Airborne Processing Library
Getting started with APL - Command line

ARSF - Data Analysis Node

August 17, 2012

Contents

1 Getting started with APL - Command line

This is a tutorial aimed to help new people get started processing data with the Airborne Processing Library. By the end of it you should be able to generate per-pixel latitude and longitude values and resample the data into a regular grid. Example data is included and can be downloaded from the ARSF Data Analysis Node website. Note that running any of the APL programs with "-help" will give brief notes about all the available options.

Note that commands that are designed to be executed are `highlighted.` It is envisaged that the tutorial will take approximately 30 minutes to complete.

1.1 Files required for tutorial: Getting started with APL - Command line

This tutorial requires the file apl_tutorial_data.zip to be downloaded from the ARSF-DAN website from:
http://arsf-dan.nerc.ac.uk/files/apl_tutorial_data.zip .
The contents of this file can be found in the appendix in section A.

2 Data description

The example data for this tutorial are in ENVI file formats. ENVI is a commercial image processing package. The data format is very simple and can be read by numerous software including the open source GDAL libraries. The data formats used are BIL (band interleaved by line) and BSQ (band sequential). These are binary file formats with an accompanying text header file that contains meta-data file descriptions.

The example data is contained in an archive structured similarly to how ARSF deliver the hyperspectral data. The layout is as follows.

2.1 Level 1 imagery

The level 1 image data is found in the flightlines/level1b directory. Each flightline has 2 files; one ending with '.bil' and one with '.hdr' as described above. These contain radiometrically calibrated "at sensor radiance" values.

2.2 Level 1 mask

When the raw hyperspectral data is calibrated a mask file is also produced. This contains information about the status of each pixel and whether it has been adversely affected, for example, by saturation or "bad" CCD pixels. The mask data is stored as 1-byte binary data, with the same dimensions as the level-1 file that it applies to. Eagle lines will have 1 mask file whereas Hawk lines will have 2 mask files. The second mask file for Hawk contains information about which method has been used to identify the bad CCD pixel. Eagle does not suffer from these as it uses a different type of CCD and so does not have the second mask file.

Information on what flag values mean can be found in the .hdr file of the corresponding mask. These are briefly:

0 = Good data.
1 = Underflows.
2 = Overflows.
4 = Bad CCD pixels.
8 = Pixel affected by uncorrected smear.
16 = Dropped scans.

Some of these are sensor dependent, such as '4' will only occur in Hawk data and '8' only in Eagle.

2.3 Level 1 navigation

The navigation data is found in the flightlines/navigation directory. Each flightline has 2 files. These contain the time, position and attitude of the aircraft (including sensor boresight and lever arm information) synchronised to the level 1 image scan lines.

2.4 Digital Elevation Model

A Digital Elevation Model (DEM) is included with the example data in the dem directory. This has been created from ASTER data and covers the area of the included flight lines. The DEM is in APL format - which is a BSQ file with the data in WGS84 Geographic latitude/longitude with heights referenced to the WGS84 spheroid.

2.5 Sensor view vectors

The sensor view vector files are in the sensor_FOV_vectors directory. These are BIL files that contain the sensor CCD view vectors for the full CCD.

Note that these may not correspond directly to the level 1 image data as the Eagle instrument CCD also captures data from the FODIS sensor.

3 Procedure description

This tutorial is designed to get new users comfortable with APL. By the end of it we shall have applied a mask to the level-1 data, created an IGM file, reprojected it into a different coordinate projection, and resampled the level 1 data into a regular, mapped grid. This can be summarised as:

- Apply a mask to the level 1 data
- Use aplcorr to generate per-pixel latitude / longitude data
- Use apltran to reproject into the desired projection
- Use aplmap to resample the level 1 data into a regular grid.

The following procedure is suitable for using either the linux or Microsoft Windows versions of APL. The graphical user interface (GUI) or command line procedures can be used on either operating system.

4 Command line procedure

The following section gives a step-by-step procedure for processing the example data using the command line. It is assumed that you have downloaded APL prior to this, if not please do so now (see section B).

4.1 Apply a mask

To apply a mask to the level-1 data use the aplmask software. This will create a new level 1 file which has the offending pixels masked out - using a value of 0. We will now construct the command that we need to use step-by-step. To skip this and go straight to the executable command go to section 4.2.

First, we need to supply the level 1 file to aplmask. This is done using the -lev1 option:

```
aplmask -lev1 flightlines/level1b/eagle_line.bil
```

We also need to give it the corresponding mask file, using the -mask option:

```
   aplmask -lev1 flightlines/level1b/eagle_line.bil\
-mask flightlines/level1b/eagle_line_mask.bil
```

Next, we need to give it an output file name to write the masked data to. We use the -output option:

```
   aplmask -lev1 flightlines/level1b/eagle_line.bil\
-mask flightlines/level1b/eagle_line_mask.bil\
-output outputs/eagle_line_masked.bil
```

If we wish to mask out every type of affected pixel (recommended) then this is all we need, and the command is ready to run.

If however we wish to only mask out certain types of affected pixel then we can use the -flags option. For example, to only mask out underflows and overflows we can use:

```
   aplmask -lev1 flightlines/level1b/eagle_line.bil \
-mask flightlines/level1b/eagle_line_mask.bil \
-output outputs/eagle_line_masked.bil \
-flags 1 2
```

4.2 Run aplmask command

Execute the following command to mask the level 1 file:

```
aplmask -lev1 flightlines/level1b/eagle_line.bil \
-mask flightlines/level1b/eagle_line_mask.bil \
-output outputs/eagle_line_masked.bil
```

This will create a BIL file called eagle_line_masked.bil and accompanying file eagle_line_masked.bil.hdr in the directory named outputs. This file contains the masked level 1 data.

4.3 Construct aplcorr command

The next step is to generate an IGM file. This is a file containing the per-pixel geocorrection information. To do this we use aplcorr - which will generate an IGM file containing data in WGS84 latitude/longitude coordinates. We will now construct the command that we need to use step-by-step. To skip this and go straight to the executable command go to section 4.4.

First, we need to give aplcorr access to the level 1 file. This is done using the -lev1file option:

```
aplcorr -lev1file flightlines/level1beagle_line.bil
```

Next we need to give aplcorr the corresponding navigation data for this file, using the -navfile option:

```
aplcorr -lev1file flightlines/level1b/eagle_line.bil\
-navfile flightlines/navigation/eagle_line_nav_post_processed.bil
```

Note that the backslash is only required because the command is split over multiple lines. If the command is on a single line then no backslash is required.

We also need to pass the view vector file using the -vvfile option:

```
aplcorr -lev1file flightlines/level1b/eagle_line.bil\
-navfile flightlines/navigation/eagle_line_nav_post_processed.bil\
-vvfile sensor_FOV_vectors/eagle_fov_fullccd_vectors.bil
```

Because we have a DEM for the area we shall use that as this will give more accurate geocorrection results.

```
aplcorr -lev1file flightlines/level1b/eagle_line.bil\
-navfile flightlines/navigation/eagle_line_nav_post_processed.bil\
-vvfile sensor_FOV_vectors/eagle_fov_fullccd_vectors.bil
-dem dem/test-site.dem
```

Finally we need to give the filename for the output IGM file. We shall save this in a pre-existing directory named outputs:

```
aplcorr -lev1file flightlines/level1b/eagle_line.bil\
-navfile flightlines/navigation/eagle_line_nav_post_processed.bil\
-vvfile sensor_FOV_vectors/eagle_fov_fullccd_vectors.bil\
-dem dem/test-site.dem\
-igmfile outputs/eagle_line.igm
```

The command is now built and is ready to be executed.

4.4 Run aplcorr command

Execute the following command to process the flight line and generate an IGM file of per-pixel latitude and longitude values:

```
aplcorr -lev1file flightlines/level1b/eagle_line.bil \
-navfile flightlines/navigation/eagle_line_nav_post_processed.bil \
-vvfile sensor_FOV_vectors/eagle_fov_fullccd_vectors.bil \
-dem dem/test-site.dem \
-igmfile outputs/eagle_line.igm
```

This will create a BIL file called eagle_line.igm and accompanying file eagle_line.igm.hdr in the directory named outputs. This file has 3 bands: longitude, latitude and height.

4.5 Construct apltran command

We have our IGM file now but want it in a different projection. The data for this tutorial was collected over an area of the UK, so we will create a map in the Ordnance Survey National Grid (OSNG) projection. To do this we must first download the OSNG projection files from http://www.ordnancesurvey.co.uk/oswebsite/gps/osnetfreeservices/furtherinfo/ostn02_ntv2.html and unzip them into a directory named "ostn02".

We use the apltran program to do all the coordinate transformations. In the following we will construct our apltran command step-by-step. To skip this and go straight to the executable command go to section 4.6.

The first thing we need to let alptran know is the name of the IGM file we want to reproject:

```
apltran -igm outputs/eagle_line.igm
```

and secondly we will give it the name of the new IGM file to create, which we shall call eagle_line_osng.igm.

```
apltran -igm outputs/eagle_line.igm \
-output outputs/eagle_line_osng.igm
```

Next we need to specify the output projection we wish to use. There are 2 projections in apltran which can be defined using special keywords, these are 'osng' for the OSNG projection, 'utm_wgs84N' for UTM Northern hemisphere projection on the WGS84 spheroid and 'utm_wgs84' for UTM Southern hemisphere projection on the WGS84 spheroid. The keywords are preceeded by the option '-outproj'. As we want to project to OSNG we add this onto our command also passing the downloaded OSTN02 .gsb file:

```
apltran -igm outputs/eagle_line.igm \
-output outputs/eagle_line_osng.igm\
-outproj osng ostn02/OSTN02_NTv2.gsb
```

The command is now built and ready to be executed.

4.6 Run apltran command

Execute the following command to reproject the IGM file into Ordance Survey National Grid coordinates:

```
apltran -igm outputs/eagle_line.igm \
-output outputs/eagle_line_osng.igm \
-outproj osng ostn02/OSTN02_NTv2.gsb
```

This will create a BIL file called eagle_line_osng.igm and accompanying file eagle_line_osng.igm.hdr in the directory named outputs. This file has 3 bands: Easting, Northing and height.

4.7 Construct aplmap command

Now we have our IGM file projected into the coordinate system we wish to use for our final map. The last stage of the processing is to resample the level 1 data into a regular grid based on the projection information in the IGM file. To do this we use the aplmap program. In the following we will construct our aplmap command step-by-step. To skip this and go straight to the executable command go to section 4.8.

The first thing to tell aplmap is where to get all the different files it requires. These are the level 1 file:

```
aplmap -lev1 outputs/eagle_line.bil
```

and the igm file:

```
aplmap -lev1 outputs/eagle_line.bil\
-igm outputs/eagle_line_osng.igm
```

Next we tell aplmap what we want to call our mapped image, here we shall call it eagle_line_osng_mapped.bil.

```
aplmap -lev1 outputs/eagle_line.bil\
-igm outputs/eagle_line_osng.igm\
-mapname outputs/eagle_line_osng_mapped.bil
```

Now we need to tell aplmap a bit more information about the final mapped image, the first of which is the size of the pixels we want. The size must be given in the same units as the input IGM file. Here we are using the OSNG projection which is in metres. You can get an estimate of spatial resolution for Eagle and Hawk data using the calculator at: http://arsf-dan.nerc.

ac.uk/pixelsize/pixelsize.html. Here we are going to use a square pixel size of 2 metres.

```
   aplmap -lev1 outputs/eagle_line.bil\
-igm outputs/eagle_line_osng.igm\
-mapname outputs/eagle_line_osng_mapped.bil\
-pixelsize 2 2
```

We also need to tell aplmap which bands we wish to map. Our test file has 4 bands of data, so lets make an RGB image using the first 3 bands:

```
   aplmap -lev1 outputs/eagle_line.bil\
-igm outputs/eagle_line_osng.igm\
-mapname outputs/eagle_line_osng_mapped.bil\
-pixelsize 2 2 \
-bandlist 3 2 1
```

The aplmap command is now built and ready to be executed.

4.8 Run aplmap command

Execute the following command to resample the level 1 file into a regular grid defined by the IGM file, the pixel size and the number of bands that are being mapped.

```
aplmap -lev1 flightlines/level1b/eagle_line.bil \
-igm outputs/eagle_line_osng.igm \
-mapname outputs/eagle_line_osng_mapped.bil \
-pixelsize 2 2 \
-bandlist 3 2 1
```

This will create a 3 band BIL file containing resampled level 1 data from bands 3, 2 and 1. The BIL file has a map projection defined by the IGM file, in this case it is in OSNG, and has pixel size of 2 metres.

This is the end of the command line tutorial. A second flight line is included in the example data, a hawk image, for you to try and process. The commands to process the hawk data are in appendix C. The following section reviews what you have learnt.

5 Review

We have gone through the 3-step procedure of mapping ARSF level 1 data using APL, and also the stage of masking the level 1 data. You should now be

confident in using APL to map data to basic pre-defined projection systems, using the command line approach. You can now:

- Use aplmask to apply a mask to the level 1 data
- Use aplcorr to generate an IGM file containing latitude and longitude data
- Use apltran to reproject the data using one of the pre-defined projection keywords
- Use aplmap to resample the level 1 data to a regular gridded product

If you wish to check your processed data you can download the file apl_basic_tutorial_data_processed.zip which contains all the files that should have been created in this tutorial.

This concludes the first tutorial in basic APL usage. For further advanced tutorials please check the ARSF-DAN website.

A Appendix - File list

Table of files contained in apl_tutorial_data.zip .

Filename	Description
test-site.dem	Digital Elevation Model made from ASTER data. In WGS84 latitude / longitude and referenced to WGS84 spheroid surface.
test-site.dem.hdr	Header file containing meta-data relating to test-site.dem.
eagle_line.bil	Level 1 processed Eagle imagery
eagle_line.bil.hdr	Header file containing some meta-data relating to eagle_line.bil
eagle_line_mask.bil	Mask file for level 1 Eagle imagery
eagle_line_mask.bil.hdr	Header file containing some meta-data relating to eagle_line_mask.bil
eagle_line_nav_post_processed.bil	Navigation synced to eagle_line.bil data
eagle_line_nav_post_processed.bil.hdr	Header file containing some meta-data relating to eagle_line_nav_post_processed.bil
eagle_fov_fullccd_vectors.bil	Eagle full CCD view vector file. Contains the Eagle CCD view vectors.
eagle_fov_fullccd_vectors.bil.hdr	Header file containing meta-data relating to eagle_fov_fullccd_vectors.bil.
eagle_line.xml	XML meta-data relating to eagle_line.bil.
hawk_line.bil	Level 1 processed Hawk imagery
hawk_line.bil.hdr	Header file containing some meta-data relating to hawk_line.bil
hawk_line_mask.bil	Mask file for level 1 processed Hawk imagery
hawk_line_mask.bil.hdr	Header file containing some meta-data relating to hawk_line_mask.bil
hawk_line_mask-badpixelmethod.bil	CCD detector method mask file for level 1 processed Hawk imagery
hawk_line_mask-badpixelmethod.bil.hdr	Header file containing some meta-data relating to hawk_line_mask-badpixelmethod.bil
hawk_line_nav_post_processed.bil	Navigation synced to hawk_line.bil data
hawk_line_nav_post_processed.bil.hdr	Header file containing some meta-data relating to hawk_line_nav_post_processed.bil
hawk_fov_fullccd_vectors.bil	Hawk full CCD view vector file. Contains the Hawk CCD view vectors.
hawk_fov_fullccd_vectors.bil.hdr	Header file containing meta-data relating to hawk_fov_fullccd_vectors.bil.
hawk_line.xml	XML meta-data relating to hawk_line.bil.

B Appendix - Installation

Visit the ARSF-DAN website and download a copy of APL for your computer system from here: http://arsf-dan.nerc.ac.uk/trac/wiki/Downloads/software

Unzip the file contents into a directory. The windows build comes with the PROJ library included, if you are running under linux then please make sure you have PROJ installed on your system. The 4 APL files aplmask, aplcorr, apltran and aplmap need to have executable rights to run them. If possible you should add the executables directory to your computer's PATH variable so that you do not have to give the full path to the executable when you run it.

C Appendix - Hawk commands

These are the commands to process the included hawk image data.

```
aplmask -lev1 flightlines/level1b/hawk_line.bil \
-mask flightlines/level1b/hawk_line_mask.bil \
-output outputs/hawk_line_masked.bil
```

```
aplcorr -lev1file flightlines/level1b/hawk_line.bil \
-navfile flightlines/navigation/hawk_line_nav_post_processed.bil \
-vvfile sensor_FOV_vectors/hawk_fov_fullccd_vectors.bil \
-dem dem/test-site.dem \
-igmfile outputs/hawk_line.igm
```

```
apltran -igm outputs/hawk_line.igm \
-output outputs/hawk_line_osng.igm \
-outproj osng ostn02/OSTN02_NTv2.gsb
```

This time we will map some different bands: 30, 15 and 7.

```
aplmap -lev1 flightlines/level1b/hawk_line.bil \
-igm outputs/hawk_line_osng.igm \
-mapname outputs/hawk_line_osng_mapped.bil \
-pixelsize 2 2 \
-bandlist 30 15 7
```