Appendix G: Alternative Plotting Package - RIP

G.1 Purpose

Program RIP is an alternative plotting utility program for the MM5 modeling system. RIP (which stands for Read/Interpolate/Plot) is a Fortran program that invokes NCAR Graphics routines for the purpose of visualizing output from the Fourth and Firth Generation versions of the PSU/NCAR Mesoscale Modeling (MM4/MM5) System. It has been under continues development since 1991, primarily by Mark Stoelinga at both NCAR and the University of Washington. The RIP program makes shaded/color plots and the overlaying of more than two fields much easier. This package become so popular among many MM5 users that it has since been incorporated into the MM5 programs and are supported by mesouser.

RIP can plot output from most of the MM5 programs; TERRAIN_DOMAINx, REGRID_DOMAINx, RAWINS_DOMAINx, LITTLE_R_DOMAINx, MMINPUT_DOMAINx, LOWBDY_DOMAINx, and MMOUT_DOMAINx.

RIP can produce the following output:

- generates horizontal plots on σ, pressure, θ, θ_e, or PV surfaces;
- generates vertical plots on σ, pressure or log pressure, π, θ, θ_e, or PV as coordinate surface;
- plot Skew-T/log p soundings at grid point or lat/lon location, with optional hodographs and sounding quantities;
- generate single-point profiles;
- generate forward and backward trajectories; and
- generate input data for Vis5D

The program is well documented. Documentation can be obtained inside the program tar file in the Doc/ directory, and on MM5 home page at *http://www.mmm.ucar.edu/mm5/doc.html*.

The newest version of this code is called RIP4. The difference between this new version and the older version, is that RIP4 can also ingest WRF model data.

G.2 RIPDP

RIP does not read MM5 output directly, so the RIP Data Preparation (RIPDP) is used to read the MM5 output files, process the data and output data in a format that RIP can read. Once RIPDP has run, every variable at each time period in the MM5 output file will have been placed in a separate file. Note, this process generates a substantial number of RIP input files.

RIPDP makes use of a namelist to extract data from the MM5 output file and process the required data:

```
&userin
 ptimes=0,-72,1,ptimeunites='h',tacc=90.,discard='LANDMASK',H2SO4',
 iexpandedout=1
&end
```

Table G-1: RIPDP namelist

Namelist Variable	Variable Type	Description
ptimes	INTEGER	Times to process. This can be a string of times or a series in the form of *A,-B,C*, which means "times from hour *A*, to hour *B*, every *C* hours"
ptimeunit	CHARACTER	Time units. This can be either 'h' (hours), 'm' (minutes), or 's' (seconds)
tacc	REAL	Time tolerance in seconds. Any time in the model output that are within *tacc* seconds of the time specified in *ptimes* will be processed.
discard	CHARACTER	All variables listed here, will not be processed.
iexpandedout	INTEGER	Only relevant for TERRAIN of REGRID output. If output is on an expanded domain and this flag is set to 1, the output will be plotted on the expanded domain.

G.3 RIP UIF

Once the data is in the correct format, the desired plots can be generated with the use of a RIP UIF (User Input File):

```
&userin
 idotitle=1,titlecolor='def.foreground',
 ptimes=0,6,12,
 ptimeunits='h',tacc=120,timezone=-7,iusdaylightrule=1,
 iinittime=1,ivalidtime=1,inearesth=0,
 flmin=.09, frmax=.92, fbmin=.10, ftmax=.85,
 ntextq=0,ntextcd=0,fcoffset=0.0,idotser=0,
 idescriptive=1,icgmsplit=0,maxfld=10,itrajcalc=0,imakev5d=0
&end
&trajcalc
 rtim=15,ctim=6,dtfile=3600.,dttraj=600.,vctraj='s',
 xjtraj=95,90,85,80,75,70,65,80.6,80.6,80.6,80.6,80.6,80.6,
 yitraj=50,55,60,65,70,75,80,77,77,77,77,77,77,
 zktraj=.9,.9,.9,.9,.9,.9,.9,.99,.9,.8,.7,.6,.5,
 ihydrometeor=0
&end
================================================================================
----------------------    Plot Specification Table    ----------------------
================================================================================
feld=xlus; ptyp=hh; chfl; cosq=1,dark.gray,2,light.yellow,3,light.green,>
   4,yellow,5,yellow,6,light.green,7,light.yellow,8,light.green,>
   9,light.green,10,light.yellow,11,green,12,dark.green,13,green,>
   14,dark.green,15,green,16,light.blue,17,green,18,green,>
   19,light.gray,20,light.gray,21,dark.green,22,light.gray,>
   23,light.gray,24,white
feld=map; ptyp=hb
```

```
feld=tic; ptyp=hb
time=0
================================================================================
feld=ter; ptyp=hc; cint=100; colr=red
feld=map; ptyp=hb
feld=tic; ptyp=hb
time=0
================================================================================
feld=ter; ptyp=hc; cint=50; cmth=fill; cosq=-1e-5,light.blue,1e-5,white,>
   3000,brown
feld=map; ptyp=hb
feld=tic; ptyp=hb
time=0
================================================================================
feld=tmc; ptyp=hc; vcor=s; levs=b1; cint=2; cmth=fill;>
   cosq=-32,light.violet,-24,violet,-16,blue,-8,green,0,yellow,8,red,>
   16,orange,24,brown,32,light.gray
feld=slp; ptyp=hc; cint=2; linw=2
feld=uuu,vvv; ptyp=hv; vcmx=-1; colr=white; intv=5
feld=map; ptyp=hb
feld=tic; ptyp=hb
================================================================================
feld=tmc; ptyp=hc; vcor=p; levs=850,700,-300,100; cint=2; cmth=fill;>
   cosq=-32,light.violet,-24,violet,-16,blue,-8,green,0,yellow,8,red,>
   16,orange,24,brown,32,light.gray
feld=ght; ptyp=hc; cint=30; linw=2
feld=uuu,vvv; ptyp=hv; vcmx=-1; colr=white; intv=5
feld=map; ptyp=hb
feld=tic; ptyp=hb
================================================================================
feld=pvo; ptyp=vc; crsa=10,30; crsb=30,10; vcor=p; vwin=1050,200; cint=.25;>
   cmth=fill; cosq=0,white,4,dark.gray; cbeg=0; cend=5
feld=the; ptyp=vc; cint=2; colr=red
feld=uuu,vvv,omg; ptyp=vv
feld=tic; ptyp=vb
================================================================================
feld=tic; ptyp=sb; sloc=KORD; hodo; sndg
feld=tmc; ptyp=sc; colr=red
feld=tdp; ptyp=sc; colr=blue
feld=uuu,vvv; ptyp=sv; colr=dark.green; hodo; sndg
================================================================================
```

A RIP UIF consists of two namelists (*userin* - which control the general input specifications; and *trajcalc* - which control the creation of trajectories), and a Plot Specification Table (PST), which control the plotting of the required frames.

Table G-2: RIP namelist: *userin*

Namelist Variable	Variable Type	Description
idotitle	INTEGER	Control of first part of title line.
titlecolor	CHARACTER	Control color of the title lines

Table G-2: RIP namelist: *userin*

Namelist Variable	Variable Type	Description
ptimes	INTEGER	Times to process. This can be a string of times or a series in the form of *A,-B,C,* which means "times from hour *A*, to hour *B*, every *C* hours"
ptimeunits	CHARACTER	Time units. This can be either 'h' (hours), 'm' (minutes), or 's' (seconds)
tacc	REAL	Time tolerance in seconds. Any time in the model output that are within *tacc* seconds of the time specified in *ptimes* will be processed.
timezone	INTEGER	Specifies the offset from Greenwich time.
iusdaylightrule	INTEGER	Flag to determine if US daylight saving is applied.
iinittime	INTEGER	Controls the plotting of the initial time on the plots.
ivalidtime	INTEGER	Controls the plotting of the plot valid time.
inearesth	INTEGER	Plot time as two digits rather than 4 digits.
flmin	REAL	Left frame limit
flmax	REAL	Right frame limit
fbmin	REAL	Bottom frame limit
ftmax	REAL	Top frame limit
ntextq	INTEGER	Quality of the text
ntextcd	INTEGER	Text font
fcoffset	INTEGER	Change initial time to something other than output initial time.
idotser	INTEGER	Generate time series output files (not plots only ASCII file that can be used as input to a plotting program).
idescriptive	INTEGER	Use more descriptive plot titles.
icgmsplit	INTEGER	Split metacode into several files.
maxfld	INTEGER	Reserve memory for RIP.
ittrajcalc	INTEGER	Generate trajectory output files (use namelist *trajcalc* when this is set).
imakev5d	INTEGER	Generate output for Vis5D

G.4 RIP PST

The second part of the RIP UIF consists of the Plot Specification Table. The PST provides all of the user control over particular aspects of individual frames and overlays. The basic structure of the PST is as follows:

- The first line of the PST is a line of consecutive equal signs. This line as well as the next two lines are ignored by RIP, it is simply a banner that says this is the start of the PST section.
- After that there are several groups of one or more lines separated by a full line of equal signs. Each group of lines is a frame specification group (FSG), and it describes what will be plotted in a single frame of metacode. Each FSG must be ended with a full line of equal signs, so that RIP can determine where individual frames starts and ends.
- Each line within a FGS is referred to as a plot specification line (PSL). A FSG that consists of three PSL lines, will result in a single metacode frame with three overlaid plots.

Examples of frame specification groups (FSG's):

```
================================================================================
feld=tmc; ptyp=hc; vcor=p; levs=850,700,-300,100; cint=2; cmth=fill;>
   cosq=-32,light.violet,-24,violet,-16,blue,-8,green,0,yellow,8,red,>
   16,orange,24,brown,32,light.gray
feld=ght; ptyp=hc; cint=30; linw=2
feld=uuu,vvv; ptyp=hv; vcmx=-1; colr=white; intv=5
feld=map; ptyp=hb
feld=tic; ptyp=hb
================================================================================
```

This FSG will generate 5 overlaid plots:

- Temperature in degrees C (*feld=tmc*). This will be plotted as a horizontal contour plot (*ptyp=hc*), on pressure levels (*vcor=p*). The pressure levels used will be 850 and 700 to 300 in steps of 100 mb (thus 5 plots will be generated, on 850, 700, 600, 500, 400, and 300 mb). The contour intervals are set to 2 (*cint=2*), and shaded plots (*cmth=fill*) will be generated with a color range from light violet to light gray.
- Geopotential heights (*feld=ght*), will also be plotted as a horizontal contour plot. This time the contour intervals will be 30 (*cint=30*), and contour lines, with a line width of 2 (*linw=2*) will be used.
- Wind vectors (*feld=uuu,vvv*), plotted as barbs (*vcmax=-1*).
- A map background will be displayed (*feld=map*), and
- Tic marks will be placed on the plot (*feld=tic*).

```
================================================================================
feld=pvo; ptyp=vc; crsa=10,30; crsb=30,10; vcor=p; vwin=1050,200; cint=.25;>
   cmth=fill; cosq=0,white,4,dark.gray; cbeg=0; cend=5
feld=the; ptyp=vc; cint=2; colr=red
feld=uuu,vvv,omg; ptyp=vv
feld=tic; ptyp=vb
================================================================================
```

This FSG will generate 4 overlaid plots:

- Potential Vorticity (*feld=pvo*). This will be plotted as a vertical contour plot (*ptyp=vc*), from grid point 10,30 to grid point 30,10 (*crsa=10,30; crsb=30,10*). The vertical coordinate used is pressure (*vcor=p*), with a window from 1050 to 200 mb (*vwin=1050,200*). The contour

intervals are set to .25 (*cint=.25*), and shaded plots (*cmth=fill*) will be generated with a color range from white to dark gray. Only values between 0 and 5 will be plotted (*cbeg=0; cend=5*).

- Potential temperature (*feld=the*), will also be plotted as a vertical contour plot. This time the contour intervals will be 2 (*cint=2*), and the contour lines will be plotted in red (*colr=red*).
- 3D circulation vectors (*feld=uuu,vvv,omg*) in the plane of the cross-section, and
- Tic marks will be placed on the plot (*feld=tic*).

```
==========================================================================
feld=tic; ptyp=sb; sloc=KORD; hodo; sndg
feld=tmc; ptyp=sc; colr=red
feld=tdp; ptyp=sc; colr=blue
feld=uuu,vvv; ptyp=sv; colr=dark.green; hodo; sndg
==========================================================================
```

This FSG will generate a sounding plot:

- Temperature in degrees C (*tmc*), and dew point temperatures (*tdp*), will be plotted in red and blue on a skew-T/log p diagram.
- Wind barbs (*uuu,vvv*) will be plotted on the side in green.
- The station for which the plot is generated is ORD (*sloc=KORD ; see station list in RIP directory for station names and locations*).
- A hodograph (*hodo*) as well as sounding information (*sndg*) will be plotted.

G.5 How to Run RIP

1) Obtain the source code tar file from one of the following places:

Anonymous ftp:
ftp://ftp.ucar.edu/mesouser/MM5V3/RIP.TAR.gz

On NCAR MSS:
/MESOUSER/MM5V3/RIP.TAR.gz

2) gunzip the file, untar it. A directory RIP will be created. cd to RIP.

3) Set up the environment variables NCARG_ROOT & RIP_ROOT
setenv NCARG_ROOT /usr/local/ncarg
(note: location of NCAR Graphics library may vary on your machine)
setenv RIP_ROOT *your-rip-directory*
(example: setenv RIP_ROOT /home/usr/MM5V3/RIP)

4) Type ‘make’ to obtain a list of machines on which the code can be compiled.
Type ‘make *your-machine*’ to create an executable for your platform.

5) Edit ripdp_sample.in to set up the namelist for the data processing, and run ripdp
ripdp -n ripdp_sample.in *model-data-set-name data-file-1 data-file-2 data-file-3 ...*
(Use ripdp_mm5 for RIP4)

where: *model-data-set-name* is the name (including path if data is to be written to a directory) that will be used as a prefix to save all the newly created processed data. Since this set creates a lot of file, it is a good practice to write these files to a directory.
data-file-1 is the MM5 output file that are going to be viewed with RIP (if you have split files, list them)

6) Edit rip_sample.in to set up the namelists for plotting as well as enter the specific plots you wish to create in the PST.

7) Run RIP

rip -f *model-data-set-name* rip-sample.in

8) If successful this will create a metacode file *rip_sample.cgm*, which can be viewed with the NCAR Graphics utility 'idt'.