

McIDAS-X Workshop

September 2023 (software version 2022.1)

In existence since 1973, McIDAS (Man computer Interactive Data Access System) is a suite of sophisticated software packages that perform a wide variety of functions with satellite imagery, observational reports, numerical forecasts, and other geophysical data. Those functions include displaying, analyzing, interpreting, acquiring and managing the data.

McIDAS-X is supported for the current GOES GVAR and upcoming GOES-R satellite series (currently estimated as being in service until 2036), with no end date in sight.

In this McIDAS-X tutorial, some exercises will be completed using different methods of data access: local data files and real-time access to default remote servers. If you have access to your own real-time ADDE servers, you may also use those, but be aware that different server configurations may make the explanations in this document not quite applicable to all data that you may load. A source for free data served via ADDE is Unidata's ADDE.SSEC.WISC.EDU server.

This tutorial assumes that you have McIDAS-X installed on your machine, and that you know how to start McIDAS-X.

Getting Started - Starting and Exiting McIDAS

To start or stop a McIDAS session, follow the steps below:

1. Start McIDAS by entering the following in lowercase from the Unix prompt.

Type: **mcidas** or **mcidas -config**

If this is the first time that you've run McIDAS, the **mcidas** command will display the McIDAS Configuration GUI.

2. Change the Configuration GUI so that your options match those listed below:
 - o 6 image frames - each 480 lines by 640 elements

Click: the **Image Window** tab

Click: **10 @ 480 x 640**

Click: **Modify**

Change the number of frames to 6

- o 2 MB memory allocated for creating additional image frames

Click: the **Miscellaneous** tab

Change the amount of memory allocated to 2 MB

3. On the **Introduction** tab, click the **Show this window each time McIDAS is started** button to uncheck the box. The next time you start McIDAS, the McIDAS Configuration GUI will not show up. In the future, you can also display this GUI by typing **mcidas -config**. You can also change the display by editing the *flags* in the **\$HOME/.mcidasrc** file by using a unix editor. For more information on the start-up flags, read the documentation within the **.mcidasrc** configuration file.

4. Save your changes and exit the configuration GUI.

Click: **Save Settings**

A text warning will pop up telling you that a new version of **.mcidasrc** will be created with the current settings.

Click: **OK**

Click: **Exit**

5. Start McIDAS again. This time you will notice that mcidas started automatically without the configuration GUI.

Type: **mcidas**

When the McIDAS GUI starts, a text warning will pop up telling you that the GUI is going to analyze your Server List and that it would take a while. Click "OK". A LOGON screen will then appear. You can just click "Cancel". We will update the GUI tables in the Graphical User Interface lesson.

6. Exit and restart McIDAS one last time with the configuration GUI to turn off the option to start the main GUI each time McIDAS is started. We will get back to the GUI in the Graphical User Interface lesson.

Additionally, recheck the **Show this window each time McIDAS is started** box. The configuration GUI is a useful tool that will assist you in starting each McIDAS session as you go through the Learning Guide.

Type: **EXIT**

Type: **mcidas -config**

Under the **Introduction** tab, click the **Show this window each time McIDAS is started** box, and under the **Miscellaneous** tab, click the **Start GUI upon McIDAS startup** box to deselect this option. Save your new settings and start McIDAS.

Getting Started - Using Online Helps

Online helps list the syntax of each command including the parameters, keywords, and remarks. To access the online help, type **HELP** in the McIDAS text and command window followed by the command for which you'd like more information. You can find additional command information in the McIDAS User's Guide.

1. Find the help for the **ZLM** command.

Type: **HELP ZLM**

The help text appears in the active text frame as shown below:

2. Get a listing of all the commands and a brief description of their functions.

Type: **HELP**

3. If you are entering a command and have forgotten the syntax, you can get a brief listing of the syntax by pressing **Alt ?**. Show the syntax of the **IMGDISP** command.

Type: **IMGDISP** (but don't press Enter)

Press: **Alt ?**

The syntax of the **IMGDISP** command is listed in the current text frame.

4. Use the **Esc** key to clear the line.

Press: **Esc**

Getting Started - Raising Windows and Displaying Text Frames

Raising a window brings it to the front of any overlapping windows so you can view the entire window.

Use the Text and Command Window for entering McIDAS commands, displaying text output, and displaying workstation status information. You can reposition the Text and Command Window to a different location on the display, and you can also resize it.

The Text and Command Window displays keyboard input regardless of whether it or the Image Window is active. Click in either window to make it active.

Use the numeric keypad's plus(+) key to make the Image Window active. To make the Text and Command Window active, press a numeric keypad number from 0-9. The number you press determines which text frame is displayed in the window. 0 is the default text window displayed when McIDAS is first ran.

1. To raise the Text and Command Window and display text frame 1,

Press: **1**

2. To raise the Image Window,

Press: **+**

3. Change the Text and Command Window back to text frame 0.

Press: **0**

Getting Started - Stopping Commands

Every command being run has a Process IDentification (PID) number and a Parent Process IDentification number (PPID) associated with it. To stop a command on the workstation, you must find the PID for the command by checking the command status with the question mark (?) command and then using the KILL command. In this exercise, you will enter a command and then stop it.

1. Enter the following command.

Type: **ZLM**

2. Find the PID for the ZLM command by checking the command status.

Press: **?**

3. Find the PID for the ZLM command. Type KILL followed by the appropriate PID number and press Enter. For example, to stop PID 7921:

WARNING: Be sure that you get the correct PID number. Accidentally stopping the wrong process will produce unpredictable results.

Type: **KILL 7921**

Alternate method (use the '/' (slash) command):

Type: **/ 7921**

4. Check the command status again to make sure ZLM is no longer running.

Press: **?**

You should see that ZLM is no longer listed in the list of processes.

5. You can now EXIT McIDAS since we are at the end of this lesson.

Type: **EXIT**

ADDE

This section of the training contains information about the ADDE (Abstract Data Distribution Environment) software in the core McIDAS package. The ADDE allows your workstation to act as a client, efficiently accessing data from multiple McIDAS-X servers.

ADDE distributes data using networked servers and clients. Servers store data and distribute it to the client. Clients request and receive data and run applications that use the data. Clients and servers communicate using the TCP/IP communications protocol.

Each account running McIDAS-X acts as both a client and local server. When a client requests data from the local server, it searches for the data in the directories specified in the user's **MCPATH** environment variable or another directory specified in a REDIRECT entry.

The client can also request data from a remote server. A remote server can be any of the following:

- a different account on the same McIDAS-X workstation configured as a remote server
- another McIDAS-X workstation configured as a remote workstation

The difference between a local server and a remote server on a McIDAS-X workstation is that the data stored in a local server is available only to the McIDAS-X sessions started under that account name. Data in a remote server is accessible to all McIDAS ADDE clients.

In the non-ADDE core commands, all image, grid and point files are referenced by file numbers. If you don't know the file numbers, finding data can be difficult. The ADDE commands use dataset names (in a group/descriptor format) that map to datasets. If the descriptors and group names follow a logical convention, it's easy to locate the data.

The naming scheme for datasets consists of three parts:

- type
- group
- descriptor

Type is the top tier in ADDE's hierarchical naming scheme. Image, grid and point indicate the type of data in the dataset files. McIDAS area files are image data; McIDAS grid files are grid data; McIDAS MD files are point data.

Group is the next tier in the naming scheme. A group name can be used only once under each type.

Descriptors are the bottom tier in the naming scheme. Descriptors are not data files; they are names that point to datasets. Identical descriptors under different groups can point to the same or different datasets.

ADDE - Client and Server Look-up Tables

Each ADDE client command starts a transaction by requesting data from a server. Both the client and server must recognize the command's specified dataset name in their look-up tables. On the client, this table is called the *routing table* because it determines which server to route the data request to. On the server, this table is called the *mapping table* because it maps the group and descriptor to a specific image, grid, point, navigation or text dataset.

The client routing table contains two lists: a list of group names with their associated server IP addresses, and a list of aliases with the dataset names they represent. When you enter an ADDE command, the client routing table is scanned for an entry with the specified group name or alias. If an entry is found, the data request is routed to the server specified in the entry. If the group name or alias is not found in the client routing table, the request is routed to the local server.

The server mapping table maps dataset names to the files that make up the datasets. When the server receives a data request from a client, it reads the mapping table to locate the correct dataset. The server sends the requested data back to the client so it can list or display the data.

ADDE Administrative Commands

Server Side

DSSERVE ADD *group/descriptor format bfile efile [keywords] "description*
Defines a dataset for the server workstation. Keywords may be required to add additional file name and location information. Group and descriptor are used by client commands to access datasets. Information stored in file named RESOLV.SRV.

Client Side

DATALOC ADD *group ip_address*

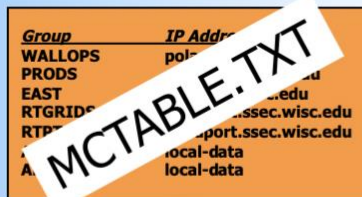
Creates a table in MCTABLE.TXT so client commands can determine which server workstation is to receive the ADDE request.

DSINFO *type group*

Lists descriptors for type and group specified.


AKA *alias group/descriptor*

Creates an alias for a group/descriptor pairing.



Group	IP Address
WALLOPS	pol...
PRODS	...
EAST	...ssec.wisc.edu
RTGRIDS	...ssec.wisc.edu
RTPTSRC	port.ssec.wisc.edu
A...	local-data
	local-data

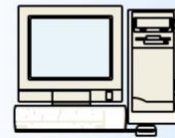
MCTABLE.TXT



Group Names
MYDATA
AZ
CO
AN
AZFIRE
11UM
V...
JOBS
AIRCRAFT
11UM
VIS
WV

RESOLV.SRV

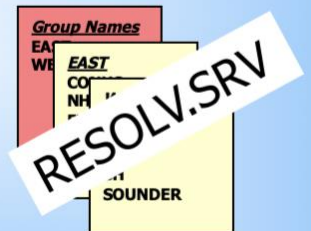
noaaport.ssec.wisc.edu



Group Names
RTPTSRC
RT
RT
RTPTSRC
SFCMOURLY
SY
SH
...

RESOLV.SRV

geo.ssec.wisc.edu



Group Names
EAST
CO
NH
...
SOUNDER

RESOLV.SRV

ADDE - Getting Started

The exercise below introduces you to ADDE concepts such as dataset names, aliases, clients and servers. Use this exercise to practice adding entries to your client routing table and local server's mapping table.

1. Start a McIDAS session.

At the Unix prompt:

Type: **mcidas -config**

Your session should still be set for six frames from the last time you changed the settings in the Configuration GUI. Set it for twenty frames, and save it for use in the workshop

2. List your client routing table's groups and associated server IP addresses.

Type: **DATALOC**

If this is the first time you are using the ADDE, the table is empty and the following listing is displayed.

Group Name	Server IP Address
-----	-----

3. Modify your client routing table so that commands requesting data from a real-time dataset containing free data from Unidata. The group RTGOESR is routed to the Unidata remote server with the IP address ADDE.SSEC.WISC.EDU.

Type: **DATALOC ADD RTGOESR ADDE.SSEC.WISC.EDU**

4. List all datasets in the group RTGOESR.

Type: **DSINFO ALL RTGOESR**

5. List all datasets of type IMAGE in the group RTGOESR.

Type: **DSINFO I RTGOESR**

Note that instead of the full word, 'IMAGE,' only an **I** is used. Each parameter in DSINFO can be abbreviated with its first letter instead of writing it out in its entirety.

The first part of the listing is displayed below. The NumPos value is the number of positions in the dataset. When setting up servers with newer data types, if DIRFILE= is used, most likely the NumPos = 99999.

```
Dataset Names of Type: IMAGE in Group: RTGOESR
```

Name	NumPos	Content
-----	-----	-----
CONUS	99999	GOES-East CONUS all bands
CONUSC01	99999	GOES-East CONUS 0.47 um VIS aerosol-over-land
CONUSC02	99999	GOES-East CONUS 0.64 um VIS clouds fog/insol/winds
CONUSC03	99999	GOES-East CONUS 0.86 um Near IR veg/burn scar/aerosol
CONUSC04	99999	GOES-East CONUS 1.37 um Near IR cirrus cloud
CONUSC05	99999	GOES-East CONUS 1.61 um Near IR cloud phase/snow
CONUSC06	99999	GOES-East CONUS 2.24 um Near IR land/cloud vege/snow
CONUSC07	99999	GOES-East CONUS 3.89 um IR Sfc/cloud/fog/fire/winds

6. List the 11 most recent images in the dataset RTGOESR/CONUSC13..

Type: **IMGLIST RTGOESR/CONUSC13.-10**

The following listing is an example of what is displayed. The Pos value is the absolute position of the image in the dataset.

```
Image file directory listing for:RTGOESR/CONUSC13
```

Pos	Satellite/ sensor	Date	Time	Center Lat Lon	Band(s)
---	-----	-----	-----	-----	-----
576	GOES-16	16 SEP 19259	02:16:14	30 87 13	
575	GOES-16	16 SEP 19259	02:11:14	30 87 13	
574	GOES-16	16 SEP 19259	02:06:14	30 87 13	

7. List the two most recent images (positions 0 and -1) in the dataset RTGOESR/FD (all bands are included in this dataset) with expanded information on the available Band (1) .

Type: **IMGLIST RTGOESR/FD.-1 BAND=1**

The following listing is an example of what is displayed. The Pos value is the absolute position of the image in the dataset. The first image is listed first because it is the most recent image, relative position 0. The next image is the second most recent image, relative position -1... The Res values are the resolution of the images' center pixels.

Image file directory listing for:RTGOESR/FD									
Pos	Satellite/ sensor	Date	Time	Center		Res (km)		Image_Size	
				Lat	Lon	Lat	Lon		
192	GOES-16	16 SEP 19259	02:10:17	0	75				
	Band: 1	0.47 um	VIS aerosol-over-land			1.00	1.00	10848	x10848
191	GOES-16	16 SEP 19259	02:00:17	0	75				
	Band: 1	0.47 um	VIS aerosol-over-land			1.00	1.00	10848	x10848
IMGLIST: done									

8. Create the aliases G16C and G16FD for the dataset names RTGOESR/CONUS and RTGOESR/FD.

Type: **AKA ADD G16C RTGOESR/CONUS**

Type: **AKA ADD G16FD RTGOESR/FD**

9. List the aliases in your client routing table.

Type: **AKA**

The following listing is displayed.

Alias Name	Group/Descriptor
G16C	RTGOESR/CONUS
G16FD	RTGOESR/FD

10. Display the most recent image in the dataset with the alias name G16C. Display the image on frame 1, band 13, centered on Athens, Georgia, and draw a map on it. The AKA listing in step 9 shows that the alias G16C represents dataset RTGOESR/CONUS.

Type: **IMGDISP G16C 1 BAND=13 STATION=KAHN SF=YES;MAP VH 2**

Assign the dataset name MYDATA/IMAGES to all of the areas available on your local workstation.

Type: **DSSERVE ADD MYDATA/IMAGES AREA 1 9999 "ALL AREA FILES**

11. Use the IMGLIST command to find five empty areas on the local server.

Type: **IMGLIST MYDATA/IMAGES.4000 4004**

If data exists in this block of areas, move or delete the areas or use the IMGLIST command to find an empty block of 5 areas.

12. Assign the dataset name MYDATA/TEST-IMAGES to the empty block of areas.

Type: **DSSERVE ADD MYDATA/TEST-IMAGES AREA 4000 4004 "Scratch areas for testing**
A listing similar to the following is displayed.

Group/Descriptor	Type	Format & Range	RT Comment
MYDATA/IMAGES	IMAGE	AREA 1-9999	ALL AREA FILES
MYDATA/TEST-IMAGES	IMAGE	AREA 4000-4004	Scratch areas for testing

13. List the groups in your client routing table.

Type: **DATALOC**

A listing similar to the following is displayed.

Group Name	Server IP Address
RTGOESR	ADDE.SSEC.WISC.EDU
MYDATA	<LOCAL-DATA>

<LOCAL-DATA> indicates that data will be accessed from the local data directory.

14. Create the alias TI for the dataset name MYDATA/TEST-IMAGES.

Type: **AKA ADD TI MYDATA/TEST-IMAGES**

15. List the aliases in your client routing table.

Type: **AKA**

The following listing is displayed.

Alias Name	Group/Descriptor
G16C	RTGOESR/CONUS
G16FD	RTGOESR/FD
TI	MYDATA/TEST-IMAGES

16. List all the images in the dataset with the alias name TI. The AKA listing in step 17 shows that the alias TI represents dataset MYDATA/TEST-IMAGES.

Type: **IMGLIST TI.ALL**

The following output is displayed because the dataset has no images.

```
Image file directory listing for:TI
IMGLIST: No images satisfy the selection criteria
```

17. Copy the most recent band 13 image in the dataset with the alias name G16C to position 5 in the dataset with the alias name TI. Place Athens, Georgia, at its center. The AKA listing in step 16 shows that the aliases G16C and TI represent datasets RTGOESR/CONUS and MYDATA/TEST-IMAGES.

Type: **IMGCOPY G16C TI.5 STATION=KAHN BAND=13**

18. List all the images in the dataset with the alias name TI.

Type: **IMGLIST TI.ALL**

Creating Local Datasets

In addition to the MYDATA/TEST-IMAGES local *area* file dataset, you can setup local datasets as defined in the DSSERVE help section. In the following exercises you will access GOES-R Series netCDF format files with the IMG* commands by using DSSERVE to assign a dataset name to the files. When doing so, specify "ABIN" in the *format* parameter, TYPE=IMAGE, and the directory and file masks in the DIRFILE keyword. The files in this exercise are mission-standard Level 1b or Level 2 files in netCDF-4 format, like those from NOAA CLASS or from the GRB data stream after they've been decoded with CSPP Geo or other software. The names of files should look similar to *ABI-L1b-RadC-M3C01_G16_s2015229195720.nc* (current CSPP Geo naming convention) or *OR_ABI-L1b-RadC-M4C16_G16_s20151702215532_e20151702220362_c20151702220394.nc* (GOES-R Ground System naming convention).

1. Create a local dataset to access the ABI netCDF files on your local machine in the *Data/ABI/NetCDF* directory.

OR_ABI-L1b-RadC-M3C02_G16_s20180041657204_e20180041659577_c20180041700012.nc
OR_ABI-L1b-RadC-M3C14_G16_s20180041657204_e20180041659577_c20180041700023.nc
OR_ABI-L2-TPWC-M3_G16_s20180041657204_e20180041659577_c20180041701161.nc

- a. Run the DSSERVE command to access the Level 1b netCDF imagery files.

```
DSSERVE ADD ABI/CONUS ABIN TYPE=IMAGE DIRFILE= '<local-path>/Data/ABI/NetCDF/*RadC*'
```

- b. Use the IMGLIST command to list the most recent image in the dataset.

```
IMGLIST ABI/CONUS
```

- c. Datasets can be organized however the user chooses. Run DSSERVE to include all the files in the data directory.

```
DSSERVE ADD ABI/DATA ABIN TYPE=IMAGE DIRFILE= '<local-path>/Data/ABI/NetCDF/*'
```

- d. IMGLIST can list the image directory and band information with the FORM=ALL keyword. Your IMGLIST command should match the image.

```
IMGLIST ABI/DATA.ALL FORM=ALL
```

```

IMGLIST ABI/DATA.ALL FORM=ALL
Image file directory listing for:ABI/DATA
Pos Satellite/      Date      Time      Center      Res (km)      Image_Size
  sensor
-----
 1 G-16 IMG          4 JAN 18004 16:57:20   30  87
Band: 2      0.64 um VIS clouds fog, insol, winds      0.50  0.50  6000 x10000
Band: 14     11.2 um IR Imagery,SST,clouds,rainfall      2.00  2.00  1500 x 2500
proj:      0 created: 2018004 165720 memo: GOES-R ConUS
type:ABIN   cal type:RAW
offsets: data= 1280 navigation= 256 calibration= 768 auxiliary= 0
doc length: 0 cal length: 0 lev length: 0 PREFIX= 0
valcod:     0 zcor: 0 avg-smp: A
lcor:      1 ecor:      1 bytes per pixel: 2 ss:186
Resolution Factors (base=1): Line= 4.0 Element= 4.0
 2 G-16 PRD          4 JAN 18004 16:57:20   30  87
Band: 13     L2: TPW - Total Precipitable Water      2.00  2.00  300 x 500
proj:      0 created: 2018004 165720 memo: GOES-R ConUS
type:ABIN   cal type:RAW
offsets: data= 1280 navigation= 256 calibration= 768 auxiliary= 0
doc length: 0 cal length: 0 lev length: 0 PREFIX= 0
valcod:     0 zcor: 0 avg-smp: A
lcor:      1 ecor:      1 bytes per pixel: 2 ss:187
Resolution Factors (base=1): Line= 20.0 Element= 20.0
IMGLIST: done

```

- e. Another IMGLIST keyword, FORM=BAND, lists the band and resolution without the file's expanded listing.

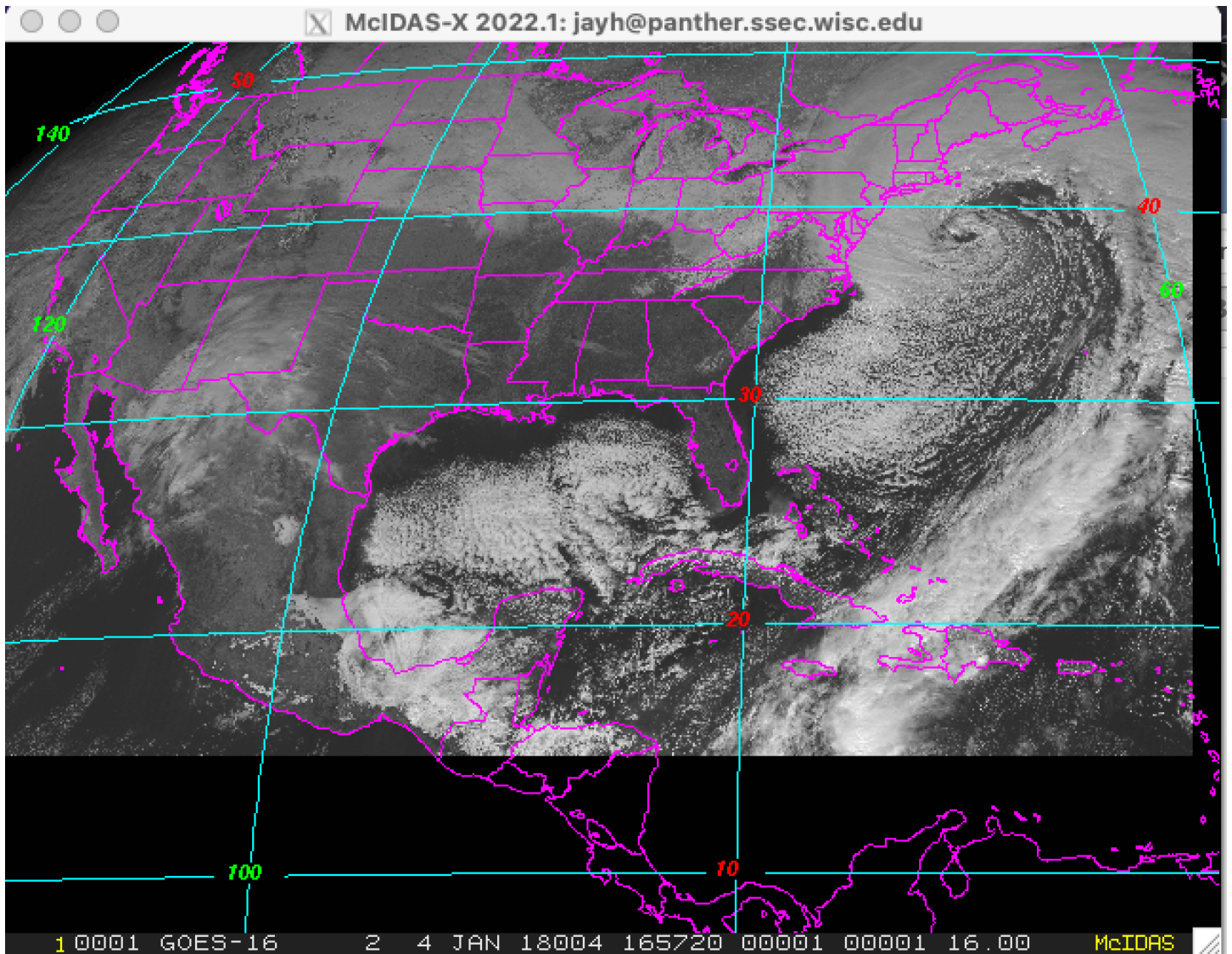
IMGLIST ABI/DATA.1 BAND=2 FORM=BAND

2. The data can be displayed and inspected with the IMGDISP and IMGPROBE commands.
 - a. Run this series of commands to display BAND 2 of an ABI Level 1b file, display a navigated map over the image, place the cursor at the center of the image and run an IMGPROBE command to inspect the pixel values under the cursor.

```

IMGDISP ABI/CONUS BAND=2 MAG= -16
MAP X X LALO
PC C
IMGPROBE MODE=N

```



3. Another way to probe the data is to use the CUR command to display statistics at the pixel under the cursor or at the top of the image.
 - a. Run the following commands to activate and turn off the STAT option of the CUR command.

```
CUR STAT CUR (Move the cursor over the image)  
CUR STAT ON  
CUR STAT OFF
```

Creating a Loop

In this exercise, you will display six images with the `IMGDISP` command and loop the frames using `Alt L`. Then, you will switch to the opposite loop with the `Alt O` command.

1. Check the loop bounds.

Press: **Alt F**

The default loop bounds are 1 through 20 as shown. Notice that the loop bounds in the status line is `1-20`.

```
Video Status for Your Workstation
                                Frames(s)
                                -----
Number Available                  20
Current                          1  (Opp = 11)
Loop Bounds                      1 to 20
Visible (K & W toggle)          Yes / Yes
Looping (L toggle)               No
Cursor parameters: Size =    31 / 31      Type = Xhair
                          Center position = 402 / 622      Color = RED
Image frames  1 - 20  with imbedded graphics  are 480 BY 640
```

2. Display six images using the following `IMGDISP` commands. `IMGDISP` is discussed further in the Satellite Imagery lesson.

Type: **IMGDISP G16C.1 1 STATION=DCA BAND=13 REPEAT=3**

Type: **IMGDISP G16FD.1 4 STATION=DCA BAND=1 REPEAT=3**

3. Set the loop bounds with the `LS` command, then loop frames 1 through 3.

Type: **LS 1-3;LS O 4-6**

Press: **Alt L**

Notice that the status line at the bottom of the Text and Command window changes to show an `L` under `Switches` and the Frame number changes as the loop progresses:

```
IMA GRA Bounds Switches          Date      Time  T
[2] [2] 1-3      L                09 Sep 2019252 15:43:31 0
```

4. Toggle to the opposite loop.

Press: **Alt O**

The loop displays frames 4 through 6 and the bounds in the status line changes to `4-6`.

5. Stop the loop.

Press: **Alt L**

The loop display returns to the first frame in the current loop sequence, in this case, frame 4. Notice that the status line now has a value of 4 in the Frame position and the L is not listed in the switches section.

IMA	GRA	Bounds	Switches	Date	Time	T
4	4	4-6		09 Sep 2019	252 15:44:01	0

6. Toggle to the primary loop.

Press: **Alt O**

7. Use the single letter commands A and B to manually step through the current loop. Alt A advances through the frames; Alt B moves backwards through the frames.

Press: **Alt A**

Press: **Alt B**

Satellite Imagery - Basic Concepts

SSEC receives real-time satellite images from geostationary and polar orbiting satellites. Geostationary satellites remain above a fixed location on the earth's surface, approximately 35,800 km above the equator. Because the satellites rotate with the earth, they always observe the same portion of the globe. Typically, there are operational geostationary satellites from the United States, the European Organisation for the Exploitation of Meteorological Satellites (EUMETSAT), and the Japan Meteorological Agency (JMA).

At most times, the United States has a satellite called GOES-East, which monitors North and South America and the western Atlantic Ocean, and one called GOES-West, which monitors North and South America and the eastern Pacific Ocean. EUMETSAT has two satellites (METEOSAT) which monitor Europe, Africa, Asia, the Indian Ocean, and the Atlantic Ocean. JMA has a satellite (Himawari) which monitors eastern Asia, Australia, the western Pacific Ocean and the eastern Indian Ocean.

Polar orbiting satellites orbit at much lower altitudes (800-900 km). Their path is 2,400 km wide centered at the orbit path. With each orbit, the satellites observe a new path. SSEC typically receives real-time imagery from the operational POES satellites plus others like JPSS, Metop, Aqua and Terra.

Satellite Data Storage

In McIDAS, satellite data is stored in various formats. McIDAS contains various servers that can read and write multiple formats of satellite data. For example, you can setup a dataset to list, display, and manipulate mission-standard Level 1b files in netCDF-4 format or Joint Polar Satellite System (JPSS) Series VIIRS SDR files in HDF5 format. Datasets can also be setup to write GeoTIFF and netCDF files. The help section of the DSSERVE command describes each type of server, and the necessary information to setup the dataset.

Type: **HELP DSSERVE**

When copying satellite data locally to your machine for use in McIDAS, the resultant file format written is called *areas*. You can copy, change, display, and delete areas. Areas contain both data and area directories, similar to how each satellite server assembles the data in memory for use in McIDAS. These satellite data can be displayed in McIDAS image frames.

The server or area directory contains descriptive information, such as the sensor source, image date, picture start time, and image coordinates. To see this information, use the `IMGLIST` command, as shown below to list a mission-standard Level 1b files in netCDF-4 format setup in a dataset which is using the ABIN server format:

```

IMGLIST G16C.1 FORM=ALL
Image file directory listing for:G16C
Pos Satellite/          Date      Time      Center      Res (km)  Image_Size
  sensor
-----
 1 GOES-16             14 SEP 19257 10:51:13    30   87
Band: 1    0.47 um VIS aerosol-over-land      1.00  1.00  3000 x 5000
Band: 2    0.64 um VIS clouds fog, insol, winds  0.50  0.50  6000 x10000
Band: 3    0.86 um Near IR veg/burn scar,aerosol, w  1.00  1.00  3000 x 5000
Band: 4    1.37 um Near IR cirrus cloud          2.00  2.00  1500 x 2500
Band: 5    1.6 um Near IR cloud phase, snow        1.00  1.00  3000 x 5000
Band: 6    2.2 um Near IR land/cloud, vege, snow    2.00  2.00  1500 x 2500
Band: 7    3.9 um IR Sfc, cloud, fog, fire, winds  2.00  2.00  1500 x 2500
Band: 8    6.2 um IR High-level WV, winds, rainfall  2.00  2.00  1500 x 2500
Band: 9    6.9 um IR Mid-level WV, winds, rainfall  2.00  2.00  1500 x 2500
Band: 10   7.3 um IR Lower-level WV, winds & SO2      2.00  2.00  1500 x 2500
Band: 11   8.4 um IR Total WV cloud phase, dust      2.00  2.00  1500 x 2500
Band: 12   9.6 um IR Total ozone,turbulence,wind    2.00  2.00  1500 x 2500
Band: 13  10.3 um IR Surface & cloud              2.00  2.00  1500 x 2500
Band: 14  11.2 um IR Imagery,SST,clouds,rainfall  2.00  2.00  1500 x 2500
Band: 15  12.3 um IR Total water, ash, and SST      2.00  2.00  1500 x 2500
Band: 16  13.3 um IR Air temp, cloud hgt and amt   2.00  2.00  1500 x 2500
proj:      0 created: 2019257 105113 memo: GOES ConUS - Mode 6
type:ABIN  cal type:RAW
offsets: data= 1280 navigation= 256 calibration= 768 auxiliary= 0
doc length: 0 cal length: 0 lev length: 0 PREFIX= 0
valcod:    0 zcor: 0 avg-smp: A
lcor: 1 ecor: 1 bytes per pixel: 2 ss:186
Resolution Factors (base=1): Line= 4.0 Element= 4.0
IMGLIST: done

```

Area Naming Conventions

Areas use the naming convention *AREAnnnn* where *nnnn* is the four digit area number. For example, *AREA0003* is the name of the file that contains area 3. In ADDE, directories of data files or AREA files are grouped together in a dataset. For example, in a previous lesson, you created the dataset *MYDATA/IMAGES*, which contains the AREA file numbers from 1 to 9999. Individual AREA files can be accessed with their position number within the dataset. In the case of *MYDATA/IMAGES*, position 3 (*MYDATA/IMAGES.3*) would relate to the file *AREA0003*. However, in the dataset *MYDATA/TEST-IMAGES* that was created in the previous lesson, the dataset defined the AREA files ranging from 4000 to 4004, so position 3 (*MYDATA/TEST-IMAGES.3*) would be *AREA4002*. (This would be equivalent to *MYDATA/IMAGES.4002*)

Coordinate Systems

McIDAS references image data in four different, but interconnected coordinate systems:

- Image
- Area (File)
- Earth
- TV

Image Coordinates

The image coordinate system forms the basis for the other McIDAS coordinate systems. A full image is a sequence of lines and elements arranged from top to bottom. The top line and leftmost element have image coordinates (1,1). Therefore, each pixel has a unique pair of line and element values that are its image coordinates. The figure below represents a full image, image sector and displayed area. The upper-left image coordinates of the full image are (1,1) and the upper-left coordinates of the image sector are (3500,5000).

Area (File) Coordinates

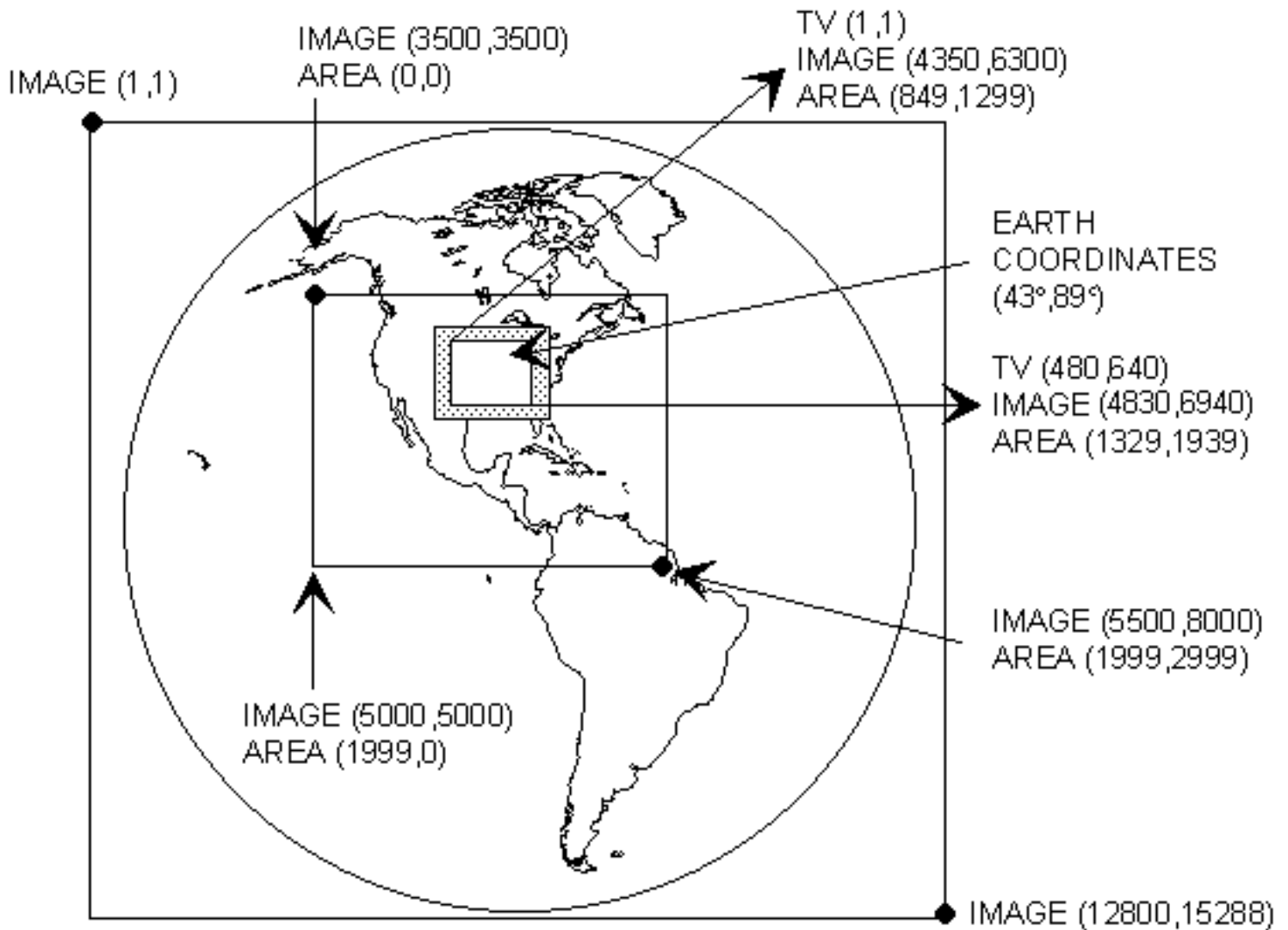
Area coordinates (file coordinates in ADDE) are based on the size of the area only. Like image coordinates, area coordinates are referenced as lines and elements. The first pixel has area coordinates (0,0) as shown in the image sector below. The bottom-right pixel has area coordinates (LSIZ-1, ESIZ-1) where LSIZ and ESIZ are the number of lines and elements in the area.

Earth Coordinates

If the displayed image is navigated, the image coordinates can be converted to earth coordinates (latitude and longitude). Earth coordinates are specified in degrees, minutes, and seconds in the form DDD:MM:SS. Southern latitudes and longitude **east** of Greenwich are negative. Latitudes run from -90:00:00 to +90:00:00 and longitudes run from -180:00:00 to +180:00:00.

TV Coordinates

The pixels on the McIDAS image frames are arranged by raster lines and pictel elements. The raster lines run horizontally across the frame and the pictel elements run vertically across the frame. The pixel in the upper-left corner of the frame is numbered (1,1) which means (raster line 1, pictel element 1). The total number of raster lines and pictel elements on the frame is determined by the frame size. The lower-right corner of the default-sized frame is (480,640) in TV coordinates.



1. Setup a new dataset of 10 Mesoscale ABI images of Hurricane Dorian. Access the Level 1B netCDF files on your local machine in the *Data/Satellite/ABI_data/Meso1* directory.

Example file name:

OR_ABI-L1b-RadM1-M6C02_G16_s20192461800251_e20192461800309_c20192461800346.nc

Run the DSSERVE command to access the Level 1b netCDF imagery files.

Type: **DSSERVE ADD DORIAN/M1 ABIN TYPE=IMAGE DIRFILE='<local-path>/Data/Satellite/ABI_data/Meso1/OR_ABI*'**

2. List the area directories for the first 3 images in the DORIAN/M1 dataset.

Type: **IMGLIST DORIAN/M1.1 3**

```

IMGLIST DORIAN/M1.1 3
Image file directory listing for:DORIAN/M1
Pos Satellite/      Date      Time      Center  Band(s)
   sensor                Lat   Lon
-----
  1 GOES-16      3 SEP 19246  18:00:25   28   79 2
  2 GOES-16      3 SEP 19246  18:01:22   28   79 2
  3 GOES-16      3 SEP 19246  18:02:22   28   79 2
IMGLIST: done

```

- 3. List the directories of all images of the DORIAN/M1 dataset that contain images between at 18:01:00 and 18:05:00 UTC on day 2019246.

Type: **IMGLIST DORIAN/M1.ALL DAY=19246 TIME=18:01:00 18:05:00**

```

IMGLIST DORIAN/M1.ALL DAY=19246 TIME=18:01:00 18:05:00
Image file directory listing for:DORIAN/M1
Pos Satellite/      Date      Time      Center  Band(s)
   sensor                Lat   Lon
-----
  2 GOES-16      3 SEP 19246  18:01:22   28   79 2
  3 GOES-16      3 SEP 19246  18:02:22   28   79 2
  4 GOES-16      3 SEP 19246  18:03:22   28   79 2
  5 GOES-16      3 SEP 19246  18:04:22   28   79 2
IMGLIST: done

```

- 4. Display the first GOES-16 visible 0.5km image on frame 1.

Type: **IMGDISP DORIAN/M1.1 1**

Notice that the bottom of the image has an annotation line which lists the frame number, satellite type, sensor source, Gregorian date, Julian date, UTC time, upper-left corner image line and element, and resolution. Note the resolution of 1.0, this is the base resolution (best) of the satellite, so it lists 1.0, not 0.5 in the frame label. Run the following command to confirm this:

Type: **IMGLIST DORIAN/M1.1 FORM=ALL**

- 5. List the frame directory for the current image on frame 4.

Type: **FRMLIST**

Information about the frame number, sensor source, band, date, time, and ADDE dataset/position number is displayed for the image.

```

FRMLIST
Frame  Satellite
[panel] Sensor      Band      Date      Time      Data Source
-----
  1[1] GOES-16      2 03 Sep 19246  18:00:25  DORIAN/M1.1
FRMLIST: Done

```

- 6. Display three areas that are in sequence using one IMGDISP command.

Type: **IMGDISP DORIAN/M1.1 4 REPEAT=3**

The repeat factor is 3; therefore this entry displays three images in sequence, on frames 4, 5, and 6.

- Step through the frames to view the images.

Press: **Alt A**

Press: **Alt B**

Listing Image Data

In this exercise, you will list image data by positioning the cursor on the image and entering the IMGPROBE command.

The IMGPROBE command lists data from the area represented on the image at the cursor center. It lists the area number, area coordinates, image coordinates, raw, brightness, and when present, temperature and radiance values. IMGPROBE can also be invoked by pressing Alt-D.

When the *region* parameter is specified as **BOX**, the IMGPROBE command lists the data from the area inside the cursor. By specifying different parameters, you can list raw, brightness, and when present, temperature and radiance values.

- Change your cursor to a smaller size with the CUR command, which is explained in more detail in the Graphics and the Cursor lesson.

Type: **CUR 5 5**

- Show the visible image on frame 4 and position the cursor in the center of the frame. List the area values for the element at the cursor center by using the IMGPROBE command in the non-interactive mode.

Type: **SF 4;PC C;IMGPROBE MODE=N**

Alternate method:

Type: **SF 4;PC C**

Press: **Alt D**

The area number, area coordinates, image coordinates, raw, and brightness values are listed. The Nominal Time is the time the scan began, and the Scan Time is the time that pixel was scanned. Because this is an infrared image, the temperature and brightness values are also listed.

Image Name	Day	Nominal Time	Scan Time	Band				
DORIAN/M1.1	3 Sep 19246	18:00:25	18:00:31	2				
Lat/Lon	File Line/Element	Nominal Image Line/Element	RAW	RAD *	ALB %	BRIT		
32:19:37/ 82:56:27	239/ 319	240/ 320	705	91.518	17.93	108		
* watts/meter**2/steradian/micron								

- List the brightness values for the area inside the cursor.

Type: **PC C;IMGPROBE LIST BOX BRIT MODE=N**

Notice that the data value at the cursor center is the same value listed as with IMGPROBE.

- Display a real time infrared image and list the values at the center of the cursor.

Type: **SF 1;ERASE**

Type: **IMGDISP G16C.-1 BAND=13**

Press: **PC C;Alt D**

Image Name	Day	Nominal Time	Scan Time	Band				
G16C.575	17 Sep 19260	01:41:14	01:41:20	13				
Lat/Lon	File	Nominal Image	RAW	RAD	TEMP	BRIT		
45:18:03/ 122:07:58	Line/Element 239/ 319	Line/Element 957/ 1277	1402	*	K 269.97	120		
* milliwatts/meter**2/steradian/(cm-1)								

Radiance and albedo values are listed when using visible data with raw and brit; whereas, temperature, radiance, raw and brightness values were listed for the infrared image in this step.

Using Coordinate Systems

In this exercise, you will display the same image using different coordinate types. The coordinate system and location where the specified coordinates are positioned on the frame define the coordinate type. The three coordinate systems are:

- Earth
- File (Area)
- Image

The two locations (PLACE) are:

- CENTER (center)
- ULEFT (upper-left corner)

The coordinate type is defined by a combination of the coordinate system keywords and location; for example, LINELE=6501 5065 I PLACE=ULEFT positions the (I) image coordinates (6501,5065) in the (ULEFT) upper-left corner. The first exercise shows that the same image can be displayed using different coordinate types.

- Show and erase frame 1.
Type: **SF 1; ERASE X 1**

- Display a GOES-16 Full Disk Infrared image on frame 1 so the pixel having image coordinates (6501, 5065) appears at the upper-left corner of the frame.

Type: **IMGDISP G16FD.-1 1 LINELE=6501 5065 I PLACE=ULEFT BAND=13**

The combination of I for line/element type and PLACE=ULEFT places the specified image coordinates in the upper-left corner. Note in the on-line help for IMGDISP that the default for PLACE is ULEFT when LINELE is specified and therefore could be omitted from the command line above.

3. Display a GOES-16 Full Disk Infrared image on frame 2 so that Puerto Escondido, Mexico, is at the center of the image frame.

Type: **IMGDISP G16FD.-1 2 STATION=MMPS BAND=13;SF 2**

The STATION parameter places the earth coordinates for Puerto Escondido in the center of the frame, as shown below. When STATION or LATLON are used, the default for PLACE is CENTER.

4. Display a GOES-16 Full Disk Infrared image on frame 3 so that file (area) coordinates (1865, 1586) appear in the center of the image frame.

Type: **IMGDISP G16FD.-1 3 LINELE=1865 1586 F PLACE=CENTER BAND=13;SF 3**

The combination of F for coordinate system and PLACE=CENTER places the specified file (area) coordinates in the center of the image frame.

5. Start and then stop the loop.

Type: **LS 1-3;LS O 4-6**

Press: **Alt L**

Press: **Alt L**

All the images are the same, except they are displayed using different coordinate types.

6. Show frame 2 and use the PC command to position the cursor in the center the frame. Then find the earth coordinates of Puerto Escondido.

Type: **SF 2;PC C**

Press: **Alt E**

Command E lists the latitude and longitude of a pixel on a navigated frame in the format DD:MM:SS. It also lists the TV coordinates and image coordinates of the pixel.

Frame	Latitude	Longitude	Tvline	Tvelem	Line	Elem
2	15:52:12	097:05:00	240	320	7457	6341

Next you will display the same image three times, but change the location of the displayed image using different coordinate types.

1. Erase frames 1 through 3 and show frame 1.

Type: **ERASE I 1 3;SF 1**

2. Display the visible image from the ABI/CONUS dataset on frame 1 so that the earth coordinates 30° latitude and 89° longitude are centered in the frame. Then position the cursor at the center of the frame and verify that the earth coordinates are 30° latitude and 89° longitude.

Type: **IMGDISP ABI/CONUS 1 LATLON=30 89 BAND=2**

Type: **PC C**

Press: **Alt E**

Notice that the center of the frame has a latitude of 30:00:07 and a longitude of 89:00:07.

3. Display the image on frame 2 so that the image coordinates (1999, 2503) are centered in the frame and show frame 2. Then position the cursor in the center of the frame.

Type: **IMGDISP ABI/CONUS 2 BAND=2 LINELE=1999 2503 I PLACE=CENTER;SF 2**

Type: **PC C**

Press: **Alt E**

Notice that the image coordinates (1999, 2503) are in the center of the frame.

4. Display the image on frame 3 so that the file (area) coordinates (0,0) appear in the upper-left corner of the frame and show frame 3.

Type: **IMGDISP ABI/CONUS 3 BAND=2 LINELE=0 0;SF 3**

5. Change the dwell rate and then start and stop the loop. Notice how the image position changes.

Type: **DR 10**

Press: **Alt L**

Press: **Alt L**

Next, you will display a sequence of real time GOES-16 visible and infrared images using the same earth coordinates to compare them. Display the GOES visible images on the primary loop and the GOES infrared images on the opposite loop. Note: The data in positions 2 through 4 on the real time server may be night time images for the visible data.

6. Display the first GOES-16 0.5km Visible image on frame 1, centering the image on earth coordinates 30° N and 75° W. Use the SF= keyword to automatically display the frame.

Type: **IMGDISP G16FD.2 1 LATLON=30 75 BAND=2 SF=YES**

7. Display the next two GOES-16 visible images on frames 2 and 3, centering the images on coordinates 30° N and 75° W.

Type: **IMGDISP G16FD.3 2 LATLON=30 75 BAND=2;IMGDISP G16FD.4 3 LATLON=30 75 BAND=2**

8. Display the three GOES-16 infrared images on frames opposite the GOES-16 visible images. The repeat factor (3) loads all three images with one IMGDISP command.

Type: **IMGDISP G16FD.2 OPP LATLON=30 75 REPEAT=3 BAND=13**

You are using the same coordinates, 30° N and 75° W, to load the images. The repeat factor 3 displays three images in sequence, on frames 4, 5, and 6.

9. Change the dwell rate and loop the images.

Type: **DR 5**

Press: **Alt L**

You should see the loop of the GOES visible images on frames 1 through 3.

10. Toggle to the opposite loop to compare the images. Then stop the loop.

Press: **Alt O**

Press: **Alt L**

Changing the Image Resolution

In this exercise, you will magnify the resolution of displayed images using the **IMGDISP** command. Then, you will use the **IMGPROBE** command to list data values inside the cursor. Many other commands, for example **IMGCOPY**, have parameters to change the resolution.

1. Display the local GOES-16 conus image, band 14, in its original resolution (2 km) on frame 1. Center the image on earth coordinates 30° and 87°

Type: **IMGDISP ABI/CONUS 1 LATLON=30 87 BAND=14**

2. Decrease (blow down) the image resolution of the GOES-16 conus image by a factor of 2 and display it on frame 3.

Type: **IMGDISP ABI/CONUS 3 LATLON=30 87 BAND=14 MAG=-2 SF=YES**

3. Magnify (blow up) the image resolution of the GOES-16 conus image by factor of 2 and display it on frame 2.

Type: **IMGDISP ABI/CONUS 2 LATLON=30 87 BAND=14 MAG=2 SF=YES**

4. Loop the frames.

Press: **Alt L**

5. Change the loop sequence to view the images in the order of increasing resolution.

Type: **LS 3 1 2**

6. Stop the loop.

Press: **Alt L**

7. Show frame 1, change the size of the cursor, and list the data and brightness values inside the cursor.

Type: **PC C;SF 1;CUR 5 5;IMGPROBE LIST BOX BRIT MODE=N**

Notice you have 5 lines and 5 elements of data.

8. Show frame 3 and list the data and brightness values inside the cursor.

Type: **SF 3;IMGPROBE LIST BOX BRIT MODE=N**

Notice that you have 10 lines and 10 elements of data. Because the image was blown down by a factor of 2, every second line and element was sampled. Therefore, the number of lines and elements read from the area is twice the amount displayed as an image. The following equation determines the number of lines and elements in the cursor area for a blow down:

*cursor size*blow down factor*

which for this example is $5 * 2 = 10$.

9. Show frame 2 and list the data and brightness values inside the cursor.

Type: **SF 2;IMGPROBE LIST BOX BRIT MODE=N**

There are 3 lines by 3 elements of data listed. The following equation determines the number of lines and elements in the cursor area for a blow up:

(cursor size-1)/blow up factor + 1

which for this example is $[(5 - 1)/2] + 1 = 3$.

Copying and Displaying Images

In this exercise, you will copy and display areas with the IMGCOPY and IMGDISP commands.

1. Show frame 4, copy a GOES-16 0.5km Vis image to position 1 in your MYDATA/TEST-IMAGES (AKA TI) dataset, and display the image on frame 4. The SIZE=SAME keyword copies the entire image to the destination dataset.

Type: **SF 4**

Type: **IMGCOPY ABI/CONUS TI.1 BAND=2 SIZE=SAME;IMGDISP TI.1 4 LAT=30 87**

2. List the two image directories

Type: **IMGLIST ABI/CONUS FORM=BAND;IMGLIST TI.1 FORM=BAND**

Notice that the images now contain the same information and the band 2 image resolutions are nearly the same.

```

Image file directory listing for:ABI/CONUS
Pos Satellite/      Date      Time      Center      Res (km)      Image_Size
  sensor                                     Lat  Lon      Lat  Lon
-----
  1  GOES-16          4 JAN 18004  16:57:20   30   87
  Band: 2    0.64 um VIS clouds fog, insol, winds    0.50  0.50  6000 x10000
  Band: 14   11.2 um IR Imagery,SST,clouds,rainfall  2.00  2.00  1500 x 2500

Image file directory listing for:TI
Pos Satellite/      Date      Time      Center      Res (km)      Image_Size
  sensor                                     Lat  Lon      Lat  Lon
-----
  1  GOES-16          4 JAN 18004  16:57:20   30   87
  Band: 2    0.64 um VIS clouds fog, insol, winds    0.63  0.54  6000 x10000

```

The Latitude and Longitude values listed above are the actual resolution values at the center point of the image.

3. Show frame 5, copy the GOES-16 2km IR image to the second position in your TI dataset, and display the image on frame 5 centered on Boston.

Type: **SF 5**

Type: **IMGCOPY ABI/CONUS.1 TI.2 BAND=14 STA=KBOS;IMGDISP TI.2 5**

4. List the two image directories.

Type: **IMGLIST ABI/CONUS.1 FORM=BAND;IMGLIST TI.2 FORM=BAND**

In ABI/CONUS.1, the image size of band 14 is 1500 x 2500. Notice that the image size for TI.2 is 480 x 640. If the SIZE= keyword is not specified, a 480 by 640 image sector (the default size of image frames) is copied to the new area.

```

Image file directory listing for:ABI/CONUS
Pos Satellite/      Date      Time      Center      Res (km)      Image_Size
  sensor                                     Lat  Lon      Lat  Lon
-----
  1  GOES-16          4 JAN 18004  16:57:20   30   87
  Band: 2    0.64 um VIS clouds fog, insol, winds    0.50  0.50  6000 x10000
  Band: 14   11.2 um IR Imagery,SST,clouds,rainfall  2.00  2.00  1500 x 2500
IMGLIST: done
Image file directory listing for:TI
Pos Satellite/      Date      Time      Center      Res (km)      Image_Size
  sensor                                     Lat  Lon      Lat  Lon
-----
  2  GOES-16          4 JAN 18004  16:57:20   42   71
  Band: 14   11.2 um IR Imagery,SST,clouds,rainfall  3.21  2.12  480 x 640
IMGLIST: done

```

5. Show frame 6, copy the GOES-16 2km infrared image to TI.3 centered on Mobile, Alabama, and increase the resolution by a factor of 2. Display it on frame 6.

Type: **SF 6;IMGCOPY ABI/CONUS.1 TI.3 BAND=14 STATION=MOB MAG=2;IMGDISP TI.3 6**

The image is displayed on frame 6 with a resolution of 1 km (2km is listed due to the base resolution of the satellite).

6. List the two images to compare image resolutions. Note that the lat and lon resolution values are different between the images.

Type: **IMGLIST ABI/CONUS.1 BAND=14 FORM=BAND;IMGLIST TI.3 FORM=BAND**

The original image (ABI/CONUS.1 had a resolution of 2 km and the new area (TI.3) has a resolution of 1 km. This is a blow up.

7. Display a loop of GOES-16 Meso1 images of water vapor from 17 to 18 UTC on frames 4 through 6 and use the REFRESH keyword to plot a high resolution map on each of the images.

Type: **IMGDISP RTGOESR/M1C10 TIME=17 18 ALL=4 6 REFRESH='MAP VH GRA=(GRA) IMA=(IMA)'**

Notice that the numbers placed into the MAP command's GRA and IMA keywords take the place of (GRA) and (IMA) and now match the image frame numbers of the loop.

Manipulating Images

A major strength of McIDAS as a meteorological data analysis/display package is its ability to display and manipulate satellite images. In this lesson, you will use the IMGREMAP and IMGOPER commands to remap images and create new image products.

1. Remap the second latest GOES-16 Visible image in the RTGOESR/FD dataset to a mercator projection centered on Washington DC, and place it in the fourth position in the TI dataset. Set the output resolution at 4 km, and remap calculates the size needed to make the default 480x640 image.

Type: **IMGREMAP RTGOESR/FD.-1 TI.4 BAND=2 STATION=DCA PRO=MERC RES=4**

2. List out the directory information for these images.

Type: **IMGLIST RTGOESR/FD.-1 FORM=BAND BAND=2;IMGLIST TI.4 FORM=BAND**

```
Image file directory listing for:RTGOESR/FD
```

Pos	Satellite/ sensor	Date	Time	Center Lat Lon	Res (km) Lat Lon	Image_Size
192	GOES-16	17 SEP 19260	10:00:18	0 75		
	Band: 2	0.64 um VIS clouds	fog, insol, winds		0.50 0.50	21696 x21696
191	GOES-16	17 SEP 19260	09:50:18	0 75		
	Band: 2	0.64 um VIS clouds	fog, insol, winds		0.50 0.50	21696 x21696


```
Image file directory listing for:TI
```

Pos	Satellite/ sensor	Date	Time	Center Lat Lon	Res (km) Lat Lon	Image_Size
4	GOES-16	17 SEP 19260	09:50:18	39 77		
	Band: 2	0.64 um VIS clouds	fog, insol, winds		3.12 3.12	480 x 640

IMGLIST: done

Notice that the second image now has a different center lat/lon and is a different size than the original image.

- Show frame 1. Display the images in frames 1 and 2 centered over Washington DC. Draw a map of dashed lat/lon lines in color 3 on each image.

Type: **SF 1**

Type: **IMGDISP RTGOESR/FD.-1 1 BAND=2 STATION=DCA REFRESH='MAP LALO -3 GRA=(GRA) IMA=(GRA)'**

Type: **IMGDISP TI.4 2 STATION=DCA REFRESH='MAP LALO -3 GRA=(GRA) IMA=(GRA)'**

- Set the loop sequence, and loop between the frames.

Type: **LS 1-2**

Press: **Alt L**

Notice that the images have different projections.

- Stop the loop.

Press: **Alt L**

- Create a new image using two DORIAN visible GOES-16 images in the DORIAN dataset using the formula DORIAN/M1.1 – DORIAN/M1.10. This will show the shift in the clouds from one image to the next. Put the new image into position 5 of the TI dataset.

Type: **IMGOPER DORIAN/M1.1 DORIAN/M1.10 TI.5 COEF=1 -1 FORM=ADD MAG=-2**

IMGOPER generates a new image by applying mathematical functions to data from one or more source images. The following equation computes the data value for each line/element pair in the destination image. This operation is performed repeatedly using data from source image line/element pairs as input values (*input*) until the entire image is completed.

output data value=

$$\text{FUNC}[\text{ACON}+(\text{MCON}*(\text{FORM}(((\text{COEF1}*(\text{OFF1}+((\text{SIGN1})*\text{input1})**\text{POW1}))$$

$$(\text{COEF2}*(\text{OFF2}+((\text{SIGN2})*\text{input2})**\text{POW2})) \dots$$
$$(\text{COEFn}*(\text{OFFn}+((\text{SIGNn})*\text{inputn})**\text{POWn}))))))]$$

The part of the equation operating on a single input data value, $(\text{COEFn}*(\text{OFFn}+((\text{SIGNn})*\text{inputn})**\text{POWn}))$, is referred to as a *term*. n represents the number of *sdataset* images. It may not be larger than 100. *input1* . . . *n* represents the individual data values from each of the source images. The **FORM** keyword determines how the terms are combined. For example, if you specify **FORM=MULT**, the terms are multiplied.

7. Erase the maps in graphics frames 4 through 6, and display the images that you just used in frames 3-5.

Type: **ERASE G 4 6**

Type: **IMGDISP DORIAN/M1.1 3 MAG=-2**

Type: **IMGDISP DORIAN.M1.10 4 MAG=-2**

Type: **IMGDISP TL5 5**

8. Loop through the images.

Type: **LS 3-5**

Press: **Alt L**

9. Stop the loop.

Press: **Alt L**

10. Add a label to the image frame.

Type: **SF 5;FRMLABEL IMA=5 "CLOUD MOVEMENT FROM 18:00:25 to 18:09:22**

Enhancements - Basic Concepts

There are two types of enhancements: color enhancements and grayscale enhancements. A color enhancement changes grayshades to colors and a grayscale enhancement changes a grayshade to a different grayshade value.

Color Enhancements

A color enhancement is a table of colors that corresponds to brightness values. Color enhancements are useful for tracking cloud features. For example, to track the tops of thunderstorms overshooting the tropopause, you can color all brightness values between 180 and 250 red.

Color enhancements are created with the **EU MAKE** command. You can create a color enhancement by assigning a color to a brightness value or a brightness range. For example, you could create an enhancement

table where the color green corresponds to the brightness range 50 to 79, the color blue corresponds to the brightness range 80 to 99, and the color red corresponds to brightness value 100.

In addition, you can create a color enhancement by specifying color intensities. The values within the brightness range are interpolated within the color intensity range. For example, if the brightness range 0 to 71 is assigned to a blue color intensity of 203 to 255, a green color intensity of 173 to 200, and a red color intensity of 3 to 100, as shown below, the pixels with a low brightness value (near 0) will have corresponding low red, green, and blue intensities, and the pixels with high brightness values (near 71) will have corresponding high red, green, and blue intensities.

Brightness		Blue		Green		Red	
min	max	min	max	min	max	min	max
---	---	---	---	---	---	---	---
0	71	203	255	173	200	3	100

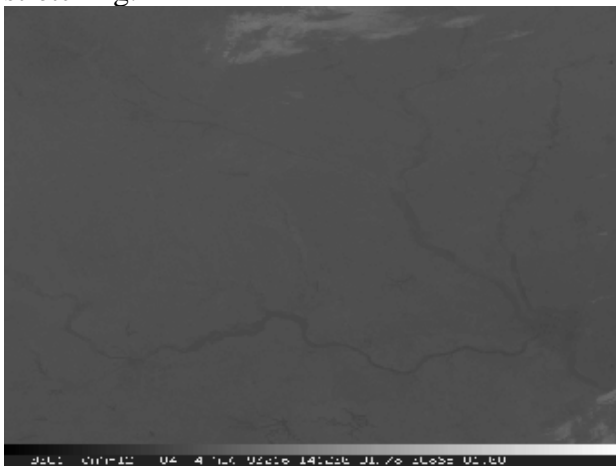
Once you create an enhancement table, you can save it using EU SAVE and then restore it using the command EU REST. You can apply the same or different enhancement tables to each frame on the workstation.

Grayscale Enhancements

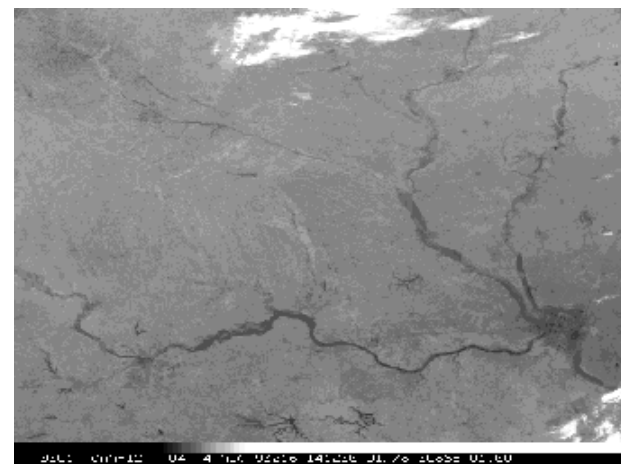
Normally, a pixel's digital value, stored in an area, correlates to a brightness value. Each brightness value appears as a different shade of gray when the image is displayed. When a grayscale enhancement is applied, the correlation between the digital values and the displayed grayshades changes. You can change the grayscale contrast of an image two ways: using image contrast stretching or using image data stretching.

Image Contrast Stretching

Image contrast stretching changes the grayscale of the displayed image; it does not change the area data values. You can change the grayscale contrast of an image using the EB command. You can run the EB command two ways: using the command line and using the mouse. Using the command line, you specify the lower and upper brightness values to be enhanced. All pixels with brightness values below the lower input values and above the upper input value remain unchanged. The brightness values between the range are linearly interpolated. Using the mouse controlled version, you move the mouse to increase or decrease the brightness of the image. You can save grayscale enhancements and apply them to other images using the EU SAVE and EU REST commands. The example below shows the original contrast of an image and the contrast of an image after contrast stretching.



Original Image



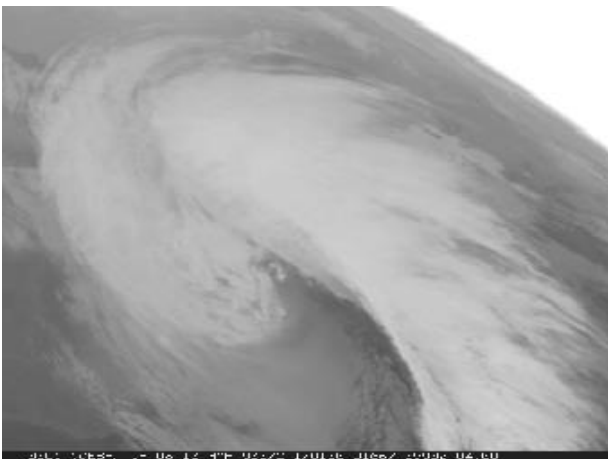
Contrast stretched image

Image Data Stretching

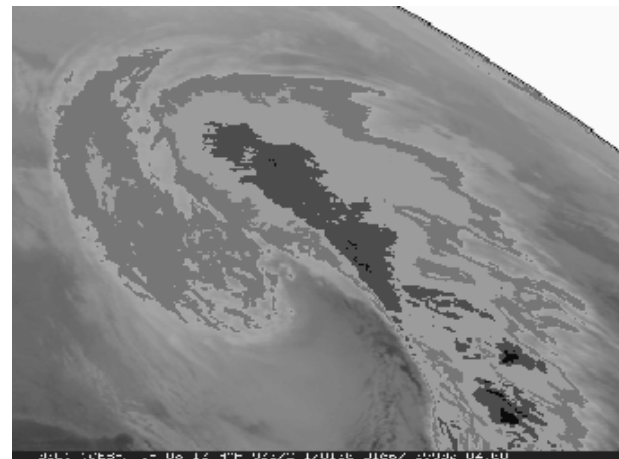
Image data stretching changes the grayscale of an image by stretching area data values to brightness values. To stretch the image data values, you must create a table that defines the values to stretch, as shown below.

```
SU TABLE MB
BREAKPOINTS STORED IN TABLE : MB.ST
INPUT        OUTPUT
-----
162.8        250
192.3        250
192.4        250
209.3        10
209.4        10
213.3        10
213.4        75
219.3        75
219.4        156
230.3        156
230.4        117
241.3        117
241.4        167
279.8        102
279.9        102
301.9        0
302          0
330          0
CALIBRATION TYPE : AAA
CALIBRATION UNITS : TEMP
BAND NUMBER      : -1
INTERPOLATION TYPE: LIN
```

The [SU](#) command defines tables to stretch raw, radiance, temperature, albedo, or brightness values (depending on the calibration type) to a user-defined brightness value. Stretch tables are used with the [IMGDISP](#) command to emphasize weather features in an image. The example below shows an image before and after an MB data stretch table was applied.



Original Image



Data Stretched image

Creating Color Enhancements

In this exercise, you will use the EU command to assign colors to brightness values. You will create a simple enhancement that assigns brightness values to colors and another enhancement that assigns various brightness ranges to color ranges.

1. Set the loop sequence to frames 1 through 6.

Type: **LS 1-6**

2. Display the first GOES-16 0.5km visible image on frame 1 centered on 29° and 78°. Decrease the image resolution by a factor of 2 and place a grayscale bar on the frame.

Type: **IMGDISP DORIAN/M1.1 1 LAT=28 79 MAG=-2 SF=YES;BAR LINT=10**

3. Assign the color red to the brightness range 180 to 220 to color enhance the cloud tops.

Type: **EU MAKE 180 220 RED**

Notice how the grayscale bar also changes to reflect the changes to the enhancement.

4. List the enhancement table. The brightness values between 180 and 220 are assigned to a red intensity range of 255 to 255.

Type: **EU TABLE**

Brightness		Blue		Green		Red	
min	max	min	max	min	max	min	max
---	---	---	---	---	---	---	---
0	179	0	179	0	179	0	179
180	220	0	0	0	0	255	255
221	255	221	255	221	255	221	255

5. Assign the brightness range 180 to 220 to the range of colors between yellow and red.

Type: **EU MAKE 180 220 YELLOW RED**

The brightness value 180 is yellow, the brightness value 220 is red, and the values in between are displayed as various shades between the two colors.

6. List the enhancement table.

Type: **EU TABLE**

The enhancement table is listed as shown below. The blue, green, and red intensity values of 0, 255, and 255 create the color yellow and are assigned to brightness value 180. The blue, green, and red intensity values of 0, 0, and 255 create the color red and are assigned to brightness value 220.

Brightness	Blue	Green	Red
------------	------	-------	-----

min	max	min	max	min	max	min	max
---	---	---	---	---	---	---	---
0	179	0	179	0	179	0	179
180	220	0	0	255	0	255	255
221	255	221	255	221	255	221	255

7. Save the enhancements with the name **STORM**.

Type: **EU SAVE STORM**

8. Restore the default enhancement to the frame.

Type: **EU REST**

9. Restore the enhancement **STORM** to the frame.

Type: **EU REST STORM**

10. List the enhancement tables in your account on the workstation.

Type: **EU LIST**

The file **STORM.ET** should be listed.

11. Delete the enhancement **STORM**.

Type: **EU DEL STORM**

12. Restore the default enhancement to frames 1 through 6.

Type: **EU REST X 1 6**

The **X** after the **REST** parameter indicates to use the default enhancement.

13. Erase the image and the graphics from frame 1.

Type: **ERASE**

Changing Grayscale Contrast

In this section, you will use two methods to create a grayscale contrast. First, you will create image contrast stretching and then you will create and apply image data stretching.

Creating Image Contrast Stretching

In this exercise, you will use the EB command to change the grayscale contrast of an image. First, you will use the mouse to move the cursor over the image and stretch the grayscale contrast. Then, you will input the values manually.

1. Display the first GOES-16 Dorian visible image on frame 1 centered on 29° and 78°, decrease the resolution by a factor of two, and add a gray scale bar.

Type: **IMGDISP DORIAN/M1.1 1 LAT=28 79 MAG=-2 GRAY=YES**

2. Initiate mouse-controlled grayscale stretching.

Type: **EB**

3. Move the cursor to the image window.

4. Move the mouse to the right to brighten the image. The range of pixels with a brightness near 255 (white) increases, as shown in the gray scale bar at the bottom of the frame.

5. Move the cursor towards the top of the frame to decrease the image brightness. The range of pixels with a brightness near 0 (black) increases, as shown in the gray scale bar at the bottom of the frame.

6. Find an enhancement that you like and press the right mouse button to end the enhancement.

7. Save the grayscale enhancement as GRAY.

Type: **EU SAVE GRAY**

8. List the brightness value at the center of the image.

Type: **PC C**

Press: **Alt D**

9. Restore the original grayscale of the image.

Type: **EU REST**

10. List the brightness value at the center of the image.

Type: **PC C**

Press: **Alt D**

Note the values are the same as those in step 8. IMGPROBE (Alt-D) does not list the values that were modified with image contrast stretching, but lists the values stored in the area.

Now, you will manually input the brightness values with the EB command.

1. Position the cursor at TV coordinates (14,262) and (277,382) to find the brightness values.

Type: **PC T 14 262;IMGPROBE MODE=N**

Type: **PC T 277 382;IMGPROBE MODE=N**

The brightness values are 165 and 249.

2. Rescale the brightness values 165 to 249 to go from 0 to 255. Brightness value 165 will become 0 and value 249 will become 255. All values in between will be linearly stretched between 0 and 255.

Type: **EB 165 249 0 255**

Since most of the brightness values of the hurricane are between 165 and 249, creating an enhancement for this range makes the image features more prominent. Note that all brightness values outside the range of 165 to 249 remain unchanged.

3. Save the enhancement as GRAY2.

Type: **EU SAVE GRAY2**

4. Restore the default enhancement table to the frame.

Type: **EU REST**

5. List the enhancement tables that start with GRAY on your workstation.

Type: **EU LIST GRAY**

PERM	SIZE	LAST	CHANGED	FILENAME	DIRECTORY
-rw-	3268	Oct 29	17:01	GRAY.ET	/home/user/mcidas/data
-rw-	3268	Oct 29	17:01	GRAY2.ET	/home/user/mcidas/data

6536 bytes in 2 files

6. Delete the saved enhancement tables.

Type: **EU DEL GRAY;EU DEL GRAY2**

Creating and Applying Image Data Stretching

Next, you will define stretch tables to stretch brightness and temperature values stored in an area. Then, you will apply the stretch tables to images and compare the stretched values to the original values.

1. Add a high resolution map to the hurricane display.

Type: **SF 1;MAP H**

- Position the cursor at the center of the frame and list the brightness value.

Type: **PC T 327 79**

Press: **Alt D**

The brightness value at the center is 87.

- Next, initialize a stretch table named LEARN to stretch brightness values. The VISR parameter specifies the data type as GOES 1-byte data.

Type: **SU INI LEARN VISR BRIT**

- Define the brightness ranges to stretch. Assign the brightness value 0 to 255 and the value 255 to 0 to make light areas dark and dark areas light.

Type: **SU MAKE LEARN 0 255 255 0**

- List the breakpoints in the stretch table.

Type: **SU TABLE LEARN**

The table lists the brightness values and the corresponding stretched values as shown below.

```
SU TABLE LEARN
BREAKPOINTS STORED IN TABLE : LEARN.ST
INPUT          OUTPUT
-----
0              255
255            0
CALIBRATION TYPE : VISR
CALIBRATION UNITS : BRIT
BAND NUMBER      : -1
INTERPOLATION TYPE: LIN
SU: DONE
```

- Display the first GOES-16 0.5km Visible image on frame 2 centered at 29° and 78°. Decrease the resolution by a factor of 2, apply the stretch table LEARN, and add a high resolution map.

Type: **IMGDISP DORIAN/M1.1 2 LAT=29 78 MAG=-2 SU=LEARN SF=YES;MAP H**

- Set the loop bounds from 1 to 2 and compare the images.

Type: **LS 1-2**

Press: **Alt A**

Press: **Alt B**

- Show frame 2 and list the areas values at the cursor's center.

Type: **SF 2;PC C;IMGPROBE MODE=N**

Notice that there is a MODB/LEARN data type listed in the output of the D command, as shown below. This lists the value of the stretched data.

Image Name	Day	Nominal Time	Scan Time	Band					
DORIAN/M1.1	3 Sep 19246	18:00:25	18:00:31	2					
Lat/Lon	File	Nominal Image	RAW	RAD	ALB	BRIT	MODB		
28:00:10/ 78:59:53	Line/Element 993/ 993	Line/Element 994/ 994	2625	*	%	77.60	LEARN	225	30
* watts/meter**2/steradian/micron									

Since the values in the table are reversed (0 is now 255 and 255 is now 0), you can calculate the stretched value of a pixel by subtracting the pixel's original brightness value from the maximum value. For example, to calculate the stretched value of the center pixel, subtract the original brightness value (225) from the maximum brightness value (255); the stretched value of the center pixel is 30 (255-225).

- List the stretch tables on the workstation.

Type: **SU LIST**

Next, you will create a Multiple Breakpoint (MB) stretch table to enhance clouds in a GOES infrared image and create an approximate MB stretch curve.

- Erase the graphics on frames 1 and 2. Display the first GOES-16 2km IR image on frame 1 centered on Washington, DC and show the frame. Use the REFRESH keyword of IMGDISP to draw a data bar on the image and label every 10th value in blue (color 6).

Type: **ERASE G 1 2;IMGDISP ABI/CONUS.1 1 BAND=14 STA=DCA SF=YES
REFRESH='BAR (GRA) LINT=10 COLOR=6'**

- Initialize a stretch table named MB to stretch temperature values to a brightness range. The AAA parameter specifies 2-byte GOES data.

Type: **SU INI MB AAA TEMP**

- Assign the temperature values between 330° K and 302° K to the brightness value 0 in the MB stretch table.

Type: **SU MAKE MB 330 302 0 0**

- Assign the temperature values between 301.9° K and 279.9° K to the brightness range 0 to 102 in the MB stretch table.

Type: **SU MAKE MB 301.9 279.9 0 102**

- Assign the temperature values between 279.8° K and 241.4° K to the brightness range 102 to 167 in the MB stretch table.

Type: **SU MAKE MB 279.8 241.4 102 167**

6. Assign the temperature values between 241.3° K and 230.4° K to the brightness value 117 in the MB stretch table.

Type: **SU MAKE MB 241.3 230.4 117 117**

7. Assign the temperature values between 230.3° K and 219.4° K to the brightness value 156 in the MB stretch table.

Type: **SU MAKE MB 230.3 219.4 156 156**

8. Assign the temperature values between 219.3° K and 213.4° K to the brightness value 75 in the MB stretch table.

Type: **SU MAKE MB 219.3 213.4 75 75**

9. Assign the temperature values between 213.3° K and 209.4° K to the brightness value 10 in the MB stretch table.

Type: **SU MAKE MB 213.3 209.4 10 10**

10. Assign the temperature values between 209.3° K and 192.4° K to the brightness range 10 to 250 in the MB stretch table.

Type: **SU MAKE MB 209.3 192.4 10 250**

11. Assign the temperature values between 192.3° K and 162.8° K to the brightness value 250 in the MB stretch table.

Type: **SU MAKE MB 192.3 162.8 250 250**

12. Verify that the stretch table contains the correct breakpoints.

Type: **SU TABLE MB**

The table MB is listed as shown below.

SU TABLE MB	
BREAKPOINTS	STORED IN TABLE : MB.ST
INPUT	OUTPUT
-----	-----
162.8	250
192.3	250
192.4	250
209.3	10
209.4	10
213.3	10
213.4	75
219.3	75
219.4	156
230.3	156
230.4	117
241.3	117

241.4	167
279.8	102
279.9	102
301.9	0
302	0
330	0
CALIBRATION TYPE : AAA	
CALIBRATION UNITS : TEMP	
BAND NUMBER : -1	
INTERPOLATION TYPE: LIN	

13. Display the first GOES-16 2km IR image on frame 2 centered on Washington, DC, and apply the stretch table MB. Draw a data bar on the image using a label interval of 10. Set the loop bounds from 1 to 2.

Type: **IMGDISP ABI/CONUS.1 2 BAND=14 STA=DCA SU=MB SF=YES REFRESH='BAR (GRA) LINT=10 COLOR=6';LS 1-2**

14. Compare the two images, one without a stretch table applied and the one with, as shown below.

Press: **Alt B**

Press: **Alt A**

MD Files - Basic Concepts

MD files normally contain observational data for a specific time period, for example, a day or an entire year. MD files store the data in individual records for a specific location at a specific time. Each MD file contains individual records. A record contains observational data for a latitude and longitude at a specific time. For example, one record may include measurements of temperature, dew point, wind speed, wind direction, and sea level pressure at 15 UTC for Houston, TX. A single MD file may contain thousands of records.

MD files use the naming convention MDXX $nnnn$ where $nnnn$ is a four-digit number. For example, MDXX0013 is the file name for MD file 13. Most McIDAS commands use only the MD file number. However, you must use the MDXX prefix with the DMAP command or when using Unix commands to copy, move, or delete MD files.

MD file Schemas

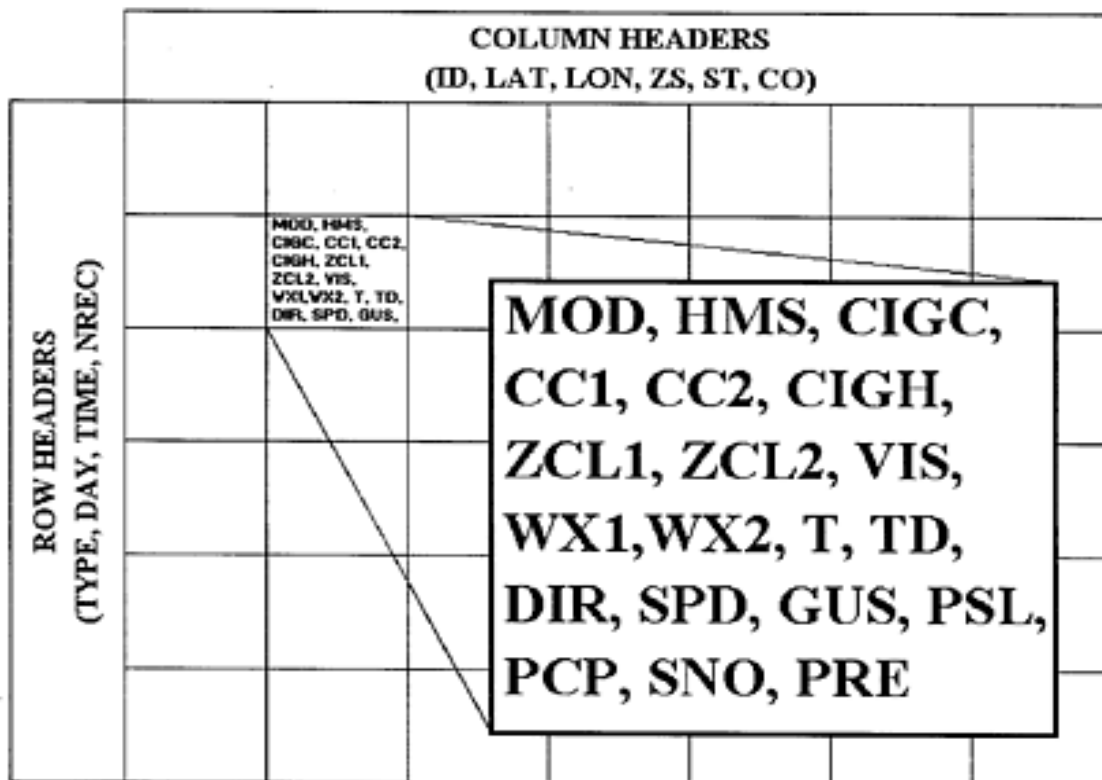
McIDAS stores MD files according to unique templates called *schemas*. SSEC receives real-time data for these nine schemas:

- ISFC (international surface hourly observations)

- IRAB (international RAOB (radiosonde) mandatory levels)
- IRSG (international RAOB (radiosonde) significant levels)
- ISHP (ship and buoy observations)
- PIRP (AIREP and PIREP reports)
- SYN (international surface synoptic observations)

MD files in the ISFC, IRAB, SYN, and GFSMOS schemas are arranged in a table of rows and columns as shown in the ISFC example below. Time and day information common to all records in the row appears in the row header. Similarly, a column header designates common information according to location. Therefore, all the records along a particular row represent the same time and all the records down a particular column are reports from the same location.

MD files in the IRSG, ISHP, and PIRP schemas have row headers, but not column headers because the reporting location changes.



The individual data values within the MD file are stored according to the *parameters* of a schema. Examples of parameters within the ISFC schema are temperature, dew point, and cloud cover. Parameters are used for searching and plotting the MD file data. For each parameter, the schema provides:

- a scaling factor, for example 2, meaning temperatures are stored in Kelvin * 10**2
- the units in which the values are stored, for example Kelvin

Real-time Point Data

The table below lists some real-time point data that is received from typical Noaaport data stream sources, processed in McIDAS-XCD and the associated dataset names.

Schema	ADDE Dataset
PIRP	RTPTSRC/AIRCRAFT
ISFC	RTPTSRC/SFCHOURLY
ISHP	RTPTSRC/SHIPBUOY
SYN	RTPTSRC/SYNOPTIC
IRAB	RTPTSRC/UPPERMAND
IRSG	RTPTSRC/UPPERSIG

Listing MD File Directories and Data Records

In this exercise, you will list an MD file directory and information about its schema, and examine individual records in the MD file.

1. Add and list the point datasets available in the RTPTSRC dataset on the ADDE.UCAR.EDU server.

Type: **DATALOC ADD RTPTSRC ADDE.SSEC.WISC.EDU**

Type: **DSINFO POINT RTPTSRC**

```

DSINFO POINT RTPTSRC

          Dataset Names of Type: POINT in Group: RTPTSRC

Name          NumPos   Content
-----
AIRCRAFT      10      Real-Time Aircraft data
ETAMOS        10      Real-Time ETA Model Output Statistics
GFSMOS        10      Real-Time GFS Model Output Statistics
PROF6MIN      10      Real-Time 6-Minute Profiler data
PROFHOURLY    10      Real-Time Hourly Profiler data
PTSRCs        130     All point data in MDXX files
SFCHOURLY     10      Real-Time SFC Hourly
SHIPBUOY      10      Real-Time Ship and Buoy data
SYNOPTIC      10      Real-Time SYNOPTIC data
TEMPSHIP      10      Real-Time TEMP SHIP data
UPPERMAND     10      Real-Time Upper Air (Mandatory)
UPPERSIG      10      Real-Time Upper Air (Significant)

DSINFO -- done

```

2. List the file directories for the RTPTSRC/SFCHOURLY dataset.

Type: **PTLIST RTPTSRC/SFCHOURLY FORM=FILE ALL**

The dataset position number, schema type, number of rows and columns, and description for each MD file are listed.

3. List all the MD files stored on your workstation.

Type: **DMAP MDXX**

4. List the structure and keys in the ISFC schema.

Type: **LSCHE ISFC**

The keys, units, and scale factors for the ISFC schema are displayed as shown below. If the schema type does not exist, register the schema with the SCHE DCISFC command, and then try the LSCHE command again. See the introduction in the McIDAS User's Guide for more information.

```
LSCHE ISFC
NAME: ISFC  VERSION:  8  DATE: 2018073  TEXTID: "SURFACE HOURLY OBSERVATIONS
-----
DEFAULT NUMBER OF ROWS:  72                INTEGER ID:  0
                   COLS: 12500            MISSING DATA VALUE: -2139062144
REPEAT GROUP:  NUMBER OF REPETITIONS:  1
                   STARTING POSITION:  11
                   SIZE: 25
NUMBER OF KEYS IN ROW HEADER:  4
                   COL HEADER:  6  STARTING AT POSITION  5
                   DATA RECORD: 25  STARTING AT POSITION 11
                   -----
                   35 TOTAL

KEY  SCALE UNIT      KEY  SCALE UNIT      KEY  SCALE UNIT
-----
TYPE  0          DAY  0  CYD      TIME  0  HMS
NREC  0          ID   0  CHAR     LAT   4  DEG
LON   4  DEG     ZS   0  M        ST    0  CHAR
CO    0  CHAR    MOD   0          HMS   0  HMS
CIGC  0          CC1  0          CC2   0
CIGH -2  FT      ZCL1 -2  FT     ZCL2 -2  FT
VIS   1  MI      WX1  0  CHAR     WX2   0  CHAR
T     2  K       TD   2  K        DIR   0  DEG
SPD   1  MPS    GUS   1  MPS     PSL   2  MB
PCP   2  IN     SNO   0  IN      PRE   2  MB
P24   2  IN     WXC1  0  CODE    WXC2  0  CODE
WXC3  0  CODE    WXC4  0  CODE

LSCHE: DONE
```

5. List the parameters in the first file of the BLIZZARD/SFCHOURLY dataset.

Type: **PTLIST RTPTSRC/SFCHOURLY.1 FORM=PARAM**

Also listed are the units, storage type and format for each parameter.

- List the first record in the first file of the dataset.

Type: **PTLIST RTPTSRC/SFCHOURLY.1**

```

PTLIST RTPTSRC/SFCHOURLY.1
Row :      1  Col :      1
TYPE      =          0          | DAY          = 2019259 CYD          |
TIME      =          0 HMS      | NREC         =      8963          |
ID        =      KEFK          | LAT          = 44.8888 DEG          |
LON       = 72.2125 DEG        | ZS           =      283 M          |
ST        =          VT        | CO           =          US          |
MOD       =          0          | HMS         = 235500 HMS          |
CIGC     =          3          | CC1         = _missing_          |
CC2      = _missing_          | CIGH        = 2700. FT           |
ZCL1     = _missing_          | ZCL2        = _missing_          |
VIS       = 10.0 MI           | WX1         = _missing_          |
WX2      = _missing_          | T           = 286.66 K            |
TD        = 283.26 K           | DIR         = 290 DEG             |
SPD       = 3.1 MPS            | GUS         = _missing_          |
PSL       = 1018.62 MB         | PCP         = _missing_          |
SNO       = _missing_          | PRE         = _missing_          |
P24       = _missing_          | WXC1        = _missing_          |
WXC2     = _missing_          | WXC3        = _missing_          |
WXC4     = _missing_          |
-----
Number of matches found = 1
PTLIST: Done

```

Searching MD Files

The SFCLIST and UALIST commands list out weather observations in an easy to read tabular format. The default for these commands is to list the observations from the current day and time from the real-time datasets. By specifying the TIME, DAY, and DATASET keywords, you will be able to access the Storm of the Century data in the BLIZZARD dataset. If you need to specify which parameters to list, or want to change the format of the output extensively, you will have to use the PTLIST command.

In this exercise, you will use the SFCLIST and UALIST commands to list out weather data for Cape Hatteras, North Carolina, as well as for the entire state of North Carolina. You will then use the PTLIST command to format the data more specifically.

- List 12 hours of surface hourly data for Madison, WI for today (DAY=#Y uses string replacement for the current day). The default dataset used is RTPTSRC/SFCHOURLY.

Type: **SFCLIST KMSN TIME=0 12 DAY=#Y**

- List the surface hourly data for North Carolina at 18 UTC, including the windchill, heat, and precipitation information.

Type: **SFCLIST NC TIME=18 DAY=#Y OPT=CHILL HEAT PRECIP**

3. List the upper-air data from 00 UTC at Green Bay, WI, the default dataset is RTPSRC/UPPERMAND and RTPSRC/UPPERSIG.

Type: **UALIST KGRB 00 #Y**

4. List the station identifier (ID), temperature (T), and dew point (TD) of the matching observations for North Carolina at 15 UTC today. Display the temperature and dew point in Fahrenheit. The PARAM keyword defines what data types in the matching records to list and the SELECT keyword defines which records to match. You must surround the SELECT variable with single quotes and separate individual select clauses with semicolons.

Type: **PTLIST RTPSRC/SFCHOURLY.9 PARAM=ID T[F] TD[F] SELECT='ST NC;TIME 15'
FORMAT=X I6 I6 NUM=ALL**

Copying and Deleting MD Files

In this exercise, you will copy MD file data from one file to another on the workstation and delete an MD file.

1. Use the DMAP command to see if there are any local MD files in the range 4000 to 4005.

Type: **DMAP MDXX400**

If you already have MD files in this range, select another range of 6 files you can use. (try 4010-4015, then 4020-4025, etc).

2. Create a local dataset name for the test MD files in the MYDATA group.

Type: **DSSERVE ADD MYDATA/TEST-PTSRCs MD 4000 4005**

3. Copy the contents of RTPSRC/SFCHOURLY.9 to MYDATA/TEST-PTSRCs.1 (MD file 4000), and change the file description to CASE STUDY.

Type: **PTCOPY RTPSRC/SFCHOURLY.9 MYDATA/TEST-PTSRCs.1 DEL=YES
TITLE='CASE STUDY'**

4. Once the PTCOPY is done, list the MD file headers to verify that the MD file was copied.

Type: **PTLIST RTPSRC/SFCHOURLY.9 FORM=FILE**
Type: **PTLIST MYDATA/TEST-PTSRCs.1 FORM=FILE**

Plotting and Contouring Point Data

The SFCCON, SFCPLOT, RAOBCON, and RAOBPLOT commands display surface and upper-air point data. These commands are designed to allow for shorter command entries by having commonly used defaults for

displaying their data types, and using positional parameters instead of keywords for values that are always or often specified.

For example, the default for the DATASET keyword is to use the real-time datasets. When looking at real-time data you will never have to specify the DATASET keyword when using McIDAS-XCD default named datasets.

These commands are easier to use for plotting or contouring their specific data types than the generic "toolbox" commands PTDISP and PTCAN. However, the "toolbox" commands are still useful for calculating complicated mathematical equations and for creating output that is completely controlled by the user instead of by the command defaults.

In this exercise, you will plot and contour observed parameters, as well as calculate the changes in these parameters over time and between different levels in the atmosphere. You will also plot some derived parameters and plot the results of your own mathematical equation.

Displaying Surface Data

1. Erase both the images and graphics in frames 1 through 6.

Type: **ERASE F 1 6**

2. Display the two latest GOES-16 band 2 visible images on frames 1 and 2 centered on Florence, South Carolina, with the resolution reduced by a factor of 2. Show each frame and add a map in graphics color level 5.

Type: **IMGDISP RTGOESR/CONUSC02.-2 1 STA=KFLO MAG=-2 SF=YES
REFRESH='MAP H 5'; IMGDISP RTGOESR/CONUSC02.-1 2 STA=KFLO MAG=-2 SF=YES
REFRESH='MAP H 5'**

3. Show frame 1 and plot the temperature on the satellite image using data from the RTPTSRC/SFCHOURLY dataset. The **X** in the *map* positional parameter means to use the default, which is to use the frame's current navigation.

Type: **SF 1;SFCPLOT T X**

4. Contour the temperature over the plot using the same data and conditions as in the previous step.

Type: **SFCCON T X**

5. On frame 2, contour the pressure change from 1 UTC to 2 UTC.

Type: **SFCCON PRE X 1-2 GRA=2;SF 2**

Displaying Upper-air Data

1. Display a map of the United States in color level 8 (gray) on frames 3 and 4.

Type: **SF 3;MAP USA 8 GRA=3-4**

2. Contour the 850-500 mb thickness on frame 3.

Type: **RAOBCON Z 500-850 X 12 #Y**

3. Plot the 850 mb streamlines over the temperature contours in red.

Type: **RAOBCON STREAML 850 X 12 #Y COL=5**

Displaying Calculated Parameters

In addition to plotting the observed parameters, these commands can also calculate and plot some derived parameters, such as windchill, vorticity, and temperature advection. If you have a more complicated equation that you would like to use, you can use the PTCON or PTDISP commands to plot the results of your equation.

1. Contour the 850 mb temperature advection on frame 5, using a contour interval of 5.

Type: **SF 5;RAOBCON TADV 850 USA 12 #Y CINT=5**

2. Compare frames 4 and 5.

Press: **Alt B**

Press: **Alt A**

Notice that the areas of positive temperature advection (warm air advection) calculated and plotted in frame 5 match the areas in frame 4 where streamlines are passing from warmer to colder air.

3. Use the Hypsometric Equation to calculate the 1000-500 mb thickness in meters.

Type: **PTDISP RTPTSRC/UPPERMAND 6 SEL='DAY #Y;TIME 12' MAP=USA
PARAM='THIK[M]=(287/9.8)*((P1+P2)/2)*LOG(1000/500)' P1=T[K] 'P 1000' P2=T[K] 'P 500' ;
SF 6**

Only stations reporting both a 1000 mb and 500 mb temperature are included in the equation, so stations that are above 1000 mb (e.g. Rocky Mountain stations) will not appear in the display.

Grids and Grid Files - Basic Concepts

Grids

A grid is a lattice of regularly spaced data points superimposed on a projection of the earth. Grids can store numerical model data and serve as an interface between observational data stored in MD files and analyzed data displayed as contours on a frame. To contour observational data, compute derived parameters, and display manually digitized radar (MDR) data, etc., the data must first be interpolated from the observing locations to a uniform latitude, longitude lattice. McIDAS stores this geographic lattice as a grid.

Grid Files

Grids are stored in grid files, GRIB files or netCDF files. This manual will use a local grib file and real time grid servers in the real time RTGRIDS dataset. Grid files use the naming convention GR##### where ##### is the six-digit file number. If the file number is less than six digits, I and D replace the first two digits and zeros precede the number. For example, GR111111 is the name of the file for grid

111111, GRI90112 is the name for grid file 90112, and GRID0013 is the name for grid file 13. In ADDE, sequences of grid files are grouped together in a dataset.

Manipulating Grids and Grid Files

The GRDLIST and GRDCOPY commands manipulate individual grids or groups of grids. GRDLIST can list grids, while GRDCOPY can copy, move, and perform mathematical operations on grids. These operations include adding or subtracting two grids, filling a grid with a constant value, or redefining all values within a grid that are less than a specified number to be a new number (e.g., all values less than zero are set to zero). If you have u and v component grids, you can create a vorticity or divergence grid, or use the grid components to advect a parameter in another grid.

The GRDINFO command provides information about grids. You can use it to list information about the grid header and statistics on the grid data, or list data points.

The GRDLIST and GRDCOPY commands can also manipulate grid files. With these commands you can create, list, or copy a grid file.

Grids and Grid Files – Listing Grids and Grid Files

1. List the grid files in the RTGRIDS dataset.

Type: **DATALOC ADD RTGRIDS ADDE.UCAR.EDU; DSINFO GRID RTGRIDS**

2. List the grid file directory for these grid files.

Type: **GRDLIST RTGRIDS/GFS.ALL FORM=FILE**

Each grid file has a dataset position, date the grid file was created, maximum number of grids the grid file can store, and a file description, as shown below.

3. Setup a local dataset that accesses a local grib file and list the first ten grids in the first grid file in the dataset. Access the grib file on your local machine in the *Data/Grid* directory.

Example file name:

GFS_CONUS_80km_20190903_1200.grib1

Type: **DSSERVE ADD GRIB/GFS GRIB TYPE=GRID DIRFILE='<local-path>/Data/Grid/GFS_CONUS***

Type: **GRDLIST GRIB/GFS.1 NUM=10**

The first ten grids are listed in the Text and Command Window, which shows the dataset position, grid file description, parameter type, level, Julian date of the data, time of the data, source type, forecast hour, valid day/time, grid number, and projection.

Dataset position 1				Directory Title= /GFS-GLME0P5D.96.2018098.1200.12						
PAR	LEVEL	DAY	TIME	SRC	FHR	FDAY	FTIME	GRID	PRO	
U	TRO	RSRV	08 APR 18098 12:00:00	GFS	12	09 APR 18099 00:00:00		1	MERC	
U		1 HPA	08 APR 18098 12:00:00	GFS	12	09 APR 18099 00:00:00		2	MERC	

Z	2	HPA	08	APR	18098	12:00:00	GFS	12	09	APR	18099	00:00:00	3	MERC
T	3	HPA	08	APR	18098	12:00:00	GFS	12	09	APR	18099	00:00:00	4	MERC
V	1	HPA	08	APR	18098	12:00:00	GFS	12	09	APR	18099	00:00:00	5	MERC
V	3	HPA	08	APR	18098	12:00:00	GFS	12	09	APR	18099	00:00:00	6	MERC
RH	3	HPA	08	APR	18098	12:00:00	GFS	12	09	APR	18099	00:00:00	7	MERC
OZMX	5	HPA	08	APR	18098	12:00:00	GFS	12	09	APR	18099	00:00:00	8	MERC
RH	7	HPA	08	APR	18098	12:00:00	GFS	12	09	APR	18099	00:00:00	9	MERC
U	10	HPA	08	APR	18098	12:00:00	GFS	12	09	APR	18099	00:00:00	10	MERC
Number of grids listed = 10														

- Get the expanded listing of the first grid in this dataset.

Type: **GRDLIST GRIB/GFS.1 FORM=ALL**

The expanded listing includes information about the total points, the number of rows and columns, the time the grid was received, units of the parameter, scaling factor, lat/lon extents and increments.

Dataset position 1														
Directory Title= /GFS-GLME0P5D.96.2018098.1200.12														
PAR	LEVEL	DAY	TIME	SRC	FHR	FDAY	FTIME	GRID	PRO					
U	TRO	RSRV	08 APR 18098 12:00:00	GFS	12	09 APR 18099 00:00:00		1	MERC					
Total pts= 259920 Num rows= 361 Num columns= 720								received:	0 000000Z					
u-component of the wind														
GRIB ID numbers: Geographic = N/A ; PAR =N/A; Model ID = 96;														
Units of gridded variable are MPS Scale of variable is: 2														
Mercator Projection														
Min Lat= -90.00 Max Lat= 90.00 Min Lon= 0.50 Max Lon= 0.00														
Latitude Increment= 0.5000 Longitude Increment= 0.5000														
Number of grids listed = 1														
GRDLIST - done														

- List all the 500 mb height (Z) grids in the grib file.

Type: **GRDLIST GRIB/GFS LEVEL=500 PARAM=Z FHOURL=12 NUM=ALL**

The 12 hour forecast grid for this level and parameter is listed.

- List the statistical information for the 12 hour 500 mb height grid in this dataset.

Type: **GRDINFO GRIB/GFS STAT LEVEL=500 PARAM=Z FHOURL=12**

Information about the maximum and minimum values, the mean and standard deviation of the values in the grid are listed.

- List a histogram of the values for this same grid. Bin the values in 120 m segments and format the values as integers.

Type: **GRDINFO GRIB/GFS HIST LEVEL=500 PARAM=Z FHOURL=12 BINSIZE=120 FORMAT=I5**

- List the 24 hour Relative Humidity grid point values between 35° and 40° N and 80° and 85° W from this dataset. Format the data as a real value with one decimal place.

Type: **GRDINFO GRIB/GFS LIST PARAM=RH LAT=35 40 LON=80 85 FORMAT=F4.1**

Row, column, latitude, longitude, and values are listed for each grid point falling in this geographical region

Copying Grids and Grid Files

Next, you will copy grids and grid files.

- Use the DMAP command to see if there are any local grid files in the range 4000 to 4009.

Type: **DMAP GRID400**

If you already have grids in this range, select another range of 10 files you can use. (try 4010-4019, then 4020-4029, etc).

- Create a local dataset name for the test grids in the MYDATA group and add an alias for the dataset. Use the grid numbers for the first free range you located in the previous step.

Type: **DSSERVE ADD MYDATA/TEST-GRIDS GRID 4000 4009 "Test grids**

Type: **AKA ADD TG MYDATA/TEST-GRIDS**

- Copy 20 parameters of the 0Z current day F HOUR=24 model run for from the RTGRIDS/GFS dataset to the first position in the TEST-GRIDS dataset. Set the description of the destination grid file to "LEARNING HOW TO COPY GRIDS".

Type: **GRDCOPY RTGRIDS/GFS TG.1 DEL=YES DAY=#Y TIME=0 F HOUR=24 NUM=20 TITLE='LEARNING TO COPY GRIDS'**

- List grids 5 through 10 in dataset TG.1.

Type: **GRDLIST TG.1 GRID=5 10**

Dataset	position	4000	Directory	Title=							
PAR	LEVEL	DAY	TIME	SRC	FHR	FDAY	FTIME	GRID	PRO		
T	500 MB	17 SEP 19260	00:00:00	GFS	24	18 SEP 19261	00:00:00	5	LAMB		
T	60 MBAG	17 SEP 19260	00:00:00	GFS	24	18 SEP 19261	00:00:00	6	LAMB		
Z	350 MB	17 SEP 19260	00:00:00	GFS	24	18 SEP 19261	00:00:00	7	LAMB		
WP	150 MB	17 SEP 19260	00:00:00	GFS	24	18 SEP 19261	00:00:00	8	LAMB		

```

T          900 MB   17 SEP 19260 00:00:00  GFS   24 18 SEP 19261 00:00:00    9 LAMB
RH         925 MB   17 SEP 19260 00:00:00  GFS   24 18 SEP 19261 00:00:00   10 LAMB
Number of grids listed = 6
GRDLIST - done

```

Creating Grids and Grid Files

In this exercise, you will create a new grid dataset, copy and contour grids, create advection and divergence grids, and list the grids and grid files.

1. Generate a local grid dataset in position 2 of the TEST-GRID dataset with a maximum of 100 grids, and the description ADVECTION AND DIVERGENCE GRIDS. Copy in the 12 hour 1000 mb forecast grids from the GRIB/GFS dataset.

Type: **GRDCOPY GRIB/GFS TG.2 DEL=YES LEVEL=1000 F HOUR=12
TITLE='ADVECTION AND DIVERGENCE GRIDS' MAXGRD=100 NUM=ALL**

If you don't specify a maximum number of grids, the grid file is created with space for 1000 grids.

2. List the new dataset.

Type: **GRDLIST TG.2 NUM=ALL**

The dataset should contain 9 grids.

3. Check to see that TG.2 contains the grids required to calculate divergence.

Type: **GRDLIST TG.2 DER=DVG**

If the grid can be created, the following output will be shown:

```

GRDLIST TG.2 DER=DVG
PARAM  LEVEL      DAY      TIME      SRC  F HOUR      F DAY      F TIME  GRID  PRO
-----
DVG    1000 HPA 08 APR 18098 12:00:00  GFS   12 09 MAR 18099 00:00:00  N/A  MERC
Number of grids listed = 1
GRDLIST - done

```

Notice the GRID number is listed as "MATH" or "N/A". This indicates that there is no actual divergence grid but that the component grids required to create it all exist within the TG.2 grid file. See the help sections of the individual commands for more information on the common meteorological parameters you can calculate from grids by using the DERIVE keyword.

If the grid cannot be created, an error will be produced.

4. Create a divergence grid from from the grids in the dataset.

Type: **GRDCOPY TG.2 TG.2 DER=DVG**

5. List the contents of the dataset to verify that a divergence grid was added.

Type: **GRDLIST TG.2 NUM=ALL**

The new grid was filed into grid 10, which is the first available grid.

- List information about the divergence grid:

Type: **GRDINFO TG.2 PARAM=DVG**

- Create a temperature advection grid using the U, V and T grids. Use the formula $TADV = -(u*DDX(T) + v*DDY(T))$. Label the new grid TADV and set the units to K/sec.

Type: **GRDCOPY TG.2 TG.2 G1='PARAM T' G2='PARAM U' G3='PARAM V' MATH='- (G2*(DDX(G1))+G3*(DDY(G1)))' NEWPAR=TADV K/S**

The G1, G2, ..., Gn keywords are used to define the grids you want to use. Since all the grids in this dataset are from the same level, model run and forecast hour, just specifying the parameter defines the grid uniquely. You can use additional search criteria (ex: LEVEL, F HOUR) to further refine your selection. The MATH keyword defines the mathematical operation to perform on the grids specified with the Gn keywords. See the help sections of the individual commands for more information on the use of these keywords.

- List information about the temperature advection grid:

Type: **GRDINFO TG.2 PARAM=TADV**

- OR, if you would like to get statistical information on a grid created with a mathematical equation, but won't need to actually display the grid, you can run the GRDINFO command with the MATH keywords included.

Type: **GRDINFO TG.2 G1='PARAM T' G2='PARAM U' G3='PARAM V' MATH='- (G2*(DDX(G1))+G3*(DDY(G1)))' NEWPAR=TTAD K/S**

This should list the same information as the previous GRDINFO command, except with a different parameter name.

Displaying Gridded Data

In this exercise, you will use the GRDDISP command to display grids. GRDDISP can be used to draw contours, streamlines, or plots of grid data.

- Erase all frames, show frame 4, and display a GOES-16 0.5km Visible image centered on Washington DC. Add a high resolution map.

Type: **ERASE F 1 6;SF 4;IMGDISP ABI/CONUS.1 4 STA=DCA MAG=-8 BAND=2;MAP H**

- Contour the divergence grid created in the previous lesson over the satellite image with a contour interval of 5 and graphics color level 3. Because the values of divergence are small, use the POWER

keyword to scale the values. Make the convergence (negative divergence) areas dashed. Use NAV=C to use the navigation in the current frame.

Type: **GRDDISP TG.2 PARAM=DVG NAV=C CINT=5 COLOR=3 POWER=6 DASH=NEG**

3. Contour the temperature advection grid over the image using a contour interval of five degrees per sec scaled by $10^{**}5$. Make negative values (cold air advection) dashed.

Type: **GRDDISP TG.2 PARAM=TADV NAV=C CINT=5 POWER=5 DASH=NEG**

4. Create a plot on frame 1 of the 12 hour 500 mb height forecast over the U.S. from the model run in the local TG.1 dataset. Use a label size of six pixels, a 60 m contour interval and label every other contour. Switch to the frame after plotting.

Type: **GRDDISP TG.1 F HOUR=24 LEV=750 PARAM=Z MAP=USA PRO=MERC GRA=1 LSIZE=6 CINT=60 LINT=2 SF=YES**

5. Overlay the 500 mb temperatures in degrees C in yellow. Dash the contours and use a label size of six pixels and a contour interval of 5.

Type: **GRDDISP TG.1 DAY=17/SEP/19 TIME=0 F HOUR=24 LEV=500 PARAM=T UNIT=C NAV=C LSIZE=6 COLOR=3 CINT=5 DASH=ALL**

6. Create a plot of wind barbs over the US at the 250 mb level using today's 0 HR forecast from the RTGRIDS/GFS real time dataset. Draw the barbs in yellow with a size of 12 on frame 2 and show the frame after the plotting is complete.

Type: **GRDDISP TG.1 DAY=#Y TIME=0 F HOUR=12 LEV=250 PARAM=WINDB MAP=USA LSIZE=12 COLOR=3 GRA=2 SF=YES**

Notice that the density of the grid points and the size of the wind barbs makes them overlap.

7. Erase the frame and redo the previous plot, this time plotting only every other grid point.

Type: **ERASE G;GRDDISP TG.1 DAY=#Y TIME=0 F HOUR=12& LEV=250 PARAM=WINDB MAP=USA LSIZE=12 COLOR=3 PINT=2 2**

The PINT keyword can be used to declutter a dense grid plot.

In this exercise, you will display the forecasted sea level pressures for 12Z on September 3, 2019. Then, you will compute the 1000-500 mb thickness for that same time and display the thickness on top of the sea level pressure contours.

8. List the available sea level pressures from the September 3, 2019 12 UTC GFS grib file.

Type: **GRDLIST GRIB/GFS NUM=ALL TIME=12 DAY=19246 PAR=P LEV=MSL**

9. Erase the graphics in the first five frames.

Type: **ERASE G 1 5**

10. Display the sea level pressure data from the 12 UTC GFS model run. Plot the contours over a map of the United States.

Type: **GRDDISP GRIB/GFS TIME=12 DAY=19246 PAR=P LEV=MSL MAP=USA F HOUR=48**

11. Calculate and display the 1000-500 mb thickness for the same data.

Type: **GRDDISP GRIB/GFS MAP=USA F HOUR=48 MATH='G2-G1' COLOR=3
NEWPAR=THCK GPM G1='LEV 1000;PAR Z' G2='LEV 500'**

Using WXTLIST to read weather text products

In this lesson, you will be using the WXTLIST command to read various weather text products.

1. Add a real time text data server. List the available predefined text products.

Type: **DATALOC ADD RTWXTEXT ADDE.SSEC.WISC.EDU**

Type: **WXTLIST DIR**

This will produce a listing of all the predefined text products available in the group RTWXTEXT.

2. List the five most recent Severe Weather Statements, using one of the predefined text products.

Type: **WXTLIST SEVERE_WX_STATEMENT NUM=5**

WXTLIST relies on keywords to refine the search for text products. The NUM keyword indicates how many reports you want listed. The WSTN keyword can be used to match the station initiating the report.

3. List the most recent zone forecast for Madison, Wisconsin.

Type: **WXTLIST APRO=ZFP ASTN=MKX MATCH=MADISON**

The APRO keyword matches the product header. The ASTN keyword matches the station initiating the report. The MATCH keyword looks for any occurrence of the word MADISON in the reports.

Listing raw observational text reports

There are five macro commands that list observations and forecasts:

- MOSRPT (to list MOS forecasts from the GFS, NAM and NGM models)
- RAOBRPT (to list upper air observations)
- SFCRPT (to list surface hourly observations)
- SYNRPRT (to list synoptic observations)
- TAFRPT (to list terminal aerodrome forecasts)

These commands all utilize the functionality of the OBSRPT command.

1. List the upper air observations for Arizona and New Mexico for the past 24 hours.

Type: **RAOBRPT AZ NM 24**

2. List the surface hourly observations for the past 12 hours in Phoenix, Arizona.

Type: **SFCRPT KPHX 12**

This data is all listed in the raw report format.

Real-time Data Access – Watches and Warnings

In this exercise, you will list and display severe thunderstorm, tornado and winter storm watches and warnings, as well as blizzard warnings, snow advisories, and flood watches and warnings.

3. Display a map of the United States with all of today's watches, warnings, and advisories plotted.

Type: **WWDISP USA TIME=0 23:59**

4. Find a state that has a box plotted and list the reports for that state.

Type: **WWLIST MS TIME=0 23:59**

Replace MS with whatever state you would like listed.

5. Output the actual reports for the state.

Type: **WWLIST MS TIME=0 23:59 OUT=RAW**

This will output the actual textual report that was issued by the National Weather Service.

Real-time Data Access - Meteorological Diagrams

In this exercise, you will create four types of meteorological diagrams: a Skew-T diagram, a hodograph, a vertical cross section and a meteorogram.

1. Erase frames 1 through 6.

Type: **ERASE F 1 6**

2. List all of the upper-air stations reporting a 00 UTC observation today. The output is sorted alphabetically by state and then by ID.

Type: **STNLIST CO=US DAT=RTPTSRC/UPPERMAND TIME=00**

3. Choose one of the stations listed. Show frame 1 and plot a Skew-T diagram for your chosen station (in this case, 72645, which is Green Bay, Wisconsin) at 00 UTC.

Type: **SF 1;UAPLOT 72645 00**

4. Run the same command, but this time activate the interactive Skew-T.

Type: **UAPLOT 72645 00 EDIT=YES**

Follow the instructions in the command output in the text window to change the temperature or dewpoint sounding. Notice the sounding analysis parameters change when you make the atmosphere more or less stable.

5. Show frame 2 and plot a hodograph of the same station.

Type: **SF 2;HODO 72645 00**

6. Show frame 3 and display a map of the upper-air stations reporting a 00 UTC observation.

Type: **SF 3;RAOBPLOT IDN X MID 00**

7. Choose the two stations that are the farthest apart on the map. Show frame 4 and display a vertical cross section of theta, wind, and mixing ratio values from these two stations (in this case, International Falls, Minnesota (72747) to Nashville, Tennessee (72327). The MAP=YES keyword displays a map in the upper-right corner showing the cross section location and stations selected.

Type: **SF 4;UACROSS 72747 72327 MAP=YES TIME=00 PAR=THA WINDB MIX**

8. Show frame 5 and plot a 24-hour surface meteorogram for Macon, Georgia.

Type: **SF 5;SFCMG KMCN**

The default dataset in the SFCMG command is RTPTSRC/SFCHOURLY, so the command automatically accesses the real-time data on the remote server. You do not need to have the data on your local machine.

McIDAS-X Advanced Workshop

September 2023 (software version 2022.1)

In existence since 1973, McIDAS (Man computer Interactive Data Access System) is a suite of sophisticated software packages that perform a wide variety of functions with satellite imagery, observational reports, numerical forecasts, and other geophysical data. Those functions include displaying, analyzing, interpreting, acquiring and managing the data.

McIDAS-X is supported for the current GOES GVAR and upcoming GOES-R satellite series (currently estimated as being in service until 2036), with no end date in sight.

In this McIDAS-X tutorial, some exercises will be completed using different methods of data access: local data files and real-time access to default remote servers. If you have access to your own real-time ADDE servers, you may also use those, but be aware that different server configurations may make the explanations in this document not quite applicable to all data that you may load. A source for free data served via ADDE is Unidata's ADDE.SSEC.WISC.EDU server.

This tutorial assumes that you have McIDAS-X installed on your machine, and that you know how to start McIDAS-X.

GOES-R ABI Band List

```

McIDAS-X 2022.1: jayh@panther.ssec.wisc.edu
IMGLIST EAST/CONUS FORM=BAND
Image file directory listing for:EAST/CONUS
Pos Satellite/      Date      Time      Center      Res (km)      Image_Size
  sensor
-----
30 GOES-16          18 SEP 23261 01:51:17    30   87
Band: 1    0.47 um VIS Daytime aerosols over land    1.27  1.07  3000 x 5000
Band: 2    0.64 um VIS Clouds,fog,insolation,wind  0.64  0.54  6000 x10000
Band: 3    0.86 um NIR Veg/burn scar,aerosols,wind  1.27  1.07  3000 x 5000
Band: 4    1.37 um NIR Cirrus clouds                2.55  2.13  1500 x 2500
Band: 5    1.6 um NIR Cloud phase,snow              1.27  1.07  3000 x 5000
Band: 6    2.2 um NIR Land/cloud,vegetation,snow     2.55  2.13  1500 x 2500
Band: 7    3.9 um IR Sfc,cloud,fog,fire,wind        2.55  2.13  1500 x 2500
Band: 8    6.2 um IR Hi-level WV,wind,rainfall      2.55  2.13  1500 x 2500
Band: 9    6.9 um IR Mid-level WV,wind,rainfall     2.55  2.13  1500 x 2500
Band: 10   7.3 um IR Lower-level WV,wind,S02        2.55  2.13  1500 x 2500
Band: 11   8.4 um IR Total WV cloud phase,dust      2.55  2.13  1500 x 2500
Band: 12   9.6 um IR Total ozone,turbulence,wind    2.55  2.13  1500 x 2500
Band: 13  10.3 um IR Surface and cloud            2.55  2.13  1500 x 2500
Band: 14  11.2 um IR Imagery,SST,clouds,rainfall   2.55  2.13  1500 x 2500
Band: 15  12.3 um IR Total water,ash,SST          2.55  2.13  1500 x 2500
Band: 16  13.3 um IR Air temp,cloud hgt and amt    2.55  2.13  1500 x 2500
IMGLIST: done
IMA GRA Bounds Switches
10 10 1-100
Date Time T
18 Sep 2023261 01:59:40 1

```

Creating Local Datasets

You can access GOES-R Series netCDF format files with the IMG* commands by using DSSERVE to assign a dataset name to the files. When doing so, specify "ABIN" in the *format* parameter, TYPE=IMAGE, and the directory and file masks in the DIRFILE keyword. The files must be mission-standard Level 1b or Level 2 files in netCDF-4 format, like those from NOAA CLASS or from the GRB data stream after they've been decoded with CSPP Geo or other software. The names of files should look similar to *ABI-L1b-RadC-M3C01_G16_s2015229195720.nc* (current CSPP Geo naming convention) or *OR_ABI-L1b-RadC-M4C16_G16_s20151702215532_e20151702220362_c20151702220394.nc* (GOES-R Ground System naming convention).

4. Create a local dataset to access the ABI netCDF files on your local machine in the *Data/ABI/NetCDF* directory.

```

OR_ABI-L1b-RadC-M3C02_G16_s20180041657204_e20180041659577_c20180041700012.nc
OR_ABI-L1b-RadC-M3C14_G16_s20180041657204_e20180041659577_c20180041700023.nc
OR_ABI-L2-TPWC-M3_G16_s20180041657204_e20180041659577_c20180041701161.nc

```

- a. Run the DSSERVE command to access the Level 1b netCDF imagery files.

```

DSSERVE ADD ABI/CONUS ABIN TYPE=IMAGE DIRFILE= '<local-
path>/Data/ABI/NetCDF/*RadC*'

```

- b. Use the IMGLIST command to list the most recent image in the dataset.

IMGLIST ABI/CONUS

- c. Datasets can be organized however the user chooses. Run DSSERVE to include all of the files in the data directory.

DSSERVE ADD ABI/DATA ABIN TYPE=IMAGE DIRFILE= '<local-path>/Data/ABI/NetCDF/*'

- d. IMGLIST can list the image directory and band information with the FORM=ALL keyword. Your IMGLIST command should match the image.

IMGLIST ABI/DATA.ALL FORM=ALL

```

IMGLIST ABI/DATA.ALL FORM=ALL
Image file directory listing for:ABI/DATA
Pos Satellite/      Date      Time      Center      Res (km)      Image_Size
  sensor
-----
 1 G-16 IMG          4 JAN 18004 16:57:20   30  87
Band: 2      0.64 um VIS clouds fog, insol, winds      0.50  0.50  6000 x10000
Band: 14     11.2 um IR Imagery,SST,clouds,rainfall      2.00  2.00  1500 x 2500
proj: 0 created: 2018004 165720 memo: GOES-R ConUS
type:ABIN    cal type:RAW
offsets: data= 1280 navigation= 256 calibration= 768 auxiliary= 0
doc length: 0 cal length: 0 lev length: 0 PREFIX= 0
valcod:      0 zcor: 0 avg-smp: A
lcor: 1 ecor: 1 bytes per pixel: 2 ss:186
Resolution Factors (base=1): Line= 4.0 Element= 4.0
 2 G-16 PRD          4 JAN 18004 16:57:20   30  87
Band: 13     L2: TPW - Total Precipitable Water      2.00  2.00  300 x 500
proj: 0 created: 2018004 165720 memo: GOES-R ConUS
type:ABIN    cal type:RAW
offsets: data= 1280 navigation= 256 calibration= 768 auxiliary= 0
doc length: 0 cal length: 0 lev length: 0 PREFIX= 0
valcod:      0 zcor: 0 avg-smp: A
lcor: 1 ecor: 1 bytes per pixel: 2 ss:187
Resolution Factors (base=1): Line= 20.0 Element= 20.0
IMGLIST: done

```

- e. Another IMGLIST keyword, FORM=BAND, lists the band and resolution without the file's expanded listing.

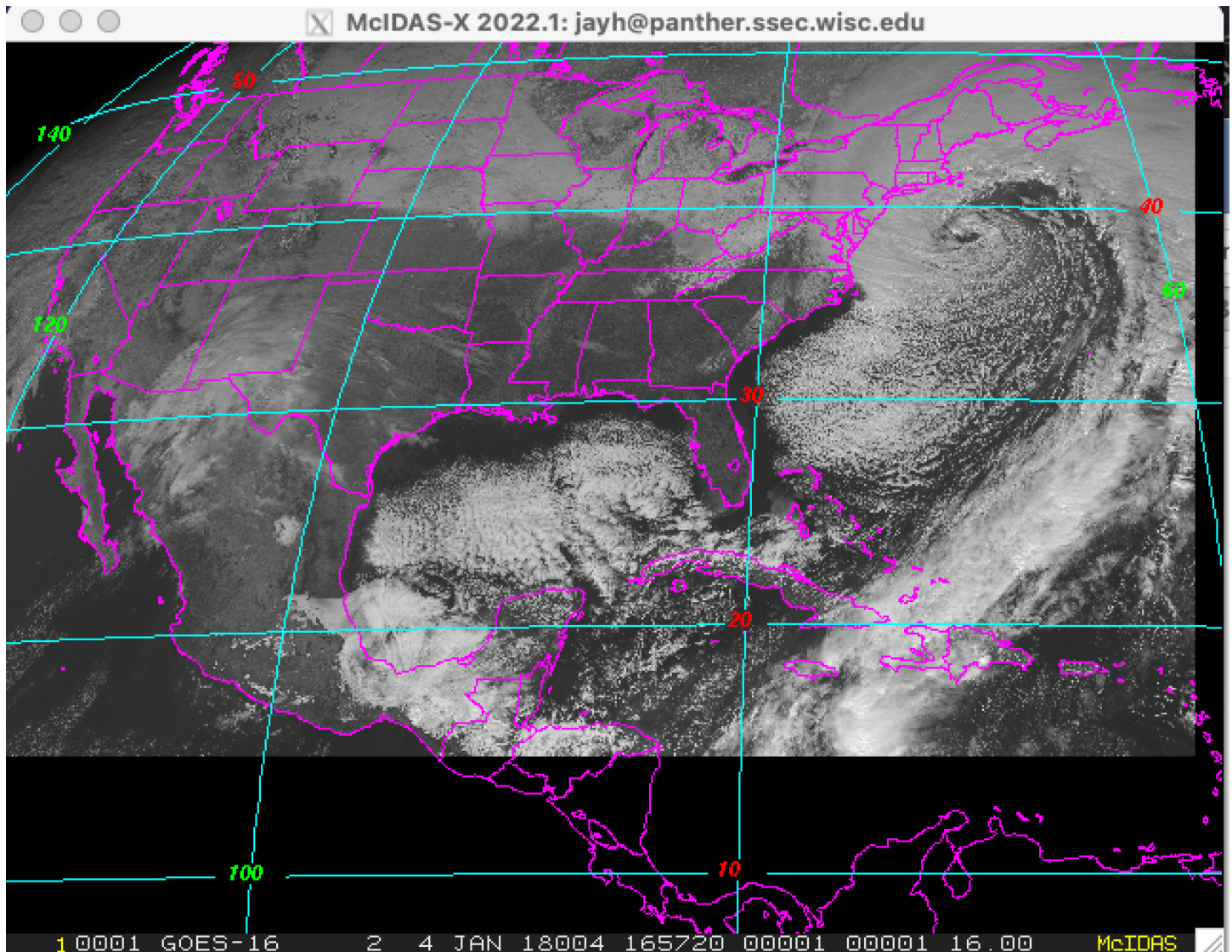
IMGLIST ABI/DATA.1 BAND=2 FORM=BAND

5. The data can be displayed and inspected with the IMGDISP and IMGPROBE commands.
 - a. Run this series of commands to display BAND 2 of an ABI Level 1b file, display a navigated map over the image, place the cursor at the center of the image and run an IMGPROBE command to inspect the pixel values under the cursor.

```

IMGDISP ABI/CONUS BAND=2 MAG= -16
MAP X X LALO
PC C
IMGPROBE MODE=N

```



- b. Display BAND 14 with a yellow map on a new frame which is automatically displayed with the SF=YES keyword. Move the cursor to a latitude/longitude point (29° N, 91° W), and list the statistics of the pixel values under the cursor.

```

IMGDISP ABI/CONUS 2 BAND=14 LATLON=29 91 SF=YES ; MAP X 3
PC E 29.4 90.7 ; IMGPROBE STAT BOX TEMP

```

Move the cursor around and press alt-g or the right mouse button over desired features. Be sure to quit each interactive IMGPROBE command with *Alt-Q* or by clicking the right and center mouse buttons.

- c. Display the Total Precipitable Water Level 2 product file in the local dataset by specifying the absolute position number of the data. Use the corresponding L2 product enhancement file to color the image. Plot a map and inspect the data in the middle of the frame with the IMGPROBE command.

```
IMGDISP ABI/DATA.2 3 EU=L2-TPW SF=YES ; MAP  
PC C ; IMGPROBE
```

Note: In McIDAS-X 2018.1, 22 L2-*.ET files were added to the McIDAS-XRD distribution. The calibration and related L2-*.ET files were updated to best match the color enhancements of AWIPS II. When McIDAS-XRD is installed, these files are added to the /home/mcidas/data directory and are named with the corresponding L2 product name. (e.g., L2-ACHA.ET, L2-TPW.ET, L2-DSI-CAPE.ET, etc.) To list the files run the DMAP command.

```
DMAP L2-*.ET
```

Accessing Remote ADDE Datasets

6. To access a remote ADDE server with Level 1b ABI netCDF image files, follow these steps.
 - a. Modify your client routing table with the DATALOC command so that commands requesting data from a dataset in the group RTGOESR are routed to the remote server with the IP address ADDE.SSEC.WISC.EDU. (You may prefer to use your own data source.)

```
DATALOC ADD RTGOESR ADDE.SSEC.WISC.EDU
```

- b. List all datasets available in the RTGOESR group with the DSINFO command.

```
DSINFO ALL RTGOESR
```

- c. Run IMGLIST commands to see the available data in the datasets. Using a position number of .-2 will list the 3 most recent times in the dataset. Not specifying a position number will default to listing the most recent time.

```
IMGLIST RTGOESR/CONUS.-2 FORM=BAND  
IMGLIST RTGOESR/CONUS FORM=ALL
```

- d. Run the following list of commands to erase the frames and display a loop of 4 BAND 9 CONUS images centered on Madison with a map. Use 4 images with the REPEAT keyword, and the REFRESH keyword to display a map on each frame. Then set the loop sequence with the LS command and animate the loop.

```
ERASE X 1 4  
IMGDISP RTGOESR/CONUS.500 1 REP=4 BAND=9 STATION=KMSN SF=YES  
REFRESH='MAP GRA=(GRA) IMA=(IMA)'  
LS 1-4
```

Alt-L (Press *Alt-L* again to stop the looping.)

- e. Run the following list of commands to erase frames 1-4 and fill the frames with the four most recent BAND 2 times in the full disk dataset and a map. The LATLON= keyword centers the image at 20° N, 80° W. The MAG= -10 blows down the data by 10. Note: the RTGOESR group has datasets for the individual bands as well.

ERASE X 1 4

**IMGDISP RTGOESR/FDC02 1 ALL=1 4 MAG= -10 LATLON=0 90
REFRESH='MAP GRA=(GRA) IMA=(IMA)'**

Alt-L (Press *Alt-L* again to stop the looping.)

- 7. To access a remote ADDE server with Level 2 ABI netCDF product data, follow these steps.
 - a. Add the group name to modify your client routing table with the NPGOESR group that exists on the server ADDE.SSEC.WISC.EDU.

DATALOC ADD NPGOESR ADDE.SSEC.WISC.EDU

- b. Run DSINFO to find the available dataset names. In this example, use “I” instead of “ALL” to list only the image datasets available in the NPGOESR group.

DSINFO I NPGOESR

```
McIDAS-X 2022.1: jayh@panther.ssec.wisc.edu
DSINFO I NPGOESR
Dataset Names of Type: IMAGE in Group: NPGOESR
Name          NumPos      Content
-----
AODCN         99999      GOES-16 Level2 CONUS Aerosol Optical Depth Product
AODFD         99999      GOES-16 Level2 FullDisk Aerosol Optical Depth Product
AODM1         99999      GOES-16 Level2 Mesoscale-1 Aerosol Optical Depth Product
AODM2         99999      GOES-16 Level2 Mesoscale-2 Aerosol Detection Optical Dep
ARSLCN        99999      GOES-16 Level2 CONUS Aerosol Detection Products
ARSLFD        99999      GOES-16 Level2 FullDisk Aerosol Detection Products
ARSLM1        99999      GOES-16 Level2 Mesoscale-1 Aerosol Detection Products
ARSLM2        99999      GOES-16 Level2 Mesoscale-2 Aerosol Detection Products
CMSKCN        99999      GOES-16 Level2 CONUS Cloud Mask Product
CMSKFD        99999      GOES-16 Level2 FullDisk Cloud Mask Product
CMSKM1        99999      GOES-16 Level2 Mesoscale-1 Cloud Mask Product
CMSKM2        99999      GOES-16 Level2 Mesoscale-2 Cloud Mask Product
CODCN         99999      GOES-16 Level2 CONUS Cloud Optical Depth Product
CODFD         99999      GOES-16 Level2 FullDisk Cloud Optical Depth Product
CODM1         99999      GOES-16 Level2 Mesoscale-1 Cloud Optical Depth Product
CODM2         99999      GOES-16 Level2 Mesoscale-2 Cloud Optical Depth Product
IMA GRA Bounds Switches          Date      Time      T
  1   1 random          18 Sep 2023 02:45:57 1[130]
```

- c. Run IMGLIST with FORM=ALL for the expanded image list for the Cloud Top Temperature product.

IMGLIST NPGOESR/CTTFD FORM=ALL

Note: All the band lists used by McIDAS-X are found in the SATBAND file, which by default is found in `~mcidas/data`. You can inspect this text file with the SEE command, use a different text window.

```
1 <enter>  
SEE SATBAND  
SEE SATBAND NLINES=ALL  
TFIND U "187 (The sensor source (SS) number for the L2 products is 187)
```

- d. Display a loop of the Cloud Top Pressure product with a map and enhancement.

```
ERASE X 1 4  
IMGDISP NPGOESR/CTPRCN ALL=1 4 LAT=29 91 REFRESH='MAP GRA=(GRA)  
IMA=(IMA)'  
EU REST L2-CTP 1 4  
Alt-L (Press Alt-L again to stop the looping.)
```

- e. The DSI (Derived Stability Indices) product has multiple bands. List the bands and display BAND 9 which is the CAPE product (Convective Available Potential Energy). Add the corresponding enhancement and use Alt-D to probe the data (Alt-D is a shortcut to execute IMGPROBE LIST POINT MODE=N).

```
IMGLIST NPGOESR/DSICN FORM=BAND  
SF 5 ; IMGDISP NPGOESR/DSICN BAND=9 LAT=30 87 MAG=2  
EU REST L2-DSI-CAPE ; BAR ; MAP  
Alt-D
```

- f. Note the BAR on frame 5 is labeled with the 0-255 BRIT values. Make a BAR that is labeled with the calibrated CAPE values.

```
SU INI DEMO X CAPE  
SU MAKE DEMO 0 5000 0 255  
EG LEV=3 ; BAR SU=DEMO RANGE=0 5000 LINT=200
```

Using Other IMG* Commands with GOES-R Data

The ADDE ABIN server reads netCDF files directly to do IMGLIST and IMGDISP display commands. IMGCOPY uses the ADDE ABIN server to read the netCDF file and to outputs a McIDAS AREA file.

8. Use the ADDE commands to copy an ABI netCDF file to a McIDAS AREA file.
(The following exercises will overwrite AREA0001 through AREA0009 files on your local machine.)

- a. Assign the MYDATA/TEST-IMAGES data to all of the AREA files on your local workstation.

```
DSSERVE ADD MYDATA/TEST-IMAGES AREA 1 9999 "ALL AREA FILES
```

- b. Create the alias TI for the MYDATA/TEST-IMAGES dataset.

```
AKA ADD TI MYDATA/TEST-IMAGES
```

- c. Copy the band 2 image from a local netCDF file to a local AREA file.

IMGCOPY ABI/DATA.1 TI.1 BAND=2 STATION=KMSN

- d. Copy the band 14 image to a local AREA file at full resolution and list your new local AREA files.

IMGCOPY ABI/DATA.1 TI.2 BAND=14 SIZE=SAME

- e. Copy a remote SST product file and list your new local AREA files.

**IMGCOPY NPGOESR/SSTFD TI.3 LAT=0 89 SIZE=960 1280
IMGLIST TI.1 3 FORM=ALL**

- f. Display the AREA files.

**IMGDISP TI.1 6 REFRESH='MAP GRA=(GRA) IMA=(IMA)'
IMGDISP TI.2 7 LAT=33 77 MAG=-2 REFRESH='MAP GRA=(GRA) IMA=(IMA)'
IMGDISP TI.3 8 MAG=-2 REFRESH='MAP X X LALO GRA=(GRA) IMA=(IMA) ; EU REST
L2-SST (IMA) (IMA)'**

- g. Use the SF command to show each frame and inspect the data with the IMGPROBE command. Be sure to quit each IMGPROBE command with *Alt-Q* or by clicking the right and center mouse buttons.

**SF 6 ; IMGPROBE
IMGPROBE HIST BOX BRIT**

- h. Use the cursor size and IMGPROBE to inspect calibrated values on the IR and product images.

**CUR 11 11
SF 7 ; IMGPROBE STAT BOX TEMP
SF 8 ; IMGPROBE HIST LINE SST OUTL=YES**

9. Use IMGREMAP to create a remapped product AREA file.

- a. Create a rectilinear remapped product image at 4 km resolution.

**IMGREMAP RTGOESR/CONUS.-1 TI.4 PRO=RECT LAT=38 91 BAND=12 SIZE=480 640
RES=4**

- b. Display the new rectilinear AREA file.

IMGDISP TI.4 9 SF=YES ; MAP X X LALO

- c. Use the PRO=DEST keyword in the IMGREMAP command to use an AREA file as the domain for a remapped product image.

IMGREMAP NPGOESR/CMSKCN TI.4 PRO=DEST

- d. Now the cloud mask data will be in the product image AREA file.

IMGDISP TI.4 10 SF=YES ; MAP X X LALO

Note: The frame label in the display window has the original GOES image frame label. Rerun the IMGREMAP command with SAVESRC=YES and redisplay and note the change in the frame label.

```
IMGREMAP NPGOESR/CMSKCN TI.4 PRO=DEST SAVESRC=YES  
IMGDISP TI.4 10
```

Problem Set

IMGOPER is used to perform mathematical operations on images. The domain of the source imagery must be the same for the operations to work correctly. The source imagery can reside on remote servers, local ABI netCDF files or local AREA files. Use IMGOPER to create a temperature difference image between two images in the ABIL2/LSTC dataset, make an enhancement for the difference, and put a color bar on the image that matches the scale of the data.

Problem Set Solution

1. Create a temperature difference image from two Land Surface Temperature product files on an ADDE server 6 hours apart (Use IMGLIST to determine the position numbers to use in the IMGOPER command). The output values are the product name DIFF and multiplied by 10. The product values from -50 to 50 are scaled from brightness 0 to 255.

```
IMGLIST NPGOESR/LSTCN.-10  
IMGOPER NPGOESR/LSTCN.282 NPGOESR/LSTCN.288 TI.5 STA=KSTL SIZE=480 640  
UNIT=LST FORM=ADD COEF=1 -1 SCALE=-50 50 0 255 MCON=10 PROD=DIFF
```

2. Display, draw a map, place the cursor, and probe the data with 4 concatenated commands. This will aid in creating the enhancement and color bar.

```
IMGDISP TI.5 11 SF=YES ; MAP X 8 ; PC E 34 93 ; D
```

3. The difference AREA file contains difference from the calibrated LST value, which is a temperature. Enhance the image from blue (cold negative values) to red (warm positive values), then make BRIT value 1 black.

```
EU MAKE 0 255 BLUE RED  
EU MAKE 0 1 BLACK
```

4. Create the stretch table for the difference area file, then use SU= in the BAR command to display the color bar. Then redo the bar to adjust the border with BORDER=.


```
SU INI DIFF X DIFF
SU MAKE DIFF -50 50 0 255
BAR SU=DIFF
BAR SU=DIFF BORDER=255 255
```

Using GLM Data with GLM* Commands

You can access GOES-R Series (GOES-16 and later) netCDF Geostationary Lightning Mapper (GLM) point files with the GLMDISP, GLMLIST and PT* commands by using DSSERVE to assign a dataset name to the files. When doing so, specify "GLMN" in the *format* parameter, TYPE=POINT, the directory and file masks to the netCDF point files in the DIRFILE keyword, and the configuration file that maps the netCDF file parameter names to McIDAS parameter names in the INFO keyword. The files must be mission-standard format and follow the GRB naming convention, and thus look similar to **OR_GLM-L2-LCFA_G16_s20172151749200_e20172151749400_c20172151749467.nc**. Three configuration files for GLM data are provided in the `~mcidas/data` directory: **GLM_EVENT.cfg**, **GLM_FLASH.cfg** and **GLM_GROUP.cfg**.

GLM data processing in the Ground System groups lightning flashes from the smallest increments (Events), to aggregates of Events (Groups) to aggregates of Groups (Flashes). Thus, Events are on a rectangular grid wherein each gridpoint is a GLM Field of View. Groups are plotted at the centroid of the Events that comprise the Group, and Flashes are plotted at the centroid of the Groups that comprise the Flash. Typically, the number of Events is greater than the number of Groups, which is greater than the number of Flashes. Real-time GLM datasets normally consist of netCDF files that each contain 20 seconds of data. For example, there are separate files at 21:25:00, 21:25:20, 21:25:40, 21:26:00, 21:26:20, etc., assuming the GLM detected lightning during each of the corresponding 20 second intervals.

1. Create local datasets to access the GLM netCDF files on your local machine in the *Data/GLM/NetCDF* directory.

```
OR_GLM-L2-LCFA_G16_s20180041656400_e20180041657000_c20180041657016.nc
OR_GLM-L2-LCFA_G16_s20180041657000_e20180041657200_c20180041657218.nc
OR_GLM-L2-LCFA_G16_s20180041657200_e20180041657400_c20180041657427.nc
```

- a. Run the DSSERVE command to access the Level 1b netCDF GLM point data files for each type of GLM data.

```
DSSERVE ADD GLM/EVENT GLMN TYPE=POINT DIRFILE= '<local-
path>/Data/GLM/NetCDF/OR_GLM*' INFO='<mcidas-path>mcidas/data/GLM_EVENT.cfg'
DSSERVE ADD GLM/GROUP GLMN TYPE=POINT DIRFILE= '<local-
path>/Data/GLM/NetCDF/OR_GLM*' INFO='<mcidas-path>mcidas/data/GLM_GROUP.cfg'
DSSERVE ADD GLM/FLASH GLMN TYPE=POINT DIRFILE= '<local-
path>/Data/GLM/NetCDF/OR_GLM*' INFO='<mcidas-path>mcidas/data/GLM_FLASH.cfg'
```

- b. Use the GLMLIST command to list each type of data and note the number of records for each type.

```
GLMLIST NA TYPE=EVENT DATASET=GLM/EVENT DAY=2018004 TIME=16:58
GLMLIST NA TYPE=GROUP DATASET=GLM/GROUP DAY=2018004 TIME=16:58
GLMLIST NA TYPE=FLASH DATASET=GLM/FLASH DAY=2018004 TIME=16:58
```

- c. Use the MAP and GLMDISP commands to display each type over a small area. Inspect the differences with Alt-A and Alt-B to switch between the frames.

```
SF 12;MAP DEF LAT=-7 0 LON=72 82 GRA=12-14  
GLMDISP TYPE=EVENT DATASET=GLM/EVENT DAY=2018004 TIME=16:58  
GLMDISP X 13 TYPE=GROUP DATASET=GLM/GROUP DAY=2018004 TIME=16:58 COL=3  
GLMDISP X 14 TYPE=FLASH DATASET=GLM/FLASH DAY=2018004 TIME=16:58 COL=2
```

- d. Display a conus image and use the time matching keywords in GLMDISP to overlay the corresponding GLM lightning GROUP display. Then use the zoom feature to inspect the data closer.

```
IMGDISP ABI/CONUS 15 BAND=2 MAG=-8 LAT=36 74 SF=YES  
GLMDISP TYPE=GROUP DATA=GLM/GROUP DAY=IMA  
Alt-Z (Move the cursor over the region with lightning strikes, press Alt-Z to toggle the zooming.)
```

- e. Display the same conus image at full resolution and use the POSITION= in GLMDISP to overlay the corresponding GLM lightning GROUP display with and without the parallax correction. (Note: the default is POSITION=GROUND which plots the lightning strikes at the corrected ground positions, keeping the native parallax correction in the GLM file. POSITION=CLOUDTOP plots the lightning strikes at the approximate cloudtop positions as viewed from the GOES-R series satellite, removing the native parallax correction in the GLM file.)

```
IMGDISP ABI/CONUS 16 BAND=2 LAT=39 63 SF=YES  
GLMDISP TYPE=GROUP DATA=GLM/GROUP DAY=IMA  
GLMDISP TYPE=GROUP DATA=GLM/GROUP DAY=IMA POSITION=CLOUDTOP COL=3
```

2. To access a remote ADDE server with GLM netCDF point files, follow these steps to use the RTGOESR dataset that was previously added and add the RTGOESS dataset for use for GOES-West data.

- a. Add RTGOESS and inspect the dataset to identify the name of the GLM realtime datasets.

```
DATALOC ADD RTGOEST ADDE.SSEC.WISC.EDU  
DSINFO P RTGOEST
```

- b. Display the most recent hour of FLASH data with a time step of 15 minutes over a specific domain. (Note: the GLM* commands default to the EAST and WEST servers in Satellite Data Services at SSEC. If using other servers, DATASET= will need to be used with GLMLIST/GLMDISP, or the GLM.SITE or GLM.USER file will need to redefine the default GLM servers. This exercise will use RTGOESR/GLM* and RTGOEST/GLM* datasets and create a GLM.USER file.)

```
SF 17; GLMDISP TYPE=FLASH INC=60 15 LAT=-25 25 LON=40 90  
DATASET=RTGOESR/GLMFLASH
```

- c. To avoid using DATASET= for all real-time commands, copy the ~mcidas/data/GLM.CORE file to your \$HOME/mcidas/data and rename it GLM.USER. Run the following commands from a terminal window in your \$HOME/mcidas/data directory. (Adjust the commands as needed for your local setup)

```
cp ~mcidas/data/GLM.CORE GLM.USER
```

Edit the GLM.USER file with your favorite text editor and change the EAST/GLM-* group/descriptor instances to RTGOESR/GLM* in the first 2 sections of the GLM.USER file. There is a “*Note*” in the file that describes how to do this correctly.

```

data — vi GLM.USER — 80x24
# --- Global definitions used by ALL APPLICATIONS
GLM      DEFTYPE=GROUP
DATASET  GROUP=RTGOESR \
          DESCRIPTOR=GLMEVENT GLMGROUP GLMFLASH \
          POSITION=ALL

#      Supports multiple satellites with region definitions
DATASETS NGROUPS=2 \
          NAMES=RTGOESR RTGOESS \
          RTGOESRLAT=-20 75 RTGOESRLON=45 105 RTGOESRSAT=186 \
          RTGOESSLAT=-20 75 RTGOESSLON=105 170 RTGOESSSAT=188 \
          DESCRIPTOR=GLMEVENT GLMGROUP GLMFLASH \
          POSITION=ALL

#      *Note* - When changing the NAMES= in the SETTINGS
#      definition above, you must also change EASTLAT=,
#      EASTLON=, EASTSAT=, WESTLAT=, WESTLON=, WESTSAT=,
#      with the same convention. E.g. if EAST is changed to
#      GOES16, the next line would change to GOES16LAT=, GOES16LON=,
#      and GOES16SAT=.

```

- d. Display a band 13 full disk image and overlay the corresponding GLM GROUP data. Display the data as asterisks (*).

```

IMGDISP RTGOESR/FD.-5 18 BAND=13 LAT=0 75 MAG=-10 SF=YES
GLMDISP DAY=IMA LOC=*

```

- e. Display a band 2 full disk image and overlay the corresponding GLM GROUP data. Display the GLMDISP data on the full image window and echo the PT* commands that are being executed to the text frame.

```

IMGDISP RTGOESR/FD.-5 19 BAND=2 LAT=0 75 MAG=-20 SF=YES
GLMDISP TYPE=GROUP TIME=IMA ECHO=YES LEGEND=OFF

```

- f. Display a loop of band 9 full disk imagery with GLM FLASH data and a map overlaid one each frame.

```

IMGDISP RTGOESR/FD BAND=9 ALL=21 24 LAT=0 75 SF=YES MAG=-8 REFRESH='MAP
XX LALO IMA=(IMA) GRA=(GRA);GLMDISP X (GRA) TYPE=FLASH TIME=IMA'

```

- g. Display a RTGOEST visible band 2 image and overlay the corresponding GLM GROUP data similar to step d. above while adjusting the MAG= and LAT= for the band 2 image. GLMDISP will default to the data from the GOES West server because you loaded a GOES West image.

```

IMGDISP RTGOEST/FD 25 BAND=2 MAG=-40 LAT=0 137 SF=YES
GLMDISP

```

- h. Determine a U.S. state with lightning data and use that in the following MAP command. GLMDISP will use the domain of the map to display lightning data from GOES East or West. Change the symbols to * with a larger size.

MAP AZ;GLMDISP LOC=* 8

Using the MCSTRETCH= Keyword When Displaying Imagery

McIDAS-X versions 2017.2 and later include improvements to visualize more bit-depth in the data and provide more contrast in the images.

This expanded stretch feature was added to increase the detail shown in visible, water vapor and short-wave infrared imagery when displayed with IMGDISP from an updated ADDE server or copied to a 1-byte Area file using commands like IMGCOPY with STYPE=VISR or IMGREMAP. This is done by expanding the range of data values stretched out over the 8-bit brightness values in the higher reflectances in visible bands, and in the very hot temperatures in short-wave IR bands. The range of data values is narrowed in both colder and warmer temperatures in the water vapor bands.

The examples and the list of satellites and bands can be found here:

http://www.ssec.wisc.edu/mcidas/software/x/mcstretch_info.html

Or the text file is included in the distribution and can be viewed with the SEE command:

SEE MCSTRETCH.TXT

1. Display a current GOES-R image in a paneled frame with and without MCSTRETCH.

PANEL 25 1 2

IMGDISP RTGOESR/FD.-1 STA=KSTL BAND=9 MAG=-2 PANEL=1

IMGDISP RTGOESR/FD.-1 STA=KSTL BAND=9 MAG=-2 PANEL=2 MCS=ORIG

2. Analyze the data by using IMGPROBE/Alt-D to compare the BRIT values between the images. (Note: The RAW/RAD/TEMP calibrated values remain the same. The only difference is in the BRIT value for the 0-255 value the pixel is displayed at.)

PC C PANEL=1 ; D

PC C PANEL=2 ; D

3. Because EU tables are based on BRIT display values, the EU command was enhanced with a conversion option to convert old *.ET files to match the new stretched display BRIT values. This option is EU CONV.

The *Data/MCSTRETCH* directory contains a pre-2017.2 enhancement table:

SPECTRUM_ORIG.ET

- a. Copy the *SPECTRUM_ORIG.ET* file to your local mcidas/data directory in a Unix terminal.
- b. Display a band 2 GOES-R image with an original stretch and enhance it with *SPECTRUM_ORIG.ET*.

**IMGDISP ABI/CONUS 26 SF=YES BAND=2 LAT=15:38 75:08 MCS=ORIG
EU REST SPECTRUM_ORIG**

- c. Convert the SPECTRUM_ORIG file with EU CONV and use the converted *.ET file to display an enhanced (the default) display for the GOES-R image. (Note: The original file is saved in SPECTRUM_ORIG_OLD.ET.) Compare frames 26 and 27 and place the cursor over the whitish areas and compare BRIT values

**EU CONV SPECTRUM_ORIG VIS
IMGDISP ABI/CONUS 27 SF=YES BAND=2 LAT=15:38 75:08
EU REST SPECTRUM_ORIG
D**

- d. Display a current band 7 GOES-R image in a paneled frame with and without MCSTRETCH and inspect the data.

**PANEL 28 1 2
IMGDISP RTGOESR/FD.-1 STA=KMIA BAND=7 MAG=2 PANEL=1
IMGDISP RTGOESR/FD.-1 STA=KMIA BAND=7 MAG=2 PANEL=2 MCS=ORIG
PC L KMIA PANEL=1 ; D
PC L KMIA PANEL=2 ; D
SU TAB SWIRCORE**

RGBDISP – Displaying and Looping RGB Imagery

The RGBDISP command works similar to the COMBINE command, except that it displays its output color image in a regular image frame in the Image Window rather than in a separate Combine Window. RGBDISP uses the brightness (BRIT) values of the three input frames or datasets to calculate the red, green and blue components, respectively, of the output color image. This allows the user to interactively use the RGB displays and create loops.

The *Data/AREAS/* directory contains AREA files that are used in the following examples.

***AREA2700 – GOES-16 bands 2 and 13
AREA2800 – All bands GOES-16
AREA2900 – All bands GOES-16
AREA3000 – 0.64um Himawari Band 3
AREA3001 – 0.51um Himawari Band 2
AREA3002 – 0.47um Himawari Band 1
AREA3010 – 0.70um Meteosat-11 HRV Band 12
AREA3011 – All bands Meteosat-11 Band 1-11***

Copy these files to your local mcidas/data directory or use the REDIRECT command to point to them:

REDIRECT ADD AREA30* “<local-path>/Data/AREAS

4. Use the RGBDISP command to do a variety of RGB displays and loops.

- a. Display a True-Color RGB composite with the 0.64, 0.51 and 0.47 um images from the Himawari satellite, and echo the IMGDISP commands being executed.

SF 29 ; RGBDISP TL.3000 TL.3001 TL.3002 MAG=-40 -20 -20 BAND=3 2 1 ECHO=YES

- b. Display the RGB composite E-View display from Meteosat-11, a procedure developed by Eumetsat. It is dedicated to detailed cloud monitoring including water clouds. Use different LINELE= values to match up the images.

**SF 30 ; RGBDISP TL.3010 TL.3010 TL.3011 TIME=12 BAND=X X 9 MAG=-15 -15 -5
PLACE=CENTER LIN='5580 5573' '5580 5573' '5583 5583' I**

- c. Create a loop of 'Icing RGB' in AWIPS (also called the 'Day Land Cloud' RGB) images using the real time GOES-R server.

**SF 31 ; RGBDISP RTGOESR/FD.-4 MAG=-6 -6 -12 BAND=5 3 2 STA=KSTL
RGBDISP RTGOESR/FD.-3 32 MAG=-6 -6 -12 BAND=5 3 2 STA=KSTL
RGBDISP RTGOESR/FD.-2 33 MAG=-6 -6 -12 BAND=5 3 2 STA=KSTL
RGBDISP RTGOESR/FD.-1 34 MAG=-6 -6 -12 BAND=5 3 2 STA=KSTL
LS 31-34 ; Alt-L**

Problem Set

Create a day cloud convection RGB jpeg image with the IMGOPER, IMGDISP, RGBDISP and FRMSAVE commands for the AREA2800 from the <local-path>/Data/AREAS directory (TI.2800).

EUMETSAT and NOAA have created many different useful RGB composites for their SEVIRI and ABI instrument respectively. Many can be applied to both satellites.

The Day Cloud Convection product is a simple RGB that uses the traditional visible and infrared channels that forecasters are familiar with. The contribution from the visible is related to the illumination and reflectance of cloud and surface features, while the infrared is related to temperature. The combination helps to distinguish between high and low clouds and can help reveal wind shear when animated. As a heritage product it can be produced for any meteorological satellite that carries one visible and one longwave infrared channel. It can be composed from a scaled combination of GOES ABI bands 2 and 13 along with using the GAMMA= keyword in IMGOPER.

From

https://rammb.cira.colostate.edu/training/visit/quick_guides/QuickGuide_DayCloudConvectionRGB_final.pdf:

Color beam	Channel	Range		Gamma
Red	VIS 0.64	0	100%	1.7
Green	VIS 0.64	0	100%	1.7
Blue	IR10.3	323	203K	1.0

Problem Set Solution

1. Run the following commands to create and display the scaled products for the RGB combination.

```
IMGOPER TI.2800 TI.6 BAND=2 COEF=1 -1 MAG= -3 UNIT=ALB MISS=NONE SCALE=0 100 0  
255 GAMMA=1.7
```

```
IMGOPER TI.2800 TI.7 BAND=2 COEF=1 -1 MAG= -3 UNIT=ALB MISS=NONE SCALE=0 100 0  
255 GAMMA=1.7
```

```
IMGOPER TI.2800 TI.8 MAG= -3 BAND=13 UNIT=TEMP SCALE=323 203 0 255
```

```
ERASE X 1 34 ; EU REST X 1 34
```

```
IMGDISP TI.6 1 REP=3 SF=YES
```

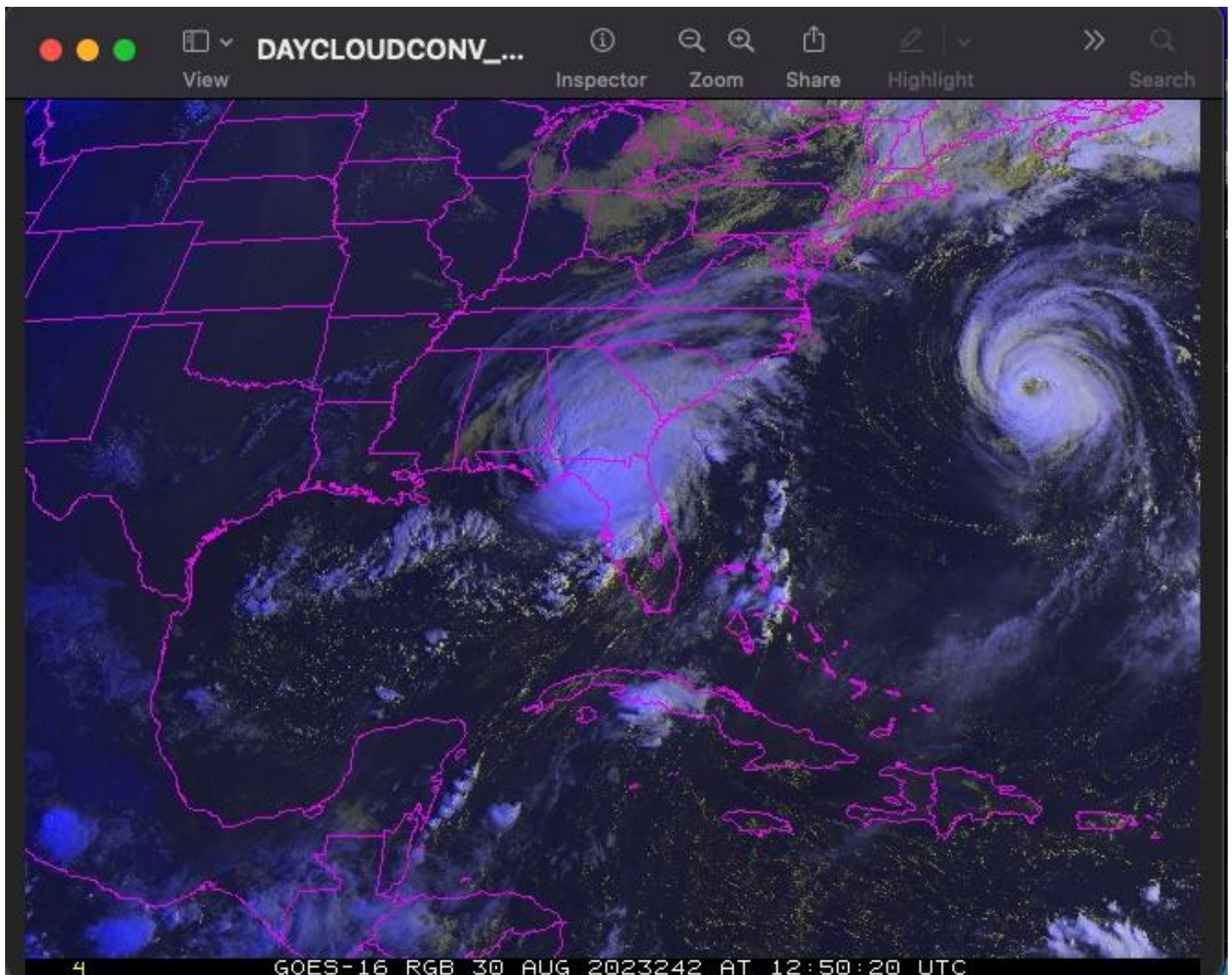
2. Use RGBDISP to combine the displays in three frames into a RGB image then FRMSAVE will convert the image into a jpeg file.

```
RGBDISP 1 2 3 4
```

```
SF 4
```

```
FRMSAVE 4 DAYCLOUDCONV_RGB
```

FRMSAVE will list the output file path where the jpeg file is created. Go to that directory and display it with a jpeg viewing application.



Note the following in the image:

- Yellow represents low to mid-level water clouds
- Blue to white represents deep convection

Servers for VIIRS/NOAA-20 Data

Now in core since version 2018.2, VIIRS instrument data from the Suomi NPP and NOAA 20-24 JPSS satellites is supported. When using DSSERVE to create a VIIRS dataset, specify "VIIR" in the *format* parameter, TYPE=IMAGE, the directory and file masks to the data files in the DIRFILE= keyword, and the directory and file masks to the geolocation files in the INFO= keyword. Also, the data (SVM*, SVI*) and terrain-corrected geolocation (GMTCO*, GITCO*) must be in separate files in the NOAA CLASS naming format. Since version 2022.1, EDR (Environmental Data Record) files are supported as well (VM* with GMGTO* geolocation files).

1. Create a local dataset to access the NOAA-20 HDF files on your local machine in the *Data/JPSS/RGB* directory.

```
GMTCO_npp_d20190902_t1815293_e1816535_b40667_c20190912225547190976_noac_ops.h5
GMTCO_npp_d20190902_t1816547_e1818189_b40667_c20190912225544540738_noac_ops.h5
SVM03_npp_d20190902_t1815293_e1816535_b40667_c20190912225554927796_noac_ops.h5
SVM03_npp_d20190902_t1816547_e1818189_b40667_c20190912225600849460_noac_ops.h5
SVM04_npp_d20190902_t1815293_e1816535_b40667_c20190912225624188549_noac_ops.h5
SVM04_npp_d20190902_t1816547_e1818189_b40667_c20190912225551829907_noac_ops.h5
SVM05_npp_d20190902_t1815293_e1816535_b40667_c20190912225611188308_noac_ops.h5
SVM05_npp_d20190902_t1816547_e1818189_b40667_c20190912225603671755_noac_ops.h5
```

```
DSSERVE ADD SUOMI/DATA VIIR TYPE=IMAGE DIRFILE='<local-path>
/Data/JPSS/RGB/SVM*' INFO='<local-path>/Data/JPSS/RGB/GMTCO*'
```

2. Use the IMG* commands to display and manipulate the data.

- a. Use IMGLIST to list out NOAA-20 band information.

```
IMGLIST SUOMI/DATA.1 FORM=BAND
```

- b. Display an image with IMGDISP and use Alt-D to inspect it.

```
IMGDISP SUOMI/DATA.2 5 SF=YES LAT=41 80 BAND=10
Alt-D
```

- c. Use IMGCOPY with STYP=VISR, IMGREMAP, and IMGOPER to merge together Band 9 for two consecutive granules.

```
IMGCOPY SUOMI/DATA.2 TI.9 BAND=9 SIZE=SAME STYPE=VISR
IMGCOPY SUOMI/DATA.1 TI.10 BAND=9 SIZE=SAME STYPE=VISR
IMGREMAP TI.9 TI.11 PRO=RECT RES=8 BAND=9 LAT=36 78
IMGREMAP TI.10 TI.12 PRO=RECT RES=8 BAND=9 LAT=36 78
IMGOPER TI.11 TI.12 TI.13 FORM=CMIN MISS=NONE
IMGDISP TI.13 6 SF=YES ; MAP
```

- d. The image is dark, IMGFILT will brighten the image with the default CLEAN filter.

```
IMGFILT TI.13 TI.14
IMGDISP TI.14 7 SF=YES ; MAP
```

3. Create a local dataset to access the NOAA-20 HDF files on your local machine in the *Data/JPSS/SDR_EDR* directory.

```
GMGTO_j01_d20230830_t1831168_e1833076_b29956_c20230919182931983224_oebc_ops.h5
GMTCO_j01_d20230830_t1831311_e1832556_b29956_c20230919182932823099_oebc_ops.h5
VM150_j01_d20230830_t1831168_e1833076_b29956_c20230919182931983224_oebc_ops.h5
SVM15_j01_d20230830_t1831311_e1832556_b29956_c20230919182935694590_oebc_ops.h5
```

```
DSSERVE ADD SDR/DATA VIIR TYPE=IMAGE DIRFILE='<local-path>
/Data/JPSS/SDR_EDR/SVM*' INFO='<local-path>/Data/JPSS/SDR_EDR/GMTCO*'
```

```
DSSERVE ADD EDR/DATA VIIR TYPE=IMAGE DIRFILE='<local-path>
/Data/JPSS/SDR_EDR/VM*' INFO='<local-path>/Data/JPSS/SDR_EDR/GMGTO*'
```

4. Use the IMG* commands to display and compare the data.
 - a. Use IMGLIST to list out NOAA-20 band information. Notice the Satellite/Sensor is denoted for each file type.

```
IMGLIST SDR/DATA.1 FORM=BAND
IMGLIST EDR/DATA.1 FORM=BAND
```

- b. Remap and display an image with IMGDISP and use Alt-D to inspect it. Notice the EDR imagery doesn't have the bowtie effect.

```
IMGREMAP SDR/DATA.1 TI.15 PRO=MERC RES=2 LAT=34 82 SIZE=960 1280 BAND=20
IMGREMAP EDR/DATA.1 TI.16 PRO=MERC RES=2 LAT=34 82 SIZE=960 1280 BAND=20
IMGDISP TI.15 MAG=-2 REP=2 REFRESH='MAP GRA=(GRA) IMA=(IMA)'
```

McIDAS-XRD: Creating GLM Density Plots with GLMIMG

The GLMIMG command in the McIDAS-XRD package will take GLMLIST data, aggregate the lightning reports into bins and create an AREA file for display.

1. Run GLMIMG to create a density image and merge it with the corresponding imagery with an enhancement.
 - a. Run GLMIMG to create the density AREA file and display it.

```
GLMIMG 40 'TYPE=EVENT TIME=1 INC=5 POS=CLOUDTOP DAY=#Y' ECHO=Y
IMGDISP TI.40 8 SF=YES
```

- b. Use the density image from the previous step and combine it with the corresponding image using IMGDISP, IMGREMAP and IMGOPER. Then enhance the lightning density points.

```
IMGOPER TI.40 TI.41 SCALE=0 255 0 50
IMGCOPY RTGOESR/CONUS TI.42 TIME=00:55 1:05 DAY=#Y BAND=13 STYPE=VISR
LAT=40 96 SIZE=1200 1200
IMGREMAP TI.41 TI.43 PRO=RECT RES=6 LAT=40 96
IMGREMAP TI.42 TI.44 PRO=RECT RES=6 LAT=40 96
IMGOPER TI.43 TI.44 TI.45 FORM=CMIN MISS=NONE
IMGDISP TI.45 9 SF=YES; MAP
EU MAKE 0 50 WHITE YELLOW
```

McIDAS-XRD: SANDWICH and RGB

SANDWICH combines image and basemap into multi-banded AREA files for use with RGBDISP.

RGB is a McBASI script that executes McIDAS commands to copy the necessary bands (IMGCOPY), manipulate them into the same resolution (REMAP2), apply the defined equations and user-defined Gamma corrections (IMGOPER), and use the RGBDISP command to display the RGB image. We will also use the EAST dataset on ADDE.SSEC.WISC.EDU for these displays.

1. Run IMGCOPY and SANDWICH to create the files to make an RGB image.

```
IMGCOPY MYDATA/IMAGES.2700 TI.1 BAND=2 SIZE=SAME STYPE=VISR  
IMGCOPY MYDATA/IMAGES.2700 TI.2 BAND=13 SIZE=SAME STYPE=VISR  
SANDWICH TI.1 TI.2 TI.3  
IMGLIST TI.3 FORM=BAND  
SF 10;RGBDISP TI.3 BAND=1 2 3 MAG=-4 LAT=17 119;MAP X 7
```

2. Run RGB LIST to see the products defined in the command, run RGB to create an ABI true-color RGB image from a local GOES-16 file.

```
DATALOC ADD EAST ADDE.SSEC.WISC.EDU  
DATALOC ADD BASEMAP ARCHIVE.SSEC.WISC.EDU  
RGB LIST  
SF 11; RGB MAKE ABI TRU MYDATA/IMAGES.2900 X X 'LAT=29 91 MAG=-2'
```

3. Run RGB to create a true-color RGB image from the most recent data.

```
SF 12;RGB
```

4. Run RGB to create a day cloud convection RGB image from the most recent data.

```
SF 13; RGB MAKE ABI DAYCLOUDCONV X X 'DAY=#Y TIME=17 18' 'LAT=32 85 MAG=-4'
```

5. Run RGB to create an airmass RGB image from a local GOES-16 file.

```
SF 13; RGB MAKE ABI AIRMASS TI.2800 X X 'MAG=-3'
```

DAS-X Workshop – Scripting Examples

September 2023 (software version 2022.1)

In existence since 1973, McIDAS (Man computer Interactive Data Access System) is a suite of sophisticated software packages that perform a wide variety of functions with satellite imagery, observational reports, numerical forecasts, and other geophysical data. Those functions include displaying, analyzing, interpreting, acquiring and managing the data.

McIDAS-X is supported for the current GOES GVAR and upcoming GOES-R satellite series (currently estimated as being in service until 2036), with no end date in sight.

In this McIDAS-X tutorial, some exercises will be completed using different methods of data access: local data files and real-time access to default remote servers. If you have access to your own real-time ADDE servers, you may also use those, but be aware that different server configurations may make the explanations in this document not quite applicable to all data that you may load. A source for free data served via ADDE is Unidata's ADDE.SSEC.WISC.EDU server.

This tutorial assumes that you have McIDAS-X installed on your machine, and that you know how to start McIDAS-X.

Getting Started – Unix Scripting

Unix scripting is a popular way to execute and schedule commands for automated output and product creation. You can run McIDAS-X commands without starting a McIDAS-X session by entering commands at the Unix prompt or using the **mcenv** process. When you run commands without starting McIDAS-X, a McIDAS environment is created, like the one created for a McIDAS session.

7. This example is a unix script that calls nested McIDAS-X BATCH files to create either band 13 or RGB imagery over 3 sectors of Mexico. The script is annotated with comments to describe what it is doing. The 9 BATCH files that are included are:

```
TRAINING_NORTE.BAT  
TRAINING_NORESTE.BAT  
TRAINING_BAJIO.BAT  
TRAINING_BAND_13_NORTE.BAT  
TRAINING_BAND_17_NORTE.BAT  
TRAINING_BAND_13_NORESTE.BAT  
TRAINING_BAND_17_NORESTE.BAT  
TRAINING_BAND_13_BAJIO.BAT  
TRAINING_BAND_17_BAJIO.BAT
```

The BATCH files for the NORTE sector are listed after the training2023.sh implementation script.

training2023.sh

```
-----  
#!/bin/sh  
# This script will use the format:  
#  
# training2023.sh band_number  
#  
# band_number - channel of data to copy  
# for training options are 13 (10.4um) or 17 (true color RGB)  
  
band_number=$1  
export band_number  
  
# Setup mcidas environment and sizes of frames  
  
# export PATH=/home/mcidas/bin:$PATH  
export MCPATH=$HOME/mcidas/data:/home/mcidas/data  
  
cd $HOME/mcidas/data  
  
mcenv -f 3@1500x1500 -g 15 -i 232 << 'EOF'  
  
logon.k ??? ????
```

Run batch files for each sector and rename output jpeg with day/time in filename

First - Norte

batch.k \${band_number} TRAINING_NORTE.BAT

time1=0

imglist.k NORTE.1 FORM=OPS > timelist

row1=`expr \${count} + 4`

day1=`cat \$HOME/mcidas/data/timelist|awk -v jl=\${row1} '{if (NR==jl) print \$3}'`

time1=`cat \$HOME/mcidas/data/timelist|awk -v jl=\${row1} '{if (NR==jl) print \$4}'`

if [\${time1} -ge 100000]

then

mv NORTE_TEMP.JPG NORTE_BANDS\${band_number}_20\${day1}_\${time1}.JPG

elif [\${time1} -ge 6000]

then

mv NORTE_TEMP.JPG NORTE_BANDS\${band_number}_20\${day1}_0\${time1}.JPG

else

mv NORTE_TEMP.JPG NORTE_BANDS\${band_number}_20\${day1}_00\${time1}.JPG

fi

Second - Noreste sector

batch.k \${band_number} TRAINING_NORESTE.BAT

time1=0

imglist.k NORESTE.1 FORM=OPS > timelist

row1=`expr \${count} + 4`

day1=`cat \$HOME/mcidas/data/timelist|awk -v jl=\${row1} '{if (NR==jl) print \$3}'`

time1=`cat \$HOME/mcidas/data/timelist|awk -v jl=\${row1} '{if (NR==jl) print \$4}'`

if [\${time1} -ge 100000]

then

mv NORESTE_TEMP.JPG

NORESTE_BANDS\${band_number}_20\${day1}_\${time1}.JPG

elif [\${time1} -ge 6000]

then

mv NORESTE_TEMP.JPG

NORESTE_BANDS\${band_number}_20\${day1}_0\${time1}.JPG

else

mv NORESTE_TEMP.JPG

NORESTE_BANDS\${band_number}_20\${day1}_00\${time1}.JPG

fi

Third - Bajio Y Altiplano Sector

batch.k \${band_number} TRAINING_BAJIO.BAT

time1=0

imglist.k BAJIO.1 FORM=OPS > timelist

row1=`expr \${count} + 4`

day1=`cat \$HOME/mcidas/data/timelist|awk -v jl=\${row1} '{if (NR==jl) print \$3}`

time1=`cat \$HOME/mcidas/data/timelist|awk -v jl=\${row1} '{if (NR==jl) print \$4}`

if [\${time1} -ge 100000]

then

mv BAJIO_TEMP.JPG BAJIO_BAND\${band_number}_20\${day1}_\${time1}.JPG

elif [\${time1} -ge 6000]

then

mv BAJIO_TEMP.JPG BAJIO_BAND\${band_number}_20\${day1}_0\${time1}.JPG

else

mv BAJIO_TEMP.JPG BAJIO_BAND\${band_number}_20\${day1}_00\${time1}.JPG

fi

exit

EOF

exit 0

TRAINING_NORTE.BAT

SET BAND=%1

REM DATALOC entries are saved in MCTABLE.TXT

REM and do not need to be run each time, so

REM I commented out the following DATALOC ADD

REM to your SDI machine. This command would

REM need to be run once on each new client

REM that would run these processing scripts.

REM DATALOC ADD EAST (your sdi IP address)

REM Also the following commands are saved from

REM session to session. DSSERVE entries are

REM saved in the RESOLV.SRV file. These I have

REM left to be run each time as they are quick

REM to execute and you can make changes as needed.

REM Band 17 does a true color image for

**REM the specified domain and is named with BAND17
REM in the filename and uses the RGB local dataset
REM for processing**

**DSSERVE ADD A/A AREA 1 9999
DSSERVE ADD A/NORTE AREA 301 305
DSSERVE ADD RGB/COMP AREA 401 580
AKA ADD A A/A
AKA ADD NORTE A/NORTE**

REM Processing script chosen per band

**IF %BAND%==13 BATCH TRAINING_BAND_13_NORTE.BAT
IF %BAND%==17 BATCH TRAINING_BAND_17_NORTE.BAT**

**-----
TRAINING_BAND_13_NORTE.BAT**

**IMGCOPY EAST/CONUS NORTE.1 BAND=13 SIZE=SAME
IMGREMAP NORTE.1 NORTE.2 PRO=MERC LAT=27 104.5 RES=.9 SIZE=1500 1500
IMGDISP NORTE.2 1 LAT=27 104.5
GD 2
MAP
MAP H
MAP NAME=OUTLHPOL
MAP NAME=OUTMEXICOST
GU MAKE 1 WHITE
FRMSAVE 1 NORTE_TEMP.JPG**

**-----
TRAINING_BAND_17_NORTE.BAT**

**REM IMGCOPY ABI bands 1, 2, and 3
IMGCOPY EAST/FD RGB/COMP.3 BAND=1 SIZE=SAME STYPE=VISR
IMGCOPY EAST/FD RGB/COMP.4 BAND=3 SIZE=SAME STYPE=VISR
IMGCOPY EAST/FD RGB/COMP.5 BAND=2 SIZE=SAME STYPE=VISR**

**REM Do this IMGCOPY to get time and day for filename later in script
IMGCOPY RGB/COMP.3 NORTE.1**

**REM Remap the higher resolution band into the domain of the
REM lower resolution bands
IMGCOPY RGB/COMP.4 RGB/COMP.1 SIZE=ALL
IMGREMAP RGB/COMP.5 RGB/COMP.1 SSIZE=ALL**

**REM To create an approximate green band, start by combining
REM a percentage of the BRIT values of ABI bands 1, 2, and 3
IMGOPER RGB/COMP.1 RGB/COMP.3 RGB/COMP.4 RGB/COMP.2 SIZE=ALL
COEF=.45 .45 .1**

**REM Scale the lower (darker) pixels for each band using SCALE= and ULMT=
REM and stretch the upper values with SCALE= and LLMT=
IMGOPER RGB/COMP.1 RGB/COMP.6 SIZE=ALL SCALE=0 33 0 10 ULMT=33
IMGOPER RGB/COMP.2 RGB/COMP.7 SIZE=ALL SCALE=0 40 0 10 ULMT=40
IMGOPER RGB/COMP.3 RGB/COMP.8 SIZE=ALL SCALE=0 50 0 10 ULMT=50
IMGOPER RGB/COMP.1 RGB/COMP.9 SIZE=ALL SCALE=33 255 10 255 LLMT=33
IMGOPER RGB/COMP.2 RGB/COMP.10 SIZE=ALL SCALE=40 255 10 255 LLMT=40
IMGOPER RGB/COMP.3 RGB/COMP.11 SIZE=ALL SCALE=50 255 10 255 LLMT=50**

**REM This last IMGOPER step combines the maximum values from the scaled red,
REM green, and blue images from the previous step
IMGOPER RGB/COMP.6 RGB/COMP.9 RGB/COMP.12 SIZE=ALL FORM=MAX
MISS=NONE
IMGOPER RGB/COMP.7 RGB/COMP.10 RGB/COMP.13 SIZE=ALL FORM=MAX
MISS=NONE
IMGOPER RGB/COMP.8 RGB/COMP.11 RGB/COMP.14 SIZE=ALL FORM=MAX
MISS=NONE**

**REM Remap into the correct sector domain
IMGREMAP RGB/COMP.12 RGB/COMP.15 PRO=MERC LAT=27 104.5 RES=.9
SIZE=1500 1500
IMGREMAP RGB/COMP.13 RGB/COMP.16 PRO=MERC LAT=27 104.5 RES=.9
SIZE=1500 1500
IMGREMAP RGB/COMP.14 RGB/COMP.17 PRO=MERC LAT=27 104.5 RES=.9
SIZE=1500 1500**

**REM The resultant AREA files are the approximated RED, GREEN, and BLUE
REM images to use for the RGBDISP command.**

REM

**REM The last step combines these bands and displays
REM the RGB image with the RGBDISP command.**

**RGBDISP RGB/COMP.15 RGB/COMP.16 RGB/COMP.17 1 LAT=27 104.5
SF 1
GD 2
MAP
MAP H
MAP NAME=OUTLHPOL
MAP NAME=OUTMEXICOST**

**GU MAKE 1 WHITE
FRMSAVE 1 NORTE_TEMP.JPG**

Python Scripting in a McIDAS-X Environment (-XRD)

Why Run McIDAS-X in a Python Environment?

The advantages of running McIDAS-X in a Python environment include but are not limited to:

- Setting up the “mcenv” environment is simpler and removes the shell scripting concepts of EOF and exit 0.
- Users can take advantage of Python’s superior text handling capabilities.
- Users can take advantage of Python’s superior date/time functionality.
- Python has many libraries for doing math, image manipulation and other data transformations.
- Python is more like a programming language than other traditional McIDAS scripting languages.

This tutorial assumes you have run the mcxpyinstall script in your environment to set up the Python environment.

1. This example starts with a unix script that calls Python scripts to create a RGB True Color image.

start-make-rgb.bash

(to invoke this script, e.g. ./start-make-rgb.bash 2023263 17:30:00 4000 RTGOESR/FD)

```
#!/bin/bash -e
```

```
DAY=$1
```

```
TIME=$2
```

```
FILE=$3
```

```
ADDEDATASET=$4
```

```
export ROOT_DIR=${HOME}/rgb
```

```
export LOCAL_MCIDAS_BIN=${HOME}/mcidas/bin
```

```
export LOCAL_DATA=${ROOT_DIR}/data
```

```
export LOCAL_LOG=${ROOT_DIR}/log
```

```
export FINAL_DIR=${ROOT_DIR}/final
```

```
export TEMP_DIR=${ROOT_DIR}/temp
```

```
export PATH=${LOCAL_MCIDAS_BIN}:/home/mcidas/bin:${PATH}
```

```
export USER_MCPATH=${HOME}/mcidas/data
```

```
export CORE_MCPATH=/Users/mcidas/data
export LOG_LEVEL=debug
```

```
echo ${PATH}
echo ${ROOT_MCPATH}
echo ${USER_MCPATH}
echo ${CORE_MCPATH}
echo ${FINAL_DIR}
echo ${TEMP_DIR}
```

```
python ${ROOT_DIR}/bin/make-rgb.py ${DAY} ${TIME} ${FILE} ${ADDEDATASET}
```

2. Below is the Python script called by the bash script that runs the commands to create the True Color image from the user inputted dataset and day/time.

```
<local-path>/rgb/bin/make-rgb.py
```

```
#!/usr/bin/python
```

```
import ast
```

```
import datetime
```

```
import glob
```

```
import logging
```

```
import os
```

```
import shutil
```

```
import socket
```

```
import subprocess
```

```
import sys
```

```
import time
```

```
def main(cmdArgs):
```

```
# The following script is triggered by RabbitMQ messages using amqpfind
```

This script uses ADDE to get bands 1,2 and 3 to create an ABI image

Create rotating log file

create_rotating_log()

LOG = logging.getLogger(__name__)

eventInfo = {}

This changes the input from amqpfnd to a dictionary

eventInfo['start_time'] = cmdArgs[1] + ' ' + cmdArgs[2]

eventInfo['file'] = os.path.join(os.environ['FINAL_DIR'],cmdArgs[3])

eventInfo['addeDataset'] = cmdArgs[4]

Create McIDAS environment

environReturn = create_mcidas_environment(eventInfo)

if not environReturn:

return

mcidasReturn = run_mcidas(eventInfo)

LOG.debug('Returning from McIDAS ' + str(mcidasReturn))

def create_mcidas_environment(eventInfo):

If GOES-16 goes into mode 4, processing between images might overlap.

```
# This could cause nasty collisions between sessions.  
# To avoid the situation, each time a new image is complete, a mcidas  
# session is started using a unique directory for MCPATH based on the  
# ip name of the machine and the pid  
  
import logging  
  
LOG = logging.getLogger(__name__)  
  
# Create MCPATH variable  
  
# Your final MCPATH should look something this:  
  
# /data/{user}/rgb/temp-files/nat-satbuf1/{pid}:/home/mcidas/data  
  
pid = str(os.getpid())  
  
shortIP = socket.gethostname().split('.')[0]  
  
rootMCPATH = os.path.join(os.environ['TEMP_DIR'], shortIP)  
  
rootMCPATH = os.path.join(rootMCPATH, pid)  
  
if not os.path.isdir(os.environ['ROOT_DIR']):  
  
    try:  
  
        os.makedirs(os.environ['ROOT_DIR'])  
  
    except Exception as e:  
  
        msg = 'cannot make destination directory: ' + os.environ['ROOT_DIR']  
  
        LOG.error(msg)  
  
        LOG.error(e)
```

return False

if not os.path.isdir(os.environ['LOCAL_DATA']):

try:

os.makedirs(os.environ['LOCAL_DATA'])

except Exception as e:

msg = 'cannot make destination directory: ' + os.environ['LOCAL_DATA']

LOG.error(msg)

LOG.error(e)

return False

if not os.path.isdir(os.environ['FINAL_DIR']):

try:

os.makedirs(os.environ['FINAL_DIR'])

except Exception as e:

msg = 'cannot make destination directory: ' + os.environ['FINAL_DIR']

LOG.error(msg)

LOG.error(e)

return False

if not os.path.isdir(os.environ['TEMP_DIR']):

try:

os.makedirs(os.environ['TEMP_DIR'])

except Exception as e:

```
msg = 'cannot make destination directory: ' + os.environ['TEMP_DIR']
```

```
LOG.error(msg)
```

```
LOG.error(e)
```

```
return False
```

```
userMCPATH = os.environ['USER_MCPATH']
```

```
coreMCPATH = os.environ['CORE_MCPATH']
```

```
mcpathDirs = [userMCPATH,rootMCPATH, coreMCPATH]
```

```
MCPATH = ':'.join(mcpathDirs)
```

```
os.environ['MCPATH'] = MCPATH
```

```
msg = 'MCPATH = ' + str(os.environ['MCPATH'])
```

```
LOG.debug(msg)
```

```
msg = 'FINAL_DIR= ' + str(os.environ['FINAL_DIR'])
```

```
LOG.debug(msg)
```

```
msg = 'TEMP_DIR= ' + str(os.environ['TEMP_DIR'])
```

```
LOG.debug(msg)
```

```
return True
```

```
def run_mcidas(mcidasInfo):
```

```
import logging
```

```
LOG = logging.getLogger(__name__)
```

```
try:
```

```
import mcidasx
```

```
except:
```

```
LOG.error('Unable to find mcidas-x python modules')
```

```
return False
```

```
# Define a local dataset used when copying data from the ABI server
```

```
localDataset = ' RGB/TEMP'
```

```
remoteGroup = mcidasInfo['addeDataset'].split('/')[0]
```

```
remoteDescriptor = mcidasInfo['addeDataset'].split('/')[1]
```

```
remoteDataset = mcidasInfo['addeDataset']
```

```
remoteServer = 'lead.unidata.ucar.edu'
```

```
# Day and Time for ABI resolve down to the minute. Use the information
```

```
# from the event to create the day and time keywords. IMGCOPY is used for
```

```
# most images, so created an imgcopy keyword just to make the string
```

```
# a little shorter. STYPE of VISR is only needed as this we only use
```


1 byte data

dayKeyword = ' DAY= ' + mcidasInfo['start_time'].split(' ')[0]

initTimeValue = mcidasInfo['start_time'].split(' ')[1]

initObj = datetime.datetime.strptime(initTimeValue, '%H:%M:%S')

firstTimeObj = initObj + datetime.timedelta(minutes=-5)

secondTimeObj = initObj + datetime.timedelta(minutes=5)

print(firstTimeObj)

print(secondTimeObj)

firstTimeValue = datetime.datetime.strftime(firstTimeObj, '%H:%M:%S')

secondTimeValue = datetime.datetime.strftime(secondTimeObj, '%H:%M:%S')

print(firstTimeValue)

print(secondTimeValue)

timeKeyword = ' TIME= ' + firstTimeValue + ' ' + secondTimeValue

timeKeyword = ' TIME=17 18 '

**imgcopyKeyword = dayKeyword + timeKeyword + ' STYPE=VISR SIZE=SAME
DEV=CCC '**

print(timeKeyword)

if 'FD' in remoteDescriptor:

centerPoint = '5428 5428'

numberLines = 10848

mcenv = mcidasx.mcenv(f=['3@10848x10848'], i=228, g=8)

LOG.error('started mcenv')

elif 'CONUS' in remoteDescriptor:

centerPoint = '1500 2500'

numberLines = 3000

mcenv = mcidasx.mcenv(f=['3@3000x5000'], i=228, g=8)

LOG.error('started mcenv')

else:

centerPoint = '500 500'

numberLines = 1000

mcenv = mcidasx.mcenv(f=['3@1000x1000'], i=228, g=8)

LOG.error('started mcenv')

imgdispKeywords = ' LINE=' + centerPoint + ' F PLACE=CENTER REP=3 TEXT=XXXX NO '

checkMCTABLE = os.path.join(os.environ['LOCAL_DATA'],'MCTABLE.TXT')

if not os.path.isfile(checkMCTABLE):

mccmdString = 'ADD ' + remoteGroup + ' ' + remoteServer

mccmdOut = mcenv.dataloc(mccmdString)

if not mcidas_command_output('DATALOC', mccmdString, mccmdOut):

return False

checkRESOLV = os.path.join(os.environ['LOCAL_DATA'], 'RESOLV.SRV')

```
if not os.path.isfile(checkRESOLV):
```

```
    mccmdString = 'ADD ' + localDataset + ' AREA 1 9999 '
```

```
    mccmdOut = mcenv.dsserve(mccmdString)
```

```
    if not mcidas_command_output('DSSERVE', mccmdString, mccmdOut):
```

```
        return False
```

```
# Initial and final AREA files definitions
```

```
# AREA0001 is for Red band (McIDAS band 2)
```

```
# AREA0002 is the final Green band (combination of bands 1, 2 and 3)
```

```
# AREA0003 is for the Blue Band (McIDAS band 1)
```

```
# AREA0004 is only used to create the simulated Green Band
```

```
# AREA0005 Temporary file to reduce the resolution of band 2
```

```
    mccmdString = remoteDataset + ' ' + localDataset + '.3 ' + imgcopyKeyword + '  
BAND=1'
```

```
    print(imgcopyKeyword)
```

```
    print(mccmdString)
```

```
    mccmdOut = mcenv.imgcopy(mccmdString)
```

```
    print(mccmdOut)
```

```
    if not mcidas_command_output('IMGCOPY', mccmdString, mccmdOut):
```

```
        return False
```

```
    mccmdString = remoteDataset + ' ' + localDataset + '.4 ' + imgcopyKeyword + '  
BAND=3'
```

```
mccmdOut = mcenv.imgcopy(mccmdString)
```

```
if not mcidas_command_output('IMGCOPY', mccmdString, mccmdOut):
```

```
    return False
```

```
mccmdString = remoteDataset + ' ' + localDataset + '.5 ' + imgcopyKeyword + '  
BAND=2'
```

```
mccmdOut = mcenv.imgcopy(mccmdString)
```

```
if not mcidas_command_output('IMGCOPY', mccmdString, mccmdOut):
```

```
    return False
```

```
# Create an image for remapping the high resolution band 2
```

```
mccmdString = localDataset + '.4 ' + localDataset + '.1 SIZE=ALL'
```

```
mccmdOut = mcenv.imgcopy(mccmdString)
```

```
if not mcidas_command_output('IMGCOPY', mccmdString, mccmdOut):
```

```
    return False
```

```
mccmdString = localDataset + '.5 ' + localDataset + '.1 SSIZE=ALL'
```

```
mccmdOut = mcenv.imgremap(mccmdString)
```

```
if not mcidas_command_output('IMGREMAP', mccmdString, mccmdOut):
```

```
    return False
```

```
# Create Green Band
```

```
mccmdString = localDataset + '.1 ' + localDataset + '.3 ' + localDataset + '.4 ' +  
localDataset + '.2 SIZE=ALL COEF=.45 .45 .1'
```

```
mccmdOut = mcenv.imgoper(mccmdString)
```

```
if not mcidas_command_output('IMGOPER', mcmdString, mcmdOut):
```

```
    if 'Full' in mcidasInfo['coverage']:
```

```
        mcmdString = localDataset + '.1 5 FORM=OPS'
```

```
        mcmdOut = mcenv.imglist(mcmdString)
```

```
        LOG.debug(mcmdOut.stdout)
```

```
    return False
```

```
# Based on some histogram values, A stretch is applied to each of the bands
```

```
# First pull out the low brightness values'
```

```
mcmdString = localDataset + '.1 ' + localDataset + '.11 SIZE=ALL SCALE=0 33 0 10  
ULMT=33'
```

```
mcmdOut = mcenv.imgoper(mcmdString)
```

```
if not mcidas_command_output('IMGOPER', mcmdString, mcmdOut):
```

```
    return False
```

```
mcmdString = localDataset + '.2 ' + localDataset + '.12 SIZE=ALL SCALE=0 40 0 10  
ULMT=40'
```

```
mcmdOut = mcenv.imgoper(mcmdString)
```

```
if not mcidas_command_output('IMGOPER', mcmdString, mcmdOut):
```

```
    return False
```

```
mcmdString = localDataset + '.3 ' + localDataset + '.13 SIZE=ALL SCALE=0 50 0 10  
ULMT=50'
```

```
mcmdOut = mcenv.imgoper(mcmdString)
```

```
if not mcidas_command_output('IMGOPER', mcmdString, mcmdOut):
```

return False

Stretch the higher brightness values'

**mccmdString = localDataset + '.1 ' + localDataset + '.21 SIZE=ALL SCALE=33 255 10
255 LLMT=33'**

mccmdOut = mcenv.imgoper(mccmdString)

if not mcidas_command_output('IMGOPER', mccmdString, mccmdOut):

return False

**mccmdString = localDataset + '.2 ' + localDataset + '.22 SIZE=ALL SCALE=40 255 10
255 LLMT=40'**

mccmdOut = mcenv.imgoper(mccmdString)

if not mcidas_command_output('IMGOPER', mccmdString, mccmdOut):

return False

**mccmdString = localDataset + '.3 ' + localDataset + '.23 SIZE=ALL SCALE=50 255 10
255 LLMT=50'**

mccmdOut = mcenv.imgoper(mccmdString)

if not mcidas_command_output('IMGOPER', mccmdString, mccmdOut):

return False

Combine the lower and higher brightness values

**mccmdString = localDataset + '.11 ' + localDataset + '.21 ' + localDataset + '.3001
SIZE=ALL FORM=MAX ZERO=DATA'**

mccmdOut = mcenv.imgoper(mccmdString)

if not mcidas_command_output('IMGOPER', mccmdString, mccmdOut):

return False

**mccmdString = localDataset + '.12 ' + localDataset + '.22 ' + localDataset + '.3002
SIZE=ALL FORM=MAX ZERO=DATA'**

mccmdOut = mcenv.imgoper(mccmdString)

if not mcidas_command_output('IMGOPER', mccmdString, mccmdOut):

return False

**mccmdString = localDataset + '.13 ' + localDataset + '.23 ' + localDataset + '.3003
SIZE=ALL FORM=MAX ZERO=DATA'**

mccmdOut = mcenv.imgoper(mccmdString)

if not mcidas_command_output('IMGOPER', mccmdString, mccmdOut):

return False

mccmdString = localDataset + '.3001 1 REP=3 ' + imgdispKeywords

mccmdOut = mcenv.imgdisp(mccmdString)

if not mcidas_command_output('IMGDISP', mccmdString, mccmdOut):

return False

mccmdString = 'X 1 GRA=1-3 LINE=11 ' + str(numberLines)

mccmdOut = mcenv.map(mccmdString)

if not mcidas_command_output('MAP', mccmdString, mccmdOut):

return False

mccmdString = 'MAKE 1 BLACK 1 3'

mccmdOut = mcenv.gu(mccmdString)

if not mcidas_command_output('GU', mccmdString, mccmdOut):

return False

mccmdString = 'OFF 3 1 3'

mccmdOut = mcenv.gu(mccmdString)

if not mcidas_command_output('GU', mccmdString, mccmdOut):

return False

mccmdString = ' 1 '

mccmdOut = mcenv.sf(mccmdString)

if not mcidas_command_output('SF', mccmdString, mccmdOut):

return False

mccmdString = ' 2 '

mccmdOut = mcenv.sf(mccmdString)

if not mcidas_command_output('SF', mccmdString, mccmdOut):

return False

mccmdString = ' 3 '

mccmdOut = mcenv.sf(mccmdString)

if not mcidas_command_output('SF', mccmdString, mccmdOut):

return False


```
mccmdString = '0 1 2 3 VIEW=NO'
```

```
mccmdOut = mcenv.combine(mccmdString)
```

```
if not mcidas_command_output('COMBINE', mccmdString, mccmdOut):
```

```
    return False
```

```
fileName = mcidasInfo['file']
```

```
mccmdString = '0 ' + fileName + ' TYPE=COMBINE FORM=JPG'
```

```
mccmdOut = mcenv.frmsave(mccmdString)
```

```
if not mcidas_command_output('FRMSAVE', mccmdString, mccmdOut):
```

```
    return False
```

```
mccmdString = ' AREA3003 '
```

```
mccmdOut = mcenv.dmap(mccmdString)
```

```
if not mcidas_command_output('DMAP', mccmdString, mccmdOut):
```

```
    LOG.info(str(mccmdOut.retcode))
```

```
    LOG.info(mccmdOut.stdout)
```

```
    LOG.info(mccmdOut.stderr)
```

```
    return False
```

```
else:
```

```
    LOG.info(str(mccmdOut.retcode))
```

```
    LOG.info(mccmdOut.stdout)
```

```
    LOG.info(mccmdOut.stderr)
```

```
# clean_up()
```

```
return True
```

```
def clean_up():
```

```
    pid = str(os.getpid())
```

```
    shortIP = socket.gethostname().split('.')[0]
```

```
    TEMP_DIR = os.path.join(os.environ['TEMP_DIR'], '-' + shortIP)
```

```
    TEMP_DIR = os.path.join(TEMP_DIR, pid)
```

```
    for f in glob.glob(os.path.join(TEMP_DIR, '*')):
```

```
        try:
```

```
            os.remove(f)
```

```
        except:
```

```
            pass
```

```
    os.rmdir(TEMP_DIR)
```

```
    return True
```

```
def mcidas_command_output(commandName, commandString, commandOutput):
```

```
    # Simplify McIDAS code by consolodating output
```

```
    import logging
```

```
LOG = logging.getLogger(__name__)

LOG.info(commandName + ' ' + commandString)

if commandOutput.retcode != 0:

    LOG.error(commandName + ' failed - return code: ' + str(commandOutput.retcode))

    LOG.error(commandOutput.stdout)

    LOG.error(commandOutput.stderr)

    return False

return True
```

```
def create_rotating_log():

    import logging.handlers

    #

    # setup logging

    #

    if not os.path.isdir(os.environ['LOCAL_LOG']):

        try:

            os.makedirs(os.environ['LOCAL_LOG'])

        except Exception as e:

            msg = 'cannot make destination directory: ' + os.environ['LOCAL_LOG']

            LOG.error(msg)

            LOG.error(e)

            return False
```

```
outLogFile = os.path.join(os.environ['LOCAL_LOG'],'make-rgb.log')
```

```
logFormat = '%(asctime)s - %(levelname)s - %(message)s'
```

```
logger = logging.getLogger()
```

```
logger.setLevel(logging.DEBUG)
```

```
handler = logging.handlers.RotatingFileHandler(outLogFile, maxBytes=1000000,  
backupCount=5)
```

```
formatter = logging.Formatter(logFormat, datefmt='%m/%d/%Y %H:%M:%S')
```

```
handler.setFormatter(formatter)
```

```
logger.addHandler(handler)
```

```
return
```

```
if __name__ == '__main__':
```

```
    main(sys.argv)
```