A Quick Guide to Programs in the ARPS System

First written: 5/30/2002 by *Ming Xue* to match ARPS Version 5.0.0IHOP_2
Last updated 06/21/2005 00:05:30 by *Yunheng Wang*

Summary of requirements to run ARPS programs

Platform and Compiler:
All programs inside ARPS package should work on all common Unix platforms and their intrinsic Fortran 90 compilers. Beside a Fortran 90 compiler, some programs require the following utilities which are usually installed with the Operational Systems of the Unix Platforms.

- C language compiler;
- C preprocessor, CPP (/lib/cpp);
- make (GNU make is recommended)l;
- cut;
- awk;
- perl;
- MPI implementation.

The following lists all Unix platforms and their compiler has been tested recently. If you have worked on another platform or with another Fortran compiler, or you would like us to add support for your platform, please let us know (arpssupport@ou.edu).

Platform	Fortran 90 Compiler	C Compiler	Tested MPI Implementations
IBM/AIX	xlf90, xlfg95, xlfxx_r, mpxlfxx, mpxlfxx_r etc.	xlc, or xlc_r or gcc	IBM MPI implementation
Linux (Intel IA-32 processors, Itanium processors, AMD 64 bit processors)	Interl compiler (ifc, ifort) or PGI compiler (pgf90) or GNU compiler (g95)	Interl compiler (icc) or PGI compiler (pgcc) or GNU compiler (gcc)	• MPICH • LAM/MPI
PC windows/Cygwin	GNU compiler (g95)	GNU compiler (gcc)	
DEC Alpha	Intrinsic compiler (f90)	GNU compiler (gcc)	Intrinsic MPI implementation
SGI 2000	Intrinsic compiler (f90)	GNU compiler (gcc)	Intrinsic MPI implementation
SUN	Intrinsic compiler (f90)	GNU compiler (gcc)	

Mac/MacOSX	GNU compiler (g95)	GNU compiler (gcc)	

External Libraries and Packages:

Library	*maekarps* Switchs	Programs	Note
HDF4	-io hdf (default) *-io nohdf* (to disable HDF support)	Most ARPS programs	HDF4 & HDF5 are totally different format. see NCSA HDF home page.
NetCDF	-io net -io nonet (default)	Most ARPS programs	• arps2wrf & wrf2arps always require NetCDF library. • NetCDF 3.0 and HDF 4 format cannot work interchangeably because of library conflicts.
HDF5	-io phdf5 Not required by default	arps2wrf_mpi & wrf2arps_mpi	• Only support PHDF5 WRF files; • MPI-IO required; • Parallel HDF5 required.
NCAR Graphics	-zxncar (arpstrn, arpspltmax, mergetrn, arpsenscv, platradcol, pltgrid etc.) -ncar (arpssfc, arpstern) Not required by default	arpspltncar, ARPSPLTNCAR_mpi Radarpltncar skewtncar, etc	

Conventions

Names of program	capitalized, such as ARPS, ADAS etc.
Executble command	Italic, such as *bin/arps*, *makearps* etc.
File and directory names	Italic, such as *arps.input*, *docs/arps2wrf.pdf* etc.
Compiler or program options	dash following with yellow letters, such as -io nohdf.

ARPS

Program Name:	**ARPS**
Purpose:	NWP model.
Function:	ARPS forward prediction model. Sometimes it also refers to the entire ARPS system, including all supporting packages.
Applications:	Starting from an initiation condition, it performs forward time integration of the governing equations of the atmosphere and produces a forecast of the future state of the atmosphere. The initial condition can be specified by using analytical functions with parameters specified in the input file, by using a single sounding, or by reading in ARPS history or restart format data.
Location of source code:	*src/arps*
Compilation and Linking:	*makearps arps* for a single-processor executable. makearps –p arps for a shared-memory multi-processor executable. It requires a Fortran compiler capable of automatically parallelizing the source codes. The compile typically perform a preprocessing step that inserts loop-level parallelization directives (OpenMP is most common) into the source code first. *makearps arps_mpi* for a distributed memory multi-processor executable. It requires that MPI is set up properly on the system. *makearps* makes some assumptions, such as the location of the MPI library. Check the correctness of the assumptions for the particular platform you are using. *makearps* takes additional option parameters, with *–p* given above as an example. Enter *makearps –help* for additional information on the options.
Execution:	*bin/arps < input/arps.input >! arps.output* for non-MPI runs. To use multiple shared memory processors, certain environmental parameters usually need to be set first, telling the arps job how many processors to use. On SGI system, for example, it is *setenv MP_SET_NUMTHREADS n,* where *n* is the number of processors to use. *mpirun –np n_proc bin/arps_mpi < input/arps.input >! arps.output*. For the MPI run that initializes from an input data set, if you do not use auto-split & atuo-join feature (see *inisplited & dmp_out_joined* in *input/arps.input*), program *splitfiles* needs to be run first. After *arps_mpi* is run, *joinfiles* is run to join together files written out by different processors.
Platform supported:	All common Unix platforms.

External package/ library required :	MPI library is required for distributed-memory parallel runs. HDF Version 4.0 library required *when –io hdf* option is invoked for makearps. ARPS will then be able to read and write HDF format history dumps and boundary condition files. Option *–io hdf* is currently the default. If *–io nohdf* is used, the link to HDF library is bypassed, as a result, no HDF format output will be produced if HDF dump option is chosen (in *arps.input*). HDF format is becoming the preferred data format for ARPS because it is portable cross platforms and contains two levels to building compression so that the file size is typical ¼ of the size of the binary format. NetCDF – Option "-io net" trigger NetCDF support. No NetCDF support by default. Vis5D – history dumps in Vis5D format can be directly written by ARPS by including *–io v5d* option for makearps. It requires a C compiler. GrADS – ARPS can directly write output in GrADS format without the need for any external library. No additional makearps option is necessary. GRIB – ARPS does not support GRIB format since ARPS5.1.0. However, the GRID I/O subroutines were kept to read ARPS GRIB data generated before ARPS5.1.0. Currently, ARPS GRIB files are not portable across big endian (SGI, IBM, SUN, HP and Cray vector machines) and little endian machines (Alpha and Intel processor machines).

ADAS

Program Name:	**ADAS**
Purpose:	Data analysis program based on Bretseth successive correction scheme.
Function:	ARPS Data Analysis System – a system that analyzes observational data onto the ARPS grid.
Applications:	The 3D analysis can be used for diagnostic studies, model initialization and for providing model boundary conditions in the case of simulation (not true forecast). The output data is in ARPS history format which can be directly read by ARPS and many other ARPS programs including ARPSPLT .
Location of source code:	*src/adas*
Compilation and Linking:	*makearps adas* for a single-processor executable. *makearps –p adas* for a shared-memory multi-processor executable.
Execution:	*bin/adas < input/arps.input >! adas.output* ADAS shares an input file with ARPS.

Platform supported:	All common Unix platforms.
External package/ library required :	Necessary libraries, such as HDF, NetCDF, for data I/O.

ARPS3DVAR

Program Name:	**ARPS3DVAR**
Purpose:	3Dvar analysis program.
Function:	Perform objective analysis using three-dimensional variational analysis scheme.
Applications:	To obtain a best estimate of the current state of the atmospheric to serve as the initial condition for forward prediction.
	The 3D analysis can be used for diagnostic studies, model initialization and for providing model boundary conditions in the case of simulation (not true forecast). The output data is in ARPS history format which can be directly read by ARPS and many other ARPS programs including ARPSPLT.
Location of source code:	*src/arps3dvar*
Compilation and Linking:	*makearps arps3dvar* for a single-processor executable. Distributed-memory parallelization is currently not yet supported.
Execution:	*bin/arps3dvar < input/arps.input >! 3dvar.output* 3DVAR shares an input file with ARPS.
Platform supported:	All common Unix platforms.
External package/ library required :	Necessary libraries, - ARPS internal libraries, libarps.a, libadas.a, libradtn.a - External libraries, such as HDF, NetCDF etc. for data I/O.

EXT2ARPS

Program Name:	**EXT2ARPS**
Purpose:	Generates ARPS initial condition history dump, lateral boundary condition files, terrain, surface characteristic, and soil variable files based on external model grid (Eta, RUC, etc.)
Function:	Read in gridded data from external models and interpolate the fields onto the ARPS grid. Write out the fields on the ARPS grid in a standard ARPS history dump format and/or the external boundary condition format.
Applications:	Used to provide analysis background and/or forecast lateral boundary conditions if the ARPS analysis background is from an external model and/or the ARPS is nested inside an external model. Currently EXT2ARPS handles NCEP ETA, RUC, AVN gridded data in various formats and coordinates. NCAR/NCEP global reanalysis data is also loosely supported.
Location of source code:	*src/ext2arps*
Compilation and Linking:	*makearps ext2arps* *makearps ext2arps_mpi*
Execution:	*bin/ext2arps < input/arps.input >! ext2arps.output*
Platform supported:	All common Unix platforms.
External package/ library required :	Libraries needed by specific choice of history I/O format.

ARPSTRN	
Program Name:	**ARPSTRN**
Purpose:	Generates ARPS terrain file using NCAR 5 min global data, USGS 30 arc-second global data, or USGS 3 arc-second north US/Alaska data. Uses bi-linear or bi-quadratic interpolation.
Function:	Prepare a terrain data file on the ARPS grid for use by ARPS and other programs that need to set up the ARPS grid.
Applications:	It reads in one of several terrain data sets and interpolates the data to the ARPS grid. Smoothing is optionally applied to the interpolated terrain field to remove 2 grid-spacing features. The field is written out into a file.
Location of source code:	*src/arpstrn*

Compilation and Linking:	*makearps arpstrn* Compile without linking to ZXPLOT graphics library. No graphic output will be generated by the program.
	makearps –zxncar arpstrn Compile and link with ZXPLOT and NCAR graphics program to produce at the same time a color contour plot of the terrain field in meta file format.
	makearps –zxpost arpstrn Compile and link with ZXPLOT graphics program to produce at the same time a color contour plot of the terrain field in Postscript format.
Execution:	*bin/arpstrn < input/arpstrn.input >! arpstrn.output*
	Currently 5min and 30 second global, and 3-second North American (covers continental US and Alaska) data sets are supported. When using 3 and 30-second data sets, the data will be directly downloaded the CAPS ftp server (caps.ou.edu), respectively. Make sure your computer have FTP access to the internet and network speed is reasonably fast, and you need to set up a *.netrc* file in your home directory for the ftp to work. See *arpstrn.input* for more details.
Platform supported:	All common Unix platforms.
External package/ library required :	NCAR graphics libraries when *–zxncar* option is included for *makearps*.

MERGETRN

Program Name:	**MERGETRN**
Purpose:	Blends two terrain files, insuring continuity for external grids near boundaries while accepting smaller-scale terrain features in the interior.
Function:	To create a terrain file with gradual transition to another terrain field in a boundary zone.
	It reads in two ARPS terrain data files, interpolates the first terrain field to the grid of the second field when necessary, and generates a new terrain field that is same as the second one except in the lateral boundary zone of specified width, where the terrain transitions from that of the second file at the interior of the boundary to that of the first one at the domain boundary.

Applications:	Typically used to ensure that the high-resolution terrain generated for nested grid matches that of the coarse grid at the nesting boundary and that the transition from coarse grid to fine grid is gradual. This function can also be realized in EXT2ARPS and ARPSINTRP programs, in which the fine-grid terrain is merged with that of the coarse grid before interpolation.
Location of source code:	*src/arpstrn*
Compilation and Linking:	*makearps mergetrn*
Execution:	*bin/mergetrn.input < input/mergetrn.input >! mergetrn.output*
Platform supported:	All common Unix platforms.
External package/ library required :	Other libraries needed for the choice of data I/O format.

ARPSINTRP

Program Name:	**ARPSINTRP**
Purpose:	Interpolates data from one ARPS grid to another.
Function:	Read in ARPS gridded data set(s) in history format, and interpolate the fields to another (output) ARPS grid, and write them out in one of the history dump formats for this output grid.
Applications:	Mostly used for generating initial (or background for initial condition analysis) and boundary condition files for one-way nested grid runs inside a coarser ARPS grid. In this case, the new output grid has a higher spatial resolution. The output grid should be no bigger than the input grid. The input and output grids have to have the same map projections, but do not have to have the same vertical coordinates. The fine grid can use higher-resolution terrain or flat terrain. The latter situation is useful when one wants to examine the fields on the ARPS terrain-following grid using software, such as GrADS and Vis5D, that does not support non-rectangular grid. For such a purpose, it is best to choose the option that directly extends surface values below ground level. This program can also be used to sub-sample the ARPS output on a coarser resolution and/or smaller grid for easier post-processing, especially when the original grid is very large. When running ARPSINTRP, an terrain merge option is available, with which the terrain of the output (typically finer resolution) grid is 'merged' with that of the input (typically coarser resolution) grid, in a way similar to what is done in MERGETRN.
Location of source code:	*src/arpsintrp*

Compilation and Linking:	*makearps arpsintrp*
Execution:	*bin/arpsintrp* < *arpsintrp.input* >! *arpsintrp.output*
Platform supported:	All common Unix platforms.
External package/ library required :	Libraries needed for reading and/or writing special format history files.

ARPSTINTRP

Program Name:	**ARPSTINTRP**
Purpose:	Interpolates two ARPS history dumps on grid of the same to a time in-between. The output is in a new history dump file.
Function:	Read in ARPS history format data at two different times and interpolates them to a time between the two. The output is written out into another history format file. It is assumed that both input data are on the same grid.
Applications:	Mostly used to provide a background field for analysis at times when output used for the analysis background (e.g., ETA model output) is not available.
Location of source code:	*src/arpstintrp*
Compilation and Linking:	*makearps arpstintrp*
Execution:	*bin/arpstintrp* < *input/arpstintrp.input* >! *arpstintrp.output*
Platform supported:	All common Unix platforms.
External package/ library required :	Libraries required by the specific history data format.

ARPSPLT

Program Name:	**ARPSPLT**
Purpose:	Main plotting program from ARPS.
Function:	Vector-based plotting program for processing ARPS history-format data.
Applications:	Generates contour and vector plots of 2D cross sections and vertical profiles. The graphical output is either in NCAR graphics meta file format or Postscript format.
Location of source code:	*src/arpsplt*
Compilation and Linking:	*makearps arpspltncar* links with ZXPLOT library and NCAR graphics low-level routines to generate NCAR graphics metafile file. *makearps arpspltpost* links with ZXPLOT library to generate *–p* option can be included for the executable to run on shared-memory multi-processors. For distributed-memory multi-processing, *makearps arpspltncar_mpi* or *makearps arpspltpost_mpi* .
Execution:	*bin/arpspltncar < input/arpsplt.input >! arpsplt.output* *bin/arpspltpost < input/arpsplt.input >! arpsplt.output* Necessary *–io* option should be included when processing special format (e.g., *-io hdf*) history data.
Platform supported:	All common Unix platforms.
External package/ library required :	ZXPLOT graphics library (see http://www.caps.ou.edu/ZXPLOT) is not available with source code inside ARPS package since ARPS5.1.0. NCAR graphics library with *arpspltncar* . NCAR Graphics is freely available from http://ngwww.ucar.edu/ngdoc/ng/ . Other libraries needed for specific history dump format.

PLTGRID

Program Name:	**PLTGRID**
Purpose:	Generate simple graphics plot of a model grid.
Function:	Plot a grid map and nested grid boxes given the grid configuration parameters, including the central longitude and latitude, map projection and grid sizes.
Applications:	To help configure the model domains or a quick look at the model grid given the configuration parameters.
Location of source code:	*src/arpsplt*
Compilation and Linking:	*makearps –zxncar pltgrid* or *makearps –zxpost pltgrid*
Execution:	*bin/pltgrid < input/pltgrid.input*
Platform supported:	All common Unix platforms.
External package/ library required :	NCAR graphics libraries with *–zxncar* option.

ARPSDIFF

Program Name:	**ARPSDIFF**
Purpose:	Reads in two history dumps and outputs differences in the fields. "Analysis" history data is interpolated to "forecast" grid.
Function:	Reads in two sets of ARPS history format data, calculate the difference fields and write out the difference fields into a file in the ARPS history dump format. The two data sets can be on different grids. The 'verification' grid data are interpolated to the 'forecast' grid first before the differences are calculated and the output will be on the forecast grid. If the two grids are the same, no interpolation will be performed.
Applications:	For comparing two sets of ARPS history format data or for 'verifying' one set of ARPS fields against the other (say analysis). It can be used to find the difference, if any, between the outputs of two ARPS runs.
Location of source code:	*src/arpsdiff*
Compilation and Linking:	*makearps arpsdiff*
Execution:	*bin/arpsdiff < input/arpsdiff.input >! arpsdiff.output*
Platform supported:	All common Unix platforms.
External package/ library required :	Libraries required by the specific history data format.

ARPSSFC

Program Name:	**ARPSSFC**
Purpose:	Generates surface characteristic files with soil types and fractions, vegetation types and fractions, LAI, surface roughness, etc.
Function:	Read in soil type, vegetation type and vegetation fraction data files and construct a set of surface and vegetation characteristics fields for the ARPS grid. ARPSSFC and ARPS soil models supports up to 4 different soil types in each grid cell, each caries its own percentage.
Applications:	Prepare land use/land cover (or surface and vegetation) characteristics data file to be used by ARPS soil model.
Location of source code:	*src/arpssfc*
Compilation and Linking:	*makearps arpssfc* *makearps –ncarg arpssfc*
Execution:	*bin/arpssfc < input/arpssfc.input > arpssfc.output*
Platform supported:	All common Unix platforms.
External package/ library required :	NCAR graphics library when *–ncarg* option is included with *makearps*.

ARPSSOIL

Program Name:	**ARPSSOIL**
Purpose:	Generates soil temperature and moisture file using either user-specified values, API data, or NCEP precipitation data. (***Analysis scheme needs to be reworked to allow for more than 2 soil levels!!!***)
Function:	Read in ARPS initial condition data (in history format) and, for the API case, the precipitation data for a period proceeding the initial time in the API case, and creates and writes out an initial condition file for ARPS soil model.
Applications:	Prepare an initial condition file for the ARPS soil model, when such initial conditions are based on offsets from surface atmospheric conditions and the soil moisture content can be derived using API (antecedent precipitation index) method. This program may not be needed when the soil model is initialized by other means, such as interpolating from the soil model state of another model. The soil model variables can be carried in the history file.
Location of source code:	*src/arpssoil*
Compilation and Linking:	*makearps arpssoil*
Execution:	*bin/arpssoil < input/arpssoil.input >! arpssoil.output*

Platform supported:	All common Unix platforms.
External package/ library required :	Libraries required by your specific choice of history data format.

ARPSCVT

Program Name:	**ARPSCVT**
Purpose:	Converts ARPS history dump from one version (binary, HDF, etc.) to another.
Function:	Convert ARPS history dumps among ARPS supported formats.
Applications:	This program can be used to convert the history dump from ARPS into other formats for various purposes. E.g., it can convert a set of history dumps into one Vis5D file for visualization, or Grads format for display. While ARPS can output data in Vis5D or GrADS format directly but it is not recommended because most other programs expect data at individual times.
Location of source code:	*src/arpscvt*
Compilation and Linking:	*makearps –io io_options arpscvt* *io_options* can be *hdf* , *v5d* or *net* .
Execution:	*bin/arpscvt < input/arpscvt.input >! arpscvt.output*
Platform supported:	All common Unix platforms.
External package/ library required :	Libraries required by your specific choice of history data format. Need to include *–io v5d* option for *makearps* in order to write Vis5D format data.

ARPSEXTSND

Program Name:	**ARPSEXTSND**
Purpose:	Interpolates data from ARPS grid to column (generates sounding).
Function:	Extract columns (profiles) from ARPS history dumps, and write out these profiles in a text format.
Applications:	The program can be used to extract profiles for plotting skew-T diagrams using program SKEWT.
Location of source code:	*src/arpsextsnd*
Compilation and Linking:	*makearps arpsextsnd* *makearps arpsextsnd_mpi*
Execution:	
Platform supported:	All common Unix platforms.
External package/ library required :	Libraries required by your specific choice of history data format.

ARPSENSIC, ARPSENSBC

Program Name:	**ARPSENSIC, ARPSENSBC**
Purpose:	**ARPSENSBC:** Generates one set of perturbation LBC's from two sets of ARPS external boundary files and write out for use in ensemble forecast. **ARPSENSIC:** Generates one perturbation IC history dump from two sets of ARPS output files.
Function:	Generate perturbed initial/boundary conditions for ARPS ensemble forecast, using the SLAF (Scaled Lagged Average Forecast) or the BGM (Breeding Fast Growing Mode) method. It reads in three sets of data files in ARPS history format and derive perturbations from the first two and add/subtract the perturbation to the third to generate perturbed initial conditions/boundary conditions.
Applications:	For creating initial and boundary conditions for ARPS ensemble members.
Location of source code:	*src/arpsens*
Compilation and Linking:	*makearps arpsensic* *makearps arpsensbc*
Execution:	*bin/arpsensinc < input/arpsensic.input > arpsensic.output* *bin/arpsensnc < input/arpsensbc.input > arpsensbc.output*
Platform supported:	All common Unix platforms.

External package/ library required :	Libraries needed by specific choice of history I/O format.

ARPSENSCV

Program Name:	**ARPSENSCV**
Purpose:	
Function:	Reads in a series of ARPS history dumps and generate ensemble forecast products, and write the 2D fields out for plotting.
Applications:	Used to process ensemble forecast output.
Location of source code:	*src/arpsens*
Compilation and Linking:	*makearps arpsenscv*
Execution:	*bin/arpsenscv < input/arpsenscv.input >! arpsenscv.output*
Platform supported:	All common Unix platforms.
External package/ library required :	Libraries needed by specific choice of history I/O format.

ARPSVERIF

Program Name:	**ARPSVERIF**
Purpose:	Main verification program.
Function:	
Applications:	
Location of source code:	*src/arpsverif*
Compilation and Linking:	
Execution:	
Platform supported:	All common Unix platforms.
External package/ library required :	

88D2ARPS

Program Name:	**88D2ARPS**
Purpose:	Remaps Level-II data to ARPS grid.
Function:	
Applications:	
Location of source code:	*src/88d2arps* Detail document is in *docs/88d2arps.pdf*.
Compilation and Linking:	*makearps 88d2arps*
Execution:	
Platform supported:	All common Unix platforms.
External package/ library required :	Require C compilers.

NIDS2ARPS

Program Name:	**NIDS2ARPS**
Purpose:	Remaps NIDS Level-III data to ARPS grid.
Function:	
Applications:	
Location of source code:	*src/88d2arps* Document is in *docs/nids2arps.pdf*.
Compilation and Linking:	*makearps nids2arps*
Execution:	*bin/nids2arps < arps.input >! nids2arps.output*
Platform supported:	All common Unix platforms.
External package/ library required :	

ARPSREAD

Program Name:	**ARPSREAD**
Purpose:	Sample program for reading in history dump.
Function:	A template program for reading ARPS history format data.
Applications:	An example for reading ARPS history format data.
Location of source code:	*src/arps*
Compilation and Linking:	*makearps arpsread*
Execution:	*bin/arpsread* < *input/arpsread.input* >*! arpsread.output*
Platform supported:	All common Unix platforms.
External package/ library required :	Libraries needed by specific choice of history I/O format.

ARPSASSIM

Program Name:	**ARPSASSIM** (no support any more)
Purpose:	Assimilation version of ARPS. (Older version, doesn't work?)
Function:	
Applications:	
Location of source code:	*src/arpsassim*
Compilation and Linking:	
Execution:	
Platform supported:	All common Unix platforms.
External package/ library required :	

ARPSADJ

Program Name:	**ARPSADJ** (Still not released)
Purpose:	
Function:	
Applications:	
Location of source code:	
Compilation and Linking:	
Execution:	
Platform supported:	All common Unix platforms.
External package/ library required :	

MCI2ARPS

Program Name:	**MCI2ARPS**
Purpose:	Remaps McIDAS AREA satellite data file to ARPS.
Function:	
Applications:	
Location of source code:	
Compilation and Linking:	*makearps mci2arps*
Execution:	
Platform supported:	All common Unix platforms.
External package/ library required :	Require C compilers.

ARPSAGR

Program Name:	**ARPSAGR**
Purpose:	3D adaptive model interface. (***Doesn't work. Not updated to use new soil model.***)
Function:	
Applications:	
Location of source code:	
Compilation and Linking:	
Execution:	
Platform supported:	All common Unix platforms.
External package/ library required :	

SPLITFILES

Program Name:	**SPLITFILES**
Purpose:	Splits multiple history dumps from multiple valid times into binary or HDF 4 format.
Function:	
Applications:	
Location of source code:	*src/arps_mp*
Compilation and Linking:	*makearps splitfiles* *makearps splitfiles_mpi*
Execution:	*bin/splitfiles* < *input/arps.input* >! *splitfiles.output*
Platform supported:	All common Unix platforms.
External package/ library required :	

JOINFILES

Program Name:	**JOINFILES**
Purpose:	Joins together split binary or HDF 4 history dumps for multiple valid times and outputs in binary or HDF 4 format.
Function:	
Applications:	
Location of source code:	*src/arps_mp*
Compilation and Linking:	*makearps joinfiles*
Execution:	*bin/joinfiles* < *input/arps.input* >! *joinfiles.output*
Platform supported:	All common Unix platforms.
External package/ library required :	

PLTRADCOL

Program Name:	**PLTRADCOL**
Purpose:	Plots remapped radar columns.
Function:	
Applications:	
Location of source code:	*src/88d2arps*
Compilation and Linking:	*makearps pltradcol*
Execution:	
Platform supported:	All common Unix platforms.
External package/ library required :	NCAR Graphics if NCAR meta data is desired.

FAKERAD

Program Name:	**FAKERAD** (No support any more)
Purpose:	Generates fake radar data from an ARPS history dump.
Function:	
Applications:	
Location of source code:	*src/88d2arps*
Compilation and Linking:	
Execution:	
Platform supported:	All common Unix platforms.
External package/ library required :	

RDRDAOSTERN

Program Name:	**RDRDAPSTERN** (No support any more)
Purpose:	Reads KMA RDAPS terrain file and converts to ARPS terrain file format w/ same resolution & map projection.
Function:	
Applications:	
Location of source code:	*src/adas*
Compilation and Linking:	
Execution:	
Platform supported:	All common Unix platforms.
External package/ library required :	

RDRUCTERN

Program Name:	**RDRUCTERN** (No support any more)
Purpose:	Reads RUC terrain file converts to ARPS terrain file format w/ same resolution & map projection.
Function:	
Applications:	
Location of source code:	*src/adas*
Compilation and Linking:	
Execution:	
Platform supported:	All common Unix platforms.
External package/ library required :	

JOINBIN2HDF

Program Name:	**JOINBIN2HDF**
Purpose:	Joins together split binary history dumps and outputs in HDF4 format.
Function:	
Applications:	
Location of source code:	*src/arps_mp*
Compilation and Linking:	*makearps joinbin2hdf*
Execution:	*bin/joinbin2hdf* follows the prompt of the program
Platform supported:	All common Unix platforms.
External package/ library required :	

JOINFILE

Program Name:	**JOINFILE**
Purpose:	Joins together split binary history dumps for one valid time and outputs in binary format.
Function:	
Applications:	
Location of source code:	*src/arps_mp*
Compilation and Linking:	*makearps joinfile*
Execution:	*bin/joinfile*
Platform supported:	All common Unix platforms.
External package/ library required :	

JOINHDF

Program Name:	**JOINHDF**
Purpose:	Joins together split HDF4 history dumps for multiple valid times and outputs in HDF format.
Function:	
Applications:	
Location of source code:	*src/arps_mp*
Compilation and Linking:	*makearps joinhdf*
Execution:	*bin/joinhdf*
Platform supported:	All common Unix platforms.
External package/ library required :	

SPLITHDF

Program Name:	**SPLITHDF**
Purpose:	Splits a HDF4 history dump for one valid time and outputs in HDF4 format.
Function:	
Applications:	
Location of source code:	*src/arps_mp*
Compilation and Linking:	*makearps splithdf*
Execution:	*bin/splithdf*
Platform supported:	All common Unix platforms.
External package/ library required :	

ARPSRAINDIFF

Program Name:	**ARPSRAINDIFF**
Purpose:	Reads in two history dumps on same grid, calculates differences in accumulated rainfall (cumulus and grid-scale), and outputs differences in "wrtvar" format. (***Is wrtvar format used by any other programs?***).
Function:	
Applications:	
Location of source code:	*src/arpsdiff*
Compilation and Linking:	*makearps arpsraindiff*
Execution:	*bin/arpsraindiff*
Platform supported:	All common Unix platforms.
External package/ library required :	

ARPSINTRP_LS

Program Name:	ARPSINTRP_LS
Purpose:	"Simplified" version of ARPSINTRP. (***Not upgraded for new soil model!!!***)
Function:	
Applications:	
Location of source code:	src/arpsintrp
Compilation and Linking:	makearps arpsintrp_ls
Execution:	bin/arpsintrp_ls < input/arpsintrp_ls.input > arpsintrp_ls.output
Platform supported:	All common Unix platforms.
External package/ library required :	

ARPSPLTMAX

Program Name:	ARPSPLTMAX
Purpose:	Produces graphic plots of time series of the domain maximum and minimum of u, v, w, ptprt, pprt, and water quantities. Uses max/min data file produced by the ARPS.
Function:	
Applications:	
Location of source code:	src/arpsplt
Compilation and Linking:	makearps arpspltmax
Execution:	bin/arpspltmax < input/arpspltmax.input > arpspltmax.output
Platform supported:	All common Unix platforms.
External package/ library required :	Require NCAR Graphics if NCAR graphics meta file is desired.

PLOT_COLTAB

Program Name:	**PLOT_COLTAB** (No support any more)
Purpose:	Plots color table from a color table ASCII file.
Function:	
Applications:	
Location of source code:	*src/arpsplt*
Compilation and Linking:	*makearps plot_coltab*
Execution:	*bin/plot_coltab*
Platform supported:	All common Unix platforms.
External package/ library required :	

ARPSPRT

Program Name:	**ARPSPRT**
Purpose:	Generates simple ASCII plots of history dump data.
Function:	
Applications:	
Location of source code:	*src/arpsprt*
Compilation and Linking:	*makearps arpsprt*
Execution:	*bin/arpsprt < input/arpsprt.input > arpsprt.output*
Platform supported:	All common Unix platforms.
External package/ library required :	

MNET2ARPS

Program Name:	**MNET2ARPS**
Purpose:	Generates soil temperature and moisture file using Mesonet data. (***Not updated for new soil variables. Analysis scheme needs to be reworked to allow for more than 2 soil levels!!! Appears to be broken in any case...***)
Function:	
Applications:	
Location of source code:	*src/arpssoil*
Compilation and Linking:	*makearps arpssoil*
Execution:	*bin/arpssoil < input/arpssoil.input > arpssoil.output*
Platform supported:	All common Unix platforms.
External package/ library required :	

ARPSTERN

Program Name:	**ARPSTERN**
Purpose:	Generates ARPS terrain file using NCAR 1 deg/5min/30sec terrain data and Barnes analysis scheme.
Function:	
Applications:	
Location of source code:	*src/arpstern*
Compilation and Linking:	*makearps arpstern*
Execution:	*bin/arpstern < input/arpstern.input > arpstern.output* Need to execute *dir5min*, *dir30sec* , and *dir1deg* first.
Platform supported:	All common Unix platforms.
External package/ library required :	

DIR1DEG

Program Name:	**DIR1DEG**
Purpose:	Reformats NCAR 1 deg terrain data for use by ARPSTERN.
Function:	
Applications:	
Location of source code:	*src/arpstern*
Compilation and Linking:	*makearps dir1deg*
Execution:	*bin/dir1deg < input/arsptern.input > dir1deg.output*
Platform supported:	All common Unix platforms.
External package/ library required :	

DIR30SEC

Program Name:	**DIR30SEC**
Purpose:	Reformats NCAR 30 sec terrain data for use by ARPSTERN.
Function:	
Applications:	
Location of source code:	*src/arpstern*
Compilation and Linking:	*makearps dir30sec*
Execution:	*bin/dir30sec < input/arpstern.input > dir30sec.output*
Platform supported:	All common Unix platforms.
External package/ library required :	

DIR5MIN

Program Name:	**DIR5MIN**
Purpose:	Reformats NCAR 5 min terrain data for use by ARPSTERN.
Function:	
Applications:	
Location of source code:	*src/dir5min*
Compilation and Linking:	*makearps dir5min*
Execution:	*bin/dir5min* < *input/arpstern.input* > *dir5min.output*
Platform supported:	All common Unix platforms.
External package/ library required :	

ARPS2GEM

Program Name:	**ARPS2GEM**
Purpose:	Converts ARPS history dumps to GEMPAK format file.
Function:	
Applications:	
Location of source code:	*src/ext2arps*
Compilation and Linking:	*makearps arps2gem*
Execution:	
Platform supported:	All common Unix platforms.
External package/ library required :	

ARPS2NCDF

Program Name:	**ARPS2NCDF**
Purpose:	Converts ARPS history dumps to netCDF format file for LDADS/AWIPS 2D display of data.
Function:	
Applications:	
Location of source code:	*src/ext2arps*
Compilation and Linking:	*makearps arps2ncdf*
Execution:	
Platform supported:	All common Unix platforms.
External package/ library required :	

EXTRACT_AVN

Program Name:	**EXTRACT_AVN**
Purpose:	Extracts and writes out a section of NCEP AVN GRIB data.
Function:	
Applications:	
Location of source code:	*src/ext2arps*
Compilation and Linking:	*makearps extract_avn*
Execution:	
Platform supported:	All common Unix platforms.
External package/ library required :	

MERGESAT

Program Name:	**MERGESAT**
Purpose:	Merges multiple ARPS satellite format files over the same domain to one file.
Function:	
Applications:	
Location of source code:	*src/mci2arps*
Compilation and Linking:	*makearps mergesat*
Execution:	
Platform supported:	All common Unix platforms.
External package/ library required :	

SAT2ARPS

Program Name:	**SAT2ARPS**
Purpose:	Remaps NOAAPORT satellite images to ARPS grid.
Function:	
Applications:	
Location of source code:	*src/mci2arps*
Compilation and Linking:	*makearps sat2arps*
Execution:	
Platform supported:	All common Unix platforms.
External package/ library required :	

PLTSATFLD

Program Name:	**PLTSATFLD**
Purpose:	Plots remapped satellite data.
Function:	
Applications:	
Location of source code:	*src/mci2arps*
Compilation and Linking:	*makearps pltsatfld*
Execution:	
Platform supported:	All common Unix platforms.
External package/ library required :	

WTRETCOL

Program Name:	**WTRETCOL**
Purpose:	Writes fake retrieval columns.
Function:	
Applications:	
Location of source code:	*src/wtretcol*
Compilation and Linking:	*makearps wtretcol*
Execution:	
Platform supported:	All common Unix platforms.
External package/ library required :	

ARPS2RAD

Program Name:	**ARPS2RAD**
Purpose:	Using an ARPS history dump as input create a remapped radar file of data to simulate the atmosphere in the history file being observed by a radar.
Function:	
Applications:	
Location of source code:	*src/88d2arps*
Compilation and Linking:	*makearps arps2rad*
Execution:	*arps2rad has its own input file, ./input/arps2rad.input which includes the history file to read and the location of the observing radar.* *Need to link libarps.a & libadas.a*
Platform supported:	All common Unix platforms.
External package/ library required :	HDF 4 by default or with "-io hdf" option.

DIFOBS

Program Name:	**DIFOBS**
Purpose:	Calculates the difference between a set of observations and the gridded fields in an ARPS history file. Statistics of bias and rms are reported, broken down by data source. This program allows calculation of verification statistics without running the entire arpsverif code.
Function:	
Applications:	
Location of source code:	*src/adas*
Compilation and Linking:	*makearps difobs*
Execution:	*difobs uses the arps.input file set up as if the data were to be used in adas. That is, the data files are named in the input file and the "iuse" switches for passes 1-to-npass are used to determine which data sources contribute to the global statistics. Output is written to a file in the present working directory with name runname.difobs. If a file with that name already exists, a sequential number is appended to the name.* *Need to link with libarps.a & libadas.a*

Platform supported:	All common Unix platforms.
External package/ library required :	

RADMOSAIC

Program Name:	**RADMOSAIC**
Purpose:	Create a mosaic of radar reflectivity from the remapped radar file in the same manner as is done for the cloud analysis. The mosaic file can be displayed in arpsplt using the "arbvar" option. Plots can be used for verification figures or comparison to analysis output. This program is NOT required for ADAS. ADAS does the same mosaicking within the cloud analysis.
Function:	
Applications:	
Location of source code:	*src/adas*
Compilation and Linking:	*makearps radmosaic*
Execution:	radmosaic has its own input file, ./input/radmosaic.input which consists primarily of a list of radar files to include in the mosaic. See documentation in radmosaic.input *Need to link with libarps.a & libadas.a*
Platform supported:	All common Unix platforms.
External package/ library required :	

ARPS2WRF

Program Name:	**ARPS2WRF**
Purpose:	Prepare ARPS data to run WRF model.
Function:	Will replace both WRFSI & *real.exe* steps starting from ARPS data. ARPS2WRF will generate • WRF initialization file in NetCDF format; • WRF lateral boundary file in NetCDF format; • WRF namelist input file.

Applications:	To run WRF model from ARPS data which includes ADAS analysis data.
Location of source code:	*src/arps2wrf* *docs/arps2wrf.hdf* for documentation.
Compilation and Linking:	*makearps arps2wrf* *makearps arps2wrf_mpi*
Execution:	arps2wrf has its own input file, ./input/arps2wrf.input. See documentation in arps2wrf.input. bin/arps2wrf < input/arps2wrf.input >&! arps2wrf.output *Need to link with libarps.a & libadas.a*
Platform supported:	All common Unix platforms.
External package/ library required :	NetCDF 3.0 or PHDF5 if "-io phdf5" option is specified with *makearps*.

WRF2ARPS

Program Name:	**WRF2ARPS**
Purpose:	Convert WRF output to ARPS history format file.
Function:	WRF2ARPS does the same work as EXT2ARPS, but just for WRF outputs.
Applications:	To run ARPS model from WRF data or use ARPS post-processing utilities to process WRF data.
Location of source code:	*src/wrf2arps*

Compilation and Linking:	*makearps wrf2arps* *makearps wrf2arps_mpi*
Execution:	wrf2arps has its own input file, ./input/wrf2arps.input. See documentation in wrf2arps.input. bin/wrf2arps < input/wrf2arps.input >&! wrf2arps.output *Need to link with libarps.a & libadas.a*
Platform supported:	All common Unix platforms.
External package/library required :	NetCDF 3.0 or PHDF5 library if using "-io phdf5" wtih *makearps*.

ARPS2ETA212

Program Name:	**ARPS2ETA212**
Purpose:	Convert ARPS history format file to Eta #212 grib format.
Function:	
Applications:	Note: ARPS domain must cover the whole Eta #212 domain. See NCEP #212 specification for details.
Location of source code:	*src/ext2arps*
Compilation and Linking:	*makearps arps2eta212*
Execution:	arps2eta212 has its own input file, ./input/arps2eta212.input. See documentation in arps2eta212.input. bin/arps2eta212 < input/arps2eta212.input >&! arps2eta212.output *Need to link with libarps.a & libadas.a*

Platform supported:	All common Unix platforms.
External package/ library required :	HDF 4 by default or with "-io hdf" option.

HDF2GRADS

Program Name:	**HDF2GRADS**
Purpose:	Processes ARPS HDF format history dumps and generates a GrADS control file for the HDF file.
Function:	hdf2grads will change the HDF dimension definitions on-line. The changed file should be also readable by the ARPS system.
Applications:	Use GrADS on ARPS HDF format dumps.
Location of source code:	*src/arpscvt*
Compilation and Linking:	*makearps hdf2grads*
Execution:	bin/hdf2grads then enter the file name to be processed. After the GrADS control file is generated. Use *gradshdf* to enter GrADS graphical tools and then ga-> xdfopen control_file_name Display the data in the HDF file as usually in GrADS.
Platform supported:	All common Unix platforms.
External package/ library required :	HDF 4

SKEWT

Program Name:	**SKEWT**
Purpose:	Produces a Skew-T diagram for a raw or model sounding.
Function:	
Applications:	
Location of source code:	*src/skewt*
Compilation and Linking:	*makearps skewtncar* to generate NCAR gmeta file. *makearps skewtpost* to generate Postscript file.
Execution:	*bin/skewtncar* or *bin/skewtpost* without arguments to print help about usage. *bin/skewtncar* [options] *sounding_file[s]*
Platform supported:	All common Unix platforms.
External package/ library required :	Necessary libraries, • ARPS internal libraries, libzxpost.a or libzxncar.a

RADARPLT

Program Name:	**RADARPLT**
Purpose:	Plots radar reflectivity and radial velocity fields for individual elevations
Function:	
Applications:	A program to plot radial velocity and reflectivity data in PPI (in elevation plane or tilt) and RHI (range-height cross-section) mode for individual radars. The input files can be output from 88d2arps or xxxx (to be added later), and the radar emulation program. A new call in 88d2arps was added to create the tilt files. Use switch "-wrttilts_on" to turn it on when running 88d2arps.
Location of source code:	*src/88d2arps/pltradarscan.f90*
Compilation and Linking:	*makearps radarpltncar* to generate NCAR gmeta plots. *makearps radarpltpost* to generate PostScript files.
Execution:	*bin/radarpltncar* or *bin/radarpltpost* < *radarplt.input*
Platform supported:	All common Unix platforms.

External package/ library required :	Necessary libraries, • ARPS internal libraries, libarps.a and libzxpost.a or libzxncar.a

NEWPROGRAM

Program Name:	**NEW**
Purpose:	*Add new program here*
Function:	
Applications:	
Location of source code:	
Compilation and Linking:	
Execution:	
Platform supported:	
External package/ library required :	

Program Name:	**F77TOF90**
Purpose:	convert_inc_to_mod: Converts FORTRAN 77 include file to Fortran 90 module. convert_src_to_f90: Converts FORTRAN 77 source file to Fortran 90 file.
Function:	Convert Fortran 77 code into the Fortran 90 free form and stylize the code to conform to the ARPS Fortran 90 coding standard.
Applications:	Used to convert F77 ARPS 4.5.x into F90 ARPS 5.0.
Location of source code:	*src/f77tof90*
Compilation and Linking:	*f90 –o bin/f77tof90 src/f77tof90*

Execution:	/bin/ls *.f > f77_filelist ; bin/f77tof90 < f77_filelist
Platform supported:	All common Unix platforms.
External package/ library required :	None

Program Name:	**Perl Scripts**
Purpose:	
Function:	
Applications:	
Location of source code:	*scripts/*
Compilation and Linking:	
Execution:	
Platform supported:	All common Unix platforms with Perl 5.0 and above.
External package/ library required :	

CAPS © June 2002.