REVU

RAMS / HYPACT Evaluation and Visualization Utilities

Version 2.3.1

User's Guide

August 20, 2001

by

Craig J. Tremback
Robert L. Walko
Martin J. Bell

*ASTER Division
Mission Research Corporation
P.O. Box 466
Fort Collins, CO 80525-0466

REVU User's Guide

Contents

About REVU

This chapter describes the RAMS/HYPACT Evaluation and Visualization Utilities (***REVU***), which is the standard supported package for generating graphical representations and reformatting ***RAMS*** model output (***VAN***, the old ***RAMS*** Visualization and ANalysis Package, is no longer supported). ***REVU***'s function is to read "analysis" files written from a ***RAMS*** or ***HYPACT*** simulation, select user-specified fields and cross sections from the file data, and plot the field cross sections, or output the selected data in one of several available formats (e.g. Vis5D, GrADS, GRIB). ***REVU*** utilizes NCAR Graphics to perform most of the plotting functions. ***REVU*** can also pluck interpolated point data from the analysis files.

Similar to ***RAMS***, ***REVU*** is in a state of continual evolution. This document is the third edition of the ***REVU*** User's Guide since it was redeveloped from ***VAN*** and split from the ***RAMS*** version 3b User's Guide. It describes ***REVU*** as of August 2001. ***REVU*** version 2.3.1 is FORTRAN 90 compliant and includes several c modules, several ***RAMS*** modules and the ***RAMS*** and ***Utilities*** libraries. UNIX/Linux make commands and a system of make files and makefile includes are used for compiling the code. The make files are detailed in the section describing how to compile the ***REVU*** model.

Execution of ***REVU*** is controlled by a set of namelist variables usually contained in a file named *REVU_IN*. In order to operate ***REVU***, the user should be acquainted with the two primary means of setting parameters that control its functions. These are:

1. The various configuration parameters contained in *vcomm2.h*, which define several array dimensions controlling the capacity of the model.
2. The variables in the *REVU_IN* namelist file.

The following sections of this document describes the function and use of each of the parameters and namelist variables, and how to set appropriate values for them.

New in REVU 2.3.1

New features since ***REVU*** version 2.3.0 include:

- GRIB file format output option (works the same way as Vis5D and GrADS).
- The overlay filed may now be filled. Filling does not go beyond the specified high/low range. This allows the user to see the map in the unfilled portions. Map boundaries can be placed under or on top of the color fills. There is also an option for enhanced map boundaries.
- Color options include an array of new color fill schemes for tiles and contours and line colors for contours and vectors. They also include user defined color options, control on axis and tile colors, white background and grayscale color options and control on landmark plotting.
- Panels option (draws up to 4 plots per frame).
- Accepts 1 argument, -f, for pointing to non-standard *REVU_IN* file names, i.e.

```
revu-2.3.1 -f <namelist file>
```

- Dump option and correct plotting of soil variables.
- In addition, many of the routines now use "implicit none" statements and all c iralloc memory allocations are now done with FORTRAN 90 allocations. This has allowed us to do bounds checking on arrays has which in turn has resulted in the cleaning up of a number of known and unknown bugs.

Installing and Running REVU

The ***REVU*** version 2.3.1 code is FORTRAN 90 compliant. ***REVU*** has been compiled and run on SGI, HP, IBM, SUN, DEC-Alpha and PC-Linux workstations. If you are compiling on other platforms we would like to know what compiler options you use. To convey this information, please contact:

`rams-support@aster.com`

Keeping Up To Date with REVU

The latest version of ***REVU*** is available from:

`http://www.aster.com/revu.shtml`

Patches will be made available at:

`http://www.aster.com/revu-2.3.1/patch.shtml`

as they are made. Notification of new patches will be sent to the rams-users mailing list and are available for viewing on the Announce archive:

`http://www.aster.com/lists/announce`

You can apply to join the rams-users mailing list at:

`http://www.aster.com/lists/index.shtml`

As always, we appreciate any bug reports and compilation options that are different or not offered in the distributed make files. Please see the notes on reporting bugs before seeking help.

Installing REVU

This section describes the installation of ***REVU*** version 2.3.1. This version is compatible with ***RAMS*** version 4.3.0, ***UTILS*** version 1.0.0 and ***HYPACT*** version 1.1.0. The reader should also check the release notes in the *README_REVU-2.3.1* file. Note that all MRC/*ASTER software should be installed under a common "rams" directory (usually this directory is named *rams*, but it can be anything you wish). This directory is referred to as the *RAMSROOT* in this user manual.

Downloading the Latest Version of REVU

1. Obtain the latest version of the software:

   Either, download the UNIX gzipped tar file *revu-2.3.1.tar.gz* from the web address:

   `http://www.aster.com/revu-2.3.1/revu-2.3.1.tar.gz`

   or, download *mrc-4.3.0.tar.gz*, which contains ***REVU*** version 2.3.1 and the current ***UTILS*** and ***RAMS*** distributions, from the web address:

   `http://www.aster.com/rams-4.3.0/mrc-4.3.0.tar.gz`

2. Move the tar file into *RAMSROOT*:

   ```
   mv revu-2.3.1.tar.gz RAMSROOT
   ```

3. Unpack the contents of the tar file, ***either***:

```
gunzip -c revu-2.3.1.tar.gz | tar -xf -
```

or, if you are on a Linux machine:

```
tar -zxf revu-2.3.1.tar.gz
```

Upon completion of this step you should find the latest versions of the *README* file, source code, make files, and configuration files in the *src*, *bin* and *run* directories in *RAMSROOT*:

```
RAMSROOT/bin/README_REVU-2.3.1              - build and install files
           ./Make.revu-2.3.1
           ./dep_revu-2.3.1.mk
           ./Makefile-std
           ./include.mk-std
           ./check
           ./versions
           ./Copyright

RAMSROOT/run/REVU_IN                        - configuration files
           ./grab_in
           ./LANDMARKS

RAMSROOT/src/post/2.3.1/revu/[source]       - source code
                      ./common/[source]
                      ./common/modules/[modules]
                      ./include/[includes]
```

Downloading the Latest Patch for HYPACT

1. Obtain the latest version of the software, by ***either*** downloading the UNIX gzipped tar file *revu-2.3.1-patch.tar.gz* from the web address:

   http://www.aster.com/revu-2.3.1/revu-2.3.1-patch.tar.gz

   - Make a temporary directory and move the tar file in to that directory:

   ```
   mkdir tmp
   mv revu-2.3.1-patch.tar.gz tmp
   ```

   - Unpack the contents of the patch tar file, ***either***:

   ```
   gunzip -c revu-2.3.1-patch.tar.gz | tar -xf -
   ```

   - ***or***, if you are on a Linux machine:

   ```
   tar -zxf revu-2.3.1-patch.tar.gz
   ```

   or, download patched modules individually from:

   http://www.aster.com/revu-2.3.1/patch.shtml

4. Replace the modules in your distribution with those new versions contained in the patch, noting that since the source comes with read only permissions, you will need to modify the permissions of the module in your *RAMSROOT/src* directory before replacing it with the patch version. For example:

```
chmod u+w RAMSROOT/src/post/2.3.1/common/vplt.f90
mv src/post/2.3.1/common/vplt.f90 RAMSROOT/src/post/2.3.1/common
```

5. You may then wish to remove read permissions from the new module:

```
chmod u-w RAMSROOT/src/post/2.3.1/common/vplt.f90
```

Preparing to Compile REVU

Before you compile the software:

1. Go to the bin directory:

   ```
   cd RAMSROOT/bin
   ```

2. If you have not done so already, move the *Makefile-std* to *Makefile* (unless you are already using a non-standard version of this file, i.e. *Makefile-hypact)*:

   ```
   mv Makefile-std Makefile
   ```

3. ***Either***, <u>if this is your first time using *include.mk*</u>, move the *include.mk-std* to *include.mk* and modify it (with vi, for example) to suit your system:

   ```
   mv include.mk-std include.mk
   vi include.mk
   ```

   include.mk contains all the make environment variables that a user might need to change in order to compile the code on their machine. It is included in all of the make files using the include command.

   FIRST TIME USERS MUST CUSTOMIZE THIS FILE BEFORE THEY ATTEMPING TO MAKE

   By default, *include.mk-std* does not have the NCAR Graphics libraries or the compiler flags for any machine type switched on.

   Or, <u>if you already have a copy of *include.mk*</u> that you have modified to suit your system, check your *include.mk* with the new *include.mk-std* for software system changes in *include.mk-std* and make those changes to your *include.mk* (with vi, for example):

   ```
   diff include.mk include.mk-std
   vi include.mk
   ```

- *dep_revu-2.3.1.mk* contains all the dependencies within the ***REVU*** distribution. This means that if a file such as *vcomm2.h* is updated, all those modules that use this file will be recompiled (noting that *vcomm2.h* is not itself compiled, but included in whatever modules require it when they are compiled). All include (.h) and module (.mod) files are treated in the same way.

- If you have multiple versions of ***RAMS***, ***REVU*** or the ***UTILS*** library, the version built by default make command will be that indicated near the top of the *include.mk* file. The set of version numbers in the *include.mk* file should match a set of version numbers in the *versions* file.

- All make commands will run the shell script *check* which outputs some advice if it finds a version mismatch (it does not stop the make command from completing, although you may then get compilation errors). *check* compares the versions indicated in the *include.mk* file with the list of compatible versions in the *versions* file and the versions installed under *RAMSROOT* on your machine. If the check script fails on your machine you can remove the "check" dependency from the "all" target in each of the make files.

- If the include command does not appear to work on your machine, try replacing the include command line in each of the make files with the contents of the include file. Alternatively, you can download "GNU make" and use that instead of your platform version. "GNU make" is available from the URL:

  `http://www.gnu.org/software/make/make.html`

- All readme, make and dependency files are distributed with a version number appended. You can optionally remove this from the make file names (do not remove them from the dependency file names).

Compiling REVU

- To compile the software, use ***either***:

  ```
  Make -f Make.revu-2.3.1
  ```

  or, to use the global make file *Makefile*, enter:

  ```
  make revu
  ```

  This should produce the ***REVU*** archive libraries *revu-2.3.1.a*, the ***REVU*** executable *revu-2.3.1*, and a link from *revu* to *revu-2.3.1*.

  or, to use the global make file *Makefile* to update all the executables in your *bin* directory:

  ```
  make
  ```

  This will ensure that all the executables are up to date, noting that there are a number of source code cross dependencies in the software system.

- If you need the dummy NCAR Graphics routines, also enter:

  ```
  make ncargd
  ```

  This should produce the NCAR Graphics dummy archive library *libncarg-1.0.0.a*. Note that you need to select an alternative NCAR Graphics library set if you do not wish to use the dummies library. NCAR now distributes the NCAR Graphics libraries and utilities used by MRC/*ASTER free of charge under the GNU general public license from the URL:

  `http://ngwww.ucar.edu/ng4.2/download`

- To recompile ***REVU*** when any of the ***RAMS*** or ***UTILS*** library modules are updated, repeat the make command. If you do not "clean" your build, this will update the executable, recompiling only those codes that have been updated, or that depend on codes that have been updated.

Installing REVU

- The *Make.revu-2.3.1* "install" target will install (as a symbolic link) the executable to the *run* and *test* directories:

  ```
  make -f Make.revu-2.3.1 install
  ```

  You can modify *Make.revu-2.3.1* if you wish to install to alternative locations. You can also manually link to the executables (does the same as above):

  ```
  ln -s ../bin/revu .
  ```

  On some machines you may need to make a physical copy of the executable in your test directory:

  ```
  cp ../bin/revu-2.3.1 revu
  ```

  Don't forget to update copied executables when you remake in the bin directory otherwise your changes will not appear in your runs.

Cleaning the REVU Installation

- *Make.revu-2.3.1* also has a "clean" target that will remove built components. To clean out the compiled libraries and executable:

  ```
  make -f Make.revu-2.3.1 clean
  ```

  This should remove the ***REVU*** archive library *revu-2.3.1.a*, the ***REVU*** executable *revu-2.3.1* and the link from *revu* to *revu-2.3.1*.

- *Makefile* also contains a "clean_all" target. Entering the following will result in all libraries and executables being removed (not just the ***REVU*** components):

  ```
  make clean_all
  ```

Running REVU

1. Check the settings of the variables in the parameter statements in *vcomm2.h*, and change if necessary. If you change any of these setting you will need to recompile ***REVU***.
2. Run ***RAMS***.
3. Copy and configure *REVU_IN* for the simulation to your working directory.
4. Run the ***REVU*** executable in your working directory using ***either***:

   ```
   revu-1.3.1
   ```

   or, if using a *REVU_IN* file with a non-standard name:

   ```
   revu-2.3.1 –f REVU_IN-non-standard
   ```

6. To view or inspect the ***REVU*** output files, refer to the examples set out in the ***REVU*** test runs.

REVU Test Runs

This section describes the running of REVU version 2.3.1 for the test run posted at the URL:

http://www.aster.com/revu-2.3.1/revu-test-2.3.1.tar.gz

The UNIX gzipped tar file contains those files required to run the simulation, view the results with NCAR Graphics, Vis5D or GrADS and extract data from the analysis files. The results of each of these and the list files containing the run time screen output are post on our web site at:

`http://www.aster.com/revu-2.3.1/test`

Because of their size, these have not been made available for downloading in a single archive file. If you wish to compare you output with the output created here (on our SGI Origin 200) you can download the specific files.

Preparing the Test Run

1. Download the UNIX gzipped tar file from the above URL:

```
revu-test-2.3.1.tar.gz
```

2. Unpack the contents of the tar file in your rams root directory:

```
gunzip -c revu-test-2.3.1.tar.gz | tar -xf -
```

3. Go to the test directory and check its contents:

```
cd test
ls -l
```

This should produce the following:

```
README_REVU_TEST          Latest test release notes
LANDMARKS                 locations file for NCAR Graphics plots
REVU_IN-space-sigma-A     REVU_IN for sigma level 2 surface NCAR Graphics
                          plots from the standard analysis files
REVU_IN-space-sigma-L     REVU_IN for sigma level 2 surface NCAR Graphics
                          plots from the 'lite' analysis files
REVU_IN-space-pressure-A  REVU_IN for 700 mb pressure surface NCAR Graphics
                          plots from the standard analysis files
REVU_IN-space-vert-A      REVU_IN for vertical slice NCAR Graphics plots
                          from the standard analysis files
REVU_IN-space-tile-A      REVU_IN for land-cover surface NCAR Graphics
                          plots from the standard analysis files
REVU_IN-v5d-sigma         REVU_IN for Vis5D output of the sigma levels
REVU_IN-v5d-cartesian     REVU_IN for Vis5D output of the Cartesian levels
REVU_IN-v5d-pressure      REVU_IN for Vis5D output of the pressure levels
REVU_IN-grib-sigma        REVU_IN for GRIB output of the sigma levels
REVU_IN-grib-cartesian    REVU_IN for GRIB output of the Cartesian levels
REVU_IN-grib-pressure     REVU_IN for GRIB output of the pressure levels
REVU_IN-grads-sigma       REVU_IN for GrADS output of the sigma levels
REVU_IN-grads-cartesian   REVU_IN for GrADS output of the Cartesian levels
REVU_IN-grads-pressure    REVU_IN for GrADS output of the pressure levels
REVU_IN-dump-sigma        REVU_IN for sigma level data extraction
REVU_IN-dump-cartesian    REVU_IN for Cartesian level data extraction
REVU_IN-dump-pressure     REVU_IN for pressure level data extraction
grab_in                   locations file for data extraction (grab)
REVU_IN-grab              REVU_IN for point data extraction
ralph_in                  locations file for data extraction (ralph)
REVU_IN-ralph             REVU_IN for point extraction
revu-test                 script containing the command for the REVU test
```

If you have not put your *RAMSROOT/bin* directory in your path, go to your *RAMSROOT/bin* directory and install the executables in your test directory, or use full or relative path references in the following commands.

Viewing with NCAR Graphics

Run REVU for the example input namelist files:

```
RAMSROOT/bin/revu -f REVU_IN-space-sigma-A
RAMSROOT/bin/revu -f REVU_IN-space-sigma-L
RAMSROOT/bin/revu -f REVU_IN-space-pressure-A
RAMSROOT/bin/revu -f REVU_IN-space-vert-A
RAMSROOT/bin/revu -f REVU_IN-space-tile-A
```

Each should produce the NCAR Graphics *gmeta* files:

```
gmeta-space-sigma-A
gmeta-space-sigma-L
gmeta-space-pressure-A
gmeta-space-vert-A
gmeta-space-tile-A
```

Each time you run ***REVU*** you will overwrite the *gmeta* file. If you do not have NCAR Graphics, or compiled without it, the *gmeta* file would not be produced, even though ***REVU*** appears to run to a normal completion.

To view a "gmeta" file, run ***either***:

```
idt gmeta
```

or:

```
ctrans gmeta
```

NCAR now distributes the NCAR Graphics libraries and utilities used by MRC/*ASTER free of charge under the GNU general public license from the URL:

`http://ngwww.ucar.edu/ng4.2/download`

The NCAR Graphics **ANATYPE** = 'SPACE' runs will use the *LANDMARKS* file (if you have one) to chart locations on your horizontal slab plots.

If you wish to compare your output with ours, the *gmeta* files and output listings created here on our SGI Origin 200 are posted at:

`http://www.aster.com/revu-2.3.1/test`

Viewing with Vis5D

Run REVU for the example input namelist files:

```
RAMSROOT/bin/revu -f REVU_IN-v5d-sigma
RAMSROOT/bin/revu -f REVU_IN-v5d-cartesian
RAMSROOT/bin/revu -f REVU_IN-v5d-pressure
```

This should produce the Vis5D files from the standard analysis files:

```
a-AS-2000-07-30-120000-g1.v5d
a-AS-2000-07-30-120000-g2.v5d
a-AC-2000-07-30-120000-g1.v5d
a-AC-2000-07-30-120000-g2.v5d
a-AP-2000-07-30-120000-g1.v5d
a-AP-2000-07-30-120000-g2.v5d
```

Note the following filename conventions:

```
a-A  is the file name prefix of the analysis files (excluding path)
S    indicates that the file is on sigma surfaces
C    indicates that the file is on Cartesian surfaces
P    indicates that the file is on pressure surfaces
```

and the date indicates the beginning of the RAMS simulation.

To view the Vis5D files, run (for example):

```
<Vis5D path>/vis5d <options> v5d-c_2000-07-30-1200.g1
```

To do this you need to download and install Vis5D (free):

http://www.ssec.wisc.edu/~billh/vis5d.html

If you wish to compare your output with ours, the Vis5D files and output listings created here on our SGI Origin 200 are posted at:

http://www.aster.com/revu-2.3.1/test

Viewing with GrADS

Run REVU for the example input namelist files:

```
RAMSROOT/bin/revu -f REVU_IN-grads-sigma
RAMSROOT/bin/revu -f REVU_IN-grads-cartesian
RAMSROOT/bin/revu -f REVU_IN-grads-pressure
```

This should produce the GrADS files from the standard analysis files:

```
a-AS-2000-07-30-120000-g1.ctl
a-AS-2000-07-30-120000-g1.gra
a-AS-2000-07-30-120000-g2.ctl
a-AS-2000-07-30-120000-g2.gra
a-AC-2000-07-30-120000-g1.ctl
a-AC-2000-07-30-120000-g1.gra
a-AC-2000-07-30-120000-g2.ctl
a-AC-2000-07-30-120000-g2.gra
a-AP-2000-07-30-120000-g1.ctl
a-AP-2000-07-30-120000-g1.gra
a-AP-2000-07-30-120000-g2.ctl
a-AP-2000-07-30-120000-g2.gra
```

To view the files you need to download and install GrADS (free):

http://grads.iges.org/grads

If you wish to compare your output with ours, the GrADS files and output listings created here on our SGI Origin 200 are posted at:

http://www.aster.com/revu-2.3.1/test

Output to GRIB

Run REVU for the example input namelist files:

```
RAMSROOT/bin/revu -f REVU_IN-grib-sigma
RAMSROOT/bin/revu -f REVU_IN-grib-cartesian
RAMSROOT/bin/revu -f REVU_IN-grib-pressure
```

This should produce the GRIB files from the standard analysis files:

```
a-AC-2000-07-30-120000-g1.grb
a-AC-2000-07-30-120000-g2.grb
a-AP-2000-07-30-120000-g1.grb
a-AP-2000-07-30-120000-g2.grb
a-AS-2000-07-30-120000-g1.grb
a-AS-2000-07-30-120000-g2.grb
```

If you wish to compare your output with ours, the GRIB files and output listings created here on our SGI Origin 200 are posted at:

http://www.aster.com/revu-2.3.1/test

Dumping 2 and 3-D Fields

Extracting data from the analysis file with the **ANATYPE** = 'DUMP' option writes full 2 and 3-D analysis file fields to an ASCI file.

Run REVU for the example input namelist files:

```
RAMSROOT/bin/revu -f REVU_IN-dump-sigma
RAMSROOT/bin/revu -f REVU_IN-dump-cartesian
RAMSROOT/bin/revu -f REVU_IN-dump-pressure
```

This should produce the dumped file from the standard analysis files:

```
a-AS-2000-07-30-120000-g1.dmp
a-AS-2000-07-30-120000-g2.dmp
a-AC-2000-07-30-120000-g1.dmp
a-AC-2000-07-30-120000-g2.dmp
a-AP-2000-07-30-120000-g1.dmp
a-AP-2000-07-30-120000-g2.dmp
```

If you wish to compare your output with ours, the dump files and output listings created here on our SGI Origin 200 are posted at:

http://www.aster.com/revu-2.3.1/test

Note that the user can modify the format of the output by modifying the code in:

```
./src/post/2.3.1/common/dumpout.f90
```

Grabbing Point Data

Extracting data from the analysis file with the **ANATYPE** = 'GRAB' option interpolates the analysis file data to the locations indicated by the *grab_in* file.

Run REVU for the example input namelist files:

```
RAMSROOT/bin/revu -f REVU_IN-grab
RAMSROOT/bin/revu -f REVU_IN-ralph
```

The GRAB run obtains the locations to get data for from the *grab_in* file (*ralph_in* for the RALPH2 output option).

This should produce the grab files from the standard analysis files:

```
a-AS-2000-07-30-120000-g0.gbr
dp-s-a-AS-2000-07-30-1200     (RALPH2)
```

If you wish to compare your output with ours, the grab files and output listings created here on our SGI Origin 200 are posted at:

```
http://www.aster.com/revu-2.3.1/test
```

Reporting Bugs in REVU

Before you seek help:

1. Ensure that you have the latest version of the software. Check for patches at:

   ```
   http://www.aster.com/rams-4.3.0/patch.shtml  (RAMS)
   http://www.aster.com/revu-2.3.1/patch.shtml  (REVU)
   http://www.aster.com/utils-1.0.0/patch.shtml (UTILS)
   ```

   Notification of new patches will be sent to the rams-users mailing list and are available for viewing on the Announce archive:

   http://www.aster.com/lists/announce

   You can apply to join the rams-users mailing list at:

   http://www.aster.com/lists/index.shtml

2. If you are still having problems:

   - Note your machine type, operating system (and version) and compiler (and version if possible).

   - Copy the screen output to a file. Try using the script command:

     ```
     script -a <file>   (to start the script shell and direct the output)
     revu-2.3.1         (run programs)
     exit               (exit script shell)
     ```

     Or redirect both standard output and error to a file. For example, running rams:

     ```
     revu-2.3.1 1>&2 <file>  (Korn and Bash shells)
     revu-2.3.1 >>& <file>   (C shell)
     ```

   - Note what configuration or make files were used.

   - Send to all of this plus any other supporting information to:

     rams-support@aster.com

Future Developments in REVU

Future upgrades may include:

- Splitting of ***REVU*** into its two main components – graphics and data reformatting / extraction.
- Ability to open and operate on more that one list of files (e.g. view different runs or compute difference fields).
- Reorganization of the memory structure and addition of a buffer that that (within the buffer size), the analysis files are not reread for information that has already been read.
- Color for ***HYPACT*** particles by age, source and species, elevation or any other input parameter.
- Inclusion of a ***HYPACT*** concentration grid for more detailed viewing of inferred Lagrangian particle concentrations.
- Completion and distribution of the ***REVU*** GUI, ***RINGI***.

Further development ideas and collaborative development are welcome and can be directed to:

rams-support@aster.com

REVU Configuration Parameters

Most ***REVU*** specific parameters are set in *vcomm2.h*.

MAXFORE integer	The maximum number of variables to plot or output (default set to 30).
MAXFILS integer	The maximum number of input files (default set to 200).
MAXLOC integer	The maximum number of observation locations (default set to 1000).
MAXLEV integer	The maximum number of levels in a profile (default set to500).

REVU_IN Namelists

The namelist file *REVU_IN* contains four namelists, with the names **$CONTROL**, **$GRAB**, **$GLL** and **$STATS**. The **$CONTROL** namelist is required for all runs and specifies the general data extraction scenario and all plotting parameters if **ANATYPE** = 'SPACE'. The remaining namelists all control some portion of the non-plotting **ANATYPE** settings and are not always required.

$CONTROL Namelist

In the following documentation a background or frame refers to the plotting or output of each **CFRAME_A**, **CFRAME_B** and **CFRAME_C** element, except in the case where multiple panels are plotted into a single frame, as specified by **IPANEL**.

ANPREF charachter	The UNIX path name and prefix of the names of ***RAMS*** or ***HYPACT*** analysis files to be read and used to drive ***REVU*** (up to 128 characters). Note that the path name is optional and can be either relative or absolute (no path is equivalent to './'). Depending on what was output by RAMS or HYPACT, ANPREF could be: • <prefix>-A-<date> Standard ***RAMS*** analysis files • <prefix>-L-<date> Lite ***RAMS*** analysis files • <prefix>-M-<date> Averaged ***RAMS*** analysis files • <prefix>-B-<date> Lite averaged ***RAMS*** analysis files • <prefix>-C-<date> ***HYPACT*** analysis files <prefix> is the directory location and file prefix given by **AFILOUT** or **HYPPREF**, the ***RAMS*** and ***HYPACT*** output files name prefixes respectively. <date> is the output filename date string in which UNIX wild cards can be used, for example: <prefix>-A* will give all standard ***RAMS*** analysis files <prefix>-A-2000-10-??-000000* will give the analysis files for 0Z each day ***REVU*** finds all files in the specified directory that have that match **ANPREF** (it looks for the header files by listing **ANPREF***-head.txt) and determines their chronological order from the time suffix encoded into the filenames. The user selects some or all of these files to be plotted in the namelist variable **TVAR**.
REVPREF charachter	The UNIX path name and prefix for the ***REVU*** output files (up to 128 characters). Note that the path is optional and can be either relative or absolute (no path is equivalent to './').

ANATYPE character	Designates what type of ***REVU*** run to make. Options are: • SPACE Produce NCAR Graphics spatial axis plots of the fields specified in **CFRAME_A**, **CFRAME_B** and **CFRAME_C**. No time axis plots are available in the current version. • V5D Create Vis5D files of the fields specified in **CFRAME_A**. • GRADS Create GrADS files of the fields specified in **CFRAME_A**. • GRIB Create GRIB files of the fields specified in **CFRAME_A**. • DUMP Dump the fields specified in **CFRAME_A** in a user defined format. The format is specified in *dumpout.f90* module, which may be modified by the user. The purpose of this option is to extract fields from analysis files that are written in packed format. • GRAB Output the fields specified in **CFRAME_A** at the points specified by latitude-longitude-height locations or vertical profiles at latitude-longitude locations in RALPH2 or a user defined format. The format is specified in *grabstat.f90* module, which may be modified by the user. • STATS Do statistical comparisons between ***RAMS*** data and observations.
HEAD1 character	A 24 character long string used to title all plots.
IGRID integer array[‡]	A background dependant parameter that specifies which grid is to be processed. • If set to zero, all grids will be processed, on separate backgrounds if **ANATYPE** is set to 'SPACE'. • If set to > 0, only the specified grid will be processed. • If set to < 0, all grids finer than abs(IGRID) will be processed, on separate backgrounds if **ANATYPE** is set to 'SPACE'. If **CFRAME_A**, **CFRAME_B** and **CFRAME_C** indicate multiple backgrounds and **IGRID** is not specified for a background, then the **IGRID** for the first background will be used for the unspecified background (i.e. in most cases you only need to specify **IGRID** for the first background).
IZTRAN integer array[‡]	A background dependant parameter that, if plotting a horizontal cross-section (see **ZVAR**, **XVAR**, and **YVAR**), determines the vertical coordinate. • If set to 1, fields are output on the model's terrain-following coordinate surfaces. • If set to 2, fields are interpolated to horizontal (Cartesian) surfaces which have the same heights as the terrain-following heights of a grid point at sea level. • If set to 3, fields are interpolated to pressure surfaces. If plotting, the surface output is determined by **IPLEVEL**, otherwise the vertical extent of the field output is determined by **ZVAR**, **XVAR**, and **YVAR** acting on the standard pressure levels 1000, 925, 850, 700, 600, 500, 400, 300, 200 and 100 mb. If **CFRAME_A**, **CFRAME_B** and **CFRAME_C** indicate multiple backgrounds and **IZTRAN** is not specified for a background, then the **IZTRAN** for the first background will be used for the unspecified background (i.e. in most cases you only need to specify **IZTRAN** for the first background).

IPLEVEL integer array[†]	A background dependant parameter that, if plotting a horizontal cross-section (see **ZVAR**, **XVAR**, and **YVAR**) and **IZTRAN** is set to 3, **IPLEVEL** denotes a pressure surface (in mb) on which to generate the plot. Only standard pressure levels 1000, 925, 850, 700, 600, 500, 400, 300, 200 and 100 mb may be specified. If **CFRAME_A**, **CFRAME_B** and **CFRAME_C** indicate multiple backgrounds and **IPLEVEL** is not specified for a background, then the **IPLEVEL** for the first background will be used for the unspecified background (i.e. in most cases you only need to specify **IPLEVEL** for the first background).
MAPFILL integer	If plotting a horizontal cross-section (see **ZVAR**, **XVAR**, and **YVAR**), specifies whether a map projection will be plotted and if so, whether it will be color-filled. • If set to 0, no map will be drawn. • If set to 1, draw a map outline (filled contours and tiles will be drawn on top on the map). • If set to 2, draw a map outline and fills the land in green and water in blue. • If set to 3, as with 2, except the map outline will be drawn on top of all filled contours and tiles. • If set to <0, as with any setting of **MAPFILL**>0, but with the map outline drawn in a highlighted shadow (enhances the geographic boundaries). Users can modify the *mkmap.f90* module to draw other geographic and demographic information (the user will also need to supply the data).
IBACKGND integer	If plotting, specifies the plot background color. • If set to 1, the plot background is set to black and the foreground to white. This setting is intended for display on electronic media. • If set to 2: the plot background is set to white and the foreground to black. This setting is intended for display on electronic media. • If set to 3: the plot background is set to white and the foreground to black. This setting is intended for display on white paper. • If set to <0, as with any setting of **IBACKGND**>0, but all colors are set to the foreground color (titles, plot scales, etc) and grayscale (map fills, filled contours and tiles). Users can modify the *rcolors.f90* module to specify their own color schemes (especially relevant to setting up the color tables for printing on specific printers), noting that some user color customization is available with **COLORS** without any modifications to the code.
IPLTINFO integer	If plotting, specifies whether to draw the plot information table that appears at the bottom of each plot. • If set to 0, do not draw the information table (this maximizes the plot space for the actual plot) • If set to 1, draw the information table. • If set to 2, draw the reduced plot information (not in table).

IPANEL
integer

If plotting, specifies the number of plots drawn per frame (1 to 4).

- If set to 1, one plot is drawn on the full frame and the number of frames drawn equals the number of plots specified by **CFRAME_A**, **CFRAME_B** and **CFRAME_C**.
- If set to 2, 3 or 4, that number of plots are drawn per frame and the number of frames drawn equals the number of plots specified by **CFRAME_A**, **CFRAME_B** and **CFRAME_C** divided by the IPANEL setting. The size of each plot is one quarter the size of the plots when IPANEL is set to 1. Also, a reduced set of plot information is included for each plot and the axis appear without any labels. You can use this setting to create classic four panel plots.
- If set to 0, not plotting is done. Instead a series of tables are output with the colors that are going to be used in the first frame (i.e. taking into account the settings of **CFRAME_A**(1), **IBACKGND** and **COLORS**).

LANDMARK
character array[†]

If plotting a horizontal cross-section (see **ZVAR**, **XVAR**, and **YVAR**), controls the drawing of the landmarks specified in the *LANDMARKS* file. **LANDMARK** is a text string that has two slash delimited components, the first controlling the drawing of the markers and the second controlling the labeling of the markers. Each of these has further colon-delimited components. The maker component:

- **M** component (integer): Marker control (M0 does not mark the landmarks and M1 marks the landmarks).
- **b** component (real): Landmark separation or buffer **-** min separation between markers (b.03 gives a spacious separation and b.01 gives a tight packing).
- **t** component (integer): Controls the marker type (t1: dot, t2: plus, t3: asterisk, t4: circle, t5: cross).
- **s** component (real): Controls the scale of the markers (s1. is a good setting).
- **x** component (character string): The color of the marker. This can be any in the color table (below), or any user defined color specified by **COLORS**.

The Label components:

- **L** component (integer): Label control (L0 does not label the landmarks and L1 labels the landmarks).
- **s** component (real): Controls the size of the label characters (s.01 is a good setting).
- **x** component (character string): The color of the label. This can be any in the color table (below), or any user defined color specified by **COLORS**.

For example, a **LANDMARK** setting of

```
    LANDMARK(1)='/M1:b.03:t2:s1.:xyellow/L1:s.01:xred/',
```

Draws yellow + makers and red labels with sufficient buffer as to avoid a clutter of markers on the plots.

If **CFRAME_A**, **CFRAME_B** and **CFRAME_C** indicate multiple backgrounds and **LANDMARK** is not specified for a background, then the **LANDMARK** for the first background will be used for the unspecified background (i.e. in most cases you only need to specify **LANDMARK** for the first background).

COLORS

character array†

If plotting, specifies the re-mapping of colors in the color table. **COLORS** is a slash delimited text string. Each slash delimited component has a further two or four colon delimited components as illustrated below:

```
COLORS(1)='/axis:yellow/title0:rgb:0.0:0.4:0.0/'
```

In the first example of **COLORS(1)**, the color table element 'axis' (defined below) is redefined from its default of white on a black background, or back on a white background, to the color table element yellow (also defined below).

In the second example of **COLORS(1)**, the color table element 'title0' (defined below) is redefined in red-green-blue (RGB) color space as a dark green, a shade of green not previously defined in the color table.

```
COLORS(2)='/red:rgb:0.8:0.0:0.0/dkred:rgb:1.0:0.0:0.0/'
```

In the first example of **COLORS(2)**, the color table element 'red' (defined below) is redefined from its default RBG value to a slightly darker shade of red.

In the second example of **COLORS(2)**, a new color table element, 'dkred', defined in RGB color space.

You can redefine up to around 8 colors (the string has a maximum length of 128 characters) in either RGB of HLS (hue-lightness-saturation) color space, each as a slash delimited item.

Background independent colors:

Color	*RGB Values*			*Color*	*RGB Values*		
white	1.0	1.0	1.0	purple	1.0	0.0	1.0
black	0.0	0.0	0.0	cyan	0.0	1.0	1.0
grayblack	0.3	0.3	0.3	tan	0.86	0.58	0.44
darkgray	0.65	0.65	0.65	sienna	0.63	0.32	0.18
gray	0.5	0.5	0.5	brown	0.65	0.16	0.16
lightgray	0.8	0.8	0.8	orangered	1.0	0.0	0.2
darkred	0.5	0.0	0.0	orange	1.0	0.5	0.0
red	1.0	0.0	0.0	gold	1.0	0.85	0.0
midred	1.0	0.65	0.65	greenyellow	0.70	1.0	0.2
lightred	1.0	0.8	0.8	forestgreen	0.14	0.56	0.14
darkgreen	0.0	0.5	0.0	aqua	0.1	1.0	0.85
green	0.0	1.0	0.0	deepskyblue	0.0	0.75	1.0
midgreen	0.65	1.0	0.65	skyblue	0.2	0.56	0.8
lightgreen	0.8	1.0	0.8	royalblue	0.25	0.45	0.95
darkblue	0.0	0.0	0.5	slateblue	0.4	0.35	0.8
blue	0.0	0.0	1.0	bluemagenta	0.5	0.0	1.0
midblue	0.65	0.65	1.0	darkviolet	0.6	0.0	0.8
lightblue	0.8	0.8	1.0	magenta	1.0	0.0	1.0
yellow	1.0	1.0	0.0	lavender	0.8	0.8	1.0
yellowgreen	0.5	1.0	0.0				

COLORS

(cont)

Background dependent colors:

Name	*Black Backgnd*			*White Backgnd*			*Description*
title0	1.0	1.0	1.0	0.0	0.0	0.0	information box
title1	1.0	0.7	0.7	0.1	0.0	0.0	first info line
title2	0.0	0.95	0.8	0.0	0.5	0.2	second info line
axis	1.0	1.0	1.0	0.0	0.0	0.0	axis and labels
roads0	0.65	0.65	0.65	0.65	0.65	0.65	roads (filled)
roads1	0.6	0.6	0.6	0.6	0.6	0.6	roads (outline)
land	0.0	0.5	0.0	0.8	1.0	0.8	land
water	0.0	0.0	0.5	0.8	0.8	1.0	water
bound0	0.0	0.3	0.0	0.45	1.0	0.45	map lines (filled)
bound1	0.0	0.5	0.0	0.6	1.0	0.6	map lines (outline)
shadow0	0.0	1.0	0.0	0.0	0.0	0.0	map shadow (filled)
shadow1	0.0	1.0	0.0	0.0	0.0	0.0	map shadow (outline)

If CFRAME_A, CFRAME_B and CFRAME_C indicate multiple backgrounds and **COLORS** is not specified for a background, then the **COLORS** for the first background will be used for the unspecified background (i.e. in most cases you only need to specify **COLORS** for the first background).

TVAR

ZVAR

YVAR

XVAR

character array[†]

Character strings that specify the orientation, location, and size of the two-dimensional slab to be plotted or 3 dimensional field to be extracted. Each is a slash delimited text string with two components, the second of which has further colon delimiting.

The first component is a single character that describes how this direction will appear on the background and is only relevant if plotting. The characters and their meanings are described as follows:

- H : direction will be plotted horizontally
- V : direction will be plotted vertically
- F : fix the direction to the following value

Note that the option of plotting variables with respect to time is not yet implemented.

The second component specifies the range information for the directions and time and is applicable to all types of ***REVU*** runs. The syntax is:

```
      left_value : right_value : increment
```

where the values, if positive, are actual grid point values. If the left or right values are negative or zero, they are interpreted as an offset from the boundaries of the grid. For the time direction, the values refer to the analysis file number found according to the filtering of ANPREF. If CFRAME_A, CFRAME_B and CFRAME_C indicate multiple backgrounds and **TVAR** indicate multiple times, then the settings for each background apply to each time. If plotting, the increment is only applied to T in **TVAR**. If the increment is not specified, it defaults to 1.

TVAR
ZVAR
YVAR
XVAR
(cont)

For example and series of vertical Y-Z slabs may be specified as:

```
TVAR(1)=‘/F/1:5:1/’,
ZVAR(1)=‘/V/0:0:1/’,
YVAR(1)=‘/F/8:8:1/’,
XVAR(1)=‘/H/0:0:1/’,
```

This means that this background will have X horizontally on the plots and Z vertical (both will have the full domain of grid points for the grid specified by **IGRID** above). The Y direction will be fixed to the 8th grid point from the southern edge of the domain and time T will range from the 1st to the 5th file found after filtering with **ANPREF**.

Or a series of horizontal X-Y slabs of may be specified as:

```
TVAR(1)=‘/F/ 1:25:2/’,
ZVAR(1)=‘/F/ 2: 2:1/’,
YVAR(1)=‘/V/-1:-2:1/’,
XVAR(1)=‘/H/ 3:15:1/’,
```

This means that this background will have X horizontally on the plots and Y vertical. For the grid specified by **IGRID** X will span from the 3rd to the 15th grid point from the western edge of the domain. Y will span from the 2nd grid point from the southern edge of the domain to the 3rd grid point from the northern edge of the domain. Time T will range from the 1st to the 25th file found after filtering with **ANPREF**, skipping every 2nd file (so 13 plots will be drawn).

If not plotting, the following settings may be appropriate (noting that the F, V and H values are ignored):

```
TVAR(1)=‘/F/ 1:25:1/’,
ZVAR(1)=‘/F/ 2: 0:1/’,
YVAR(1)=‘/V/-1:-1:2/’,
XVAR(1)=‘/H/-1:-1:2/’,
```

Which will output all 25 times for a horizontal (X-Y) domain that does not include the boundary grid points for the grid specified by **IGRID** and skips every other grid point in between, and includes all vertical levels other that the lowest which is below the terrain surface.

If **CFRAME_A**, **CFRAME_B** and **CFRAME_C** indicate multiple backgrounds and **TVAR**, **ZVAR**, **YVAR** and **XVAR** are not specified for a background, then the settings for the first background will be used for the unspecified background (i.e. in most cases you only need to specify them for the first background).

CFRAME_A
character array[†]

CFRAME_A specifies the variables to plot if **ANATYPE** is set 'SPACE', or extract for other **ANATYPE** settings. All available variables are detailed below (default is 'none').

If not plotting, only the variable is required. For example, a set of **CFRAME_A** settings for **ANATYPE** = 'V5D' might be:

```
CFRAME_A(1)='/u/',
CFRAME_A(2)='/v/',
CFRAME_A(3)='/w/',
CFRAME_A(4)='/tempk/',
CFRAME_A(5)='/dewptk/',
```

If plotting, a series of slash-delimited components containing plotting options can follow the variable. These control the way in which the variable is represented in the plot. Order is important and, although you can leave components at the end of the list unspecified, you must specify each component (at least with placeholders) up to the last component you wish to specify (an example of this is shown below). Options that are left unspecified are given sensible defaults. Note that the subscript of **CFRAME_A** denotes the background number (equals the frame number if **IPANEL** is set to 0). The subscript on **CFRAME_A** is tied to those on **CFRAME_B** and **CFRAME_C** in that sets of these parameters with the same subscript appear on the same plots. Describing each slash delimited component following the variable:

2. A 1 or 2 character string that defines the method of plotting (first character), and whether or not to include a color scale bar (second character) if the viable is to be plotted with a range of colors. First character setting may be:

   - **c** : draw contour lines
   - **f** : draw and fill contours
   - **t** : fill tiles

   Second character settings may be:

   - **n** or blank : no color bar
   - **b** : draw color bar

   For example:

   ```
   CFRAME_A(n)='/tempc/fb/',
   ```

   will result in a filled contour plot of temperature in degrees centigrade and include color bar (default is **fb**).

3. Minimum contour level for the field named in component 1. A value of zero allows the code to choose its own minimum.
4. Maximum contour level for the field named in component 1. A value of zero allows the code to choose its own maximum.

CFRAME_A

(cont)

5. If positive, this is the contour interval for the field named in component 1. If negative, then it indicates about how many contour lines will be drawn between the lowest and highest values. If set to zero, a value will be chosen in the code.

   For example:

   ```
   CFRAME_A(n)='/tempc/fb/10.0/30.0/2.0/',
   ```

   will set the contour levels for temperature from 10.0 to 30.0 degrees with a contouring interval of 2.0 degrees centigrade. If values exist outside of this range they will not be contoured.

6. Options for plotting the field named in component 1. This is a string of colon-delimited options. Only those options you wish to specify need be included. An array of examples follow the definitions:

   - **m** : Graduated color method (integer). There are a variety of ways to specify the color increments with contour or tile values (all methods interpolate between colors in HLS space):

     m0 : Default color table (mixture of predefined and easily distinguished colors).

     Single color schemes:

     m1 : Single color scheme with component 7 (first specified color) at the maximum slab value, fading to the background color at the minimum slab value.

     m2 : Single color scheme that is the same as **m1**, but fades to the foreground color.

     m3 : Single color scheme with component 7 (first specified color) at the middle value, lightening above to white at the maximum slab value and darkening to black at the minimum slab value.

     Two color schemes:

     m4 : Two color scheme with component 7 (first specified color) covering slab values greater than the central color value (specified in option '**c**') and component 8 (second specified color) covering slab values less than the central color value. The maximum saturation for the specified colors occurs at the maximum and minimum slab values, fading to the background color at the interface (the central color value) (default).

     m5 : Two color scheme that is the same as **m4**, but fades to the foreground color.

     m6 : Two color blending scheme with component 7 (first specified color) at the maximum slab value and component 8 (second specified color) at the minimum slab value. Colors are blended between the maximum and minimum without going through either the foreground or background colors. This can lead to some odd color ranges if the user is not careful with the bounding color selections.

CFRAME_A

(cont)

Three color schemes:

m7 : Three color blending scheme with component 7 (first specified color) at the maximum slab value, component 8 (second specified color) at the minimum slab value and component 9 (third specified color) at the central slab value. Colors are blended between the maximum and minimum and the center without going through either the foreground or background colors. This can lead to some odd color ranges if the user is not careful with the bounding color selections.

Negative values for m switches the order of color graduation (what appeared at the maximum values appear at the minimum values and vise versa).

- **b** : Graduated color bias (real) that gives a departure from linear the interpolation between colors since bias towards lighter colors is preferred as this gives more intervals in the human eyes sensitive range (e.g. b1. = linear, b1.5 = lighten and b.75 = darken) (default is 1.5).
- **c** : Central color value (real) for splitting colors and contour lines in the scheme described above (default = 0.).
- **i** : Major contour line interval (integer) (default is 5).
- **g** : Graduated contour line color offset (real).

  >0.0 : Graduate contour line colors> the value is used for the lightness color offset (default is 30).

  0.0 : Use component 7 (first specified color) and component 8 (second specified color) for positive and negative contours respectively.

  <0.0 : Use contour and high/low labels color parameter (option **x**) for the contour line color.
- **t** : Contour line thickness (real). t1.0 is the standard line thickness. Major contour lines are drawn at 1.5*t (default is 1.0)
- **d** : Dashed / solid line control (integer)

  d0 : Solid lines for all contours.

  d1 : Dashed lines for contours with values less that the central color values (option **c**) and solid lines for all other contours (default).

  d2 : Dashed lines for all contours.
- **o** : Overlay lines on top of any filled contours and tiles (integer).

  o0 : Do not overlay contour lines.

  o1 : Overlay contour lines (default)
- **l** : Contour line label controls (integer).

  l-1: No labels.

  l0 : Draw labels with the color defined in **x** below, but without filling a background behind the label or outlining the labels with boxes (default).

  l1 : Draw labels in box outlined with the foreground color.

  l2 : Draw labels in box filled with the background color.

  l3 : Draw labels in box outlined with the foreground color and led with the background color.

CFRAME_A
(cont)

- **h** : High and low label controls (integer). The options for h are identical to those for ‘**l**’.
- **x** : Color of contour and high/low labels (character), and contour lines if option **g** is set to less that zero (default is lightgray).
- **s** : Size of the contour and high/low labels (real) (default is 0.01).

7. First specified color used for color contours and fills (character). Use of this color is determined by the **m** and **g** options above (default is red).
8. Second specified color used for color contours and fills (character). Use of this color is determined by the **m** and **g** options above (default is blue).
9. Third specified color used for color contours and fills (character). Use of this color is determined by the **m** and **g** options above (default is green).

The default setting for CFRAME_A is:

```
CFRAME_A(1)='/none/fb/0./0./0./m4:c0.:b1.5:i5:d1:g30.:
                 t1.:o1:l0:h0:s.01:xlightgray/red/blue/green/'
```

Examples:

```
CFRAME_A(1)='/tempc/fb/10./30./2.0/m5:c20.:xgray/red/blue/'
```

will fill contours of temperature from 10 to 20 degrees centigrade at 2 degree intervals with the 28 to 30 degree area filled in red and the 10 to 12 degree area in blue. The color fill dividing line is set to 20 degrees, so on a black background with color scheme 5, the two areas either side to 20 degrees (18 to 20 and 20 to 22) will be filled in white. The colors will graduate from white to red above 20 and white to blue below. The contour lines will follow the same color pattern, but each line will have a lightness offset of 30% from the fill color. Contour line and high/low labels will be drawn in gray without boxes or fill. A color scale bar will also be drawn.

```
CFRAME_A(2)='/relhum/f/50./100./5./m1:g20.:h-1:xblue/midblue/'
```

will fill contours of relative humidity from 50 to 100% at 5% intervals. The 95 to 100% interval will be filled with midblue. From there the fill colors will fade to the background color in the 50 to 55% interval. Contour lines will be drawn with a 20% lightness offset from the adjacent fill color and contour line labels will be drawn in blue. No high/low labels or color scale bar will be drawn.

```
CFRAME_A(3)='/relhum/c////c50.:g-1.:h-1/yellow/purple/'
```

will draw contours of relative humidity from the minimum value in the plotted slab to the maximum value with some convenient interval (still centered around a central color value of 50%). With the central color set at 50%, yellow contour lines of the same color will be drawn above 50% and purple below. Not high/low labels or color fill scale bar will be drawn. Note that the /’s after the /c/ are placeholders for the contour range and interval settings. They will take on the default values. The placeholders are required as order and occurrence of each component in the slash-delimited string is important.

CFRAME_B
character array†

If plotting, **CFRAME_B** specifies what wind information should be plotted. It is a slash-delimited string with up to 10 components. Order is important and, although you can leave components at the end of the list unspecified, you must specify each component (at least with placeholders) up to the last component you wish to specify (an example of this is in CFRAME_A). Options that are left unspecified are given sensible defaults. The subscript on **CFRAME_B** is tied to those on CFRAME_A and CFRAME_C in that sets of these parameters with the same subscript appear on the same plots. Describing each slash delimited component:

1. Character value to denote how to depict the wind field:

   n : none (default)

   s : draw streamlines.

   b : draw wind barbs.

   v : draw wind vectors.

   t : draw turbulence with vectors.

   r : draw relative vorticity vectors.
2. Variable indicating the number of horizontal grid points between adjacent wind vector or barb plots (integer ***intindi***) (default is 1).
3. Variable indicating the number of vertical grid points between adjacent wind vector or barb plots (integer ***intindj***) (default is 1).
4. Variable specifying the wind barb flags, staff and half staff values (integer) (default is (not used for vectors, but place holder still required):

   1 : 50, 10, 5 m/s

   2 : 20, 4, 2 m/s

   3 : 10, 2, 1 m/s (default)

   4 : 5, 1, 0.5 m/s
5. Wind marker color (test)
6. Stem length for vector or barb (scales the whole barb) (real) (default is ***intindi***). The default behavior for the size of the vectors and barbs is to double in size when ***intindi*** is doubled, in which case they tend to dominate the picture. You can prevent this by halving the stem length when ***intindi*** is doubled.
7. Maximum wind vector scale (real) (default is –1. which get recomputed to the maximum wind speed).
8. Wind vector head length (real) (default is –1. which get recomputed to 0.1* ***intindi***).
9. Wind vector head length for 0 wind speed (real) (default is –1. which get recomputed to 0.2* ***intindi***).

Wind vector head angle (real) (default is 25.0 degrees).

CFRAME_B
(cont)
	The default setting for **CFRAME_B** is:
	`CFRAME_B(1)='/n/1/1/3/yellow/1./0./-1./-1./-1./'`
	Examples:
	`CFRAME_B(1)='/b/2/2/4/green/.5/ '`
	will draw green wind barbs on every other grid point in both the horizontal and vertical with flag, staff and half staff values of 5, 1 and 0.5 m/s respectively. The size of the barbs will be maintained the same as for drawing them at every grid point with the 0.5 stem length setting.

CFRAME_C
character array[†]
	If plotting, **CFRAME_C** is a second variable that can be contoured or tiled. It can also be used to color wind vectors with or without a color scale bar (second component setting of 'vb' or 'vn'). The subscript on **CFRAME_C** is tied to those on **CFRAME_A** and **CFRAME_B** in that sets of these parameters with the same subscript appear on the same plots.
	The default setting for **CFRAME_C** is:
	`CFRAME_C(1)='/none/cn/0./0./0./m2:c0.:b1.5:i5:d1:g30.:`
	`t1.:o1:l0:h0:s.01:xlightgray/darkgreen/purple/'`

[†] Each array may contain entries for each background. In general, if array elements for a background are missing, the first array element is used. If nothing is specified, the default value is used. If nothing is specified in CFRAME_A, CFRAME_B and CFRAME_C for a background, nothing is plotted on that background.

$GRAB Namelist

Required only for run types **ANATYPE** = 'GRAB' and described the I/O of the grabbed data.

GRABIN character	Specifies the filename that contains the locations where we want to grab the data. Each latitude-longitude location can be specified as either a point in space (requires a height AGL to be given), or a profile of the model levels. The output data for each latitude-longitude-height location is then interpolated from the surrounding ***RAMS*** grid points on the finest possible grid specified by **IGRID**. The format of this file is specified below.
IGRABFMT integer	Specifies the output data format: • If set to 1, output selected **CFRAME_A** variables in an ASCI file (format is specified below). • If set to 2, output the data in the RALPH2 ASCI format (requires that **CFRAME_A** be set, in order, to the following fields 'u', 'v', 'tempc', 'dewptc' and 'press').

$GLL Namelist

Required only for run types **ANATYPE** = 'GRIB', 'V5D' or 'GrADS' and described the latitude-longitude grid that the output data is interpolated to.

IGRIDLL integer	For output that is interpolated to a latitude-longitude grid (e.g. GRIB, GrADS and Vis5D), this specifies the method by which the size and resolution of the output grid are determined. • If set to 0, the remaining variables in this namelist specify the size and resolution of the latitude-longitude grid. • If set to 1, the maximum sized latitude-longitude grid that fits within the RAMS grid specified by **IGRID** is determined in the code. • If set to 2, the minimum sized latitude-longitude grid that contains the RAMS grid specified by **IGRID** is determined in the code.
GLLDLLAT real	If **IGRIDLL** = 0, specifies the latitude grid spacing of the latitude-longitude grid.
GLLDLLON real	If **IGRIDLL** = 0, specifies the longitude grid spacing of the latitude-longitude grid.
GLLWLON real	If **IGRIDLL** = 0, specifies the western edge of the latitude-longitude grid
GLLELON real	If **IGRIDLL** = 0, specifies the eastern edge of the latitude-longitude grid
GLLSLAT real	If **IGRIDLL** = 0, specifies the southern edge of the latitude-longitude grid
GLLNLAT real	If **IGRIDLL** = 0, specifies the northern edge of the latitude-longitude grid

$STATS Namelist

Required only for run types **ANATYPE** = 'STATS' and describes the run mode and I/O.

CMODE character	Specifies the observations to process: • If set to 'all', process both surface and upper air observations. • If set to 'sfc', process surface only. • If set to 'rawin', process upper air observations only.
SFCPREF character	Prefix for the surface observational file names (file must be in RALPH2 format).
RWNPREF character	Prefix for the upper air observational file names (file must be in RALPH2 format).
NOQ character	Quality control suffix for the file names. **-NOQ** is tagged to the original data file if quality control has been run on it.
IPDIFF **IPVALS** **IPWIND** **IPHIST** **IPCONT** integer	Switches for plotting statistical results (not fully functional): • If set to 0, switch is off. • If set to 1, switch is on. **IPDIFF** specifies whether to plot predicted-observed difference values. **IPVALS** specifies whether to plot actual predicted and observed values. **IPWIND** specifies whether to plot winds comparison. **IPHIST** specifies whether to plot histograms of predicted-observed. **IPCONT** specifies whether to plot contours of predicted-observed differences.

REVU Variables for Plotting or Output

The list of variables that are currently configured in ***REVU*** version 2.3.1 (for ***RAMS*** version 4.3.0) is described below. These may be selected for plotting or output into the GrADS, GRIB or Vis5D files (except for the LEAF patch variables).

The field name is what you may use in **CFRAME_A** or **CFRAME_C** in your *REVU_IN* file. The model variables indicate what you need to have selected in your *VTABLES* file when you run ***RAMS*** to ensure that you can look at field or extract the data with ***REVU***. To select a *VTABLE* field for output to the analysis files, ensure that the "anal" tag is included in the ctables list. Likewise, for output to the "lite" files, ensure that the "lite" tag is included in the ctables list.

Some of the model variables indicated are not directly listed in the *VTABLE* file. Specifically selecting PP in *VTABLE* will output PI and VKM will output HKH. The model variables indicated for the LEAF fields are relevant to the future ***RAMS*** version 4.4. If you wish to view any of these select all of TGP, WGP, SCHAR and GS. SCHAR is also required to get the patch area that is used to obtain land percentages for most plots.

3-D Atmospheric Variables

The following variables are defined on the 3-D atmospheric grid and may be plotted in either horizontal or vertical cross section. Obviously, many of these variables are dependent on which options were activated for a particular run.

Field Name	***Description [units]***	***Model Variables***
u	x-direction wind component [m/s]	UP
v	y-direction wind component [m/s]	VP
u_avg	eastward wind component averaged to T point [m/s]	UP, VP
u_avg	northward wind component averaged to T point [m/s]	UP, VP
ue	earth rotated eastward wind component [m/s]	UP, VP
ve	earth rotated northward wind component [m/s]	UP, VP
ue_avg	eastward wind component earth rotated and averaged to T point [m/s]	UP, VP
ve_avg	northward wind component earth rotated averaged to T point [m/s]	UP, VP
w	z-direction wind component [m/s]	WP
wcms	z-direction wind component [cm/s]	WP
w_avg	z-direction wind component averaged to T point [m/s]	WP
speed	horizontal wind speed averaged to T point [m/s]	UP, VP
speed_mph	horizontal wind speed averaged to T point [mph]	UP, VP
direction	horizontal wind direction averaged to T point [deg]	UP, VP
relvortx	x-component of relative vorticity [rad/s]	UP, VP, TOPT
relvorty	y-component of relative vorticity [rad/s]	UP, VP, TOPT
relvortz	z-component of relative vorticity [rad/s]	UP, VP, TOPT
absvortz	z-component of absolute vorticity [rad/s]	UP, VP, TOPT

Field Name	*Description [units]*	*Model Variables*
potvortz	z-component of potential vorticity [rad/s]	UP, VP, TOPT, THETA
horiz_div	horizontal divergence [s^-1]	WP
pi	Exner function [J/(kg K)]	PI
press	pressure [mb]	PI
theta	potential temperature [K]	THETA
dn0	reference state density [kg/m^3]	TOPT
pi0	reference state Exner function [J/(kg K)]	TOPT
th0	reference state virtual potential temperature [K]	TOPT
pert_pressure	perturbation pressure [mb]	TOPT, PI
tempk	temperature [K]	THETA, PI
tempc	temperature [deg C]	THETA, PI
tempf	temperature [deg F]	THETA, PI
theta_e	equivalent potential temperature [K]	RV, THETA, PI
theta_v	virtual potential temperature [K]	THETA, PI
vapor	water vapor mixing ratio [g/kg]	RV
cloud	cloud water mixing ratio [g/kg]	RCP
rain	rain mixing ratio [g/kg]	RRP
pristine	pristine ice mixing ratio [g/kg]	RPP
snow	snow mixing ratio [g/kg]	RSP
aggregates	aggregates mixing ratio [g/kg]	RAP
graupel	graupel mixing ratio [g/kg]	RPP
hail	hail mixing ratio [g/kg]	RHP
liquid	liquid water mixing ratio [g/kg]	RCP, RRP, RGP, Q6, RHP, Q7
ice	ice mixing ratio [g/kg]	RPP, RSP, RAP, RGP, Q6, RHP, Q7
total_cond	total condensate mixing ratio [g/kg]	RPP, RSP, RAP, RGP, Q6, RHP, Q7
r_total	total water mixing ratio [g/kg]	RV, RCP, RRP, RPP, RSP, RAP, RGP, RHP
rtotal_orig	total water mixing ratio (original method) [g/kg]	RTP
dewptk	dew point temperature [K]	RV, PI, THETA
dewptf	dew point temperature [deg F]	RV, PI, THETA
dewptc	dew point temperature [deg C]	RV, PI, THETA
relhum	relative humidity [percent]	RV, PI, THETA
relhum_frac	relative humidity [fraction]	RV, PI, THETA
cloud_concen_mg	cloud droplet number concentration [#/mg]	CCP
rain_concen_kg	rain number concentration [#/kg]	CRP
pris_concen_kg	pristine ice number concentration [#/kg]	CPP
snow_concen_kg	snow number concentration [#/kg]	CSP
agg_concen_kg	aggregates number concentration [#/kg]	CAP
graup_concen_kg	graupel number concentration [#/kg]	CGP
hail_concen_kg	hail number concentration [#/kg]	CHP
cloud_concen_cm3	cloud droplet number concentration [#/cm^3]	CCP, TOPT
rain_concen_m3	rain number concentration [#/m^3]	CRP, TOPT

Field Name	***Description [units]***	***Model Variables***
pris_concen_m3	pristine ice number concentration [#/m^3]	CPP, TOPT
snow_concen_m3	snow number concentration [#/m^3]	CSP, TOPT
agg_concen_m3	aggregates number concentration [#/m^3]	CAP, TOPT
graup_concen_m3	graupel number concentration [#/m^3]	CGP, TOPT
hail_concen_m3	hail number concentration [#/m^3]	CHP, TOPT
ccn_concen	CCN number concentration [#/mg]	CCCNP
ifn_conc	IFN number concentration [#/kg]	CIFNP
cloud_diam	cloud droplet mean-mass diameter [microns]	RCP, CCP
rain_diam	rain mean-mass diameter [mm]	RRP, CRP
pris_diam	pristine ice mean-mass diameter [microns]	RPP, CPP
snow_diam	snow mean-mass diameter [mm]	RSP, CSP
agg_diam	aggregates mean-mass diameter [mm]	RAP, CAP
graup_diam	graupel mean-mass diameter [mm]	RGP, CGP
hail_diam	hail mean-mass diameter [mm]	RHP, CHP
q2	rain internal energy parameter [J/kg]	Q2
q6	graupel internal energy parameter [J/kg]	Q6
q7	hail internal energy parameter [J/kg]	Q7
rain_temp	rain temperature [deg C]	Q2
graup_temp	graupel temperature [deg C]	Q6
hail_temp	hail temperature [deg C]	Q7
rain_air_tempdif	rain-air temperature difference [K]	Q2, THETA, PI
graup_air_tempdf	graupel-air temperature difference [K]	Q6, THETA, PI
hail_air_tempdif	hail-air temperature difference [K]	Q7, THETA, PI
graup_fracliq	liquid fraction in graupel [ ]	Q6
hail_fracliq	liquid fraction in hail [ ]	Q7
geo	geopotential height [m]	TOPT
tke	turbulent kinetic energy [m^2/s^2]	TKEP
scalar1_mixrat	scalar # 1 mixing ratio [units/kg]	SCLP1
scalar2_mixrat	scalar # 2 mixing ratio [units/kg]	SCLP2
cuparm_thetasrc	cumulus parameterization heating rate {K/s]	THSRC
cuparm_rtsrc	cumulus parameterization moistening rate [kg/kg/s]	RTSRC
rad_thetasrc	radiative transfer heating rate [K/s]	FTHRD
khh	horizontal scalar mixing coefficient [m^2/s]	HKH
khv	vertical scalar mixing coefficient [m^2/s]	VKH

2-D Variables

The following variables are defined as a function of horizontal coordinates only and may only be plotted in horizontal cross section.

Field Name	*Description [units]*	*Model Variables*
tempf2m	2-meter-height air temperature [deg F.]	UP, VP, THETA, TOPT, TGP, SCHAR, GSF, PI
speed10m	10-meter-height wind speed [m/s]	UP, VP, THETA, TOPT, GSF, SCHAR, TGP
clear_frac	clear sky fraction [fraction]	RV, PI, THETA
cloud_frac	cloud cover fraction [fraction]	RV, PI, THETA
pbl_ht	planetary boundary layer height [m]	TOPT, TKE
accpr	surface accumulated rain [kg/m^2]	ACCPR
accpp	surface accumulated pristine ice [kg/m^2]	ACCPP
accps	surface accumulated snow [kg/m^2]	ACCPS
accpa	surface accumulated aggregates [kg/m^2]	ACCPA
accpg	surface accumulated graupel [kg/m^2]	ACCPG
accph	surface accumulated hail [kg/m^2]	ACCPH
totpcp	surface accumulated resolved precipitation [mm liquid equivalent]	ACCPR, ACCPP, ACCPS, ACCPA, ACCPG, ACCPH
totpcp_in	surface accumulated resolved precipitation [inches liquid equivalent]	ACCPR, ACCPP, ACCPS, ACCPA, ACCPG, ACCPH
precip	surface accumulated resolved plus convective precipitation [mm liquid equivalent]	ACCPR, ACCPP, ACCPS, ACCPA, ACCPG, ACCPH, ACONPR
precip_in	surface accumulated resolved plus convective precipitation [inches liquid equivalent]	ACCPR, ACCPP, ACCPS, ACCPA, ACCPG, ACCPH, ACONPR
pcprr	surface precipitation rate of rain [mm/hr liquid equivalent]	PCPRR
pcprp	surface precipitation rate of pristine ice [mm/hr liquid equivalent]	PCPRP
psprs	surface precipitation rate of snow [mm/hr liquid equivalent]	PCPRS
pcpra	surface precipitation rate of aggregates [mm/hr liquid equivalent]	PCPRA
pcprg	surface precipitation rate of graupel [mm/hr liquid equivalent]	PCPRG
pcprh	surface precipitation rate of hail [mm/hr liquid equivalent]	PCPRH
pcpg	total surface precipitation falling this timestep [kg/m^2]	PCPG
qpcpg	total internal energy of surface precipitation falling this timestep [J/m^2]	QPCPG
dpcpg	total added depth of surface precipitation falling this timestep [m]	DPCPG

Field Name	***Description [units]***	***Model Variables***
pcprate	resolved surface precipitation [mm/hr liquid equivalent]	PCPRR, PCPRP, PCPRS, PCPRA, PCPRH, PCPRG, CONPRR
pcprate_in	resolved surface precipitation [inches/hr liquid equivalent]	PCPRR, PCPRP, PCPRS, PCPRA, PCPRH, PCPRG, CONPRR
precipr	resolved plus convective surface precipitation [mm/hr liquid equivalent]	PCPRR, PCPRP, PCPRS, PCPRA, PCPRH, PCPRG, CONPRR
precipr_in	resolved plus convective surface precipitation [inches/hr liquid equivalent]	PCPRR, PCPRP, PCPRS, PCPRA, PCPRH, PCPRG, CONPRR
conpcp	cumulus parameterization precipitation rate [mm/hr]	CONPRR
acccon	cumulus parameterization accumulated surface precipitation [mm]	CONPRR
vertint_rt	vertically-integrated total water mixing ratio [mm liquid equivalent]	TOPT, RCP, RRP, RPP, RSP, RAP, RGP, RHP,RV
vertint_cond	vertically-integrated total condensate mixing ratio [mm liquid equivalent]	TOPT, RCP, RRP, RPP, RSP, RAP, RGP, RHP
tfz	surface sensible heat flux [K m/s]	TFZ
qfz	surface water vapor flux [kg/kg m/s]	QFZ
uw	surface x-component momentum flux [m^2/s^2]	UW
vw	surface y-component momentum flux [m^2/s^2]	VW
wfz	surface y-component momentum flux [m^2/s^2]	WFZ
sens_flux	surface sensible heat flux [W/m^2]	TFZ, TOPT
lat_flux	surface latent heat flux [W/m^2]	QFZ, TOPT
etrans	evapotranspiration rate [mm/hr]	QFZ, TOPT
etrans_in	evapotranspiration rate [in/hr]	QFZ, TOPT
umom_flx	surface x-component momentum flux [Pa]	UW, TOPT
vmom_flx	surface y-component momentum flux [Pa]	VW, TOPT
wmom_flx	surface x-component momentum flux [Pa]	WFZ, TOPT
bowen	Bowen ratio [ ]	TFZ, QFZ
rshort	incident surface flux of shortwave radiation [W/m^2]	RSHORT
rlong	incident surface flux of longwave radiation [W/m^2]	RLONG
rlongup	upward surface flux of longwave radiation [W/m^2]	RLONGUP
albedt	grid-cell-averaged surface albedo [ ]	ALBEDT
topo	topography height [m]	TOPT
topo_ft	topography height [ft]	TOPT
lat	latitude [deg]	GLAT
lon	longitude [deg]	GLON
sea_press	sea level pressure [mb]	TOPT, PI, THETA
sfc_div	horizontal divergence at surface [s^-1]	WP
land	land fractional area [ ]	SCHAR
ctprof	cloud top height [m]	????
sst	water temperature [deg C]	TGP

LEAF Variables

These variables are from the LEAF2 parameterization and comprise all the soil and vegetation parameters.

The following variables are defined in ***RAMS*** for land subgrid patches only. They are averaged over all land patches and are defined for the entire grid cell regardless of how little area of the grid cell is occupied by land. *In grid cells that contain no land, these field values will retain their initial values in the RAMS simulation.* These fields may be plotted only in horizontal cross section.

Field Name	***Description [units]***	***Model Variables***
soil_z0_ps	soil roughness height by grid cell [m]	SCHAR
veg_fracarea_ps	vegetation fractional area by grid cell [ ]	SCHAR
veg_lai_ps	vegetation leaf area index by grid cell [ ]	SCHAR
veg_z0_ps	vegetation roughness height by grid cell [m]	SCHAR
veg_disp_ps	vegetation displacement height by grid cell [m]	SCHAR
grnd_mixrat_ps	ground mixing ratio by grid cell [g/kg]	SCHAR, WGP
soil_mixrat_ps	soil mixing ratio by grid cell [g/kg]	SCHAR, TGP
veg_moist_ps	vegetation surface moisture by grid cell [kg/m^2]	SCHAR, WGP
veg_temp_ps	vegetation temperature by grid cell [deg C]	SCHAR, TGP
snow_depth_ps	snow depth by grid cell [m]	SCHAR, TGP
snowcover_ps	snowcover content by grid cell [kg/m^2]	SCHAR, TGP

The following variables are defined in ***RAMS*** for all subgrid patches. They are averaged over all patches here. They may be plotted only in horizontal cross section.

Field Name	***Description [units]***	***Model Variables***
net_z0_ps	net roughness height by grid cell [m]	SCHAR
canopy_mixrat_ps	canopy mixing ratio by grid cell [g/kg]	SCHAR, WGP
canopy_temp_ps	canopy temperature by grid cell [deg C]	SCHAR, TGP
ustar_ps	ustar by grid cell [m/s]	SCHAR, GSF
tstar_ps	tstar by grid cell [K]	SCHAR, GSF
rstar_ps	rstar by grid cell [kg/kg]	SCHAR, GSF
sens_heat_flux_ps	surface sensible heat flux by grid cell [W/m^2]	SCHAR, GSF, TOPT
lat_heat_flux_ps	surface latent heat flux by grid cell [W/m^2]	SCHAR, GSF, TOPT
5050_temp_ps	average of k = 2 air and canopy air temperatures by grid cell [deg C]	SCHAR, TGP, THETA, PI
5050_tempf_ps	average of k = 2 air and canopy air temperatures by grid cell [deg F]	SCHAR, TGP, THETA, PI

The following variables are defined in ***RAMS*** for all grid cells, but they are not quantities that can be averaged. A single value is taken here for the entire grid cell from the subgrid patch having the largest area in the cell. They may be plotted only in horizontal cross section.

Field Name	***Description [units]***	***Model Variables***
veg_class_bp	vegetation class of the biggest patch in the grid cell [ ]	SCHAR
qveg_class_bp	"q" vegetation class of the biggest patch in the grid cell [ ]	SCHAR

The following variables are defined in ***RAMS*** for subgrid patches and remain defined by patch here. They may be plotted only in horizontal cross section and only with the tile plot function. *These variables cannot be output to GrADS, GRIB or Vis5D.*

Field Name	***Description [units]***	***Model Variables***
patch_area	fractional area of patches [ ]	SCHAR
soil_z0_p	soil roughness height by patch [m]	SCHAR
veg_class_p	vegetation class by patch [ ]	SCHAR
qveg_class_p	"q" vegetation class by patch	SCHAR
veg_fracarea_p	vegetation fractional area by patch [ ]	SCHAR
veg_lai_p	vegetation leaf area index by patch [ ]	SCHAR
net_z0_p	net roughness height by patch [m]	SCHAR
veg_z0_p	vegetation roughness height by patch [m]	SCHAR
veg_disp_p	vegetation displacement height by patch [m]	SCHAR
patch_wetind	wetness index by patch [ ]	SCHAR
snowlevels	number of snow levels by patch [ ]	SCHAR
grnd_mixrat_p	ground mixing ratio by patch [g/kg]	SCHAR, WGP
soil_mixrat_p	soil mixing ratio by patch [g/kg]	SCHAR, TGP
veg_moist_p	vegetation surface moisture by patch [kg/m^2]	SCHAR, WGP
canopy_mixrat_p	canopy mixing ratio by patch [g/kg]	SCHAR, WGP
veg_temp_p	vegetation temperature by patch [deg C]	SCHAR, TGP
canopy_temp_p	canopy temperature by patch [deg C]	SCHAR, TGP
ustar_p	ustar by patch [m/s]	SCHAR, GSF
tstar_p	tstar by patch [K]	SCHAR, GSF
rstar_p	rstar by patch [kg/kg]	SCHAR, GSF
sens_heat_flux_p	surface sensible heat flux by patch [W/m^2]	SCHAR, GSF, TOPT
lat_heat_flux_p	surface latent heat flux by patch [W/m^2]	SCHAR, GSF, TOPT
snow_depth_p	snow depth by patch [m]	SCHAR, TGP
snowcover_p	snowcover content by patch [kg/m^2]	SCHAR, TGP

The following variables are defined in ***RAMS*** for land subgrid patches only and as a function of depth. They are averaged over land patches here and are expanded to cover the entire grid regardless of how little area of the grid cell is occupied by land. *In grid cells that contain no land, these field values will retain* their initial values in the ***RAMS*** simulation. These fields may be plotted either in horizontal or vertical cross section. *These variables cannot be output to GrADS, GRIB or Vis5D.*

Field Name	***Description [units]***	***Model Variables***
sltex_bp	soil textural class of biggest patch in grid cell [ ]	SCHAR, TGP
soilq_ps	soil internal energy parameter by grid cell [j/m^2]	SCHAR, TGP
soil_temp_ps	soil/sea temperature by grid cell [deg C]	SCHAR, TGP
soil_moist_ps	soil moisture by grid cell [m^3/m^3]	SCHAR, TGP
soil_moistf_ps	soil moisture fraction by grid cell [m^3/m^3]	SCHAR, TGP

The following variables are defined in ***RAMS*** for all subgrid patches and as a function of depth. They remain defined in that way here. They may be plotted either in horizontal or vertical cross section, but only with the tile plot function. *These variables cannot be output to GrADS, GRIB or Vis5D.*

Field Name	***Description [units]***	***Model Variables***
sltex_p	soil textural class by patch [ ]	SCHAR, TGP
soilq_p	soil internal energy parameter by patch [J/m^3]	SCHAR, TGP
soil_temp_p	soil/sea temperature by patch [deg C]	SCHAR, TGP
soil_moist_p	soil moisture by patch [m^3/m^3]	SCHAR, TGP
soil_moistf_p	soil moisture fraction by patch [m^3/m^3]	SCHAR, TGP

The following variables are made up from a combination of leaf2 variables defined by both patch and depth and those defined by patch only, plus atmospheric variables from the lowest model level. They remain defined in that way here. They may be plotted only in vertical cross section, and only with the tile plot function. *These variables cannot be output to GrADS, GRIB or Vis5D.*

Field Name	***Description [units]***	***Model Variables***
leaf2_moisture	moisture of all leaf2 components	SCHAR, TGP, WGP
leaf2_temp	temperature of all leaf2 components	SCHAR

REVU Input Files

The *REVU_IN* namelist file, required for all ***REVU*** runs, is described in full above. Two other input files are described below.

LANDMARKS

If plotting a horizontal cross-section (see **ZVAR**, **XVAR**, and **YVAR**) and the **LANDMARK** parameter is set to draw landmarks, the locations available for plotting are specified in the *LANDMARKS* file which must reside in the run directory. Landmark locations are specified by a name (up to 16 characters in length) and its latitude-longitude coordinates (as either degree minute second integer qualities, or degree real quantities), noting that line beginning with # are ignored. For example:

```
      # ------- NY --------------------------
      ISP              40 47 38   -073 06 06
      MTP              41 04 23   -071 55 24
      LGA              40 46 45   -073 52 48
      POU              41 37 32   -073 52 55
```

Is equivalent to:

```
      # ------- NY --------------------------
      ISP              40.7939    -73.1017
      MTP              41.0731    -71.9233
      LGA              40.7792    -73.8800
      POU              41.6256    -73.8819
```

Since not all the locations specified in the *LANDMARKS* file are plotted when doing so will cause label text to overlap or the plot to be cluttered (controlled by the **LANDMARK** '**b**' component setting), those locations that appear nearer the beginning of the file may get plotted at the expense of those later in the file. Thus, to ensure you see the locations you wish to see, ensure those locations are at the top of the file, in order of drawing priority.

grab_in

The **$GRAB** namelist parameter **GRABIN** by default points to the file *grab_in*. This file specifies the locations where we want to grab the data for an **ANATYPE** 'GRAB'. Each latitude-longitude location can be specified as either a point in space (requires a height AGL to be given), or a profile of the model levels. The first line contains the file format version number (integer). The remaining lines are either comments or space delimited station location data (one station per line):

```
stat lat lon elev type
```

where:

```
stat = station id number (integer)
lat  = latitude [degrees] (real)
lon  = longitude [degrees] (real)
elev = height AGL in m (real) [=0. for itype=2]
type = 1, get point data at lat,lon,elev (integer)
     = 2, get profile data at lat,lon (RAMS levels)
```

Noting that anything after a # or ! is ignored. For example:

```
1
# some test data (! and # indicate comments)
! st id   lat         lon        elev  itype
  1    40.79388889 -73.10166667   10.    1   ! point data at 10m AGL
  2    41.07305556 -71.92333333   10.    1
  3    41.07305556 -71.92333333   0.     2   ! profile data
  4    40.77916667 -73.88000000   500.   1   ! point data at 500m AGL
  5    41.62555556 -73.88194444   10.    1
```

REVU Output Files

If the type of run defined by **ANATYPE** is 'SPACE', an NCAR Graphics NCGM file named *gmeta* will be output in the current directory (not the directory specified by **REVPREF**). For all other run types the file format:

```
REVPREF<analpref>[SCP]-<date>-g<grid>.<format>
```

where:

- <analpref> is the file prefix given by **AFILOUT** or **HYPPREF**, the ***RAMS*** and ***HYPACT*** output filename prefixes respectively, without the path component.
- [SCP] is determined by **IZTRAN**, i.e.
  S – for **IZTRAN**=1, output on sigma surfaces.
  C – for **IZTRAN**=2, output on Cartesian surfaces.
  P – for **IZTRAN**=3, output on pressure surfaces.
- <date> is the standard date string of the analysis files for the beginning of the ***RAMS*** simulation.
- <grid> is the ***RAMS*** grid number of the data. A grid number of 0 indicates that the file contains information on all grids.
- <format> is the type of output file, as determined by **ANATYPE** below, i.e.
  v5d – for **ANATYPE**=V5D, Vis5d files.
  gra – for **ANATYPE**=GRADS, GrADS files.
  grb – for **ANATYPE**=GRIB, GRIB files.
  gbr – for **ANATYPE**=GRAB, grabber files.
  dmp – for **ANATYPE**=DUMP, dumped files.

If the types of run defined by **ANATYPE** is 'GRAB' and **IGRABFMT** is 0, the selected **CFRAME_A** variables are output in an ASCI file with the following format:

```
date time istat lat lon elev x y z grd (var(i),(i=1,nvars))
```

where:

```
date = YYYYMMDD (integer, i10)
time = HHMMSS (integer, i10)
stat = station id number (integer, i10)
lat  = latitude [degrees] (real, e15.6)
lon  = longitude [degrees] (real, e15.6)
elev = height AGL [m] (real, e15.6) [=0. for itype=2]
x    = RAMS x coordinate [m] (real, e15.6)
y    = RAMS y coordinate [m]  (real, e15.6)
z    = RAMS z coordinate [m AGL] (real, e15.6)
grd  = RAMS grid used (integer, i10)
var  = variable values  (real, e15.6) [up to 30]
```

If the types of run defined by **ANATYPE** = 'GRAB' and **IGRABFMT** = 1, the selected **CFRAME_A** are output in a RALPH2 format ASCI file.