPTRAN water
- calt19.F: v3 & v4 AIRS RTA module calt4
- calt20.F: v3 & v4 AIRS RTA module calt5
- calt21.F: v3 & v4 AIRS RTA module calt6
- calt22.F: v3 & v4 AIRS RTA module calt7
- calt23.F: v6 & v7 & v8 AIRS, CrIS, IASI model calt1
 - * calokw.F, OPTRAN water
- calt24.F: v6 & v7 & v8 AIRS, CrIS, IASI model calt2
- calt25.F: v6 & v7 & v8 AIRS, CrIS, IASI model calt3
 - * calokw.F, OPTRAN water
- calt26.F: v6 & v7 & v8 AIRS, CrIS, IASI model calt4
- calt27.F: v6 & v7 & v8 AIRS, CrIS, IASI model calt5
- calt28.F: v6 & v7 & v8 AIRS, CrIS, IASI model calt6
- calt29.F: v6 & v7 & v8 AIRS, CrIS, IASI model calt7
- comp.down.F: module to compute down-welling radiance for levels other than the surface. Used for cloudy radiance computations.

diagnostic subroutine – not required for normal use

- ir_rta3.F: module to compute FIXED, WATER, and OZONE transmittances separately (diagnostic and printout uses only)

12.4.3 State variable definitions and units

The user is responsible for providing the state variables, in the proper units, to the radiative transfer programs. The arrays must be dimensioned to ≥ 100 . All the state variables are real*4.

NOTE: A layer column density is the column density between Pobs(L-1) and Pobs(L) and are given in units of molecules/cm².

- Pobs(L): for L = 1, 100, are the lower boundary pressures of the 100 layer model. This variable is not used, but it is verified to be consistent with the RTA file.

- temp(L): are the temperatures at Pobs(L)
- wcd(L): are the water vapor layer column densities
- wlcd(L): are the liquid water layer column densities (not used by IR RTA)
- ocd(L): are the ozone column densities
- cocd(L): (optional for AIRS, CrIS, IASI) are the carbon monoxide layer column densities
- ch4cd(L): (optional for AIRS, CrIS, IASI) are the methane layer column densities
- co2per: (optional for AIRS, CrIS, IASI) is the % change in CO2 abundance (co2per=0.0 → 370 ppm (in v6-v8 RTA) or 363 ppm (in v3-v4 RTA))
- Psurf: is the surface pressure
- Tsurf: is the surface skin temperature
- pland: is the fraction land coverage, 0.0=all water (ocean), 1.0=land
- emiss(n): is the surface emissivity for channel n, at frequency, freq(n)
- rho_solar or reflect(n): is the surface bi-directional reflectivity for channel n. In the radiative transfer code this value is multiplied by the solar radiance. This value should be solved for during daytime conditions. The value is $(1-\text{emiss}(n))/3.14$ for Lambertian surfaces and $(1-\text{emiss}(n))$ for specular surfaces.
- rho_therm: is the surface thermal reflectivity for channel n. This value is multiplied by the thermal down-welling radiation. It can either be solved for or set equal to $(1-\text{emiss}(n))/3.14$.
- viewang: is the instrument angle. 0 is directly down, varies +/- 49.5 degrees for this simulation (AIRS pattern)
- satheight: is the satellite altitude in km
- alat: is the latitude (degrees) at the center of the footprint
- alon: is the longitude (degrees) at the center of the footprint
- solzang: is the solar zenith angle (degrees). 0 is overhead, 90 is at the horizon.
- AU: is the Earth-Sun distance in Astronomical Units, 1.0 AU is 1.496E8 km.

12.4.4 Computational efficiency issues

An integer*4 bitwise flag, called "comprta", has been implemented to improve the performance of the new RTA. This flag can be set to -1 ensure that everything is computed. If computational enhancements are desired, this flag can direct `init_rta()` and `ir_rta()` to NOT recompute all the predictors (`init_rta()`) or transmittances (`ir_rta()`). This is particularly useful when finite difference Jacobians are computed or when approximate Jacobians are desired. The bit parameters are defined below.

At the present time, only the SOLAR and down-welling bits are used. Computation of the trace gases is automatically linked to the calls of `load_ir()`, `setang_ir()` and `init_trace()` - so the individual bits for CO₂, CH₄, and CO are not used at this time. In the future, the other bits may be implemented for water and ozone. To ensure upward compatibility the user should assume that if a bit is not set, then the predictors and transmittance associated with that state variable may NOT be updated from a previous computation.

| | | |
|-------|----|-------------|
| bit.1 | 1 | secant etc. |
| bit.2 | 2 | T(p) terms |
| bit.3 | 4 | q(p) terms |
| bit.4 | 8 | O3(p) terms |
| bit.5 | 16 | CO2 terms |

```

bit.6      32   CO terms
bit.7      64   CH4 terms
bit.8     128   spare
bit.9     256   spare
bit.10    512   spare
bit.11   1024   spare
bit.12   2048   spare
bit.13   4096   spare
bit.14   8192   SOLAR surface transmittance
bit.15  16384   DOWN-WELLING terms
bit.16  32768   SOLAR profile terms

```

```

integer*4 crta_A, crta_T, crta_D, crta_S, crta_SP
integer*4 crta_Q, crta_O3, crta_CO2, crta_CO, crta_CH4
parameter (crta_A=1, crta_T=2, crta_Q=4, crta_O3=8)
parameter (crta_CO2=16, crta_CO=32, crta_CH4=64)
parameter (crta_S=8192, crta_D=16384, crta_SP=32768)

integer*4 crta_all
parameter (crta_all=32767)

```

For example, to compute all the predictors and transmittances necessary for a cloud free radiance (all levels at the viewing angle and the solar bidirectional transmittance at the surface) the value of `comprta` would be

```
comprta = crta_all
```

For cloudy radiances we need the bidirectional transmittance at the solar+spacecraft secant angle for all levels. In this case, the flag should be set to

```
comprta = crta_all + crta_SP
```

If a finite difference Jacobian is needed, then the second call (w/ perturbation) to `init_rta()` and `ir_rta()` could save time by NOT re-computing predictors and transmittances of the unperturbed state. For example, for water Jacobians,

```
comprta = crta_Q + crta_S + crta_D
```

For long-wave channels, additional time could be saved by not computing the bidirectional transmittance

```
comprta = crta_Q + crta_D
```

12.4.5 UMBC predictor definitions

These are the definitions of the variables used in the UMBC RTA code and in the document tables below:

| code | doc | description |
|-------|------|---|
| ----- | ---- | ----- |
| PTEMP | | is the profile's temperature at P_{eff} of the layer |

RTEMP is the reference temperature
 PWAMNT is the profile's water amount in molecules/cm²
 NOTE: molecules/cm² = Na*1000*kilo-moles/cm², where
 Na = 6.0221367E+23 molecules/mole
 RWAMNT is the reference water amount in molecules/cm²
 POAMNT is the profile's ozone amount in molecules/cm²
 ROAMNT is the reference ozone amount in molecules/cm²
 PMAMNT is the profile's methane amount in molecules/cm²
 RMAMNT is the reference methane amount in molecules/cm²
 PCAMNT is the profile's carbon monoxide amount in molecules/cm²
 RCAMNT is the reference carbon monoxide amount in molecules/cm²

SECANG a is the secant of the viewing angle SECANG
 TR Tr is the temperature ratio PTEMP/RTEMP
 TRZ Trz is the pressure weighted temperature ratio above, i.e.
 the sum i=2 to i=L of { P(i) * (P(i) - P(i-1)) * Tr(i-1) }
 where "P" is the pressure PRES and "L" is the layer number, and
 Trz(L=1)=0

A_W W is the water amount ratio PWAMNT/RWAMNT
 dT dT is the temperature offset PTEMP-RTEMP
 AZ_W Wz is the pressure weighted water amount above ratio, the
 sum i=1 to i=L of { P(i) * ((P(i)-P(i-1)) * PWAMNT(i)) },
 divided by the same sum except using RWAMNT instead of PWAMNT.
 For these sums, term P(0) is defined as P(0)=2*P(1) - P(2).

A_O O is the ozone amount ratio POAMNT/ROAMNT
 AZ_O Oz is the pressure weighted ozone amount above ratio, the
 sum i=1 to i=L of { P(i) * ((P(i)-P(i-1)) * POAMNT(i)) },
 divided by the same sum except using ROAMNT instead of POAMNT.
 For these sums, term P(0) is defined as P(0)=2*P(1) - P(2)

XZ_O Ox is the unweighted ozone amount above ratio, the
 sum i=1 to i=L of { POAMNT(i) },
 divided by the same sum except using ROAMNT instead of POAMNT.
 For these sums, term P(0) is defined as P(0)=2*P(1) - P(2).

TAZ_0 TOz is the pressure and ozone weighted temperature ratio above,
 sum i=2 to i=L of { P(i) * (P(i)-P(i-1)) * dT(i-1) * O(i-1) }
 and TOz(L=1)=0

A_C C is the carbon monoxide amount ratio PCAMNT/RCAMNT
 AZ_C Cz is the pressure weighted CO amount above ratio, the
 sum i=1 to i=L of { P(i) * ((P(i)-P(i-1)) * PCAMNT(i)) },
 divided by the same sum except using RCAMNT instead of PCAMNT.
 For these sums, term P(0) is defined as P(0)=2*P(1) - P(2).

A_M M is the methane amount ratio PMAMNT/RMAMNT
 AZ_M Mz is the pressure weighted methane amount above ratio, the
 sum i=1 to i=L of { P(i) * ((P(i)-P(i-1)) * PMAMNT(i)) },
 divided by the same sum except using RMAMNT instead of PMAMNT.
 For these sums, term P(0) is defined as P(0)=2*P(1) - P(2).

TAZ_M TMz is the pressure and methane weighted temperature ratio above,
 sum i=2 to i=L of { P(i) * (P(i)-P(i-1)) * Tr(i-1) * M(i-1) }
 and TMz(L=1)=0

Table 12.6: Number of Predictors for the Seven UMBC Fitting Models

| | | AIRS RTA version 5 | | | | | | | | | |
|---|----------|--------------------|---|----|----|----|----|-----|-----|-------|--|
| | | # chl | C | F | W | O | CO | CH4 | TOT | N*TOT | |
| 1 | FOW | 1193 | 5 | 8 | 11 | 5 | 0 | 0 | 29 | 34597 | |
| 2 | FOW | 279 | 5 | 8 | 11 | 10 | 0 | 0 | 34 | 9486 | |
| 3 | FMW | 287 | 5 | 8 | 11 | 0 | 0 | 9 | 33 | 12936 | |
| 4 | FCOW | 61 | 5 | 11 | 13 | 3 | 11 | 0 | 43 | 2623 | |
| 5 | FOW-bfsw | 189 | 5 | 11 | 3 | 1 | 0 | 0 | 20 | 3780 | |
| 6 | FOW-mfnw | 112 | 5 | 8 | 7 | 1 | 0 | 0 | 21 | 2352 | |
| 7 | FOW-mfbw | 152 | 5 | 8 | 13 | 1 | 0 | 0 | 27 | 4104 | |
| | | 2378 | | | | | | | | 69878 | |

In addition, the following variables are defined within the code:

```

TJUNKS = Tr^2

WJUNKA = W*a
WJUNKR = sqrt(W*a)
WJUNKS = (W*a)^2
WJUNKZ = W^2*a/Wz
WJUNK4 = (W*a)^0.25

OJUNKA = O*a
OJUNKR = sqrt(O*a)
OJUNKZ = O*a/Ox
OJUNKX = a*Ox

CJUNKA = C*a
CJUNKR = sqrt(C*a)
CJUNKS = (C*a)^2
CJUNKZ = (C*a)*C/Cz

MJUNKA = M*a
MJUNKR = sqrt(M*a)
MJUNKZ = a*Mz

```

The original codes used a unique array for predictor for each RTA type. This complicated merging of the RTA's for AIRS, CrIS, IASI, and NAST. In my version of their code, I defined a single super-set of predictors which all the modules use. These are the definitions of those predictors and the indexing back to the original predictor definition in the UMBC code.

In the Table 12.6 the number of coefficients for each module for each optical depth component is listed. In Tables 12.7 through Table 12.12 the mapping from the original predictor indice to my super-set of all predictors (*i.e.*, all predictors used by ANY AIRS, IASI, CrIS, or NAST RTA) is given.

The coefficients are multiplied by predictors, each predictor is a function of the following parameters.

- PTEMP is the profile's temperature at P_{eff} of the layer, $T(L)$
- RTEMP is the reference temperature, $T_0(L)$

- PWAMNT is the profile's water layer column density given in molecules/cm². NOTE: molecules/cm² = $N_a \cdot 1000$ kilo-moles/cm², where $N_a = 6.0221367 \cdot 10^{23}$ molecules/mole
- RWAMNT is the reference water layer column density in molecules/cm²
- POAMNT is the profile's ozone layer column density in molecules/cm²
- ROAMNT is the reference ozone layer column density in molecules/cm²
- PMAMNT is the profile's methane layer column density in molecules/cm²
- RMAMNT is the reference methane layer column density in molecules/cm²
- PCAMNT is the profile's carbon monoxide layer column density in molecules/cm²
- RCAMNT is the reference carbon monoxide layer column density in molecules/cm²
- a is the secant of the viewing angle (SECANG)
- Tr is the temperature ratio of $T(L)/T_0(L)$, where $T_0(L)$ is the reference profile. (PTEMP/RTEMP).
- Trz is the pressure weighted temperature ratio above, *i.e.*,

$$\sum_{i=2}^L P(i) \cdot (P(i) - P(i-1)) \cdot Tr(i-1) \quad (12.12)$$

where "P" is the pressure (PRES) and "L" is the layer number. $Trz(L=1) = 0$

- W is the ratio of water layer column density to the reference water layer column density (PWAMNT/RWAMNT)
- dT is the temperature offset $T(L) - T_0(L)$ (PTEMP-RTEMP)
- Wz is the pressure weighted water layer column density above ratio, the

$$\frac{\sum_{i=1}^L P(i) \cdot ((P(i) - P(i-1)) \cdot PWAMNT(i))}{\sum_{i=1}^L P(i) \cdot ((P(i) - P(i-1)) \cdot RWAMNT(i))} \quad (12.13)$$

For these sums, term $P(0)$ is defined as $P(0) = 2 \cdot P(1) - P(2)$.

- O is the ratio of ozone layer column density to the reference ozone layer column density (POAMNT/ROAMNT)
- Oz is the pressure weighted ozone layer column density above ratio, the

$$\frac{\sum_{i=1}^L P(i) \cdot ((P(i) - P(i-1)) \cdot POAMNT(i))}{\sum_{i=1}^L P(i) \cdot ((P(i) - P(i-1)) \cdot ROAMNT(i))} \quad (12.14)$$

For these sums, term $P(0)$ is defined as $P(0) = 2 \cdot P(1) - P(2)$

- Ox is the unweighted ozone layer column density above ratio, the

$$\frac{\sum_{i=1}^L POAMNT(i)}{\sum_{i=1}^L ROAMNT(i)} \quad (12.15)$$

- TOz is the pressure and ozone weighted temperature ratio above,

$$\sum_{i=2}^L P(i) \cdot ((P(i) - P(i-1)) \cdot dT(i-1) \cdot O(i-1)) \quad (12.16)$$

and $TOz(L=1) = 0$.

Table 12.7: UMBC AIRS RTA: Fixed Gas Predictors

| table to go from FPREDX(i,L) to FPRED(j,L) | | | | | | | | | |
|--|---------------------|---|---|---|----|----|---|---|----------------------------|
| (values of i are given in table) | | | | | | | | | |
| j | equation | 1 | 2 | 3 | 4 | 5 | 6 | 7 | code |
| 1 | a | 1 | 1 | 1 | 1 | 1 | 1 | 1 | SECANG |
| 2 | a ² | 2 | 2 | 2 | 2 | 2 | 2 | 2 | SECANG(L)*SECANG(L) |
| 3 | a*Tr | 3 | 3 | 3 | 3 | 3 | 3 | 3 | SECANG(L)*TR |
| 4 | a*Tr ² | 4 | 4 | 4 | 4 | 4 | 4 | 4 | SECANG(L)*TJUNKS |
| 5 | Tr | 5 | 5 | 5 | 5 | 5 | 5 | 5 | TR |
| 6 | Tr ² | 6 | 6 | 6 | 6 | 6 | 6 | 6 | TJUNKS |
| 7 | a*Trz | 7 | 7 | 7 | 7 | 7 | 7 | 7 | SECANG(L)*TRZ |
| 8 | a*Trz/Tr | 8 | 8 | 8 | . | 8 | . | . | SECANG(L)*TRZ/TR |
| 9 | a ² *Tr | . | . | . | 9 | 9 | . | . | TR*SECANG(L) ² |
| 10 | a ³ | . | . | . | 10 | . | . | . | SECANG(L) ³ |
| 11 | sqrt(a) | . | . | . | 11 | 10 | 8 | 8 | SQRT(SECANG(L)) |
| 12 | a ² *Trz | . | . | . | 8 | . | . | . | TRZ*SECANG(L) ² |
| 13 | Trz | . | . | . | . | 11 | . | . | TRZ |
| 14 | sqrt(Tr) | . | . | . | . | . | . | . | sqrt(Tr) |
| 15 | | . | . | . | . | . | . | . | |
| 16 | | . | . | . | . | . | . | . | |

Table 12.8: UMBC AIRS RTA: Water Continuum Predictors

| table to go from CPREDX(i,L) to CPRED(j,L) | | | | | | | | | |
|--|-------------------------------------|---|---|---|---|---|---|---|------------------------|
| (values of i are given in table) | | | | | | | | | |
| j | equation | 1 | 2 | 3 | 4 | 5 | 6 | 7 | code |
| 1 | a*W/Tr ² | 1 | 1 | 1 | 1 | 1 | 1 | 1 | WJUNKA/TJUNKS |
| 2 | a*(W/Tr ²) ² | 2 | 2 | 2 | 2 | 2 | 2 | 2 | CONPRD(1,L)*A_W/TJUNKS |
| 3 | a*W/Tr | 3 | 3 | 3 | 3 | 3 | 3 | 3 | WJUNKA/TR |
| 4 | a*W ² /Tr | 4 | 4 | 4 | 4 | 4 | 4 | 4 | CONPRD(3,L)*A_W |
| 5 | a*(W/Tr) ² | 5 | 5 | 5 | 5 | 5 | 5 | 5 | CONPRD(1,L)*A_W |

Table 12.9: UMBC AIRS RTA: Carbon Dioxide Predictors

| table to go from CO2PRDX(i,L) to CO2PRD(j,L) | | | | | | | | | |
|--|-------------------|---|---|---|---|---|---|---|------------------|
| (values of i are given in table) | | | | | | | | | |
| j | equation | 1 | 2 | 3 | 4 | 5 | 6 | 7 | code |
| 1 | a | 1 | 1 | . | 1 | 1 | 1 | 1 | SECANG(L) |
| 2 | Tr | 2 | 2 | . | 2 | 2 | 2 | 2 | TR |
| 3 | a*Tr | 3 | 3 | . | 3 | 3 | 3 | 3 | SECANG(L)*TR |
| 4 | a*Tr ² | 4 | 4 | . | 4 | 4 | 4 | 4 | SECANG(L)*TJUNKS |

- C is the ratio of carbon monoxide (CO) layer column density to the reference CO layer column density, PCAMNT/RCAMNT
- Cz is analogous to Oz using RCAMNT and PCAMNT.
- M is the ratio of methane (CH₄) layer column density to the reference layer column density, PMAMNT/RMAMNT
- Mz is analogous to Oz using RMAMNT and PMAMNT.
- TMz analogous to TOz using RMAMNT and PMAMNT.

Chapter 13

Infrared and Microwave Noise Models

13.1 Microwave models

| | |
|--------------|---|
| amsu_0.dat | original AMSU & MHS noise spec |
| amsu_1.dat | AMSU & MHS noise spec 7/23/90 |
| amsu_300.dat | measured EOS-AMSU & HSB noise |
| atms_0.dat | ATMS specification w/ 32 channels (1/13/00) |

13.2 Infrared models

In order to compute noise spectra a noise model needs to be loaded and, in general, the noise at each channel is computed with knowledge of the radiances in all channels.

| additional steps required to compute a CLOUDY radiance | | | |
|--|--------------|----------|----------------------------------|
| step | module | library | purpose |
| Things Done Once | | | |
| 2.1 | load_noise | src.gsfc | load the noise file |
| 2.2 | rta_apod | IR RTA | determine the APODIZATION TYPE |
| 2.3 | set_apod | comp_rad | set apodization noise parameters |
| Things Done For Each Profile | | | |
| 23 | calc_phi | src.gsfc | compute integrated # of photons |
| 24 | calc_nedn(n) | src.gsfc | compute noise for channel=n |

13.2.1 converting $NE\Delta N(n)$ to $NE\Delta T(n)$

Since the Planck function is non-linear, the conversion from radiance to temperature must be done about a reference radiance $R_{\text{ref}}(n)$.

$$NE\Delta T(n) \equiv B_{\nu(n)}^{-1} \left(R_{\text{ref}}(n) + \frac{NE\Delta N(n)}{2} \right) - B_{\nu(n)}^{-1} \left(R_{\text{ref}}(n) - \frac{NE\Delta N(n)}{2} \right) \quad (13.1)$$

which can be approximated by a single-sided finite derivative.

$$NE\Delta T(n) = B_{\nu(n)}^{-1} (R_{\text{ref}}(n) + NE\Delta N(n)) - B_{\nu(n)}^{-1} (R_{\text{ref}}(n)) \quad (13.2)$$

sometimes, the reference radiance is a Planck function of a given temperature, so that if $R_{\text{ref}}(n) \equiv B_{\nu(n)}(T_{\text{ref}})$ then NE Δ T can be written as

$$\text{NE}\Delta T_{T_{\text{ref}}}(n) = B_{\nu(n)}^{-1}(R_{\text{ref}}(n) + \text{NE}\Delta N(n)) - T_{\text{ref}} \quad (13.3)$$

We can also use the analytic derivative of the Planck function when NE Δ N is small. For a fixed reference temperature we can write

$$\text{NE}\Delta T_{T_{\text{ref}}}(n) = \frac{\text{NE}\Delta N(n)}{\left. \frac{\partial B_{\nu(n)}}{\partial T} \right|_{T_{\text{ref}}}} \quad (13.4)$$

and when computing the NE Δ T for a given spectrum we can write

$$\text{NE}\Delta T(n) = \frac{\text{NE}\Delta N(n)}{\left. \frac{\partial B_{\nu(n)}}{\partial T} \right|_{B_{\nu}^{-1}(R_{\text{ref}}(n))}} \quad (13.5)$$

where $R_{\text{ref}}(n)$ is the spectrum of interest.

Most of the AIRS noise models tend to be displayed with $T_{\text{ref}}=250$ K; however, any temperature can be used.

13.2.2 converting NE Δ T(n) to NE Δ N(n)

Most instruments will measure NE Δ T directly from their internal black body calibrator at T_{ref} . In order to use these measurements in our simulator and retrieval, these values will be converted to NE Δ N.

$$\text{NE}\Delta N(n) = B_{\nu(n)}\left(T_{\text{ref}} + \frac{\text{NE}\Delta T(n)}{2}\right) - B_{\nu(n)}\left(T_{\text{ref}} - \frac{\text{NE}\Delta T(n)}{2}\right) \quad (13.6)$$

$$\text{NE}\Delta N = \text{NE}\Delta T_{T_{\text{ref}}}(n) \cdot \left. \frac{\partial B_{\nu(n)}}{\partial T} \right|_{T_{\text{ref}}} \quad (13.7)$$

13.2.3 Noise models

There have been many noise models used over the years and it is a little difficult to keep track of the origin and format of these files. The program `src_gsfc/load_noise.F` and `idl/load_noise.pro` are used to read all of these machine readable files into a FORTRAN or IDL common block for use in the simulator, retrieval, and diagnostic codes. The routines `calc_nedn` and `calc_phi` (FORTRAN or IDL) are used to compute NE Δ N values for a given scene. NE Δ T conversion is not done in the simulator or retrieval except for diagnostic printout.

Basically, noise models fall into the following categories

- Grating Models
 1. Modeled Noise on Hingepoints. Single column (a_0 of a polynomial).
 2. Noise for each channel, specified as noise as function of different scene temperatures (“MEASURED NO” & “INTERPOLATE” option).
- Interferometer Methods

3. Modeled Noise on Hingepoints. Each hingepoint is a polynomial expansion as a function of the integrated number of photons, ϕ or phi, striking the detector. These models are used for interferometers.
 4. Noise for each channel, specified as a simple polynomial of the scene integrated number of photons. (“MEASURED NO”)
- Channel Properties Files

Most recently the load_noise program was modified to directly read and parse the JPL channel properties files. These files contain flags for various non-optimal channel conditions as well as the NE Δ T values.

| filename | valid | |
|------------------------------------|---------------|---------------|
| | from | until |
| L2.chan_prop.1990.01.01.v5.1.2.anc | pre-launch | |
| L2.chan_prop.2002.05.04.v6.6.5.anc | May 4, 2002 | July 8, 2002 |
| L2.chan_prop.2002.07.09.v6.6.6.anc | July 9, 2002 | Aug. 29, 2002 |
| L2.chan_prop.2002.08.30.v6.6.2.anc | Aug 30, 2002 | Sep. 16, 2002 |
| L2.chan_prop.2002.09.17.v6.6.3.anc | Sep. 17, 2002 | Oct. 21, 2002 |
| L2.chan_prop.2002.10.22.v6.6.4.anc | Oct. 22, 2002 | Jan. 9, 2003 |
| L2.chan_prop.2003.01.10.v6.6.8.anc | Jan. 10, 2003 | Oct. 26, 2003 |

All noise data files are machine readable, fixed field files. The remainder of this section described the NON-channel properties files. Channel property files are an ASCII format with internal documentation of the columns.

The first line of the file contains the number of frequencies (channels or hingepoints), Nfrq, and the number of columns of noise data. Then there is a block of Nfrq lines with the noise data.

The noise data is followed by line with the number of options, Nopt. The options are comment and **keywords**. The first 2 lines are required and the rest are optional. The default file options are

```

NEDTmin = 0.0
hingeflg = .TRUE. ! assume these are modelled hinge points
polyflag = .TRUE. ! assume polynomial in terms of band integrated flux
noisescale = 1.0 ! multiplier to file
Ndeahinge = -1 ! option to kick dead channels
Tref = 2.5d02 ! default reference temperature = 250 K

```

| line # | key-word | description |
|----------|---------------------|--|
| 1 | NEDN | units are mW/cm ² /sr/cm-1 |
| 1 | NEDT(250) | units are NE Δ T and $T_{ref}=250$ |
| 2 | AIRS | instrument name is AIRS (grating model) allowed names: AIRS HIRS ITS CRIS IASI NAST |
| ≥ 3 | NEDT minimum | sets the minimum value of computed NEDT |
| ≥ 3 | MEASURED NOISE | noise measurement for each channel |
| ≥ 3 | NOISE SCALE | multiplier for the noise (old files were “per resolution element”) |
| ≥ 3 | INTERPOLATE COLUMNS | value of SCENE T_{eff} for this column |
| ≥ 3 | DEAD THRESHOLD | # of hingepoints for NEDT “DEAD” threshold |

Then the band definitions are defined in a table. This table is used to compute the number of photons striking the detector.

| column | key-word | description |
|--------|----------|--|
| 1 | ib1 | starting index (within noise data) for this band |
| 2 | ib2 | ending index (within noise data) for this band |
| 3 | wlow | lower frequency cutoff for this band |
| 4 | whgh | upper frequency cutoff for this band |
| 5 | L | Optical Path Difference |
| 6 | SCALE | photon flux scale |

The final block is simple comments about the generation of the file and are ignored by the program. Here is what the file MUST contain:

```

Nfrq  Ncol
f(1)  N( 1,1), N( 1,2), ..., N( 1,Ncol)
...
f(Nfrq) N(Nfrq,1), N(Nfrq,2), ..., N(Nfrq,Ncol)
Nopt
option (1)
...
option (Nopt)
Nband  Nb_col
ib( 1)  ib2( 1)  wlow( 1)  whgh( 1)  L( 1)  SCALE( 1)
....
ib(Nband)  ib2(Nband)  wlow(Nband)  whgh(Nband)  L(Nband)  SCALE(Nband)
Ncomments

```

An example of a simple file, airt_spec.dat

```

      8      1
500.00  0.35
740.00  0.35
740.01  0.20
2374.99 0.20
2375.00 0.14
2399.99 0.14
2400.00 0.20
3000.00 0.20
      5
NEDT(250)
AIRS      EOS specification taken from Aumann+Pagano paper
NEDT minimum = 0.00
NOISE SCALE= 1.4142 noise is specified as 'per resolution element'
ib1 ib2 Wlow Whgh L SCALE
 1  5
 1  8  620.0 2750.0 1.0000 1.0E18
-1

```

Here is an example of the file noise/airs.v6a.dat. Here a computed noise is given for one Scene temperature. The INTERPOLATE flag is set; however, there is only one column (Can't remember why I did this). It is measured noise, which means that there can be DEAD channels.

```

2378      1
      649.612      0.653000
      649.851      0.709000

```

```

        650.090      0.702000
        650.328      0.661000
        650.567      0.627000
        650.807      0.667000
        651.046      0.704000
        651.285      0.717000
        651.525      0.682000
        651.765      0.660000
        652.005      0.699000
        652.245 -1008.000000
        652.486 -1008.000000
        652.727      0.640000
        ....
        2664.138     0.393000
        2665.244     0.552000
16
NEDT(250) f/ 7/09/02 Cal Team Properties File w/ v6a RTA freqs
AIRS      f/ /airs/noise/anc/L2.chan_prop.2002.08.30.v6.3.0.anc
NEDT minimum = 0.000
MEASURED NOISE, therefore DEAD channels allowed, order must equal RTA
NOISE SCALE= 1.0000  noise is specified as per resolution element
INTERPOLATE COLUMNS  250.00
DEAD THRESHOLD= 8
  500.00 0.70  ! 2 * 0.35 (2 * per resolution element specification)
  740.00 0.70
  740.01 0.40  ! 2 * 0.20 (2 * per resolution element specification)
 2374.99 0.40
 2375.00 0.28
 2399.99 0.28  ! notch @ 2380 for T(p) sounding
 2400.00 0.40
 3000.00 0.40
ib1 ib2 Wlow  Whgh      L  SCALE
 1  5
 1 2378  620.0 2750.0 1.0000 1.0E18
22
Bad channels are flagged with values for NEDT that are < -1000. These
NedT values are an accumulation of individual flags that are offset
(subtracted) from a base of -1000:
.....

```

An example file for ITT CrIS model in which 3 noise models have been computed for a Scene with $T_{eff} = 233, 269,$ and 287 K. The routine `calc_nedn()` will use ϕ computed from `calc_phi()` to interpolate between the 3 columns.

```

1429 3
 635.000 0.6593697071 0.6511981487 0.6504131556
  ....
2797.500 0.0021761644 0.0024250064 0.0027498289
2800.000 0.0021808241 0.0024302420 0.0027557381
  6
NEDN  from ITT noise model
CrIS  from nedn_710.dat.frq
NEDT minimum = 0.000

```

MEASURED NOISE

INTERPOLATE COLUMNS: 233.00 269.00 287.00

ib1 ib2 Wlow Whgh L SCALE

3 5

1 11 630.0 1102.0 0.8000 1.0E18

12 21 1210.0 1800.0 0.4000 1.0E17

22 31 2100.0 2492.0 0.2000 1.0E16

-1

| filename | type | description |
|-------------------|------|--|
| airs.v6a.dat | | post-launch f's and > 8/30/02 noise |
| airs_cp63_m16.dat | | v5.0 -16.7 μm pre-launch f's w/ post-launch > 7/20/02 noise |
| airs_cp62_m16.dat | | v5.0 -16.7 μm pre-launch f's w/ post-launch > 7/9/02 noise |
| airs_cp62_p16.dat | | v5.0 +16.7 μm pre-launch f's w/ post-launch > 7/9/02 noise |
| airs_cp60_m16.dat | | v5.0 -16.7 μm pre-launch f's w/ post-launch > 6/14/02 noise |
| airs_cp60_p16.dat | | v5.0 +16.7 μm pre-launch f's w/ post-launch > 6/14/02 noise |
| airs_cp60_v5a.dat | | v5.0 pre-launch f's w/ post-launch > 6/14/02 noise |
| airs.v5a.dat | | pre-launch model v5.0 RTA f's |
| airs.v5.dat | | pre-launch model v5.0 RTA f's |
| airs.v4b.dat | | pre-launch model v4.0 RTA f's |
| airs.v5a_m16.dat | | =v5a on +16.7 μm model |
| airs.v5a_p16.dat | | =v5a on +16.7 μm model |
| airs_fm3.dat | | f/ L2.I.channel_prop.v5.1.2.anc, 12/04/00 |
| airs_fm2.dat | | D. Hagan analysis of PFM cal. data 3/17/99 |
| airs_fm1.dat | | from flight model test 933-938 3/23/99 |
| airs_n3.dat | | f/ Aumann and Pagano 1994, Opt.Eng. v.33 |
| airs_n2.dat | | same as airs_nedt.f 3/25/94 model |
| airs_n1.dat | | file used in 1995 AIRS simulations |
| airs_spec.dat | | AIRS specification |
| noaa00.dat | | HIRS model f/ POES user guide |
| noaa14.dat | | HIRS on NOAA-14 |

13.2.4 Interferometer Noise Modeling

| filename | type | description |
|----------------|------|--|
| its_LL1.dat | 3 | 1/97 Lincoln Lab's noise model |
| its_LL0.dat | 3 | 6/95 Lincoln Lab's noise model |
| iasi_rea.dat | 3 | IASI 12/97 nominal model |
| iasi_bst.dat | 3 | IASI 12/97 best-case model |
| iasi_wor.dat | 3 | IASI 12/97 worst-case model |
| iasi_1299.dat | 3 | IASI 12/99 model |
| iasi_0311.dat | 3 | F/ itwg-13 talk by T. Phulpin |
| iasi_budg.dat | 3 | IASI Budget model (5/15/04) |
| itt_1198.dat | 4 | 11/98 ITT model on hinge points |
| itt_pdr.dat | 4 | 5/99 ITT PDR model @ 3 T's on channels |
| itt_501.dat | 4 | 5/01 ITT model @ 3 T's on channels |
| itt_501b.dat | 4 | 5/01 ITT model on hinge pts |
| cris_0404a.dat | 4 | f/ Dequi Qu 4/4/04 (sensor only) |
| cris_0404b.dat | 4 | f/ Dequi Qu 4/4/04 (all sources) |

For the JPL CrIS models we have one document, which is a LATEX document (tex/jpl-nois.tex) based on

G. Aumanns Mathcad document. This describes a model developed at JPL to estimate a version of the ITS (later to become the CrIS) model used in the IPO vendor down-select process. This document and associated program (prog/idl/jpl_noise.pro) illustrates the kind of mathematics used in these models.

| filename | type | description |
|-------------|------|--|
| jpl1_08.dat | 3 | ITS L=0.8, 1" aperture (jpl_noise.pro) |
| jpl1_12.dat | 3 | ITS L=1.2, 1" aperture (jpl_noise.pro) |
| jpl2_08.dat | 3 | ITS L=0.8, 2" aperture (jpl_noise.pro) |
| jpl3_08.dat | 3 | ITS L=0.8, 3" aperture (jpl_noise.pro) |
| jpl3_12.dat | 3 | ITS L=1.2, 3" aperture (jpl_noise.pro) |

Interferometers can be apodized which is equivalent to a running mean filter applied to the Nyquist sampled spectral data (see Barnet, C.D., J.M. Blaisdell and J. Susskind 2000. "An analytical transformation for use in computation of interferometric spectra for remote sensing applications spectra." IEEE Trans. Geosci. Remote Sens. v.38 p.169-183.) Noise models are assumed to be an un-apodized noise spectrum, which is the FFT of the raw interferogram. Many instruments can be self-apodized, so un-apodized is not the same as a theoretical sinc() function. A de-apodized spectrum is a spectrum that has the self-apodization removed.

Simulation of instrument noise

The best way to simulate an apodized noise is to apodize the random noise spectrum. The programs comp_rad.F and simrad.F both do this process. The steps are as follows:

1. compute the apodized radiance for all channels, $R^A(n)$.
2. compute the photon flux using the apodized radiance, $\phi_b(R^A(n))$ for each band. The photon flux should be independent of apodization.
3. compute the statistical unapodized noise estimate, $NE\Delta N^U$, for this profile.
4. build a random sequence with a Gaussian distribution, $r(n)$, which has the property that the average is zero, $\langle r(n) \rangle = 0$ and the standard deviation is unity, $\sigma(r(n)) = 1$.
5. If the instrument is an interferometers apodize the noise spectrum by multiplication of the cosine transform of the noise spectrum by the apodization function, $A(\delta)$. This must be done independently for each band.

$$NE\Delta N_b^A(n) = CT^{-1} [A(\delta) \cdot CT(r(n) \cdot NE\Delta N_b^U(n))] \quad (13.8)$$

6. Add the apodized noise to the apodized radiance

$$R^A(n) = R^A(n) + NE\Delta N_b^A(n) \quad (13.9)$$

The noise will have the spectral correlation and noise reduction properties of the apodization function used.

Specification of statistical properties for retrieval noise covariance matrices

In Barnet *et al.* 2000, it is shown that all apodization functions can be expressed as a cosine series and that apodization can be expressed in terms of the cosine series coefficients

$$A(\delta) = a_0 + 2 \cdot \sum_{j=1}^{J-1} a_j \cdot \cos\left(j \cdot \pi \frac{\delta}{L}\right) \quad \text{for } |\delta| \leq L \quad (13.10)$$

and

$$a_0 \equiv \left(1 - 2 \sum_{j=1}^{J-1} a_j \right). \quad (13.11)$$

It can be easily shown that for apodized spectra sampled at the Nyquist spacing of $\delta y = \pi$, the radiances are a linear combination of the unapodized radiances,

$$R_G(i) = a_0 \cdot R_U(i) + \sum_{j=1}^{J-1} a_j \cdot (R_U(i-j) + R_U(i+j)) \quad (13.12)$$

which can be expressed in matrix form as

$$R_G = \mathbf{M}_G \cdot R_U. \quad (13.13)$$

where \mathbf{M}_G $_{i,i \pm j} = a_j$.

The fact that the apodized radiances are a linear combination of the unapodized radiances allows calculation of the noise correlation in adjacent apodized channels. Equation (13.13) represents a $2J - 1$ point running mean with weights, w_k , given by

$$w_k = a_{|k|}, \quad \text{for } k = -(J-1), J-1. \quad (13.14)$$

For the Hamming apodization function ($J = 2, a_0 = 0.54, a_1 = 0.23$) the three running mean coefficients are $w_{-1} = 0.23, w_0 = 0.54, w_1 = 0.23$. In general, the random channel noise is reduced by a factor of

$$f = \left[\sum_{k=1-J}^{J-1} w_k^2 \right]^{-\frac{1}{2}} \quad (13.15)$$

and it can be shown that the channel noise correlation for channels separated by $\pm n$ Nyquist spacings is given by

$$C_n = f^2 \sum_{k=1-J}^{J-1-n} w_k \cdot w_{k+n}, \quad \text{for } n = 1, 2J-2. \quad (13.16)$$

The RTA's are expected to have a keyword which tells the simulator, retrieval, and diagnostic programs what the apodization type is. Currently the following apodization types are supported:

| APOD_type | description | COSINE coefficients | | | |
|-----------|------------------------------------|---------------------|--------|--------|---------|
| | | a_0 | a_1 | a_2 | a_3 |
| HAMM | Hamming Apodized | 0.23 | | | |
| HAMM | General Cosine | APOD_coef | | | |
| BLAC | Blackman Apodized | 0.2502 | 0.0397 | | |
| GAUS | Gaussian Apodized ($\sigma = L$) | 0.2380 | 0.0270 | 0.0025 | -0.0009 |
| K-B | Kaiser-Bessel $k = 6$ | 0.2414 | 0.0076 | 0.0017 | -0.0012 |

I used the program prog/idl/apod_fit() to determine the noise reduction factor and noise correlation that results from applying a running mean filter to the un-apodized spectrum. Barnet et al. 2000 has tables of results from those runs. Each package does it differently. Here are some examples:

- The comp_rad.F program does it by applying the appropriate apodization to the radiances using src.gsfc/apod_mat. The coefficients are determined in the local subroutine set_apod(). The radiances will have the noise reduction factor and correlation of the case dependent noise spectrum.

- The simulator (`src_util/simrad.F`) does it by applying the appropriate apodization to the radiances in `simrad/add_noise.F` using `src_gsfc/apod_mat`. The coefficients are determined in `simrad/proinit.F`. The radiances will have the noise reduction factor and correlation of the case dependent noise spectrum.
- The retrieval (`src_util/airsb.F`) does it by applying the statistical noise reduction factor and statistical noise correlation factors. These statistical estimates are read in via the `pro.nl` as `APOD_NRF`, `APOD_CCN`, and `APOD_CCV()`. Currently the following types have been supported in the retrievals

| APOD_XXX: airsb apodization paramters | | | | | | | | |
|---------------------------------------|-------------|---------|-----|--------|---------|--------|--------|--------|
| APOD_type | description | NRF | CCN | CCV(1) | CCV(2) | CCV(3) | CCV(4) | CCV(5) |
| GRAT | Grating | 1.0 | 0 | 1.0 | n/a | n/a | n/a | n/a |
| HAMM | Hamming | 1.5863 | 2 | 1.0 | 0.625 | 0.133 | n/a | n/a |
| BLAC | Blackman | 1.80913 | 4 | 1.0 | 0.75435 | 0.314 | 0.0645 | 0.005 |

Appendix A

Documentation for UMBC RTA Versions

A.1 v9 AIRS RTA: January 2004

New RTA coefficients (MODIS center frequency adjustment, non-LTE, and new trace gases: HNO₃, N₂O, and SO₂) required a new format for the RTA file. This format is called v9 and the code is backward compatible with earlier RTA versions without these coefficients.

- v9e: v9d + added replaced all columns from tunmlt_exper.7.txt. This transmittance tuning was derived from sondes and retrievals (using v9d).
- v9d: v9c + added all columns from tunmlt_exper.6.txt and new coefficient files (set1, set2, co2) in which M12 frequencies were adjusted. This transmittance tuning was derived from sondes and retrievals (using v9b).
- v9c: v9b + added non-LTE tuning (from tunmlt_exper.6.txt)
- v9b: v9a + added HNO₃, SO₂, and N₂O transmittances
- v9a: v8d + added non-LTE coefficients (new file format to allow center freq (MODIS) adjustment, non-LTE, HNO₃, N₂O, and SO₂ coefficients)

A.2 v8 AIRS RTA: January 2004

The differences between v8 & v7 RTA's are:

- v8d: v8c + added Ozone tuning to adjust Ozone for new HITRAN
- v8c: v8b + added new down-welling term (see rs_notes, DOWNVER=2)
- v8b: v8a + used empirical transmittance tuning to ARM-TWP sondes
- v8a: differences from v7 are (see Larrabee's AIRS STM talk on Mar. 30, 2004):
 - AIRS was shut down in Oct. 2003 to avoid damage due to a solar storm forecast. The instrument warmed up and upon cooling down the focal plane and spectrometer temperatures were re-set as closely as possible; however, there were differences.
 - improvements to water continuum
 - CO₂ 4.3 μ m band head spectroscopy improvements
 - CFC concentrations
 - M5 had incorrect frequencies in v7 - these were fixed.

Table A.1: Available UMBC RTA's for AIRS
All convert programs in AIRS_code/umbc_cnv

| original delivery | date | convert | final binary file |
|-------------------|--------|---------------------|--|
| sartav107 | Nov 05 | cnv_v9a.F | binary.trcoef.airs.v9e (v9c+M12 f's, e7 tuning) |
| sartav107 | Nov 05 | cnv_v9a.F | binary.trcoef.airs.v9d (v9c+M12 f's, e6 tuning) |
| sartav107 | Nov 05 | cnv_v9a.F | binary.trcoef.airs.v9c (v9b+nLTE tuning) |
| sartav107 | Jul 05 | cnv_v9a.F | binary.trcoef.airs.v9b (v9a+HNO ₃ ,N ₂ O,SO ₂) |
| sartav107 | Jun 05 | cnv_v9a.F | binary.trcoef.airs.v9a (v8d+non-LTE) |
| sartav105 | May 05 | cnv_v8c.F | binary.trcoef.airs.v8d (v8c+ozone) |
| sartav105 | Apr 05 | cnv_v8c.F | binary.trcoef.airs.v8c (v8b+Rdown) |
| sartav105 | Jan 04 | cnv_v80.F | binary.trcoef.airs.v8b (v8a+tuning) |
| sartav105 | Jan 04 | cnv_v80.F | binary.trcoef.airs.v8a (w/o tuning) (v8, freq's after 11/2003 warmup) |
| sartav104 | Jan 03 | cnv_v70.F | binary.trcoef.airs.v7a (v6+fringing) |
| sartav103 | Sep 02 | cnv_v60.F | binary.trcoef.airs.v6a (new freq's) |
| AIRS_FTC_Aug00/ | Aug 00 | cnv_800_m16.F | binary.trcoef_m16.airs.v5a (-16 μ m FPA) |
| AIRS_FTC_Aug00/ | Aug 00 | cnv_800_p16.F | binary.trcoef_p16.airs.v5a (+16 μ m FPA) |
| AIRS_FTC_Aug00/ | Aug 00 | cnv_800.F | binary.trcoef.airs.v5a |
| AIRS_FTC_Sep99/ | Sep 99 | cnv_999.F | binary.trcoef.airs.v4b (11 CO chl's) |
| AIRS_FTC_Sep99/ | Sep 99 | cnv_999.F | binary.trcoef.airs.v4a |
| AIRS_FTC_Mar98/ | Mar 98 | cnv_598.F | binary.trcoef.airs.v3d |
| modis_v00 | Jul 05 | cnv_modis.F | binary.trcoef.modis.v7c |
| modis_v00 | Jun 05 | cnv_modis.F | binary.trcoef.modis.v7b |
| modis_v00 | Mar 05 | cnv_modis.F | binary.trcoef.modis.v7a |
| uses RTA files | Nov 04 | merge_cris.F | binary.trcoef.cris.080404 |
| uses RTA files | Nov 04 | merge_cris.F | binary.trcoef.cris.080408 |
| Data_CrIS_Nov2004 | Nov 04 | cnv_crisL08.F | binary.trcoef.cris.080808 |
| Data_CrIS_Nov2004 | Nov 04 | cnv_crisL08.F | binary.trcoef.cris.080804 |
| Data_cris_jun03 | Jun 03 | cnv_crisv7a.F | binary.trcoef.cris.v7a |
| cris_9810/ | Oct 98 | cnv_1098.F | binary.cris08.v3b |
| cris_9810/ | Oct 98 | cnv_1098.F | binary.cris10.v3b |
| cris_9810/ | Oct 98 | cnv_1098.F | binary.cris12.v3b |
| CrisUMBCFMv1.1/ | May 98 | cnv_598.F | binary.trcoef.cris.v3b |
| Data_iasi_sep03 | Sep 03 | cnv_iasiv7a.F | binary.trcoef.iasi.v7a |
| iasi_9810/ | Oct 98 | cnv_iasi.F | binary.IASI.v3b |
| nast_9810/ | Oct 98 | cnv_iasi.F | binary.NAST.v3b |
| hffp_9906/ | Jun 99 | hffp0104/cnv_hffp.F | hirs07.bin \rightarrow hirs15.bin |
| OSSFwdModel/ | | | |

A.3 v7 AIRS, IASI, CrIS RTA's: January 2003

v7a = v6a + added AIRS fringing (due to the interference effect with the germanium entrance filter's).

A.4 v6a AIRS RTA : August 2002

See Larrabee Strow's AIRS STM talk on Feb. 25, 2003.

- This is the first RTA with measured post-launch frequencies.
- use 5 coef's for water continuum (was 4), therefore, left v5 as calt1-calt7 and made v6 set calt23-calt29

Table A.2: Available IASI & CrIS RTA files

| RTA | OPD of Band | | |
|---------------------------|-------------|-----|-----|
| | LW | MW | SW |
| binary.trcoef.cris.080404 | 0.8 | 0.4 | 0.4 |
| binary.trcoef.cris.080408 | 0.8 | 0.4 | 0.8 |
| binary.trcoef.cris.080808 | 0.8 | 0.8 | 0.8 |
| binary.trcoef.cris.080804 | 0.8 | 0.8 | 0.4 |
| binary.trcoef.cris.v7a | 0.8 | 0.4 | 0.2 |
| binary.cris08.v3b | 0.8 | 0.4 | 0.2 |
| binary.cris10.v3b | 1.0 | 0.4 | 0.2 |
| binary.cris12.v3b | 1.2 | 0.4 | 0.2 |

Table A.3: Thermal Down-welling version in UMBC RTA's

| NOAA version | down ver | calt set |
|--------------|----------|----------|
| v9 & v8 | 0 | 23-29 |
| v7 | 0 | 23-29 |
| v6 | 0 | 23-29 |
| v5 | 0 | 1-7 |
| v4 | 0 | 16-22 |
| AIRS v3 | 0 | 16-22 |
| Cris v3 | 0 | 8-11 |
| IASI v3 | 1 | 12-15 |
| NAST v3 | 1 | 12-15 |

Table A.4: 66 level RTA's developed at GSFC (Amita Mehta)

| Other RTA files (see /airs/RTA/README.DOC) | | | |
|--|--------------|----|-----------------------------|
| /airs/RTA/ | Author | L | description |
| AIRS48_v2.bi | Amita, 7/98 | 66 | AIRS Gaussian CSRF built |
| TRAP48_v2.bi | Amita, 7/98 | 66 | AIRS Trapezoidal CSRF, 6/97 |
| binary.trcoef.airs.v1a | JOHN, 7/98 | 66 | original AIRS |
| trcoef.airs.bi | Lena, 3/95 | 64 | original AIRS, no gaps |
| HAMM48_v1.bi | Amita, 10/97 | 66 | ITS Hamming (L=0.8,0.4,0.2) |
| HAMM48_L12_v1.bi | Amita, 1/98 | 66 | ITS Hamming (L=1.2,0.6,0.3) |
| HIRS_v1.bi | Amita, 4/96 | 66 | HIRS RTA |

- changes to CO2 channels based on laboratory data (not based on validation data).
- changes in H2O continuum in window regions based on variation of biases with total column water in ECMWF model fields.
- modification to coefficients in strong H2O band based on ARM and ABOVE RS-90 sondes (Sep. to Oct. timeframe).
- modification to coefficients weak water lines above 2500 cm^{-1} based on ARM and ABOVE RS-90 sondes (Sep. to Oct. timeframe).

A.5 v5 AIRS RTA: August 2000

v5: This is the final pre-launch RTA with the best estimate of fringing (due to the interference effect with the germanium entrance filter's). Three sets of frequencies were made to be prepared for whatever the final set-points of the AIRS focal plane and spectrometer temperatures would be. The frequencies were set at the nominal frequencies (v5a) and frequencies with a +16 μm (_p16)

and a $-16 \mu\text{m}$ (`_m16`). A $1 \mu\text{m}$ shift in the focal plane is equivalent to about a 1% shift in the centroid w.r.t. the FWHM of the channel. Since for AIRS $\text{FWHM} \simeq \nu/1200$ and AIRS is Nyquist sampled and $\Delta\nu \simeq \nu/2400$ this means that between these three RTA's we have RTA coefficients defined at every $\frac{1}{3}$ of an AIRS sample. As luck would have it, the $-16 \mu\text{m}$ RTA was the closest to the configuration in July of 2002.

The differences between v5 & v4 RTA's are:

- new (launch) frequencies
- `co2std = 370 ppm` (was 363) reference profiles have changed.
- IDEAL spectral channel response functions (i.e., accounted for AIRS fringing)
- `MXCONPRD=6`, `CONPRD()` uses 5th term
- `MXCHNW` needs to be incremented from 640 to 680 (`_i665`)
- `MXCHNC` needs to be incremented from 1000 to 1024 (`_i1004`)

The channel RTA blocks are given by

| | | AIRS v5a | | | | | | | | | |
|---|----------|----------|---|----|----|----|----|-----|-----|-------|--|
| | | # chl | C | F | W | O | CO | CH4 | TOT | N*TOT | |
| 1 | FWO | 1193 | 5 | 8 | 11 | 5 | 0 | 0 | 29 | 34597 | |
| 2 | FOW | 279 | 5 | 8 | 11 | 10 | 0 | 0 | 34 | 9486 | |
| 3 | FMW | 287 | 5 | 8 | 11 | 0 | 0 | 9 | 33 | 12936 | |
| 4 | FCOW | 61 | 5 | 11 | 13 | 3 | 11 | 0 | 43 | 2623 | |
| 5 | FWO-bfsw | 189 | 5 | 11 | 3 | 1 | 0 | 0 | 20 | 3780 | |
| 6 | FWO-mfmw | 112 | 5 | 8 | 7 | 1 | 0 | 0 | 21 | 2352 | |
| 7 | FWO-mfbw | 152 | 5 | 8 | 13 | 1 | 0 | 0 | 27 | 4104 | |
| | | 2378 | | | | | | | | 69878 | |

Note:

- `bfsm == big fixed, small water`
- `mfmw == medium fixed, medium water`
- `mfbw == medium fixed, big water`

| | | AIRS v5b (w/ extra CO channels) | | | | | | | | | |
|---|----------|---------------------------------|---|----|----|----|----|-----|-----|-------|--|
| | | # chl | C | F | W | O | CO | CH4 | TOT | N*TOT | |
| | | AIRS | C | F | W | O | CO | CH4 | TOT | N*TOT | |
| 1 | FWO | 1193 | 5 | 8 | 11 | 5 | 0 | 0 | 29 | 34597 | |
| 2 | FOW | 279 | 5 | 8 | 11 | 10 | 0 | 0 | 34 | 9486 | |
| 3 | FMW | 287 | 5 | 8 | 11 | 0 | 0 | 9 | 33 | 12936 | |
| 4 | FCOW | 72 | 5 | 11 | 13 | 3 | 11 | 0 | 43 | 3096 | |
| 5 | FWO-bfsw | 189 | 5 | 11 | 3 | 1 | 0 | 0 | 20 | 3780 | |
| 6 | FWO-mfmw | 112 | 5 | 8 | 7 | 1 | 0 | 0 | 21 | 2352 | |
| 7 | FWO-mfbw | 152 | 5 | 8 | 13 | 1 | 0 | 0 | 27 | 4104 | |
| | | 2389 | | | | | | | | 70389 | |

A.6 v4 AIRS RTA: Sep. 1999

The differences between v3b & v4a are:

- Used instrument line shapes as measured in pre-flight calibration at LMIRIS (for definitions see the file /airs/RTA/ AIRS_FTC_Sep99/CSRF/fortran_srf.tar)
- new channel frequencies
 - lost 2169.42-2181.37 CO channels
 - reclassified 2243.72-2270.62 as non-CO channels
- make 575 channels OPTRAN vs 419 in v3d
- add more CH4 channels (calt03)
 - 1217.01-1272.62 added 105 channels
 - 1351.91-1383.33 added 60 channels
- v4b has 11 extra CO channels (calt4) in order to perform a trade off study to evaluate the shifting of the SW band to (a) improve the CO retrieval and (b) assess the impact on the temperature retrieval.
- in Sep. 2000, upon deliver of v5 RTA the v4a RTA was rebuilt and the indexing in within the file was altered from calt.1 to calt.7 to calt.16 to calt.22. The v3 RTA's have their coefficients moved in load_ir.F

| UMBC calt | v3 | v4 |
|-----------|------|------|
| calt.1 | 1364 | 1561 |
| calt.2 | 213 | 180 |
| calt.3 | 287 | 122 |
| calt.4 | 67 | 108 |
| calt.5 | 183 | 131 |
| calt.6 | 125 | 144 |
| calt.7 | 139 | 127 |
| co2 | 956 | 954 |
| optran | 575 | 419 |

| AIRS v4a | | | | | | | | | | | |
|----------|----------|-------|---|----|----|----|----|-----|-----|-------|--|
| | | # chl | C | F | W | O | CO | CH4 | TOT | N*TOT | |
| 1 | FWO | 1364 | 4 | 8 | 11 | 5 | 0 | 0 | 28 | 38192 | |
| 2 | FOW | 213 | 4 | 8 | 11 | 10 | 0 | 0 | 33 | 7029 | |
| 3 | FMW | 287 | 4 | 8 | 11 | 0 | 0 | 9 | 32 | 9184 | |
| 4 | FCOW | 67 | 4 | 11 | 13 | 3 | 11 | 0 | 42 | 2814 | |
| 5 | FWO-bfsw | 183 | 4 | 11 | 3 | 1 | 0 | 0 | 19 | 3477 | |
| 6 | FWO-mfmw | 125 | 4 | 8 | 7 | 1 | 0 | 0 | 20 | 2500 | |
| 7 | FWO-mfbw | 139 | 4 | 8 | 13 | 1 | 0 | 0 | 26 | 3614 | |
| | | 2378 | | | | | | | | 66810 | |

A.7 v3 RTA's for AIRS (5/98), CrIS (5/98 & 10/98), NAST (10/98) & IASI (10/98)

- v3 used an empirical function with four parameters (a_n, b_n, c_n, d_n). The parameters were determined by estimates derived from the optical design of AIRS.

$$\begin{aligned} \Phi_n(\nu) &= \exp(-a_n \cdot (\nu - \nu_n)^2) \\ &+ b_n \cdot (1.0 - \exp(-a_n \cdot (\nu - \nu_n)^2)) \cdot [d_n + |(\nu - \nu_n)|]^{c_n} \end{aligned} \quad (\text{A.1})$$

v3d: is ordered in ascending *wavenumber* (per change requested by the AIRS science team). This was done with a re-mapping file located in umbc_cnv/airs.v3d.map

v3c: has 5 missing channels added into set to make up 2378 channels. The numbering scheme flipped 4 modules such that the AIRS channel ordering was in ascending *wavelength*. These channels were simply copies of 5 other channels. The file umbc.cnv/airs_v3c.map controlled the addition of the new channels (*e.g.*, channels at ends of some modules were copied to end of adjacent module). Channels duplicated were 118=120 (2552.6699) 248=364 (2309.4951) 364=234 (2432.5825) 514=502 (2177.5149) 515=913 (1337.5775)

v3b: is equal to v3a with additional information in header (apodization)

v3a: has original 2373 channels of the UMBC set

~/RTA/AIRS_CrIS_kludge/AIRS_FTC_Mar98 == V3A

| | AIRS | C | F | W | O | CO | CH4 | TOT | N*TOT |
|------------|------|---|----|----|----|----|-----|-----|-------|
| 1 FWO | 1561 | 4 | 8 | 11 | 5 | 0 | 0 | 28 | 43708 |
| 2 FOW | 180 | 4 | 8 | 11 | 10 | 0 | 0 | 33 | 5940 |
| 3 FMW | 122 | 4 | 8 | 11 | 0 | 0 | 9 | 32 | 3904 |
| 4 FCOW | 108 | 4 | 11 | 13 | 3 | 11 | 0 | 42 | 4536 |
| 5 FWO-bfsw | 131 | 4 | 11 | 3 | 1 | 0 | 0 | 19 | 2489 |
| 6 FWO-mfmw | 144 | 4 | 8 | 7 | 1 | 0 | 0 | 20 | 2880 |
| 7 FWO-mfbw | 127 | 4 | 8 | 13 | 1 | 0 | 0 | 26 | 3302 |
| | ---- | | | | | | | | ----- |
| | 2373 | | | | | | | | 66759 |

66759 * 100 levels, real*4 --> 26,703,600 bytes

OPTRAN coefficients

419 channels, 300 layers, 9 COEFS, 4 PROFILE AVERAGES?

INDH20 2373

WAZOP 419

WAVGOP 4*300 ==> 4,541,168b

COFH20 9*300*419

CO2 coefficients

INDCO2 = 2373 ==> 9,492b

COFCO2 = 954*4*100 ==> 1,526,400b

DOWNWELLING: COEFF, LABOVE

5*2373 = 11865 + 2373 = 14238 ==> 56,952b

AIRS 12/95 RTA

12/95 RTA

| | AIRS | C | F | W | O | CO | CH4 | TOT | N*TOT |
|---|------|---|---|----|---|----|-----|-----|-------|
| 0 | 2380 | 4 | 8 | 13 | 9 | 0 | 0 | 34 | 80920 |

2380*34 = 80920 * 100 levels, real*4 --> 32,368,000

108*42 4536 * 100 levels, real*4 --> 1,814,400

```

(3/98 is 26,703,600 bytes)                34,182,400 bytes

2380*34 = 80920 * 100 levels, real*4 --> 32,368,000
150*42 = 6300 * 100 levels, real*4 -->  2,520,000
-----
                                34,888,000 bytes

```

```

2373*42 = 99666 * 100 levels, real*4 --> 39,866,400 bytes

```

```

-----
                        CrIS 1/98 RTA
-----

```

```

~/RTA/AIRS_CrIS_kludge/CrisUMBCFMv1.1

```

- L=0.79994, 761 chl's from 630.05 (#1008) to 1105.08 (#1778)
- L=0.39997 481 chl's from 1205.09 (#964) to 1805.13 (#1444)
- L=0.20008 161 chl's from 2104.15 (#842) to 2503.99 (#1002)

```

CrIS RTA

```

| | CrIS | C | F | W | O | CO | CH4 | TOT | N*TOT | |
|----|-------|------|---|----|----|----|-----|-----|-------|-------------------------------|
| 8 | FWO | 1105 | 4 | 8 | 11 | 5 | 0 | 0 | 28 | 19890 (uses #1's coef memory) |
| 9 | FWO | 137 | 4 | 8 | 11 | 10 | 0 | 0 | 33 | 4521 (uses #2's coef memory) |
| 10 | | 44 | 4 | 8 | 11 | 11 | 0 | 0 | 34 | 1496 (uses #4's coef memory) |
| 11 | FWO-b | 117 | 4 | 11 | 3 | 1 | 0 | 0 | 19 | 2223 (uses #5's coef memory) |
| | | ---- | | | | | | | ----- | |
| | | 1403 | | | | | | | | 28130 |

```

100 levels, real*4 --> 11,252,000 bytes

```

```

-----
~/RTA/CrIS_FTC_Oct98    10/98 CrIS RTA at L=0.8, 1.0, 1.2
~/RTA/IASI_FTC_Sep98   IASI RTA
~/RTA/NAST_FTC_Aug98   NAST RTA
-----

```

```

NOTE: AIRS_FTC_Mar98  WPRED3(11)
      documnet says: W*a*Mz*a, but is really
      sqrt(W*a)*Mz*a == WPRED3(11,L)=WJUNKR*MJUNKZ
NOTE: NAST_FTC_Aug98 has a documentation error for OPRED1(5)
      document says: sqrt(0*a)*dT, but it is really: 0*a^2
NOTE: CrisUMBCFMv1.1 has a documentation error for FRED4(8)
      document says: a*Trz/Tr, but it is really: sqrt(Tr)
      otherwise, calpar.f, calt1.f, calt1.2, calt3.f, calt4.f
      are identical between these 3 RTA's.

```

```

NAST
-----

```

```

from NAST_FTC_Aug98/Data/Lists/list_band?.txt. The original
UMBC delivery (8/98) had 4149 channels in band.2, of which, 853

```

between 1974.39 and 2180.03 were deleted because they were duplicates.
 All the deleted channels were from the original set #3 (now calt14)
 which had 6172 channels and now has 5319 channels. The original
 set had 11305. The program cnv_iasi.F was used to delete the channels.

band #1 has 3526 channels 605.095 to 1454.879 0.241074 2.07406
 band #2 has 3296 channels 1180.055 to 1974.393 0.241074 2.07406
 band #3 has 3630 channels 1955.107 to 2829.963 0.241074 2.07406

 10452

65 layers, real*4 --> 86,204,300 bytes

NAST 100
 35 p(35) = 51.528
 1 <-> 36 p(36) = 56.126
 62 <-> 97
 65 <-> 100

IASI

605.095-2829.96 9230 channels L = 2.0 Gaussian Apodized

f = 605.095 + dv*(n-1), dv=0.241074, L = 2.07406

CrIS

| | band.1 | band.2 | band.3 | total |
|------------|--------|--------|--------|-------|
| CrIS L=0.8 | 939 | 483 | 294 | 1716 |
| CrIS L=1.0 | 1171 | 601 | 367 | 2139 |
| CrIS L=1.2 | 1408 | 723 | 441 | 2572 |

- L=0.79994 761 chl's from 630.05 (#1008) to 1105.08 (#1778)
 - L=0.39997 481 chl's from 1205.09 (#964) to 1805.13 (#1444)
 - L=0.20008 161 chl's from 2104.15 (#842) to 2503.99 (#1002)

CrIS L=0.8

- L=0.80210 939 chls from 605.29(# 971) to 1190.00(#1909) dv=0.6233637
 - L=0.40100 483 chls from 1199.50(# 962) to 1800.50(#1444) dv=1.2468829
 - L=0.20050 294 chls from 2099.75(# 842) to 2830.42(#1135) dv=2.4937649

CrIS L=1.0

- L=1.00060 1171 chls from 605.14(#1211) to 1189.79(#2381) 0.4997002
 - L=0.50030 601 chls from 1200.28(#1201) to 1799.92(#1801) 0.9994004
 - L=0.25010 367 chls from 2099.16(#1050) to 2830.87(#1416) 1.9992002

CrIS L=1.2

- L=1.20310 1408 chls from 605.10(#1456) to 1189.84(#2863) 0.4155930

- L=0.60160 723 chls from 1200.13(#1444) to 1800.20(#2166) 0.8311171
 - L=0.30080 441 chls from 2099.40(#1263) to 2830.78(#1703) 1.6622342

| new calt# | orig calt# | | | | | | | # of channels | | | | |
|--------------|---------------|---|----|----|----|----|-----|---------------|-------|-------|-------|-------|
| | | C | F | W | 0 | CO | tot | L=0.8 | L=1.0 | L=1.2 | IASI | NAST |
| 12 | 1 | 4 | 11 | 7 | 5 | - | 27 | 729 | 940 | 1166 | 4371 | 3730 |
| 13 | 2 | 4 | 8 | 11 | 10 | - | 33 | 236 | 301 | 358 | 737 | 621 |
| 14 | 3 | 4 | 8 | 13 | 3 | - | 28 | 702 | 837 | 974 | 3405 | 5319 |
| 15 | 4 | 4 | 11 | 13 | 9 | 11 | 48 | 49 | 61 | 74 | 717 | 782 |
| | | | | | | | | ----- | ----- | ----- | ----- | ----- |
| | | | | | | | | 1716 | 2139 | 2572 | 9230 | 10452 |

In the following tables the predictors for the CrIS models are given. The columns for labeled 8,9,10,11 refer to the L=0.8 model for CrIS (calt's 8 through 11) while the columns labeled 12,13,14,15 refer to the CrIS, IASI, and NAST models from Oct. 1998 and are given in calt's 12 through 15.

Fixed Gases:

| j | predictor | CrIS | | | | NAST | | | | |
|----|---------------------|------------------|---|----|----|------------------|----|----|----|---------------------|
| | | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | |
| 1 | a | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | SECANG |
| 2 | a ² | 4 | 4 | 4 | 4 | 4 | 4 | 4 | 4 | SECANG(L)*SECANG(L) |
| 3 | a*Tr | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | SECANG(L)*TR |
| 4 | a*Tr ² | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | SECANG(L)*TJUNKS |
| 5 | Tr | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | TR |
| 6 | Tr ² | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | TJUNKS |
| 7 | a*Trz | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | SECANG(L)*TRZ |
| 8 | a*Trz/Tr | 8 | 8 | . | . | 8 | 8 | . | 8 | SECANG(L)*TRZ/TR |
| 9 | a ² *Tr | . | . | . | 9 | 9 | . | . | 9 | TR*SECANG(L)**2 |
| 10 | a ³ | . | . | . | . | . | . | . | . | SECANG(L)**3 |
| 11 | sqrt(a) | . | . | 8 | 10 | 10 | . | 8 | 10 | SQRT(SECANG(L)) |
| 12 | a ² *Trz | . | . | . | . | . | . | . | . | TRZ*SECANG(L)**2 |
| 13 | Trz | . | . | . | 11 | 11 | . | . | 11 | TRZ |
| 14 | sqrt(Tr) | . | . | . | 8 | . | . | . | . | |
| 15 | | . | . | . | . | . | . | . | . | |
| 16 | | . | . | . | . | . | . | . | . | |

Water Vapor:

| j | predictor | CrIS | | | | NAST | | | | |
|---|-----------|------------------|---|----|----|------------------|----|----|----|--------|
| | | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | |
| 1 | W*a | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | WJUNKA |
| 2 | sqrt(W*a) | 2 | . | . | . | . | . | 2 | 3 | WJUNKR |

| | | | | | | | | | | | | | |
|----|----------------|--|----|----|----|---|--|---|----|----|----|--|-----------------------|
| 3 | W*a*W/Wz | | 3 | 8 | 8 | . | | . | 8 | 8 | 8 | | WJUNKZ |
| 4 | W*a*dT | | 4 | 3 | 3 | 3 | | 3 | 3 | 3 | 4 | | WJUNKA*DT |
| 5 | (W*a)^2 | | 5 | 5 | 5 | . | | 4 | 5 | 4 | 5 | | WJUNKS |
| 6 | sqrt(W*a)*dT | | 6 | . | . | . | | . | . | 5 | 6 | | WJUNKR*DT |
| 7 | root^4(W*a) | | 7 | . | . | . | | . | . | 10 | 7 | | WJUNK4 |
| 8 | sqrt(W*a)*W/Wz | | 8 | . | . | . | | . | . | 9 | 12 | | WJUNKZ/WJUNKR |
| 9 | (W*a)^3 | | 9 | 9 | 9 | . | | . | 9 | 6 | 10 | | WJUNKS*WJUNKA |
| 10 | W | | 10 | 10 | 10 | . | | . | . | . | 2 | | A_W |
| 11 | W*a*dT* dT | | . | . | . | . | | . | . | . | . | | WJUNKA*DT*ABS(DT) |
| 12 | W*a*Ox*a | | . | 4 | 4 | . | | . | 4 | . | . | | WJUNKA*OJUNKX |
| 13 | W*a*(Ox*a)^2 | | . | . | . | . | | . | . | . | . | | WJUNKA*OJUNKX**2 |
| 14 | sqrt(W*a)*Mz*a | | . | . | . | . | | . | . | . | . | | WJUNKR*MJUNKZ |
| 15 | W*a^2 | | . | . | . | . | | 7 | 10 | 7 | 9 | | WJUNKA*SECANG(L) |
| 16 | W*a*Cz*a | | . | . | . | . | | . | . | . | 11 | | WJUNKA*AZ_C*SECANG(L) |
| 17 | W*a^2*dT | | . | . | . | . | | . | . | . | 13 | | WJUNKA*DT*SECANG(L) |
| 18 | (W*a)^3/2 | | 11 | 2 | 2 | 2 | | 2 | 2 | . | . | | WJUNKA*WJUNKR |
| 19 | (W*a)^3/2*dT | | . | 7 | 7 | . | | 5 | 7 | . | . | | WJUNKA*WJUNKR*DT |
| 20 | (W*a)^3/2*W/Wz | | . | 11 | 11 | . | | . | 11 | . | . | | WJUNKZ*WJUNKR |
| 21 | (W*a)^5/4 | | . | 6 | 6 | . | | 6 | 6 | . | . | | WJUNKA*WJUNK4 |
| 22 | (W*a)^2*W/Wz | | . | . | . | . | | . | . | 11 | . | | WJUNKA*WJUNKZ |
| 23 | W^2*a | | . | . | . | . | | . | . | 13 | . | | WJUNKA*A_W |
| 24 | (W*a)^7/4 | | . | . | . | . | | . | . | . | . | | WJUNKS/WJUNK4 |
| 25 | (W*a)^3/4 | | . | . | . | . | | . | . | 12 | . | | WJUNKA/WJUNK4 |
| 26 | | | . | . | . | . | | . | . | . | . | | |
| 27 | | | . | . | . | . | | . | . | . | . | | |
| 28 | | | . | . | . | . | | . | . | . | . | | |
| 29 | | | . | . | . | . | | . | . | . | . | | |
| 30 | | | . | . | . | . | | . | . | . | . | | |

Ozone:

| j | predictor | CrIS | | | | NAST | | | | | | | |
|----|----------------|------------------|---|----|----|------------------|----|----|----|---|---|--|------------------------|
| | | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | | | | |
| 1 | O*a | | 1 | 1 | 1 | 1 | | 1 | 1 | 1 | 1 | | OJUNKA |
| 2 | sqrt(O*a) | | 2 | 2 | 2 | . | | 2 | 2 | 2 | 2 | | OJUNKR |
| 3 | O*a*dT | | 3 | 3 | 3 | . | | 3 | 3 | 3 | 3 | | OJUNKA*DT |
| 4 | (O*a)^2 | | 4 | 4 | 4 | . | | 4 | 4 | . | 4 | | OJUNKA*OJUNKA |
| 5 | sqrt(O*a)*dT | | 5 | 5 | 5 | . | | . | 5 | . | 5 | | OJUNKR*DT |
| 6 | O*a*O/Ox | | . | 6 | 6 | . | | . | 6 | . | 6 | | OJUNKZ*A_0 |
| 7 | sqrt(O*a)*O/Ox | | . | 7 | 7 | . | | . | 7 | . | . | | OJUNKR*A_0/XZ_0 |
| 8 | O*a*Oz/Ox | | . | 8 | 8 | . | | . | . | . | . | | OJUNKZ*AZ_0 |
| 9 | O*a*sqrt(Ox*a) | | . | 9 | 9 | . | | . | 9 | . | . | | OJUNKA*SQRT(OJUNKX) |
| 10 | O*a*TOz*a | | . | 10 | 10 | . | | . | 10 | . | . | | OJUNKA*TAZ_0*SECANG(L) |
| 11 | (O*a)^3/2 | | . | . | 11 | . | | . | . | . | 9 | | OJUNKA*OJUNKR |
| 12 | a^2*O | | . | . | . | . | | 5 | . | . | . | | OJUNKA*SECANG(L) |
| 13 | (O*a)^2*O/Ox | | . | . | . | . | | . | 8 | . | . | | OJUNKA*OJUNKZ*A_0 |
| 14 | (O*a)^3 | | . | . | . | . | | . | . | . | 7 | | OJUNKA**3 |
| 15 | O | | . | . | . | . | | . | . | . | 8 | | A_0 |
| 16 | | | . | . | . | . | | . | . | . | . | | |

Carbon Monoxide:

Appendix B

Description of the Phillip's dataset

Phillips, N., J. Susskind and L.M. McMillin 1988. Results of a joint NOAA/NASA sounder simulation study. J. Atmos. Oceanic Tech. v.5 p.44- 56.

This is a set of 1600 profiles. 1 profile (2717) is excluded because it is a bad profile (Tsurf, latitude, and %land were set to be invalid)

On 4/24/97 I rebuilt the dataset and ensured that no profile had a relative humidity greater than 98%.

| | | Land | | Land | |
|----------|-------|--------|--------|--------|--------|
| latitude | total | Winter | Summer | Winter | Summer |
| -30.00 | 170 | 41 | 44 | 36 | 49 |
| -20.00 | 116 | 25 | 25 | 31 | 35 |
| -10.00 | 92 | 31 | 30 | 22 | 9 |
| .00 | 136 | 21 | 2 | 60 | 53 |
| 10.00 | 108 | 28 | 27 | 27 | 26 |
| 20.00 | 178 | 38 | 57 | 40 | 43 |
| 30.00 | 266 | 66 | 66 | 67 | 67 |
| 40.00 | 268 | 68 | 68 | 66 | 66 |
| 50.00 | 265 | 66 | 66 | 67 | 66 |
| all | 1599 | 384 | 385 | 416 | 414 |

100 profiles for selected for simulation experiments. These profiles have the following characteristics.

| | | Land | | Land | |
|----------|-------|--------|--------|--------|--------|
| latitude | total | Winter | Summer | Winter | Summer |
| -30.00 | 8 | 2 | 2 | 1 | 3 |
| -20.00 | 9 | 1 | 2 | 3 | 3 |
| -10.00 | 8 | 3 | 3 | 2 | 0 |
| 0.00 | 8 | 1 | 0 | 4 | 3 |
| 10.00 | 8 | 2 | 0 | 3 | 3 |
| 20.00 | 12 | 3 | 6 | 3 | 0 |
| 30.00 | 17 | 5 | 5 | 2 | 5 |
| 40.00 | 15 | 3 | 5 | 3 | 4 |
| 50.00 | 15 | 3 | 3 | 4 | 5 |
| all | 100 | 23 | 26 | 25 | 26 |

Appendix C

Description of η model: NA,NB,NC,DB,DC,DD,DP

Each file in this dataset is a single scan lines of 45 AMSU FOR's. They were generated from the NCEP ETA model and contain 2-layer and 2-formation clouds for daytime 'D' and nighttime 'N' cases 'A', 'B', 'C', 'D', and 'DP'. There is a training ensemble used to train the NOAA regression of that era.

| file | description |
|-----------|-------------|
| cld2lr.NA | |
| cld2lr.NC | |
| cld2lr.DC | |
| cld2lr.DD | |
| cld2lr.DP | |
| cld2fm.NA | |
| cld2fm.NB | |
| cld2fm.NC | |
| cld2fm.DB | |
| cld2fm.DC | |
| cld2fm.DD | |

Appendix D

Description of the Nov. 1996 dataset (16 scan lines)

The AIRS orbital simulation is derived from a global simulation which is based on the output from a version of the operational GCM from NOAA NCEP. The fields in the GCM are 2.5 degree by 2.5 degree gridded data from a spectral model of NCEP (*e.g.*, Juang, H.-M. H., S.-Y. Hong and M. Kanamitsu, 1997, "The NCEP regional spectral model: an update." *Bulletin Amer. Meteor. Soc.* **78**, p.2125). The temperature, ozone and the liquid water are at 29 levels from 1015 mb to 3 mb. The water vapor profiles are at 12 levels from 1015 to 300 mb. A GOES IR image for Nov. 5, 1996 is shown in Figure 1.

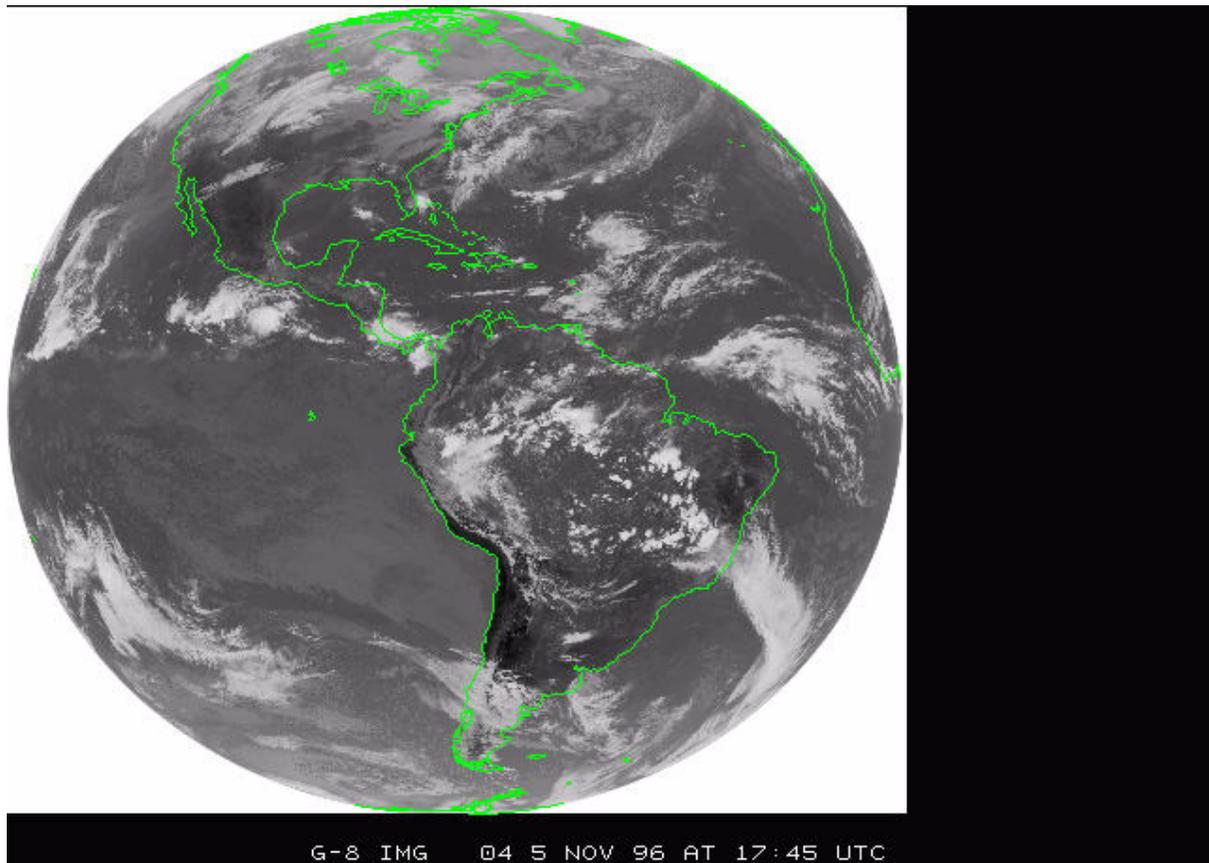


Figure 1: GOES IR image for Nov. 5, 1996

In order to generate a profile, the four surrounding grid points have to be found for the time period immediately preceding and the time period following the given time. A bilinear interpolation method is used to compute the forecast fields at the given location for the two time periods. Then the time interpolation of the forecast data is performed to get the data at the location and at the given time. A vertical interpolation of forecast fields was performed to generate the fields in the vertical layers of the AIRS retrieval software.

Note that this linear interpolation of 2.5 degree by 2.5 degree gridded data produces smooth fields. However the cloud clearing algorithm depends on the contrasts between AIRS/MHS footprints within an AMSU footprint. Therefore the cloud fraction was randomized at the end of the interpolation process. The artificially smooth fields, especially water vapor fields and surface parameters, will be studied and fixed later, if necessary.

The top of the forecast model data is at 300 mb for water vapor profile and 3 mb for others. The water vapor is extrapolated above 300 mb by multiplying the mixing ratio by $(p/300)^{**3}$, where p is pressure in milli-bar, and then converted to molecular column density. The UARS climatology is appended to the temperature profile above 3 mb. The AIRS orbital dataset has the following salient features:

- implementation of surface topography and variable surface pressure
- daytime and nighttime conditions
- $T(p)$, $q(p)$, $O_3(p)$ from surface to 0.005 mB
- cloud liquid water profiles (only affects microwave)

- multiple level cloudy conditions with spectrally varying cloud emissivity and reflectivity, consistent with atmospheric conditions.
- variable surface skin temperature, surface emissivity, and surface reflectivity.
- variable land coverage, with coast-lines, islands, lakes, etc.
- orbital simulation with simulated scan lines with variable viewing angle and solar zenith angle.

One simulation simplification has been made to help the vendors. In a real scan line of data, each of the 9 MHS and 9 CrIS spots that are co-located within a given AMSU-A footprint will have its own zenith angle. Some algorithms (*e.g.*, cloud clearing) need to “correct” the IR radiances to a single zenith angle. An algorithm to “correct” the group of 9 AIRS spots to the center AIRS footprint (*i.e.*, footprint# 5 of the 9 co-located footprints) has been developed by the AIRS Science Team for the AIRS instrument; however, we assume the vendors have not developed this algorithm. Even if the vendor algorithm existed, it would be for their CrIS footprint pattern and NOT the AIRS pattern that has been simulated here.

For these reasons, the data in each file are grouped sequentially (*i.e.*, 9 spots of the MHS file and 9 spots of CrIS file are associated with 1 AMSU spot). All 9 of the CrIS spots are simulated at the zenith angle of the central spot of CrIS (# 5/9) and all 9 of the MHS spots are simulated at the zenith angle of the MHS’s central spot (which is identical to CrIS in this simulation). The AMSU-A will have one zenith angle which is not equal to the either CrIS or MHS. This is because AMSU-A steps at 3 1/3 degree and MHS steps at 1.1 degree.

In all cases, the geophysical water vapor and ozone profiles were converted from layer column densities back to mixing ratio (units of gram per gram of dry air) at the specified 100 levels. These mixing ratio files replace the original column density files as the truth. The new level.2 files contain the following information

The new AIRS Science Team dataset is quite challenging, especially with regard to clouds

D.1 16 track training ensemble, ~/level2/TR.trn

The AIRS Science Team simulated a full orbit of data (6h forecast for Nov. 6, 1996) for the 705 km EOS PM-1 orbit. This orbit consists of 22,248 AMSU footprints (741.6 AMSU scan lines) and 200,232 AIRS and HSB footprints (2224.8 AIRS scan lines).

For training purposes, given the smooth nature of the horizontal structure in this simulation, the full orbit is not necessary. To make the data file and training process more manageable, we have selected the center CrIS (AIRS) footprint (footprint # 5 of the 9 CrIS footprints co-located within the AMSU footprint) of every 5th AMSU scan line from the orbit. Therefore, there are 30 CrIS footprints per AMSU scan line or 4470 profiles in total.

In Figure 2, we illustrate the distribution of the selected scan lines within the orbit. The daytime profiles are shown in red and nighttime profiles are shown in blue. Table 1 shows the distribution of day/night and land/ocean profiles.

No radiances will accompany this ensemble of data. We have provided all the code (*e.g.*, see ~/src/main/comp_rad.F or ~/src/main/simrad.F) and data-files necessary to compute radiances consistent with the data described in item # 2, below.

The distribution of profiles within the training dataset file, ~/level2/TR.trn, is shown in Table D.1. There are a total of 4470 profiles distributed throughout the orbit shown in Figure 1. The column labeled “total/ocean+land” shows the distribution of all profiles as a function of latitude. The table further divides the profiles into ocean and land and daytime or nighttime profiles. For example, between 30.0 to 39.9 northern latitude there are 253 profiles, of which 2 are ocean (1 day and 1 night) and 251 are land (125 day and 126 night).

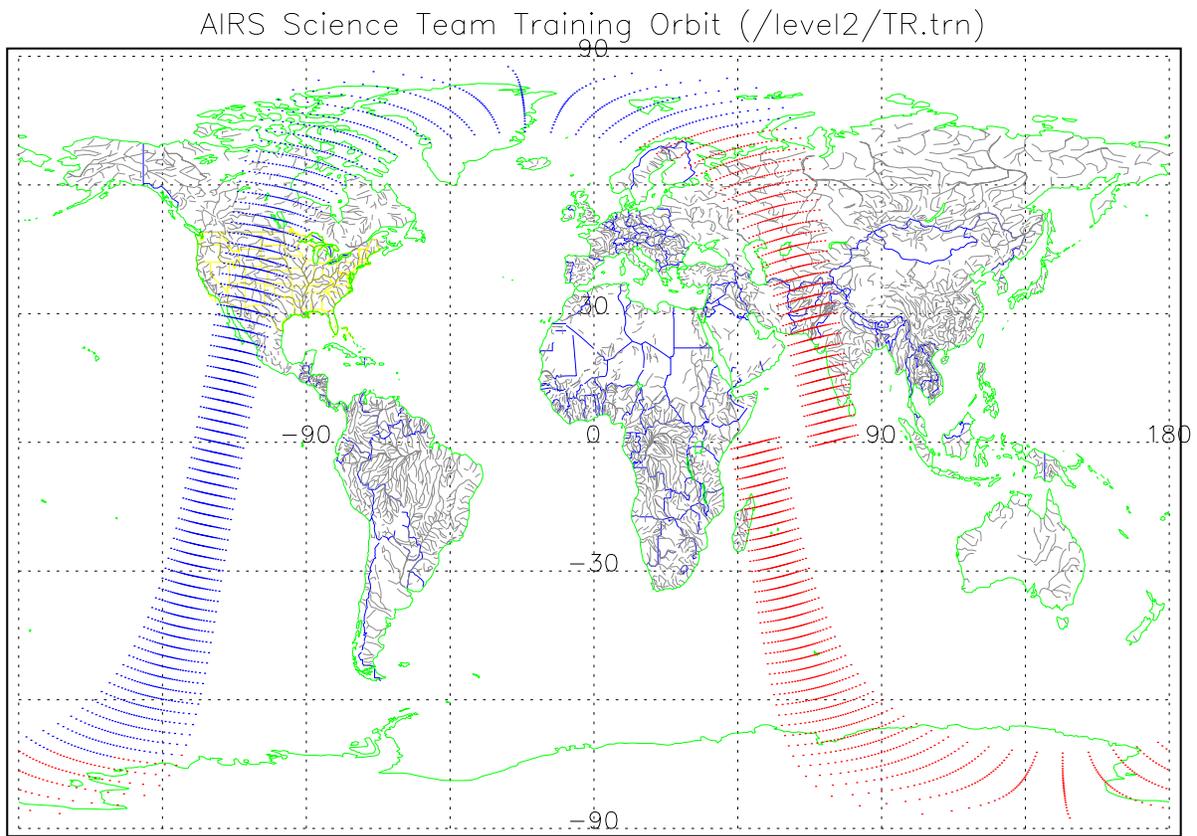


Figure D.1: Distribution of profiles in the training dataset.

Table D.1: Distribution of Training Profiles for the AST Sixteen Scan Line Set

| | | total | ocean | | | land | | |
|----------|-----|------------|-------|------|-------|-------|-----|-------|
| latitude | | ocean+land | total | day | night | total | day | night |
| -90 | -81 | 83 | 22 | . | 22 | 61 | . | 61 |
| -80 | -71 | 332 | 126 | 38 | 88 | 206 | . | 206 |
| -70 | -61 | 282 | 247 | 140 | 107 | 35 | . | 35 |
| -60 | -51 | 262 | 262 | 131 | 131 | . | . | . |
| -50 | -41 | 255 | 252 | 128 | 124 | 3 | . | 3 |
| -40 | -31 | 250 | 250 | 124 | 126 | . | . | . |
| -30 | -21 | 256 | 254 | 128 | 126 | 2 | . | 2 |
| -20 | -11 | 250 | 239 | 125 | 114 | 11 | . | 11 |
| -10 | -1 | 264 | 264 | 127 | 137 | . | . | . |
| 0 | 9 | 265 | 261 | 127 | 134 | 4 | . | 4 |
| 10 | 19 | 252 | 221 | 125 | 96 | 31 | 1 | 30 |
| 20 | 29 | 253 | 108 | 77 | 31 | 145 | 48 | 97 |
| 30 | 39 | 253 | 2 | 1 | 1 | 251 | 125 | 126 |
| 40 | 49 | 255 | 12 | . | 12 | 243 | 128 | 115 |
| 50 | 59 | 262 | 21 | 19 | 2 | 241 | 113 | 128 |
| 60 | 69 | 281 | 86 | 65 | 21 | 195 | 76 | 119 |
| 70 | 79 | 332 | 216 | 181 | 35 | 116 | 114 | 2 |
| 80 | 90 | 83 | 57 | 57 | . | 26 | 26 | . |
| total | | 4470 | 2900 | 1593 | 1307 | 1570 | 631 | 939 |

D.2 Selected 16 scan lines

Sixteen AMSU scan lines of geophysical data, selected from an orbit on the day prior to the training ensemble (6h forecast for Nov. 5, 1996 from 8:30 to 9:54), were used in IPO and AIRS Science Team simulations. The 16 selected AMSU scan lines are arranged in pairs starting with #1 and #2 in India. The location and some properties of the scan lines are shown in Table D.3 and Table D.3. The index # within the orbit (orb#) is shown in the tables for reference to the original AIRS Science Team orbital data file.

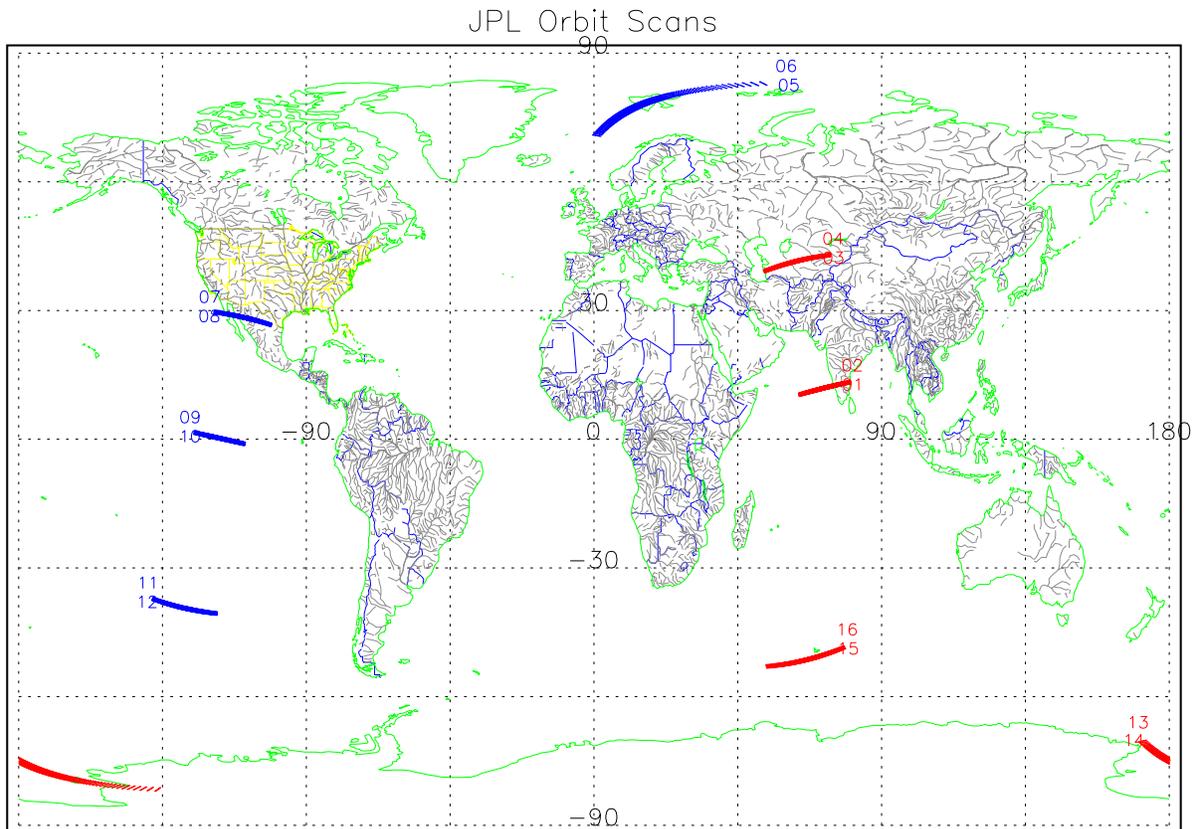


Figure D.2: Location of 16 profiles in original simulation set

The program `~/src_util/simrad.F` can be used to generate these radiances from namelist files

Table D.2: Location of the AST Sixteen Scan Lines

| scan # | orb # | Day or Night | # of land | starting | | ending | | Ts | Location |
|--------|-------|--------------|-----------|----------|------|--------|------|---------|----------------------------|
| | | | | Lat | Long | Lat | Long | | |
| 1 | 25 | D | 68 | 10 | 80 | 13 | 80 | 300 | Indian Ocean/India |
| 2 | 26 | D | 69 | 10 | 80 | 13 | 80 | 300 | Indian Ocean/India |
| 3 | 87 | D | 255 | 38 | 53 | 42 | 73 | 290-275 | Kazakh (former USSR) |
| 4 | 88 | D | 251 | 38 | 53 | 42 | 73 | 290-275 | Kazakh (former USSR) |
| 5 | 171 | N | 66 | 70 | 1 | 82 | 53 | 277-266 | N.polar, Svalbard Isl. |
| 6 | 172 | N | 45 | 70 | 1 | 82 | 53 | 277-266 | N.polar, Svalbard Isl. |
| 7 | 311 | N | 206 | -27 | -101 | -29 | -118 | 295-291 | Baha, Mexico |
| 8 | 312 | N | 188 | -27 | -101 | -29 | -118 | 295-291 | Baha, Mexico |
| 9 | 370 | N | 0 | 0 | -109 | 1 | -125 | 295-297 | Eastern Pacific |
| 10 | 371 | N | 0 | 0 | -109 | 1 | -125 | 295-297 | Eastern Pacific |
| 11 | 453 | N | 0 | -40 | -118 | -37 | -138 | 284-287 | S.E. Pacific |
| 12 | 454 | N | 0 | -40 | -118 | -37 | -138 | 284-287 | S.E. Pacific |
| 13 | 539 | D | 48 | -81 | -135 | -70 | +170 | 257-249 | Antarctica |
| 14 | 540 | D | 42 | -81 | -135 | -70 | +170 | 257-249 | Antarctica |
| 15 | 633 | D | 0 | -53 | 54 | -48 | 78 | 274-277 | S.In.Ocean, Kerguelen Isl. |
| 16 | 634 | D | 0 | -53 | 54 | -48 | 78 | 274-277 | S.In.Ocean, Kerguelen Isl. |

Table D.3: Cloud and Surface Pressure Properties of the AST Sixteen Scan Lines

| scan # | orb # | cloud fraction (%) | | | pcloud(1) (mb) | | | pcloud(2) (mb) | | | Psurf (mb) | | |
|--------|-------|--------------------|------|------|----------------|-----|-----|----------------|-----|-----|------------|------|------|
| | | min | max | avg | min | max | avg | min | max | avg | min | max | avg |
| 1 | 25 | 1.0 | 52.7 | 26.0 | 99 | 129 | 117 | 672 | 672 | 672 | 937 | 1018 | 994 |
| 2 | 26 | 1.2 | 51.6 | 21.8 | 101 | 130 | 116 | 672 | 673 | 672 | 942 | 1021 | 995 |
| 3 | 87 | 0.0 | 90.1 | 28.8 | 213 | 447 | 318 | 365 | 700 | 492 | 843 | 1022 | 977 |
| 4 | 88 | 0.0 | 92.7 | 32.3 | 215 | 447 | 321 | 362 | 700 | 478 | 871 | 1020 | 983 |
| 5 | 171 | 56.0 | 96.2 | 87.3 | 239 | 343 | 301 | 477 | 556 | 513 | 970 | 991 | 980 |
| 6 | 172 | 45.9 | 94.3 | 86.8 | 241 | 331 | 297 | 483 | 559 | 512 | 969 | 993 | 978 |
| 7 | 311 | 31.2 | 68.5 | 48.7 | 137 | 164 | 151 | 882 | 884 | 883 | 831 | 1011 | 922 |
| 8 | 312 | 26.2 | 67.3 | 43.2 | 134 | 165 | 151 | 882 | 884 | 883 | 826 | 1013 | 926 |
| 9 | 370 | 7.7 | 40.7 | 21.3 | 90 | 159 | 126 | 806 | 876 | 829 | 1005 | 1010 | 1008 |
| 10 | 371 | 2.0 | 41.9 | 18.3 | 90 | 162 | 122 | 809 | 877 | 844 | 1006 | 1010 | 1008 |
| 11 | 453 | 68.6 | 91.4 | 78.9 | 250 | 288 | 271 | 349 | 368 | 353 | 1012 | 1021 | 1017 |
| 12 | 454 | 67.8 | 92.8 | 81.5 | 249 | 288 | 271 | 348 | 374 | 352 | 1013 | 1021 | 1018 |
| 13 | 539 | 3.8 | 86.1 | 54.4 | 195 | 510 | 359 | 452 | 716 | 507 | 911 | 983 | 960 |
| 14 | 540 | 23.7 | 90.8 | 60.5 | 189 | 521 | 364 | 453 | 716 | 498 | 930 | 986 | 962 |
| 15 | 633 | 7.7 | 81.4 | 53.1 | 237 | 790 | 426 | 419 | 773 | 610 | 983 | 992 | 986 |
| 16 | 634 | 5.5 | 82.1 | 49.0 | 240 | 790 | 426 | 421 | 772 | 609 | 983 | 992 | 986 |

D.3 Selected 8track datasets

The even and odd scan lines of the 16 scan line set are quite redundant. We build a merged dataset from the even scan lines (AMSU scan lines #2, #4, ..., #16) of the original JPL orbital dataset for Nov. 5, 1996. All the truth and radiances are stored in #/8track. Once the baseline dataset was created a number of special files were built to test the retrieval sensitivity to variability in the 9 IR FOV's. The name of the dataset reflects what variability is turned on. The original data set was simulated for cloudy conditions and had variability in geophysical parameters (T(p),q(p),O3(p)), surface parameters, and in cloud parameters (Pcld and numcld was variable on 9 FOV's). This dataset did not originally include trace gas variability.

| Even data-set | CLD? | var. Geop? | var. Surf? | var. Cldy? | Trace? |
|---------------|------|---------------|---------------|---------------|--------|
| EDGSC.XX.asc | c1D | YES | YES | YES | NO |
| EDGS.XX.asc | c1D | YES | YES | NO | NO |
| EDG.XX.asc | c1D | YES | NO | NO | NO |
| ED.XX.asc | c1D | NO | NO | NO | NO |
| ERGS.XX.asc | c1R | YES | YES | n/a | NO |
| ERG.XX.asc | c1R | YES | NO | n/a | NO |
| ER.XX.asc | c1R | NO | NO | NO | NO |
| ERGST_XX.asc | c1R | NO | NO | NO | YES |

XX usage

--- -----

TR0 original truth (level.2) file f/ JPL
 MIT original MIT first guess (level-2.MW.02, ...)
 MIT2 newer MIT first guess (newerlevel-2.MW.02, ...)
 NOAA newest NOAA first guess (newestlevel-2.INIT_RET-F_M0.02, ...)
 TR truth (level.2) f/ simrad program
 AI AIRS level.1
 CR CrIS level.1
 AM AMSU level.1
 MH AMSU-B, MHS, HSB level.1
 AT ATMS level.1

original JPL truth files located in: /bc3c/wljmb/8track/truth

```
-----
L1scan_all.dat      scan information for simrad
EDGSC.TR0.asc      mergel2.exe tr_orbit.lst EDGSC.TR0.asc
EDGS.TR.asc        avgspots.exe EDGSC.TR0.asc EDGS.TR0 -geovar -surfvar
EDG.TR.asc         avgspots.exe EDGSC.TR0.asc EDGS.TR0 -geovar

ERGS.TR0.asc       mergel2.exe tr_orbit.lst ERGS.TR0.asc -clear
ERG.TR0.asc        avgspots.exe ERGS.TR0.asc ERG.TR0 -geovar
```

first guess files located in: /bc3c/wljmb/8track/fg/

```
-----
EDGSC_MIT.asc      sgiout/level-2.MW.xx
```

```

EDGSC_MIT2.asc  sgiout/newerlevel-2.MW.xx
EDGSC_NOAA.asc  sgiout/newestlevel-2.INIT_RET-F_MO.xx
ERGS_MIT2.asc   sgiout/newerlevel-2.MW.xx w/ Liq,clouds set to zero

```

```

EDGSC_NOAArcc.asc  $bin/mergel2.exe noaa_rcc.lst EDGSC_NOAArcc.asc
                   sgiout/ourcclevel-2.INIT_RET-F_MO.xx

```

AIRS level.1 files located in: /bc3c/wljmb/8track/airs/

```

-----
EDGSC.AI.bin AIRS level.1 for cloudy conditions
EDGSC.AM.asc AMSU level.1 for cloudy conditions
EDGSC.MH.asc MHS level.1 for cloudy conditions
EDGSC.TR.bin simrad truth file (echo of EDGSC.TR0.asc)

EDGS.AI.bin AIRS level.1 for cloudy conditions, cloud e's, r's, P's fixed
EDGS.AM.asc AMSU level.1 for cloudy conditions
EDGS.MH.asc MHS level.1 for cloudy conditions
EDGS.TR.bin simrad truth file (echo of EDGS.TR0.asc)

EDG.AI.bin AIRS level.1 for cloudy conditions, surf+cloud e's, r's, P's fixed
EDG.AM.asc AMSU level.1 for cloudy conditions
EDG.MH.asc MHS level.1 for cloudy conditions
EDG.TR.bin simrad truth file (echo of EDG.TR0.asc)

ERGS.AI.bin AIRS level.1 for clear conditions
ERGS.AM.asc AMSU level.1 for clear conditions
ERGS.MH.asc MHS level.1 for clear conditions
ERGS.TR.bin simrad truth file (echo of ERGS.TR0.asc)

ERG.AI.bin AIRS level.1 for clear conditions, surf e's, r's, P's fixed
ERG.AM.asc AMSU level.1 for clear conditions
ERG.MH.asc MHS level.1 for clear conditions
ERG.TR.bin simrad truth file (echo of ERG.TR0.asc)

```

CrIS level.1 files located in: /bc3c/wljmb/8track/cris/

```

-----
EDGSC.CR.bin CrIS level.1 for cloudy conditions
EDGSC.AM.asc AMSU level.1 for cloudy conditions
EDGSC.MH.asc MHS level.1 for cloudy conditions
EDGSC.TR.bin simrad truth file (echo of EDGSC.TR0.asc)

ERGS.CR.bin CrIS level.1 for clear conditions
ERGS.AM.asc AMSU level.1 for clear conditions
ERGS.MH.asc MHS level.1 for clear conditions
ERGS.TR.bin simrad truth file (echo of ERGS.TR0.asc)

```

270 profiles per AMSU scan line

```

Truth is 6 hr forecast 11/ 5/96 8:33:00 to 9:54:00
Guess is 18 hr forecast 11/ 5/96 8:33:00 to 9:54:00
Training is 11/ 6/96 8:30:00 to 10:08:00

```

| file# | orb# | D/N | #land | start | | | end | | Ts | Location |
|-------|------|-----|-------|----------|----------|----------|------|---------|---------------------------|----------|
| | | | | Lat/Long | Lat/Long | Lat/Long | | | | |
| 2 | 26 | D | 66 | 10 | 80 | 13 | 80 | 300 | Indian Ocean/India | |
| 4 | 88 | D | 251 | 38 | 53 | 42 | 73 | 290-275 | Kazakh (former USSR) | |
| 6 | 172 | N | 45 | 70 | 1 | 82 | 53 | 277-266 | N.polar Svalbard Isl. | |
| 8 | 312 | N | 188 | 27 | -101 | 29 | -118 | 295-291 | Baha California, Mexico | |
| 10 | 371 | N | 0 | 0 | -109 | 1 | -125 | 295-297 | Eastern Pacific | |
| 12 | 454 | N | 0 | -40 | -118 | -37 | -138 | 284-287 | S.E. Pacific | |
| 14 | 540 | D | 42 | -81 | -135 | -70 | +170 | 257-249 | Antartica | |
| 16 | 634 | D | 0 | -53 | 54 | -48 | 78 | 274-277 | S.In.Ocean, Kerguelen Is. | |

| file# | orb# | .cloud fraction. | | | ...pcloud(1)... | | | ...pcloud(2)... | | |Psurf.... | | |
|-------|------|------------------|------|------|-----------------|-----|-----|-----------------|-----|-----|----------------|------|------|
| | | min | max | <> | min | max | <> | min | max | <> | min | max | <> |
| 2 | 26 | 1.2 | 51.6 | 21.8 | 101 | 130 | 116 | 672 | 673 | 672 | 942 | 1021 | 995 |
| 4 | 88 | 0.0 | 92.7 | 32.3 | 215 | 447 | 321 | 362 | 700 | 478 | 871 | 1020 | 983 |
| 6 | 172 | 45.9 | 94.3 | 86.8 | 241 | 331 | 297 | 483 | 559 | 512 | 969 | 993 | 978 |
| 8 | 312 | 26.2 | 67.3 | 43.2 | 134 | 165 | 151 | 882 | 884 | 883 | 826 | 1013 | 926 |
| 10 | 371 | 2.0 | 41.9 | 18.3 | 90 | 162 | 122 | 809 | 877 | 844 | 1006 | 1010 | 1008 |
| 12 | 454 | 67.8 | 92.8 | 81.5 | 249 | 288 | 271 | 348 | 374 | 352 | 1013 | 1021 | 1018 |
| 14 | 540 | 23.7 | 90.8 | 60.5 | 189 | 521 | 364 | 453 | 716 | 498 | 930 | 986 | 962 |
| 16 | 634 | 5.5 | 82.1 | 49.0 | 240 | 790 | 426 | 421 | 772 | 609 | 983 | 992 | 986 |

Appendix E

NOAA 1988 and 1989 radiosonde dataset

In 1997 and 1998 two sets of files were constructed by Mitch Goldberg at NOAA/NESDIS for the use in simulation studies of advanced sounders. Chris Barnet at GSFC appended some additional products from Joel Susskind's TOVS processing system and distributed these files to vendors studying the NPOESS CrIS instrument.

| filename | # obs | original IPO document | delivery date | format |
|--------------|-------|-----------------------|---------------|------------|
| noaa_88.asc | 8344 | f97_WPTB.ps | 10/9/97 | 66 layers |
| noaa_89.asc | 8434 | f97_WPTB.ps | 10/9/97 | 66 layers |
| noaa_88b.asc | 7547 | f98_WPTB.ps | 9/21/98 | 100 layers |
| noaa_89b.asc | 7662 | f98_WPTB.ps | 9/21/98 | 100 layers |

E.1 The 66 level NOAA 1988 and 1989 dataset

The first dataset that was built used the 66 layer pressure grid. The files can be read using `openl2.F` and `readl2.F` (see Section E.3 and 11.1) and used the format "L2V1B" (original format, index = 1 in `openl2`). There were 2 years built, the original idea being that 1989 could be used for training (*i.e.*, the dependent dataset) and 1988 could be used for testing (*i.e.* independent dataset).

E.1.1 Statistics of the 1988 radiosonde dataset

The original 1988 profiles have the statistical distribution given in Table E.1 for profiles over ocean and Table E.2 for profiles over land. The first column (repeated in Table E.2) is the total number of profiles in the latitude region (ocean + land). The 2nd column is the total number of those profiles (day or night) which are ocean (Table E.1) or land (Table E.2). The next column is the total number of profiles that are nighttime conditions and then a column for daytime conditions for ocean (Table E.1) and land (Table E.2). The next 8 columns are the distribution of those profiles in each season (Winter, Spring, Summer, Fall) and day/night domain for ocean (Table E.1) or land (Table E.2).

- profiles w/ $P_s \leq 1000 = 1207$ 14.47%

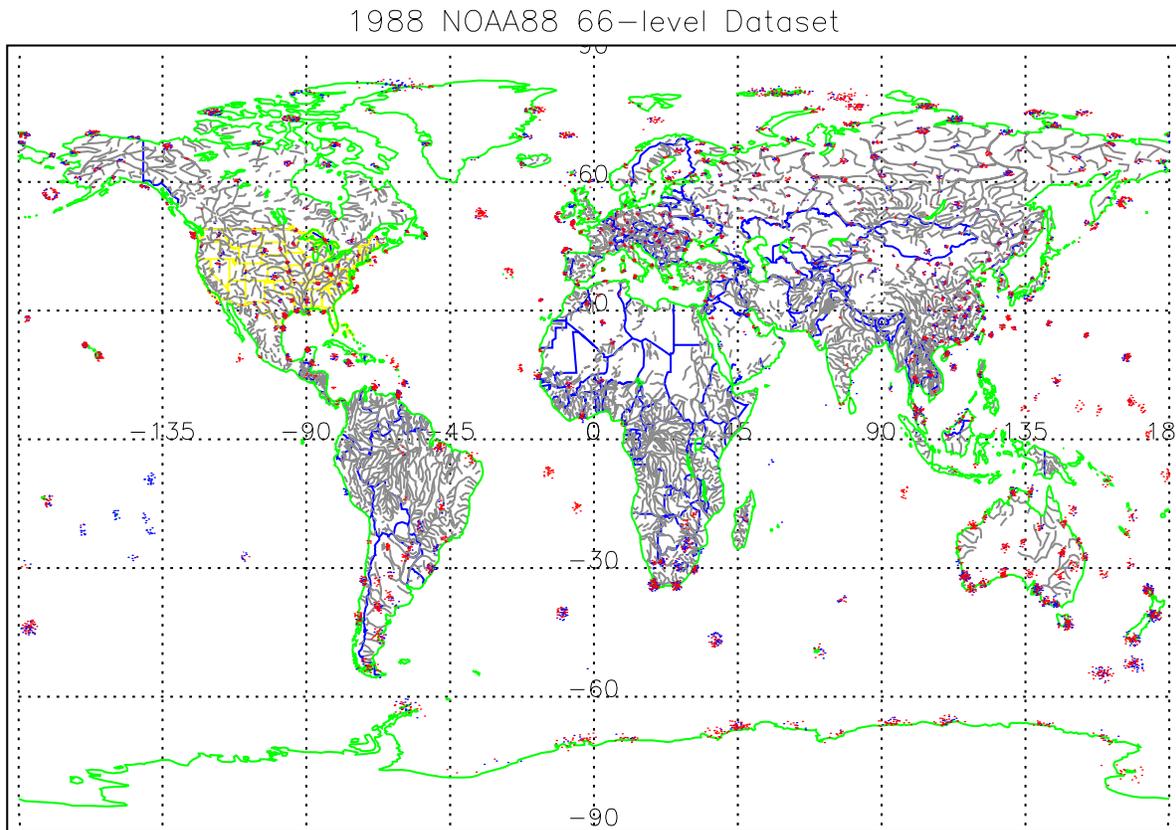


Figure E.1: Location of the original 66 level 1988 profiles

- profiles w/ LiQ water = 0 0.00%
- daytime profiles = 6607 79.18%

E.1.2 Statistics of the 1989 radiosonde dataset

- profiles w/ Ps \geq 1000 = 1192 14.13%
- profiles w/ LiQ water = 0 0.00%
- daytime profiles = 6095 72.27%

The distributions of the original 1989 dataset are given in Table E.3 for ocean and Table E.4 for land.

E.2 The 100 level NOAA 1988 and 1989 dataset.

We re-issued the NOAA-88 geophysical dataset on 100 levels. The file is now called “~/level2/noaa88b.asc”. This dataset still includes the old surface and cloud models and has no liquid water. The temperature,

moisture, and ozone profiles were extended to 0.005 mb in a reasonable manner. No radiance is provided for this product; however, the vendors can compute radiances themselves.

This file is a subset of the original data file because not all profiles could be co-located to rocketsondes. The total number of profiles is now 7457 of the original 8344 profiles. The distribution of these profiles is shown in Table 4, for ocean profiles, and Table 5, for land profiles.

The subset ensembles used in the 1997 IGS study for the IPO, consisting of 65 profiles in the winter ensemble and 81 profiles in the summer ensemble, are affected by the missing profiles. In the winter ensemble there are now 54 profiles and 74 in the summer ensemble.

For the noaa88b.asc dataset the level.2 file utilizes some of the spare fields

1. r spare(1) is the SBUV co-location distance in degrees; if this value is -999.9 then the previous profile's SBUV profile was used,
2. r spare(2) is the SBUV co-location time difference in hours

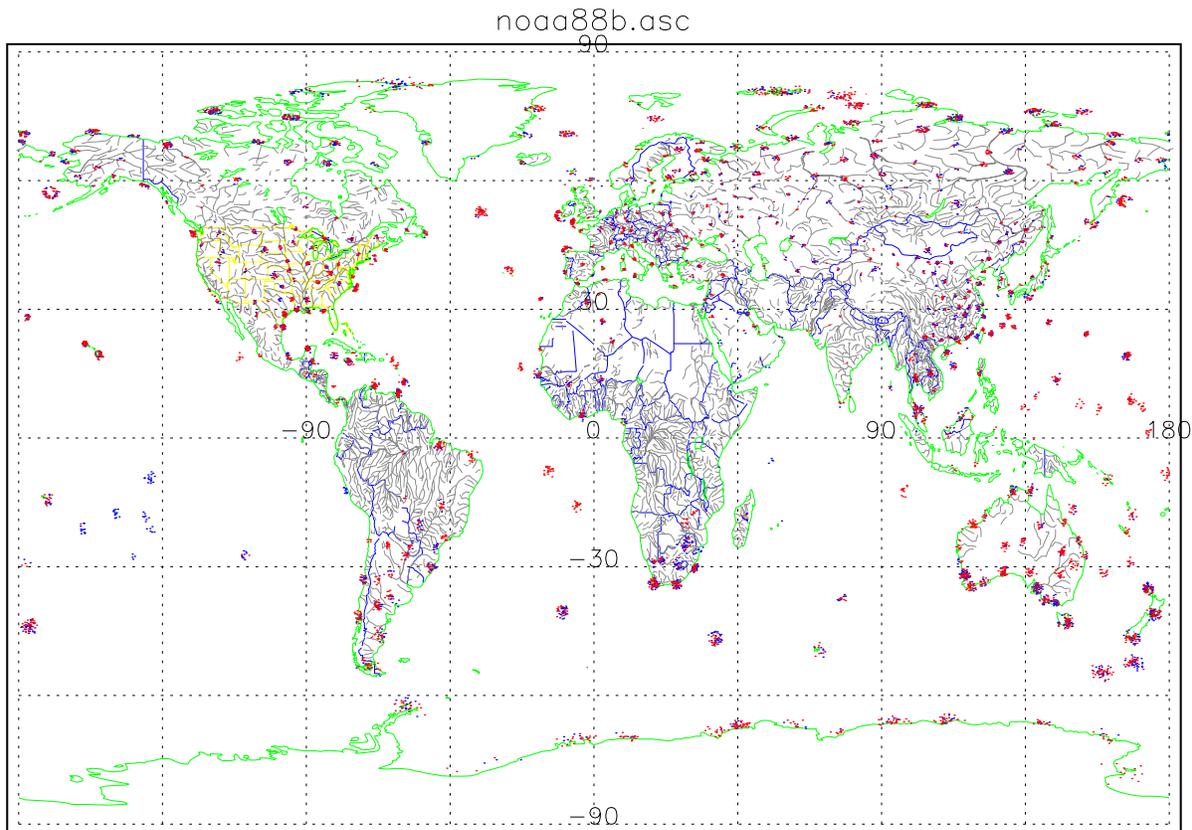


Figure E.2: Location of profiles in the NOAA 1988b dataset

E.2.1 Statistics of the 1988 radiosonde dataset

- profiles w/ Ps \leq 1000 = 1055 13.98%
- profiles w/ LiQ water = 0 0.00%

- daytime profiles = 4403 58.34%

In the following Tables, “D” are daytime profiles, as determined from the date, time, and location, and “N” are nighttime profiles. “total” refers to “ocean” plus “land” profiles.

E.2.2 Statistics of the 1989 radiosonde dataset

Table E.1: NOAA88 66 level set: distribution of profiles over OCEAN

| latitude | | total | ocean | ocean | ocean | winter | | spring | | summer | | fall | |
|----------|-------|-------|-------|-------|-------|--------|-----|--------|-----|--------|-----|------|-----|
| lower | upper | all | D+N | N | D | N | D | N | D | N | D | N | D |
| -90 | -81 | 2 | 1 | . | 1 | . | . | . | . | . | . | . | 1 |
| -80 | -71 | 55 | 11 | 2 | 9 | . | 6 | 2 | . | . | . | . | 3 |
| -70 | -61 | 288 | 73 | 25 | 48 | 6 | 24 | 14 | . | 4 | . | 1 | 24 |
| -60 | -51 | 160 | 126 | 63 | 63 | 24 | 33 | 25 | . | 6 | . | 8 | 30 |
| -50 | -41 | 439 | 156 | 84 | 72 | 12 | 42 | 34 | . | 20 | 3 | 18 | 27 |
| -40 | -31 | 801 | 138 | 71 | 67 | 19 | 31 | 22 | 4 | 11 | 10 | 19 | 22 |
| -30 | -21 | 473 | 68 | 44 | 24 | 20 | 8 | 7 | 5 | 4 | 3 | 13 | 8 |
| -20 | -11 | 300 | 110 | 43 | 67 | 12 | 25 | 13 | 14 | 9 | 14 | 9 | 14 |
| -10 | -1 | 137 | 55 | 14 | 41 | 5 | 14 | 5 | 13 | 1 | 9 | 3 | 5 |
| 0 | 9 | 261 | 59 | 9 | 50 | 2 | 11 | 2 | 12 | 3 | 14 | 2 | 13 |
| 10 | 19 | 650 | 197 | 79 | 118 | 22 | 31 | 19 | 35 | 13 | 17 | 25 | 35 |
| 20 | 29 | 732 | 72 | 33 | 39 | 18 | 11 | 4 | 6 | 4 | 18 | 7 | 4 |
| 30 | 39 | 678 | 83 | 23 | 60 | 5 | 8 | 5 | 17 | 5 | 24 | 8 | 11 |
| 40 | 49 | 821 | 111 | 33 | 78 | 15 | 7 | 1 | 31 | 3 | 30 | 14 | 10 |
| 50 | 59 | 603 | 276 | 92 | 184 | 42 | 19 | 1 | 82 | 2 | 68 | 47 | 15 |
| 60 | 69 | 924 | 8 | 6 | 2 | 3 | . | . | 1 | . | 1 | 3 | . |
| 70 | 79 | 820 | 164 | 45 | 119 | 26 | 2 | . | 57 | . | 57 | 19 | 3 |
| 80 | 90 | 200 | 29 | 10 | 19 | 7 | . | . | 9 | . | 10 | 3 | . |
| total | | 8344 | 1737 | 676 | 1061 | 238 | 272 | 154 | 286 | 85 | 278 | 199 | 225 |

Table E.2: NOAA88 66 level set: distribution of profiles over LAND

| latitude | | total | land | land | land | winter | | spring | | summer | | fall | |
|----------|-------|-------|------|------|------|--------|------|--------|------|--------|-----|------|-----|
| lower | upper | all | D+N | N | D | N | D | N | D | N | D | N | D |
| -90 | -81 | 2 | 1 | . | 1 | . | 1 | . | . | . | . | . | . |
| -80 | -71 | 55 | 44 | 7 | 37 | . | 34 | 5 | . | 1 | . | 1 | 3 |
| -70 | -61 | 288 | 215 | 47 | 168 | 8 | 126 | 31 | . | 6 | . | 2 | 42 |
| -60 | -51 | 160 | 34 | 7 | 27 | . | 20 | 5 | . | 2 | . | . | 7 |
| -50 | -41 | 439 | 283 | 131 | 152 | 24 | 89 | 50 | 1 | 38 | 8 | 19 | 54 |
| -40 | -31 | 801 | 663 | 293 | 370 | 56 | 169 | 104 | 33 | 107 | 51 | 26 | 117 |
| -30 | -21 | 473 | 405 | 158 | 247 | 28 | 101 | 50 | 25 | 70 | 51 | 10 | 70 |
| -20 | -11 | 300 | 190 | 65 | 125 | 7 | 49 | 31 | 35 | 24 | 30 | 3 | 11 |
| -10 | -1 | 137 | 82 | 18 | 64 | 3 | 20 | 4 | 15 | 9 | 22 | 2 | 7 |
| 0 | 9 | 261 | 202 | 70 | 132 | 46 | 57 | 10 | 22 | 4 | 35 | 10 | 18 |
| 10 | 19 | 650 | 453 | 198 | 255 | 77 | 99 | 36 | 59 | 35 | 43 | 50 | 54 |
| 20 | 29 | 732 | 660 | 291 | 369 | 125 | 101 | 45 | 100 | 46 | 94 | 75 | 74 |
| 30 | 39 | 678 | 595 | 253 | 342 | 126 | 99 | 7 | 100 | 13 | 69 | 107 | 74 |
| 40 | 49 | 821 | 710 | 296 | 414 | 174 | 91 | 5 | 136 | 11 | 137 | 106 | 50 |
| 50 | 59 | 603 | 327 | 145 | 182 | 89 | 25 | 1 | 76 | . | 62 | 55 | 19 |
| 60 | 69 | 924 | 916 | 442 | 474 | 269 | 20 | . | 182 | 1 | 239 | 172 | 33 |
| 70 | 79 | 820 | 656 | 318 | 338 | 202 | 6 | . | 193 | . | 123 | 116 | 16 |
| 80 | 90 | 200 | 171 | 85 | 86 | 58 | . | . | 53 | . | 29 | 27 | 4 |
| total | | 8344 | 6607 | 2824 | 3783 | 1292 | 1107 | 384 | 1030 | 367 | 993 | 781 | 653 |

Table E.3: NOAA89 66 level set: distribution of profiles over OCEAN

| latitude | | total | ocean | ocean | ocean | winter | | spring | | summer | | fall | |
|----------|-------|-------|-------|-------|-------|--------|-----|--------|-----|--------|-----|------|-----|
| lower | upper | all | D+N | N | D | N | D | N | D | N | D | N | D |
| -90 | -81 | 1 | 1 | . | 1 | . | 1 | . | . | . | . | . | . |
| -80 | -71 | 47 | 21 | 5 | 16 | 2 | 16 | 3 | . | . | . | . | . |
| -70 | -61 | 297 | 136 | 52 | 84 | 28 | 83 | 18 | . | 6 | . | . | 1 |
| -60 | -51 | 261 | 220 | 127 | 93 | 36 | 75 | 20 | 1 | 53 | 4 | 18 | 13 |
| -50 | -41 | 443 | 189 | 111 | 78 | 39 | 56 | 21 | 3 | 45 | 9 | 6 | 10 |
| -40 | -31 | 1006 | 267 | 140 | 127 | 49 | 76 | 12 | 8 | 68 | 24 | 11 | 19 |
| -30 | -21 | 601 | 164 | 89 | 75 | 41 | 31 | 7 | 5 | 34 | 20 | 7 | 19 |
| -20 | -11 | 360 | 172 | 62 | 110 | 17 | 41 | 12 | 7 | 31 | 45 | 2 | 17 |
| -10 | -1 | 129 | 62 | 25 | 37 | 10 | 10 | . | 5 | 14 | 16 | 1 | 6 |
| 0 | 9 | 302 | 116 | 20 | 96 | 4 | 54 | 3 | 12 | 8 | 24 | 5 | 6 |
| 10 | 19 | 705 | 292 | 95 | 197 | 35 | 82 | 8 | 27 | 30 | 65 | 22 | 23 |
| 20 | 29 | 563 | 102 | 45 | 57 | 15 | 13 | 1 | 5 | 10 | 21 | 19 | 18 |
| 30 | 39 | 849 | 78 | 38 | 40 | 12 | 14 | 1 | 3 | 4 | 11 | 21 | 12 |
| 40 | 49 | 728 | 101 | 29 | 72 | 17 | 14 | . | 10 | 3 | 39 | 9 | 9 |
| 50 | 59 | 651 | 161 | 60 | 101 | 47 | 28 | 1 | 14 | 4 | 53 | 8 | 6 |
| 60 | 69 | 773 | 16 | 9 | 7 | 2 | 2 | 1 | 1 | . | 3 | 6 | 1 |
| 70 | 79 | 615 | 209 | 47 | 162 | 25 | 16 | . | 22 | 1 | 112 | 21 | 12 |
| 80 | 90 | 103 | 32 | 4 | 28 | . | 6 | . | 3 | . | 19 | 4 | . |
| total | | 8434 | 2339 | 958 | 1381 | 379 | 618 | 108 | 126 | 311 | 465 | 160 | 172 |

Table E.4: NOAA89 66 level set: distribution of profiles over LAND

| latitude | | total | ocean | ocean | ocean | winter | | spring | | summer | | fall | |
|----------|-------|-------|-------|-------|-------|--------|------|--------|-----|--------|------|------|-----|
| lower | upper | all | D+N | N | D | N | D | N | D | N | D | N | D |
| -90 | -81 | 1 | . | . | . | . | . | . | . | . | . | . | . |
| -80 | -71 | 47 | 26 | 7 | 19 | 6 | 19 | . | . | 1 | . | . | . |
| -70 | -61 | 297 | 161 | 54 | 107 | 23 | 100 | 21 | . | 9 | . | 1 | 7 |
| -60 | -51 | 261 | 41 | 10 | 31 | 1 | 20 | 2 | 1 | 6 | 3 | 1 | 7 |
| -50 | -41 | 443 | 254 | 111 | 143 | 44 | 98 | 7 | 4 | 48 | 18 | 12 | 23 |
| -40 | -31 | 1006 | 739 | 324 | 415 | 95 | 229 | 37 | 25 | 156 | 62 | 36 | 99 |
| -30 | -21 | 601 | 437 | 111 | 326 | 26 | 161 | 17 | 24 | 61 | 78 | 7 | 63 |
| -20 | -11 | 360 | 188 | 64 | 124 | 24 | 44 | 9 | 7 | 25 | 50 | 6 | 23 |
| -10 | -1 | 129 | 67 | 19 | 48 | 1 | 12 | 2 | 2 | 13 | 26 | 3 | 8 |
| 0 | 9 | 302 | 186 | 70 | 116 | 32 | 51 | 4 | 10 | 20 | 34 | 14 | 21 |
| 10 | 19 | 705 | 413 | 171 | 242 | 89 | 99 | 13 | 17 | 40 | 82 | 29 | 44 |
| 20 | 29 | 563 | 461 | 228 | 233 | 87 | 71 | 13 | 15 | 39 | 95 | 89 | 52 |
| 30 | 39 | 849 | 771 | 334 | 437 | 157 | 122 | 5 | 38 | 56 | 215 | 116 | 62 |
| 40 | 49 | 728 | 627 | 280 | 347 | 142 | 81 | 7 | 39 | 17 | 171 | 114 | 56 |
| 50 | 59 | 651 | 490 | 201 | 289 | 101 | 57 | 3 | 39 | 7 | 153 | 90 | 40 |
| 60 | 69 | 773 | 757 | 231 | 526 | 145 | 58 | 1 | 107 | 9 | 331 | 76 | 30 |
| 70 | 79 | 615 | 406 | 122 | 284 | 63 | 45 | . | 50 | 1 | 169 | 58 | 20 |
| 80 | 90 | 103 | 71 | 20 | 51 | 9 | 8 | . | 9 | 2 | 30 | 9 | 4 |
| total | | 8434 | 6095 | 2357 | 3738 | 1045 | 1275 | 141 | 387 | 510 | 1517 | 661 | 559 |

Table E.5: NOAA88b (100 level set), distribution of profiles over OCEAN

| latitude | | total | ocean | ocean | ocean | winter | | spring | | summer | | fall | |
|----------|-------|-------|-------|-------|-------|--------|-----|--------|-----|--------|-----|------|-----|
| lower | upper | all | D+N | N | D | N | D | N | D | N | D | N | D |
| -90 | -81 | 2 | 1 | . | 1 | . | . | . | . | . | . | . | 1 |
| -80 | -71 | 41 | 9 | 2 | 7 | . | 4 | 2 | . | . | . | . | 3 |
| -70 | -61 | 244 | 58 | 20 | 38 | 5 | 20 | 12 | . | 2 | . | 1 | 18 |
| -60 | -51 | 148 | 117 | 59 | 58 | 24 | 31 | 23 | . | 4 | . | 8 | 27 |
| -50 | -41 | 404 | 149 | 81 | 68 | 12 | 41 | 33 | . | 18 | 3 | 18 | 24 |
| -40 | -31 | 735 | 128 | 65 | 63 | 18 | 29 | 21 | 4 | 9 | 8 | 17 | 22 |
| -30 | -21 | 420 | 60 | 37 | 23 | 16 | 7 | 6 | 5 | 4 | 3 | 11 | 8 |
| -20 | -11 | 280 | 104 | 42 | 62 | 12 | 22 | 12 | 14 | 9 | 13 | 9 | 13 |
| -10 | -1 | 124 | 49 | 12 | 37 | 4 | 12 | 5 | 12 | 1 | 9 | 2 | 4 |
| 0 | 9 | 236 | 55 | 8 | 47 | 2 | 11 | 2 | 12 | 2 | 12 | 2 | 12 |
| 10 | 19 | 584 | 176 | 67 | 109 | 18 | 25 | 16 | 34 | 10 | 15 | 23 | 35 |
| 20 | 29 | 650 | 67 | 30 | 37 | 16 | 10 | 3 | 6 | 4 | 17 | 7 | 4 |
| 30 | 39 | 597 | 75 | 19 | 56 | 4 | 6 | 4 | 16 | 5 | 24 | 6 | 10 |
| 40 | 49 | 746 | 103 | 28 | 75 | 11 | 6 | 1 | 31 | 3 | 28 | 13 | 10 |
| 50 | 59 | 554 | 255 | 85 | 170 | 38 | 18 | 1 | 77 | 2 | 62 | 44 | 13 |
| 60 | 69 | 852 | 7 | 5 | 2 | 3 | . | . | 1 | . | 1 | 2 | . |
| 70 | 79 | 754 | 156 | 40 | 116 | 22 | 1 | . | 56 | . | 56 | 18 | 3 |
| 80 | 90 | 176 | 26 | 8 | 18 | 7 | . | . | 8 | . | 10 | 1 | . |
| total | | 7547 | 1595 | 608 | 987 | 212 | 243 | 141 | 276 | 73 | 261 | 182 | 207 |

Table E.6: NOAA88b (100 level set), distribution of profiles over LAND

| latitude | | total | land | land | land | winter | | spring | | summer | | fall | |
|----------|-------|-------|------|------|------|--------|-----|--------|-----|--------|-----|------|-----|
| lower | upper | all | D+N | N | D | N | D | N | D | N | D | N | D |
| -90 | -81 | 2 | 1 | . | 1 | . | 1 | . | . | . | . | . | . |
| -80 | -71 | 41 | 32 | 5 | 27 | . | 26 | 4 | . | . | . | 1 | 1 |
| -70 | -61 | 244 | 186 | 43 | 143 | 7 | 110 | 30 | . | 4 | . | 2 | 33 |
| -60 | -51 | 148 | 31 | 7 | 24 | . | 18 | 5 | . | 2 | . | . | 6 |
| -50 | -41 | 404 | 255 | 113 | 142 | 23 | 84 | 42 | 1 | 30 | 7 | 18 | 50 |
| -40 | -31 | 735 | 607 | 273 | 334 | 53 | 154 | 97 | 29 | 98 | 43 | 25 | 108 |
| -30 | -21 | 420 | 360 | 143 | 217 | 26 | 85 | 45 | 24 | 62 | 48 | 10 | 60 |
| -20 | -11 | 280 | 176 | 62 | 114 | 7 | 44 | 29 | 32 | 23 | 29 | 3 | 9 |
| -10 | -1 | 124 | 75 | 18 | 57 | 3 | 18 | 4 | 11 | 9 | 22 | 2 | 6 |
| 0 | 9 | 236 | 181 | 59 | 122 | 39 | 53 | 8 | 20 | 2 | 33 | 10 | 16 |
| 10 | 19 | 584 | 408 | 184 | 224 | 69 | 83 | 34 | 52 | 34 | 40 | 47 | 49 |
| 20 | 29 | 650 | 583 | 248 | 335 | 104 | 91 | 40 | 89 | 36 | 88 | 68 | 67 |
| 30 | 39 | 597 | 522 | 230 | 292 | 115 | 87 | 6 | 88 | 11 | 58 | 98 | 59 |
| 40 | 49 | 746 | 643 | 268 | 375 | 156 | 80 | 5 | 125 | 11 | 124 | 96 | 46 |
| 50 | 59 | 554 | 299 | 134 | 165 | 81 | 21 | 1 | 72 | . | 57 | 52 | 15 |
| 60 | 69 | 852 | 845 | 392 | 453 | 239 | 18 | . | 179 | 1 | 227 | 152 | 29 |
| 70 | 79 | 754 | 598 | 284 | 314 | 182 | 6 | . | 187 | . | 108 | 102 | 13 |
| 80 | 90 | 176 | 150 | 73 | 77 | 47 | . | . | 47 | . | 28 | 26 | 2 |
| total | | 7547 | 5952 | 2536 | 3416 | 1151 | 979 | 350 | 956 | 323 | 912 | 712 | 569 |

E.3 How to read the NOAA88 & 89 files

The files used an early version of geophysical files for the AIRS Science Team. These files are located in a FORTRAN library called “src_gsfc”

| file | topic (root= \sim /code/src_gsfc) |
|-------------|-------------------------------------|
| openl2_b.F | open L2 file and read pressure grid |
| readl2_b.F | read L2 record |
| writel2_b.F | write L2 record |

These files contain all the information for computation of a infrared or microwave radiance.

E.3.1 openl2_b.F & readl2_b.F: Documentation for reading L2 files

All L2 files are read and written by a single I/O driver. The files are opened (for read or write) with openl2_b.F and then records are read by readl2_b.F and written by writl2_b.F. L2 files come in a variety of shapes and sizes. For example,

- Truth files for simulation contain a truth for each AIRS FOV. Therefore, there are 270 FOV’s per AMSU Scan line.
- First Guess files (f/ MIT, NOAA) contain one record per AMSU FOR.
- Retrieval files (output from airs_b.F) contain one record per AMSU FOR.

In all cases, the file contains all the geophysical information necessary to compute a radiance. For a retrieval file, this might include items that were NOT retrieved, such as cloud emissivity, but when a cloudy radiance was computed the cloud emissivity that was specified is included in the file.

level.2 header record (read with \sim /src/src_gsfc/openl2.F)

- version: a 5 character version identification (a5).
All four of these files have a version = “L2V1B” which has 1 frequency scale for cloud spectral properties and does NOT include the gas amounts for CO₂, CO, and CH₄.
- numpro: the number of profiles in this file (i10)
- nemis: number surface emissivity/reflectivity wavenumber hinge points (see **getemis1.f** for use of these hinge points) (i10)
- ncemis: number cloud emissivity/reflectivity wavenumber hinge points (i10)
- nl2lev: number of layers in the vertical profiles (i10)
- ndescr: number of description lines in header (i10)
- Pobs(L), L=1, nl2lev: pressure value at bottom level of layer (8f10.3). The top of the atmosphere is **implicitly** set to 0.005 mb (*i.e.*, Pobs(0)=0.005).
- descr(i), i=1,ndescr: 80 character labels, (a80)

level.2 data record (read with \sim /src/src_gsfc/readl2.F)

Each record (profile or FOV) within the level.2 file has the following format. This is the specification for GPV3a (iver=17):

- idprof: 8 character profile identification, usually 1st 4 characters are descriptive and the last 4 characters are a number (a8).
- latitude (± 90) and east longitude (± 180) in degrees (2f9.3)
- year, month, day, hour, minute, second: (i6,4i3,f7.3). NOTE: the year is now 4 digits and the sec is now a real*4 value.
- pland: fraction of land (0 = all ocean, 1 = 100% land) (f10.5)
- Psurf: surface pressure in milli-bar (f10.2)

- Tsurf: skin temperature (f10.2)
- emismw: microwave emissivity (f10.5) at 50.3 GHz.
- topog: topography parameter (f10.2) . This is the altitude (km) above sea level if pland < 0.4. Otherwise, it is the ocean depth in meters.
- nclد: number of cloud layers (i3) NOTE: The format of nclد has changed from (i4) to (i3).
- Smw_surfclass: the microwave surface type
- N_Smw: number of microwave spectral points
- nemis: For versions that have this entry, this value overwrites the number of surface emissivity hinge points in the header.
- ncemis: For versions that have this entry, this value overwrites the number of cloud emissivity hinge points in the header.
- satheight: The satellite altitude from the surface in kilo-meters.
- viewang: The satellite view-angle. This angle is defined to be the IDEAL view angle and is related to the satellite zenith angle, θ , at the surface by

$$\text{viewang} \equiv \frac{180}{\pi} \cdot \sin^{-1} \left[\frac{(R_e)}{(R_e + \text{satheight})} \cdot \sin \left(\frac{\pi \cdot \theta}{180} \right) \right] \quad (\text{E.1})$$

where, $R_e = 6371.004$ (see src.jpl/paramet.com, REARTH)

- solzang: The solar zenith angle in degrees.
- avgCO2: The average value of CO₂ expressed in parts-per- million. If CO2ppm(L) is not specified than CO2ppm(L)=avgCO2. Otherwise, this value is ignored.
- pclدtop(i), clدfrc(i): cloud top pressure and cloud fraction for i=1,nclد (f10.2,f10.5)
- tair(L), wcd(L), ocd(L), wlcd(L), ciw(L), cocd(L), ch4cd(L), CO2ppm(L) for L=1,numlev: (f9.3,3e15.5,i2,3e15.5)
 - Tair(L): temperature in Kelvin at Pobs(L)
 - wcd(L): water layer column density in units of molecules/cm² of water between levels at Pobs(L-1) and Pobs(L).
 - ocd(L): ozone layer column density in units of molecules/cm² of ozone between levels at Pobs(L-1) and Pobs(L).
 - wlcd(L): the liquid water layer column density in units of molecules/cm² of liq. water between levels at Pobs(L-1) and Pobs(L).
 - ciw(L): ice/liquid flag for gpдl(L) (0=water, 1=ice)
 - cocd(L) the carbon monoxide (CO) layer column density in units of molecules/cm² of CO between levels at Pobs(L-1) and Pobs(L).
 - ch4cd(L) the methane (CH₄) water layer column density in units of molecules/cm² of water between levels at Pobs(L-1) and Pobs(L).
 - co2ppm(L) is the dry mixing ratio of Carbon dioxide (CO₂) in units of parts-per-million.
- Smw_freq(1:N_Smw), Smw_emiss(1:N_Smw): If N_Smw is greater than zero then the spectral array of microwave hinge points will be read in. The emissivity for each channel is computed according to these hinge points and the function in src.gsfc/getemis2.F.
- freqemis(i), emisir(i), rhoir(i), i=1,nemis: the IR surface emissivity and reflectivity defined at frequency hinge points (f10.3,2f10.5). The emissivity for each channel is computed according to these hinge points and the function in src.gsfc/getemis2.F.
- clдfreq(i,k), clдemis(i,k), clдrho(i,k), i=1,ncemis, k=1,nclд: the IR cloud emissivity and reflectivity defined at frequency hinge points (f11.3,2f10.5). The emissivity for each channel is computed according to these hinge points and the function in src.gsfc/getemis2.F.
- nspr, nspi: number of spare real and integer values (2i3)

- `ispare(nspl)`: integer spare values (8i8)
The off-line retrieval uses this parameter to store rejection and clear flags as well as format of `rspare`.
see `doc/namelist.ema` and `src/ret_driver.F` to see details
- `rspare(nspl)`: real spare values (4e15.5)
The off-line retrieval uses this parameter to store OLR, COLR, rejection criteria and 9-spot cloud fractions
see `doc/namelist.ema` and `src/ret_driver.F` to see details

Appendix F

Sept. 13, 1998 AIRS Science Team Dataset

This dataset was build to coincide with the CAMEX-III AIRS day. There are NAST-M and NAST-I measurements around Andros Island which can be compared to the simulated AIRS radiances. A few AIRS graules (45 AMSU scanlines = 6 minutes of AIRS data) are located at `/ide400/cbarnet/airs/exercise/`. The granules we have are

| Granule | Latitude | | Longitude | | Solar | | % land | |
|---------|----------|-----|-----------|------|-------|-----|--------|-----|
| | start | end | start | end | start | end | start | end |
| 101 | -31 | -49 | -127 | -154 | Night | | 0.0 | 0.0 |
| 108 | 2 | 26 | 30 | 40 | 12 | 33 | 100 | 100 |
| 125 | 14 | 37 | 3 | 14 | 15 | 41 | 100 | 77 |
| 156 | -28 | -5 | -38 | -27 | 34 | 29 | 0 | 0 |
| 174 | 5 | 28 | -70 | -59 | 12 | 34 | 100 | 0 |
| 192 | 37 | 61 | -103 | -88 | 35 | 61 | 100 | 0 |
| 198 | 8 | -11 | 92 | 72 | Night | | 0 | 0 |

Appendix G

Dec. 15, 2000 AIRS Science Team Dataset

Various versions of this dataset exist. I am aware of Apr. 2000, Jun. 2000, Sep. 2000, Jan. 2001, Feb. 2001 (v.2.15), and May 2001 (v.2.20) versions.

| latitude | D+N total | Daytime | | | | | Nighttime | | | | |
|----------|--------------|---------------------|------|-------|-----|------|-----------------------|-----|-----|------|--|
| | | tot | Land | Ocean | tot | Land | Ocean | | | | |
| | day | Win | Sum | Win | Sum | ngt | Win | Sum | Win | Sum | |
| -90.00 | 154 | 154 | 0 | 154 | 0 | 0 | 0 | 0 | 0 | 0 | |
| -80.00 | 532 | 532 | 0 | 407 | 0 | 125 | 0 | 0 | 0 | 0 | |
| -70.00 | 450 | 343 | 0 | 97 | 0 | 246 | 107 | 0 | 0 | 107 | |
| -60.00 | 445 | 224 | 0 | 0 | 0 | 224 | 221 | 0 | 0 | 221 | |
| -50.00 | 422 | 213 | 0 | 14 | 0 | 199 | 209 | 0 | 14 | 195 | |
| -40.00 | 443 | 215 | 0 | 27 | 0 | 188 | 228 | 0 | 5 | 223 | |
| -30.00 | 412 | 206 | 0 | 50 | 0 | 156 | 206 | 0 | 48 | 158 | |
| -20.00 | 412 | 206 | 0 | 52 | 0 | 154 | 206 | 0 | 56 | 150 | |
| -10.00 | 409 | 204 | 0 | 37 | 0 | 167 | 205 | 0 | 41 | 164 | |
| 0.00 | 409 | 205 | 32 | 1 | 149 | 23 | 204 | 38 | 18 | 144 | |
| 10.00 | 381 | 176 | 56 | 0 | 120 | 0 | 205 | 46 | 0 | 159 | |
| 20.00 | 411 | 205 | 57 | 0 | 148 | 0 | 206 | 108 | 0 | 98 | |
| 30.00 | 353 | 176 | 72 | 0 | 104 | 0 | 177 | 54 | 0 | 123 | |
| 40.00 | 419 | 209 | 89 | 0 | 120 | 0 | 210 | 103 | 0 | 107 | |
| 50.00 | 366 | 184 | 130 | 0 | 54 | 0 | 182 | 114 | 0 | 68 | |
| 60.00 | 446 | 170 | 107 | 0 | 63 | 0 | 276 | 183 | 0 | 93 | |
| 70.00 | 485 | 0 | 0 | 0 | 0 | 0 | 485 | 173 | 0 | 312 | |
| 80.00 | 250 | 0 | 0 | 0 | 0 | 0 | 250 | 34 | 0 | 216 | |
| 90.00 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 1 | |
| ----- | ----- | --- | --- | --- | --- | --- | --- | --- | --- | --- | |
| all | 7200 | 3622 | 543 | 839 | 758 | 1482 | 3578 | 853 | 182 | 1321 | |
| | | 1222 | | | | | | | | | |