
GRAPH



12

-
- Purpose 12-3
 - Typical GRAPH Jobs 12-4
 - Plotting Table File: g_plots.tbl 12-5
 - Default Option Settings File: g_defaults.nml 12-7
 - Map Options File: g_map.tbl 12-8
 - Plot Color Options File: g_color.tbl 12-10
 - How to Run GRAPH 12-11
 - Available 2-D Horizontal Fields 12-14
 - Available Cross-Section Only Fields 12-17
 - Available 3-D Fields (as 2-D Horizontal or Cross-Section) 12-17
 - Some Hints for Running GRAPH 12-20
 - Sample Graph Plot File 12-21
 - Graph tar file 12-22
 - Script file to run Graph job 12-22
 - An Alternative Plotting Packages RIP 12-25

12

GRAPH

12.1 Purpose

The GRAPH program generates simple diagnostics and plots for some standard meteorological variables. The GRAPH code will process multiple times and vertical levels, computing the same diagnostics for each time and level. The GRAPH code will provide simple vertical interpolation capability, cross-section figures, and skew-T plots. The GRAPH program can overlay two plots. The GRAPH code is written to be used as a batch processor, so that all graphical choices are made from tables. The GRAPH code can process data from TERRAIN, REGRID, little_r and RAWINS, INTERPF, MM5, NESTDOWN, LOWBDY, and INTERPB. But GRAPH code cannot plot boundary condition data. The GRAPH code does not produce any standard output for use by a subsequent program.

The GRAPH code in MM5 system is built on NCAR Graphics library (which is a licensed software: <http://ngwww.ucar.edu>, but part of it has become free which is sufficient to be used by all MM5 modeling system programs that require NCAR Graphics). It can be run on IBMs, Crays, workstations, and PC running Linux where NCAR Graphics is installed. When working on an IBM, a user can run GRAPH in batch or interactive mode. Examples of the interactive GRAPH use are shown in the section 12.7.

Note on compiling GRAPH on a PC:

When compiling on a PC running Linux using Portland Group Fortran compiler, it may require a library called libf2c.a. This library is required because NCAR Graphics library is compiled with GNU f77, while the GRAPH program requires PGF77 (or PGF90 - in order to deal with pointers). This library may or may not be available on your system. If it isn't, you may obtain it from the internet for free.

12.2 Typical GRAPH Jobs

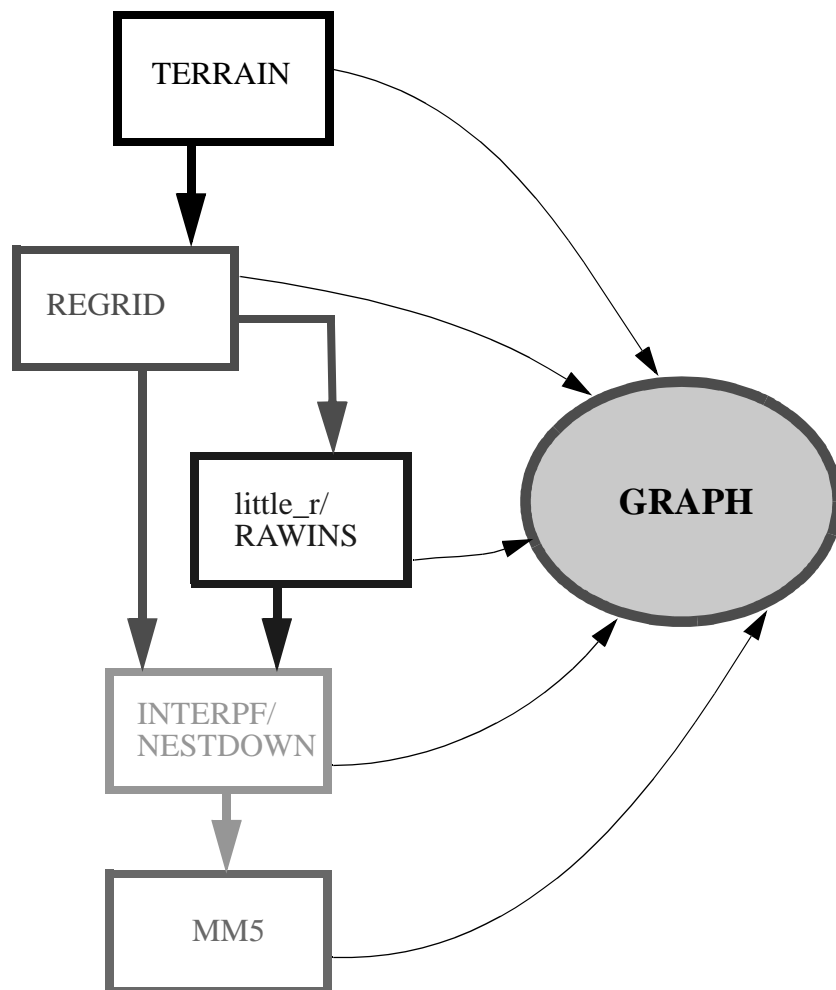


Fig. 12.1 Schematic diagram showing GRAPH accepting data from outputs of MM5 modeling system.

12.3 Plotting Table File: g_plots.tbl

This table is used to define times, levels and fields to be processed and plotted by Graph. An example is shown below:

```

TIME LEVELS: FROM 1993-03-13_00:00:00 TO 1993-03-14_00:00:00 BY 21600 (A)
PRESSURE LEVEL MANDATORY: FROM SFC TO PTOP (B)
PRESSURE LEVEL NON STANDARD: FROM SFC TO PTOP BY 3 (C)
SIGMA LEVEL: FROM 23 TO KMAX BY 5 (D)
TITLE: MM5 Tutorial (E)

```

PLOT T/F	FIELD	UNITS	CONTOUR INTERVAL	SMOOTH PASSES	OVERLAY FIELD	UNITS	CONTOUR INTERVAL	SMOOTH PASSES
T	TER	m	100	0				
T	WIND	m/s	5	0	BARB	m/s	2	0
TP500	HEIGHT	m	30	0	VOR	10**5/s	0	0
TI305	PV	PVU	1	0	P	mb	20	0
X	5	5	23	8	PSLV	mb	2	0
X	THETA	K	3	0	CXW	m/s	10	0
X	5	18	23	5	PSLV	mb	2	0
X	THETA	K	3	0	CXW	m/s	10	0
T	SKEWTLL	72469 DEN	DENVER, CO	39.75	-104.87			
T	SKEWTXY	STATION IN (X,Y)		19	30			

Description of Table Header Rows:

- (A) *TIME* with beginning and ending times given in YYYY-MM-DD_HH:MM:SS, and time increment given in seconds defined by the number after *BY*. If one doesn't use :SS, the increment should be in minutes, and if one doesn't use :MM, then the increment should be in hours. '*BY 0*' means to plot every output times.
- (B) *MANDATORY* used for pressure level dataset (such as from DATAGRID and Rawins). Will plot every mandatory level from the maximum and minimum level requested. ALL and NONE may also used to replace the entire string after the colon.
- (C) *NON STANDARD* used for pressure level dataset. Will plot every mandatory level from the maximum and minimum level requested. Optional use of *BY n* will make plots at every n levels. ALL and NONE may also be used to replace the entire string after the colon.
- (D) *SIGMA* used for σ -level data. Will plot the levels specified by indexes (K=1 is at the top of the model). An increment is required, and defined by the number after *BY*. Can also use ALL or NONE.
- (E) *TITLE* except for a colon (:), any alpha-numeric character can be used to make a simple 1 line, 80 character title

Description of Table Columns:

<i>PLOT T/F</i>	True or False to plot this field. Removing the line from the table has the same effect as F. If the user requests a cross section plot, the letter is X. If the user requests a plot on pressure level, the first 2 characters are TP, followed by the pressure value (TP500 is 500 mb level); if the user requests a plot on isentropic surface, the first 2 characters are TI, followed by the potential temperature value (TI330 is 330 K level). The last two options only work with σ data.
<i>FIELD</i>	a field name to be plotted. See complete list in Tables 8.1, 8.2 and 8.3. If the field is a skew-T, the interpretation of the following columns is changed (see the explanation below).
<i>UNITS</i>	units used on a plot. For some fields, there are different standard units available. If you don't know the unit, use '?'. If you don't know the unit, use '?'.
<i>CONTOUR INTERVAL</i>	real or integer values used to contour a plot. If you don't know what contour interval to use, use '0'. For a vector field (e.g. BARB), this value specifies the grid interval. For streamline field (VESL), this value specifies how sparse or dense streamlines are drawn.
<i>SMOOTH PASSES</i>	number of passes of the smoother-desmoother used for each horizontal plot.
<i>OVERLAY FIELD</i>	a field name for the overlay plot. May be left blank.

To create pressure-level plot from sigma-level data:

```
TP500 | HEIGHT | m | 30 | 0 || VOR | 10**5/s | 0 | 0
```

To create isentropic-level plot from sigma-level data:

```
TI305 | PV | PVU | 1 | 0 || P | mb | 20 | 0
```

To plot skew-T:

If the plot is a skew-T (SKEWTLL or SKEWXY), the UNITS column is used to define location name, and lat/long or X/Y appear in the following two columns. e.g.

```
T | SKEWTLL | 72469 DEN DENVER, CO | 39.75 | -104.87 || | | |
```

To plot a vertical cross-section:

For a cross-section plot, the location is defined by the 4 numbers in the columns following 'X', and they are in the order of X1, Y1, X2, and Y2. e.g.,

```
X | 5 | 5 | 23 | 8 || PSLV | mb | 2 | 0
X | THETA | K | 3 | 0 || CXW | m/s | 10 | 0
```

12.4 Default Option Settings File: g_defaults.nml

This is a namelist file and it is optional. If this file exists in the current working directory when the Graph program starts executing, the file's contents replace the previously set defaults in the Fortran code. Since this is a namelist structured file, lines may be removed. Comments after ';' are not allowed on most platforms, but are shown here for easy reference only.

```
&JOEDEF ; defaults for graph
; ISTART=1, ; sub-domain plot beginning I location
; JSTART=1, ; sub-domain plot beginning J location
; IEND=37, ; sub-domain plot ending I location
; JEND=49, ; sub-domain plot ending J location
LW1=2000, ; line width, 1000 is thinnest
LW2=2000, ; line width for overlay plot
DASH1=-682, ; dash pattern, standard NCAR GKS method
DASH2=-682, ; 4092, 3640, 2730, -682
COLOR1=12, COLOR4=12,
COLOR2=9, COLOR5=9,
COLOR3=8, COLOR6=8,
HDRINFO=F, ; true=print header and stop
LOGP=0, ; cross section: 0=linear in p; 1=linear in ln p
XPTOP=200., ; top of cross section plots (mb)
LABLINE=1, ; 0: no contour line labels
LABMSG=0, ; 1: no message below conrec plot
NOZERO=0, ; 0: allow zero line; 1:no min/max zero line;
; 2: no zero whatsoever
IHIRES=0, ; 1: use high resolution US county line/China coastline
&END ;
```

Description of variables in the namelist:

<i>ISTART</i>	integer	for a subdomain plot, this is the I-direction starting point
<i>JSTART</i>	integer	for a subdomain plot, this is the J-direction starting point
<i>IEND</i>	integer	for a subdomain plot, this is the I-direction ending point
<i>JEND</i>	integer	for a subdomain plot, this is the J-direction ending point
<i>LW1</i>	integer	line width for the first plot; 1000 is the thinnest
<i>LW2</i>	integer	line width for the overlay plot
<i>DASH1</i>	integer	dash pattern for the first plot; standard NCAR GKS method. A '-' before a number means contour of positive values is solid, negative values is dashed 682: shorter-dashed line 2730: short-dashed line 3640: medium-dashed line 4092: long-dashed line
<i>DASH2</i>	integer	dash pattern for the overlay plot
<i>COLOR1</i>	integer	color index for the first contour plot, labeled lines
<i>COLOR2</i>	integer	color index for the overlay plot, labeled lines
<i>COLOR3</i>	integer	color index for a dot-point plot, labeled lines
<i>COLOR4</i>	integer	color index for the first contour plot, unlabeled lines

<i>COLOR5</i>	integer	color index for the overlay plot, unlabeled lines
<i>COLOR6</i>	integer	color index for a dot-point plot, unlabeled lines
<i>HDRINFO</i>	logical	T: will only print record header
<i>LOGP</i>	integer	for cross section plots: whether the vertical coordinate is plotted in linear p (LOGP=0), or log p (LOGP=1)
<i>XPTOP</i>	real	top of a cross section plot (in mb)
<i>LABLINE</i>	integer	=0: no contour line labels
<i>LABMESG</i>	integer	=1: no message below conrec plot
<i>NOZERO</i>	integer	=1: no min/max zero line; =2: no zero line whatsoever
<i>IHIRES</i>	integer	=1: use high resolution US county/Asia coastline

To use higher resolution US county lines or Asia coastline, set *IHIRES*=1, and name the outline file to be *hipone.ascii*. These files may be downloaded from *ftp://ftp.ucar.edu/mesouser/Data/GRAPH* directory.

12.5 Map Options File: *g_map.tbl*

This table is used to modify map background specifics for a Graph plot.

MAP DETAILS													
LL	DASH	INT	LB	LSZ	LQL	P	TTL	TSZ	TQL	OUT	DOT	LW	SP
A	PB	D	M	12	00	Y	Y	8	00	PS	N	D	
MAP COLORS													
LL	LINES	LABELS	TITLE	STATES	COUNTRIES	CONTINENTS	PERIMETER						
1		1	1	1	1	1	1						

Description of variables in *g_plots.tbl*: (Text is provided by Dr. Mark Stoelinga of University of Washington.)

<i>LL</i>	lat/lon lines over land only (L), water only (W), none (N), or both land and water (D, A, or E)
<i>DASH</i>	lat/lon lines are dashed large (L), medium (M), small (SM), tiny (T), solid (SO), publ. style (P), or default (D) [LL.ne.N]
<i>INT</i>	lat/lon grid interval in degrees, or D for default [LL.ne.N]
<i>LB</i>	M for only MAPDRV labels (lat/lon on perimeter), N for none, or D or A for both
<i>LSZ</i>	lat/lon label size, 1 to 25 [LB.ne.N]

<i>LQL</i>	label quality: [LB.ne.N] 00 - Complex characters / High quality 01 - Complex characters / Medium quality 02 - Complex characters / Low quality 10 - Duplex characters / High quality 11 - Duplex characters / Medium quality 12 - Duplex characters / Low quality D - Default = 11
<i>P</i>	draw just a line perimeter (N) or a line perimeter with ticks (Y) [DASH.ne.P.or.LL.eq.N]
<i>TTL</i>	title flag: read the next two title parameters (Y) or skip to outline parameter (N)
<i>TSZ and TQL</i>	the same as LSZ and LQL except they refer to the title [both TTL.eq.Y]
<i>OUT</i>	determines which geo-political outlines will be drawn: NO - no outlines CO - continental outlines only US - U.S. State outlines only PS - Continental + International + State outlines PO - Continental + International outlines
<i>DOT</i>	determines whether geo-political boundaries will be dotted (Y) or solid (N) [OUT.ne.NO]
<i>LW</i>	gives the line width, in multiples of default (which is 1000 “units”). D gives default line width. [OUT.ne.NO.and.DOT.eq.N] (LW=2 would double the line width for geographic boundaries)
<i>SP</i>	gives dot spacing. Default (D) is 12 [OUT.ne.NO.and.DOT.eq.Y]

With each parameter is given a conditional statement. If that conditional statement is not met, then that particular box should be made blank. The most common error that occurs when the routine attempts to read this table is “Too many entries on line”, which simply means that the routine expected a box to be blank, but it wasn’t.

One can also do color-filled maps. To do so, add the following in the `g_map.tbl`:

```

MAP FILL
WATER |          SIX COLOR INDICIES WITH WHICH TO COLOR IN THE MAP
-----
1      |          2      | 2      | 2      | 2      | 2      | 2
-----

```

In this example, the water will be colored white, and land light grey according to the color label described below.

12.6 Plot Color Options File: *g_color.tbl*

This table is used to define the color codes referred in the Graph program.

COLOR TABLE				
COLOR	RED	GREEN	BLUE	NUMBER
WHITE	1.00	1.00	1.00	1
LIGHT GRAY	0.66	0.66	0.66	2
DARK GRAY	0.40	0.40	0.40	3
BLACK	0.00	0.00	0.00	4
SKY BLUE	0.20	0.56	0.80	5
BLUE	0.00	0.00	1.00	6
LIGHT YELLOW	0.80	0.80	0.00	7
MAGENTA	1.00	0.00	1.00	8
YELLOW	1.00	1.00	0.00	9
GREEN	0.00	1.00	0.00	10
FOREST GREEN	0.14	0.25	0.14	11
CYAN	0.00	1.00	1.00	12
TAN	0.40	0.30	0.20	13
BROWN	0.25	0.20	0.15	14
ORANGE	1.00	0.50	0.00	15
RED	1.00	0.00	0.00	16
MID-BLUE	0.00	0.50	1.00	17
DULL MID-BLUE	0.00	0.15	0.30	18
BRIGHT FOREST GREEN	0.20	0.40	0.20	19
DULL ORANGE	0.60	0.30	0.00	20

To make a color contour plot, change the background color from black to white using the following *g_color.tbl*:

COLOR TABLE				
COLOR	RED	GREEN	BLUE	NUMBER
WHITE	1.00	1.00	1.00	0
BLACK	0.00	0.00	0.00	1
LIGHT GRAY	0.66	0.66	0.66	2
DARK GRAY	0.40	0.40	0.40	3
BLACK	0.00	0.00	0.00	4

....

and change color used for maps in the MAP COLORS section of the *g_map.tbl* from 1 to a color code other than white for borders, tick marks, and map background.

12.7 How to Run GRAPH

Obtaining Graph tar file

To run GRAPH interactively, the first step is to obtain the GRAPH tar file. The GRAPH tar file, **GRAPH.TAR.gz**, can be obtained from `~mesouser/MM5V3` (or `/fs/othrorgs/home0/mesouser/MM5V3`) from NCAR's IBM, `/MESOUSER/MM5V3/GRAPH.TAR.gz` on MSS, or from the anonymous site (`ftp://ftp.ucar.edu:mesouser/MM5V3`). This tar file contains the GRAPH source code, makefiles, as well as the table files required to produce plots.

To get the tar file from the anonymous ftp site:

- 1) `ftp ftp.ucar.edu`
- 2) login as anonymous
- 3) use your full email address as the password
- 4) `cd mesouser/MM5V3`
- 5) set the transfer to binary (or image), usually this is just "bin"
- 6) `get GRAPH.TAR.gz`
- 7) `quit`

Or to get the tar file on NCAR's IBM:

```
cd /ptmp/$USER
msread GRAPH.TAR.gz /MESOUSER/MM5V3/GRAPH.TAR.gz
```

or

```
cp ~mesouser/MM5V3/GRAPH.TAR.gz .
```

Compiling Graph code

Once you have the *GRAPH.TAR.gz* on the IBM's working directory or on the local workstation, the building process is to gunzip the file, untar the file and make the executable.

- 1) `gunzip GRAPH.TAR.gz`
- 2) `tar -xvf GRAPH.TAR`

After untarring the file, you should find the GRAPH directory and the following in GRAPH directory among others:

```
Makefile
g_color.tbl
g_defaults.nml
g_map.tbl
g_plots.tbl
graph.csh
```

- 3) If your dataset dimensions are greater than 200x200x40, you need to edit two files in `src/` directory: *scratch.incl* and *data.incl*
- 4) type "**make**", and this will create a graph executable called *graph.exe*.
(if working on NCAR's IBM, a user can simply copy the *graph-run-ibm.tar.gz* file from

~mesouser/MM5V3/IBM, unzip and untar it. An executable is inside)

5) edit the *g_plots.tbl* and *g_defaults.nml* (if needed) files.

6) if a user is working on NCAR's IBM, he/she needs to retrieve data from MSS by typing the following:

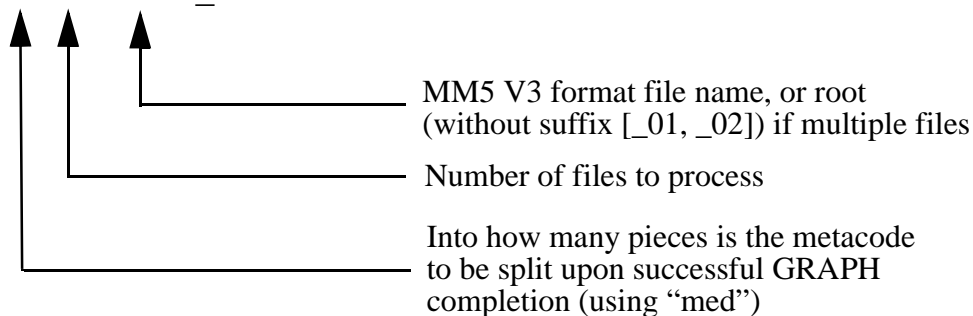
```
msread MMOUT_DOMAIN1[_01 through _99] MSSfilename &
```

The '&' puts the *msread* command in the background.

Running Graph

Program Graph can only process output from one domain at a time. To run Graph, type "graph.csh 1 1 MMOUT_DOMAIN1", or

```
graph.csh 1 3 MMOUT_DOMAIN1
```



The graph.csh tries to figure out what options you have placed on the command line. For example,

a) to run graph with one data file:

```
graph.csh 1 1 MMOUT_DOMAIN1
```

b) to run graph with 3 files named MMOUT_DOMAIN1, MMOUT_DOMAIN1_01, MMOUT_DOMAIN1_02:

```
graph.csh 1 3 MMOUT_DOMAIN1
```

c) to run graph with 3 files named MMOUT_DOMAIN1_00, MMOUT_DOMAIN1_01, MMOUT_DOMAIN1_02:

```
graph.csh 1 3 MMOUT_DOMAIN1
```

d) to run graph with 3 files named MMOUT_DOMAIN1_00, MMOUT_DOMAIN1_01, MMOUT_DOMAIN1_02:

```
graph.csh 1 3 MMOUT_DOMAIN1*
```

Viewing Graphic Output

The plot files generated by Graph are metacode files called '**gmeta**' (and `gmeta.split1`, `gmeta.split2`, etc. if you choose to split the file), which can be viewed by NCAR Graphics utility **idt**, and/or transformed to postscript files using **ctrans** (also an NCAR Graphics utility). For example, to transfer a `gmeta` file to postscript file,

```
ctrans -d ps.mono gmeta > gmeta.ps (for black-white plot),  
or  
ctrans -d ps.color gmeta > gmeta.ps (for color plot)
```

Or to view the output interactively using an interface,

```
idt gmeta
```

12.8 Available 2-D Horizontal Fields

Table 8.1 List of 2-D horizontal fields available for plotting.

Field ID	Description	Default Units	Optional Units	Optional Units	Optional Units
CORIOLIS	Coriolis parameter	1/s			
ICLW	integrated cloud water	cm	mm	in	
IRNW	integrated rain water	cm	mm	in	
LATITDOT	latitude	degrees			
LI	lifted index	K	C		
LNDUS	land use categories	(no units)			
LHFLUX	surface latent heat flux	W/m ²			
LONGIDOT	longitude	degrees			
LWDOWN	longwave downward radiation	W/m ²			
MAPFACDT	map scale factor	(no units)			
PBL HGT ⁷	PBL height	m			
PRECIPT	total accumulated precipitation	mm	cm	in	
PRECIPC	convective accumulated precip	mm	cm	in	
PRECIPN	stable accumulated precip	mm	cm	in	
PRECIPTT	total precip during time interval	mm	cm	in	
PRECIPTN	stable precip during time interval	mm	cm	in	
PRECIPTC	convective precip during interval	mm	cm	in	
PRH2O	precipitable water	cm	mm	in	
PSLV	sea level pressure	mb	hPa	Pa	inHg
PSFC	surface pressure	mb	hPa	Pa	inHg
PTEND	pressure change	mb	hPa	Pa	
RAINT	total accumulated precipitation	mm	cm	in	
RAINC	convective accumulated precip	mm	cm	in	
RAINN	stable accumulated precip	mm	cm	in	

Field ID	Description	Default Units	Optional Units	Optional Units	Optional Units
RTENDT	total precip during time interval	mm	cm	in	
RTENDC	convective precip during interval	mm	cm	in	
RTENDN	stable precip during time interval	mm	cm	in	
REGIME	PBL regimes (values 1-4)	catagory			
SHFLUX	surface sensible heat flux	W/m ²			
SOIL T 1-6	soil temp in 1/2/4/8/16 cm layer	K			
SWDOWN	shortwave downward radiation	W/m ²			
TER	terrain elevation	m	ft		
TGD	ground temperature	K	C		
THK	thickness	m			
TSEASFC	sea surface temperature	K	C		
UST	frictional velocity	m/s			
	if IPOLAR = 1				
SEAICEFR	Sea ice fraction	(no units)			
	if ISOIL = 2				
SOIL T 1-4	soil temp in 10/40/100/200 cm layer	K			
SOIL M 1-4	soil moisture in above layers	m ³ /m ³			
SOIL W 1-4	soil water in above layers	m ³ /m ³			
SFCRNOFF	surface runoff	mm			
UGDRNOFF	underground runoff	mm			
CANOPYM	canopy moisture	m			
SNODPTH	water-equivalent of snow depth	mm			
SNOWH	physical snow depth	m			
SEAICE	sea ice flag	(no units)			
ALB	albedo	fraction			
ALBSNOMX	maximum snow albedo	%			

Field ID	Description	Default Units	Optional Units	Optional Units	Optional Units
	if FRAD >= 2				
SWOUT	top outgoing shortwave radiation	W/m ²			
LWOUT	top outgoing longwave radiation	W/m ²			
	if IBLTYP = 5				
T2M/T2	2 m temperature	K	C	F	
TD2M	2 m dewpoint temperature	K	C	F	
TDD2M	2 m dewpoint depression	K	C	F	
Q2M/Q2	2 m mixing ratio	kg/kg	g/kg		
U10	10 m model u wind component	m/sec	knots		
V10	10 m model v wind component	m/sec	knots		
WIND10M	10 m wind speed	m/sec	knots		
BARB10M	10 m wind barb	m/sec	knots		
VECT10M	10 m wind vector	m/sec	knots		
VESL10M	10 m streamline				
	if ISOIL=3 and IBLTYP=7				
M-O LENG	Monin-Obukov Length	m			
NET RAD	surface net radiation	W/m ²			
GRNFLX	ground heat flux	W/m ²			
ALBEDO	surface albedo	fraction			
VEGFRG	vegetation coverage	fraction			
LAI	leaf area index	area/area			
RA	aerodynamic resistance	s/m			
RS	surface resistance	s/m			
ZNT	roughness length	m			
ISLTYP	soil texture type	category			

12.9 Available Cross-Section Only Fields

Table 8.2 List of cross-section-only fields available for plotting.

Field ID	Description	Default Units	Optional Units	Optional Units	Optional Units
AM	absolute momentum	m/s			
AXW	wind speed tangential to the cross-section	m/s			
CUV	horizontal wind barb in plane	m/s			
CXW	circulation vectors in cross-section plane	m/s			
XXW	wind speed normal to the cross-section	m/s			

12.10 Available 3-D Fields (as 2-D Horizontal or Cross-Section)

Table 8.3 List of 3-D fields available for plotting.

Field ID	Description	Default Units	Optional Units	Optional Units	Optional Units
AGL	above ground level	m	cm	Dm	
BARB	wind barbs	m/s	kt	cm/s	
CLB	cloud boundary	g/kg	kg/kg	mg/kg	
CLW	cloud water	g/kg	kg/kg	mg/kg	
DIV	divergence of horizontal wind	$10^{**5}/s$	1/s		
GRA	graupel	g/kg	kg/kg	mg/kg	
H	geopotential height	m			
HEIGHT	geopotential height	m			
ICE	cloud ice	g/kg	kg/kg	mg/kg	
MDIV	moisture divergence	$10^{**7}/s$	1/s		

Field ID	Description	Default Units	Optional Units	Optional Units	Optional Units
MSE	moist static energy	J/kg			
MSS	saturated moist static energy	J/kg			
NCI	number concentration of ice	number/m ³			
OMG	vertical motion (pressure level data only)	ub/s	mb/s	hPa/s	
P	pressure	mb	Pa	hPa	
PP	pressure perturbation	mb	Pa	hPa	
PV	potential vorticity	PVU			
QDIV	q-vector divergence (<i>p data only</i>)				
QV	mixing ratio	g/kg	kg/kg		
QVEC	q-vectors (<i>p data only</i>)				
RDTEND	atmospheric radiative tendency	K/day	K/h		
RH	relative humidity	%			
RNW	rain water	g/kg	kg/kg	mg/kg	
SLW	super-cooled liquid water	g/kg	kg/kg	mg/kg	
SNOW	snow	g/kg	kg/kg	mg/kg	
T	temperature	K	C	F	
TD	dew point temperature	K	C	F	
TDD	dew point depression	K	C	F	
THETA	potential temperature	K	C	F	
THETAE	equivalent potential temperature	K	C	F	
TKE	turbulent kinetic energy	J/kg			
U	u-component of wind	m/s	kt	cm/s	
V	v-component of wind	m/s	kt	cm/s	
VAB	absolute vorticity	10**5/s	1/s		
VECT	horizontal wind vectors	m/s	kt	cm/s	

Field ID	Description	Default Units	Optional Units	Optional Units	Optional Units
VESL	horizontal wind streamlines	m/s			
VOR	relative vorticity	$10^{**5}/s$	1/s		
W	w-component of wind	m/s	kt		
WIND	wind speed	m/s	kt	cm/s	

12.11 Some Hints for Running GRAPH

- make sure the following line is included in your `.cshrc` file on NCAR's IBM or your local computer:
`setenv NCARG_ROOT /usr/local or setenv NCARG_ROOT /usr/local/ncarg`
- NCAR Graphics has recently been upgraded to include better country/political boundaries. This is especially true over Europe. GRAPH's default is to use NCAR Graphics version 4.1, but GRAPH can also run with older/newer versions.
 - If you have an older version, remove the “-DNCARG41” directive from the Makefile.
 - If you have NCAR Graphics version 4.2, change the “-DNCARG41” directive in the Makefile to “-DNCARG42”.
- The GRAPH program uses the information in the record header to define the size and location of the data. This limits the "wrong" data that the user can provide to be related to the requested fields and levels to be plotted.
- GRAPH prints out information to allow you to track the program's status. It should inform the user that it is processing each of the requested time periods, and for each of the requested variables and levels.
- If the GRAPH program is not processing the time that you have requested, and it should be based upon the intervals that you have set, ask GRAPH to plot every time by setting time increment to be 0.
- GRAPH only vertically interpolates data that is on a σ coordinate.
- Contour intervals for precipitation are fixed for the RAIN and RTEND fields and user modifiable for the PRECIP fields.
- Do not request a subdomain and also process soundings. The GRAPH program may not place the sounding at the correct location for the large domain.
- Errors related to the NAMELIST or other temporary files are common when porting GRAPH to a different architecture. Use the NAMELIST format in the architecture's FORTRAN manual. Make sure and remove all temporary files and links prior to each initiation of the GRAPH C-shell.
- When GRAPH is compiled on a different architecture, the length of the records for the direct access files must be specified in bytes (4 or 8 per word) or words. This information is found in the include file `word_length.incl`.
- If you get the following, and Graph stops,
`NEED MORE DATA SPACE`
it means that you need to increase the dimensions in the `data.incl` and `scratch.incl` files.
- If you get error message related to ‘Direct Access Files’, it is usually an indication that the dimensions specified in `data.incl` and `scratch.incl` files are not large enough, or the running memory is not large enough, or the word length is not correct for your particular computer architecture.

12.12 Sample Graph Plot File

For some horizontal plots, please refer to Chapter 15, page 15-9 and 15-10.

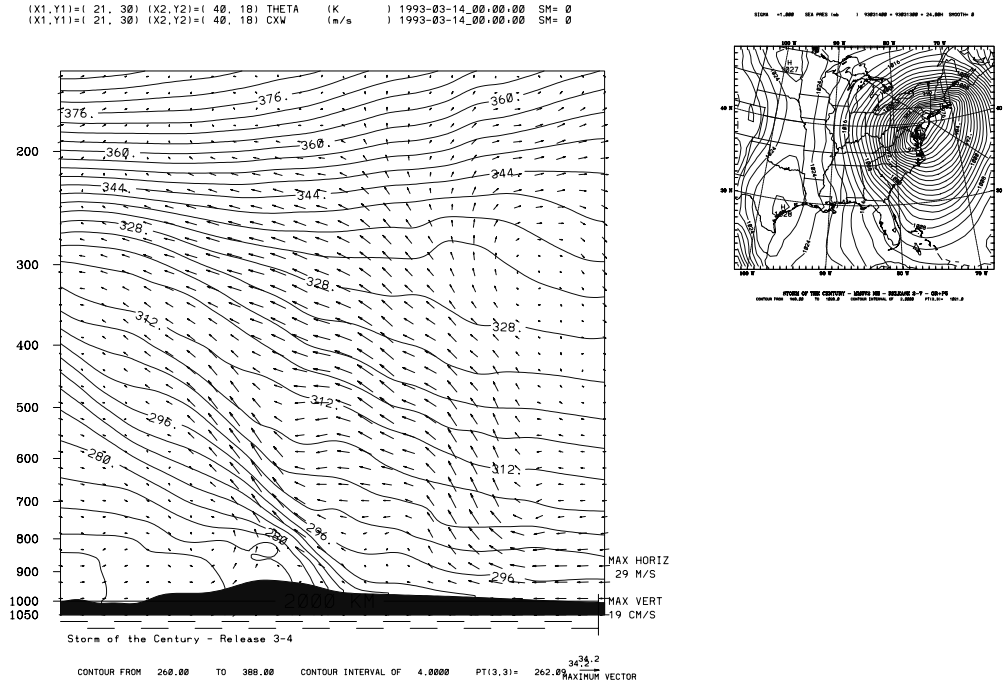


Figure 12.1 A NW-SE cross section of potential temperature (unit K and contour interval 4 K) and 2-D in-plane cir-

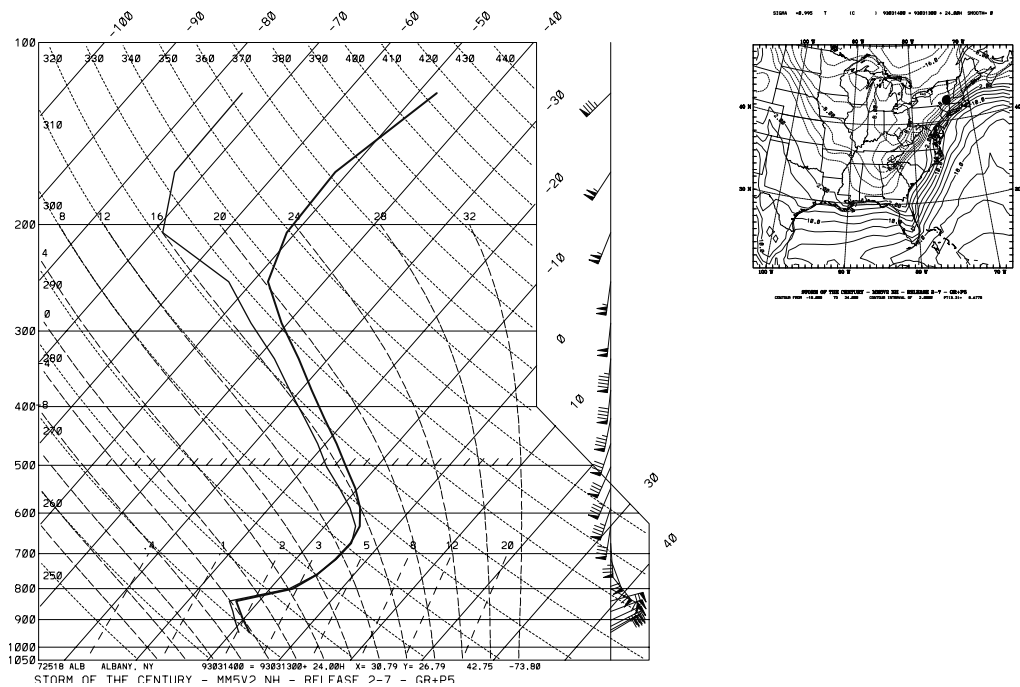


Figure 12.2 A skew-T plot from a 24-h simulation at Albany, New York.

12.13 Graph tar file

The graph.tar file contains the following files and directories:

CHANGES	Description of changes to the Graph program
Diff/	Will contain difference files between consecutive releases
Makefile	Makefile to create Graph executable
README	General information about the Graph directory and how to run Graph
Templates/	Job deck directory: batch deck for Cray only
g_color.tbl	Color table for Graph job
g_defaults.nml	NAMELIST file for Graph job
g_map.tbl	Map table for Graph job
g_plots.tbl	Table for selecting plot variables
graph.csh	C-shell script to run Graph interactively
src/	Graph source code and low-level Makefile

12.14 Script file to run Graph job

```
#!/bin/csh -f

#       this is INTERACTIVE or BATCH

if ( $?ENVIRONMENT ) then
    echo "environment variable defined as $ENVIRONMENT"
else
    setenv ENVIRONMENT INTERACTIVE
    echo "environment variable defined as $ENVIRONMENT"
endif

#       initializations, no user modification required

set FILE_EXT = ( 00 01 02 03 04 05 06 07 08 09 \
                 10 11 12 13 14 15 16 17 18 19 \
                 20 21 22 23 24 25 26 27 28 29 \
                 30 31 32 33 34 35 36 37 38 39 \
                 40 41 42 43 44 45 46 47 48 49 \
                 50 51 52 53 54 55 56 57 58 59 \
                 60 61 62 63 64 65 66 67 68 69 )

#       is it color

# set Color = BW
set Color = CO

#   If this is an HP-UX machine, the generic name for fortran files
#   starts with "ftn" not "fort".

model >& /dev/null
set OK = $status
if ( $OK == 0 ) then
    set ForUnit = ftn
#       echo "This is an HP-UX"
else
    set ForUnit = fort.
#       echo "This is not an HP-UX"
endif

if ( -e numsplit.tbl ) rm numsplit.tbl
if ( -e med.input ) rm med.input
```

```

if ( ( `uname` == AIX ) || ( `uname` == SunOS ) || ( `uname` == HP-UX ) ) then
  if ( -e ${ForUnit}20 ) rm ${ForUnit}2*
  if ( -e ${ForUnit}30 ) rm ${ForUnit}3*
  if ( -e ${ForUnit}40 ) rm ${ForUnit}4*
  if ( -e ${ForUnit}50 ) rm ${ForUnit}5*
  if ( -e ${ForUnit}60 ) rm ${ForUnit}6*
  if ( -e ${ForUnit}70 ) rm ${ForUnit}7*
  if ( -e ${ForUnit}80 ) rm ${ForUnit}8*
else
  if ( ( -e ${ForUnit}20 ) || ( -l ${ForUnit}20 ) ) rm ${ForUnit}2*
  if ( ( -e ${ForUnit}30 ) || ( -l ${ForUnit}30 ) ) rm ${ForUnit}3*
  if ( ( -e ${ForUnit}40 ) || ( -l ${ForUnit}40 ) ) rm ${ForUnit}4*
  if ( ( -e ${ForUnit}50 ) || ( -l ${ForUnit}50 ) ) rm ${ForUnit}5*
  if ( ( -e ${ForUnit}60 ) || ( -l ${ForUnit}60 ) ) rm ${ForUnit}6*
  if ( ( -e ${ForUnit}70 ) || ( -l ${ForUnit}70 ) ) rm ${ForUnit}7*
  if ( ( -e ${ForUnit}80 ) || ( -l ${ForUnit}80 ) ) rm ${ForUnit}8*
endif

#       simple error check on call

if ( ( $#argv == 0 ) && ( $ENVIRONMENT == INTERACTIVE ) ) then
  echo -n "into how many pieces is the metafile to be split (1) "
  set NumSplit = "$<"
  echo -n "how many input files are there                      (1) "
  set TotFiles = "$<"
  echo -n "what is the name of the file                        (MMOUT_DOMAIN1) "
  set FileName = "$<"
else if ( $#argv < 3 ) then
  echo "graph.deck: error in call"
  echo "usage: graph.deck ns nf filename [filename2 filename3 ...]"
  echo "       where ns is the number of files into which metacode is split"
  echo "       where nf is the number of input files"
  echo "       where filename is either the root name, or several names"
  exit (1)
else if ( $#argv == 3 ) then
  set NumSplit = $1
  set TotFiles = $2
  set FileName = $3
else if ( $#argv > 3 ) then
  set NumSplit = $1
  set TotFiles = $2
endif

#       consistency checks on input

if ( ( $NumSplit < 1 ) || ( $NumSplit > 99 ) ) then
  set NumSplit = 1
endif
cat >! numsplit.tbl << EOF
$NumSplit
EOF
if ( ( $TotFiles < 1 ) || ( $TotFiles > 69 ) ) then
  set TotFiles = 1
endif
if ( $#argv == 0 ) then
  if ( ( ! -e $FileName ) && ( ! -e ${FileName}_00 ) ) then
    echo "file $FileName does not exist"
    exit (2)
  endif
else if ( $#argv >= 3 ) then
  if ( ( ! -e $argv[3] ) && ( ! -e $argv[3]_00 ) ) then
    echo "file $argv[3] does not exist"
    exit (3)
  endif
endif
endif

```

```
if ( ! -e graph.exe ) ln -s src/graph.exe graph.exe
chmod +x graph.exe

#      make sure we have all the required tables

if ( -e g_map.tbl ) then
  echo "using local copy of g_map.tbl"
else
  echo "need a copy of g_map.tbl"
  exit (4)
endif

if ( -e g_color.tbl ) then
  echo "using local copy of g_color.tbl"
else
  echo "need a copy of g_color.tbl"
  exit (5)
endif

if ( -e g_plots.tbl ) then
  echo "using local copy of g_plots.tbl"
else
  echo "need a copy of g_plots.tbl"
  exit (6)
endif

if ( -e g_defaults.nml ) then
  echo "using local copy of g_defaults.nml"
else
  echo "need a copy of g_defaults.nml"
  exit (7)
endif

#      run graph program

if ( ( `uname` == AIX ) || ( `uname` == SunOS ) || ( `uname` == HP-UX ) ) then
  if ( -e ${ForUnit}18 ) rm ${ForUnit}18
else
  if ( ( -e ${ForUnit}18 ) || ( -l ${ForUnit}18 ) ) rm ${ForUnit}18
endif
ln -s      g_plots.tbl                ${ForUnit}18
if ( -e .assign ) rm .assign
if ((( $TotFiles == 1 ) && ( $#argv == 3 )) || \
    (( $TotFiles > 1 ) && ( $#argv > 3 ))) then
  shift
  shift
  set NUMFIL = 1
  while ( $#argv )
    @ UNIT = 19 + $NUMFIL
    ln -s      $argv[1]                ${ForUnit}$UNIT
#    assign -a $argv[1] -Ff77 -Nieee  ${ForUnit}$UNIT
    shift
    @ NUMFIL ++
  end
else if ( (( $TotFiles > 1 ) && ( $#argv == 3 )) || ( $#argv == 0 ) ) then
  if ( ( `uname` == AIX ) || ( `uname` == SunOS ) || ( `uname` == HP-UX ) )
then
    set AIX = true
  else
    set AIX = false
  endif
  set NUMFIL = 1
  while ( $NUMFIL <= $TotFiles )
    @ UNIT = 19 + $NUMFIL
    if ( $AIX == true ) then
      if ( ( $NUMFIL == 1 ) && ( -e $FileName ) ) then
```

```

        ln -s      $FileName                                ${ForUnit}$UNIT
#         assign -a $FileName                                -Ff77 -Nieee ${ForUnit}$UNIT
        @ NUMFIL ++
        else if ( ( $NUMFIL == 1 ) && ( -e ${FileName}_00 ) ) then
                ln -s      ${FileName}_$FILE_EXT[${NUMFIL}]
${ForUnit}$UNIT
#         assign -a ${FileName}_$FILE_EXT[${NUMFIL}] -Ff77 -Nieee
${ForUnit}$UNIT
        @ NUMFIL ++
        else
                ln -s      ${FileName}_$FILE_EXT[${NUMFIL}]
${ForUnit}$UNIT
#         assign -a ${FileName}_$FILE_EXT[${NUMFIL}] -Ff77 -Nieee
${ForUnit}$UNIT
        @ NUMFIL ++
        endif
    else if ( $AIX == false ) then
        if ( ( $NUMFIL == 1 ) && ( ( -e $FileName ) || ( -l $FileName ) ) )
then
        ln -s      $FileName                                ${ForUnit}$UNIT
#         assign -a $FileName                                -Ff77 -Nieee ${ForUnit}$UNIT
        @ NUMFIL ++
        else if ( ( $NUMFIL == 1 ) && ( ( -e ${FileName}_00 ) || ( -l ${File-
Name}_00 ) ) ) then
                ln -s      ${FileName}_$FILE_EXT[${NUMFIL}]
${ForUnit}$UNIT
#         assign -a ${FileName}_$FILE_EXT[${NUMFIL}] -Ff77 -Nieee
${ForUnit}$UNIT
        @ NUMFIL ++
        else
                ln -s      ${FileName}_$FILE_EXT[${NUMFIL}]
${ForUnit}$UNIT
#         assign -a ${FileName}_$FILE_EXT[${NUMFIL}] -Ff77 -Nieee
${ForUnit}$UNIT
        @ NUMFIL ++
        endif
    endif
end
endif
endif

#         run graph program

graph.exe

#         split metacode apart

if      ( $NumSplit != 1 ) then
    cat med.input
    med -f med.input
else if ( $NumSplit == 1 ) then
#     cat med.input
#     cp gmeta gmeta.split1
endif

rm med.input numsplit.tbl
rm tmp.*
echo "GRAPH run complete"

```

12.15 An Alternative Plotting Package: RIP

RIP (which stands for Read, Interpolate and Plot), is a graphical program developed by Dr. Mark Stoelinga of University of Washington. This program is a popular alternative graphics program to GRAPH, especially when it comes to color plots. This program is also based on NCAR Graphics.

A descriptions of this package is available in Appendix G.

