Read the binary table, HDU #1

In [ ]:
```
from astropy.io import fits

# this uses the url for the file
# but can alternatively use a local file instead
base_url = 'https://lasp.colorado.edu/eve/data_access/eve_data/products/level0b/megs_a'
megsa_url = f'{base_url}/2010/120/MA__L0B_4_2010120_235905_00_001_01.fit.gz'
hdul = fits.open(megsa_url)

# read the binary table in HDU #1
for name_val_pair in zip(hdul[1].data.names, hdul[1].data[0]):
    print(f"{name_val_pair[0]} = {name_val_pair[1]}")
```

```
yyyydoy = 2010120
sod = 86345
tai_sec = 1651363179
tai_subsec = 2077186843
vcdu_count = 2395
int_time = 1
hw_test = 0
sw_test = 0
reverse_clock = 0
valid = 1
ram_bank = 0
int_time_warn = 0
filter_position = 4
readout_mode = 2
ccd_temp = -103.40232849121094
led_on = 0
led0_level = 0
led1_level = 0
resolver = 0
sam_resolver = 28328
```

And peek at the header contents for the binary table HDU #1:

In [ ]:
```
hdul[1].header
```

Out[ ]:
```
XTENSION= 'BINTABLE'           / binary table extension
BITPIX  =                    8 / 8-bit bytes
NAXIS   =                    2 / 2-dimensional binary table
NAXIS1  =                   39 / width of table in bytes
NAXIS2  =                    1 / number of rows in table
PCOUNT  =                    0 / size of special data area
GCOUNT  =                    1 / one data group (required keyword)
TFIELDS =                   20 / number of fields in each row
TTYPE1  = 'yyyydoy '           / label for field   1
TFORM1  = '1J      '           / data format of field: 4-byte INTEGER
TZERO1  =           2147483648 / offset for unsigned integers
TSCAL1  =                    1 / data are not scaled
TTYPE2  = 'sod     '           / label for field   2
TFORM2  = '1J      '           / data format of field: 4-byte INTEGER
TZERO2  =           2147483648 / offset for unsigned integers
TSCAL2  =                    1 / data are not scaled
TTYPE3  = 'tai_sec '           / label for field   3
TFORM3  = '1J      '           / data format of field: 4-byte INTEGER
TZERO3  =           2147483648 / offset for unsigned integers
TSCAL3  =                    1 / data are not scaled
TTYPE4  = 'tai_subsec'         / label for field   4
TFORM4  = '1J      '           / data format of field: 4-byte INTEGER
TZERO4  =           2147483648 / offset for unsigned integers
TSCAL4  =                    1 / data are not scaled
TTYPE5  = 'vcdu_count'         / label for field   5
TFORM5  = '1I      '           / data format of field: 2-byte INTEGER
TZERO5  =                32768 / offset for unsigned integers
TSCAL5  =                    1 / data are not scaled
TTYPE6  = 'int_time'           / label for field   6
TFORM6  = '1I      '           / data format of field: 2-byte INTEGER
TZERO6  =                32768 / offset for unsigned integers
TSCAL6  =                    1 / data are not scaled
TTYPE7  = 'hw_test '           / label for field   7
TFORM7  = '1B      '           / data format of field: BYTE
TTYPE8  = 'sw_test '           / label for field   8
TFORM8  = '1B      '           / data format of field: BYTE
TTYPE9  = 'reverse_clock'      / label for field   9
TFORM9  = '1B      '           / data format of field: BYTE
TTYPE10 = 'valid   '           / label for field  10
TFORM10 = '1B      '           / data format of field: BYTE
TTYPE11 = 'ram_bank'           / label for field  11
TFORM11 = '1B      '           / data format of field: BYTE
TTYPE12 = 'int_time_warn'      / label for field  12
TFORM12 = '1B      '           / data format of field: BYTE
TTYPE13 = 'filter_position'    / label for field  13
TFORM13 = '1B      '           / data format of field: BYTE
TTYPE14 = 'readout_mode'       / label for field  14
TFORM14 = '1B      '           / data format of field: BYTE
TTYPE15 = 'ccd_temp'           / label for field  15
TFORM15 = '1E      '           / data format of field: 4-byte REAL
TTYPE16 = 'led_on  '           / label for field  16
TFORM16 = '1B      '           / data format of field: BYTE
TTYPE17 = 'led0_level'         / label for field  17
TFORM17 = '1B      '           / data format of field: BYTE
TTYPE18 = 'led1_level'         / label for field  18
TFORM18 = '1B      '           / data format of field: BYTE
TTYPE19 = 'resolver'           / label for field  19
TFORM19 = '1I      '           / data format of field: 2-byte INTEGER
TZERO19 =                32768 / offset for unsigned integers
TSCAL19 =                    1 / data are not scaled
TTYPE20 = 'sam_resolver'       / label for field  20
TFORM20 = '1I      '           / data format of field: 2-byte INTEGER
TZERO20 =                32768 / offset for unsigned integers
TSCAL20 =                    1 / data are not scaled
EXTNAME = 'MEGSA_TABLE'        / name of this binary table extension
```

We can also examine the contents of the header for the imaage in HDU #0:

In [ ]:
```
hdul[0].header
```

Out[ ]:
```
SIMPLE  =                    T / file does conform to FITS standard
BITPIX  =                   16 / number of bits per data pixel
NAXIS   =                    2 / number of data axes
NAXIS1  =                 2048 / length of data axis 1
NAXIS2  =                 1024 / length of data axis 2
EXTEND  =                    T / FITS dataset may contain extensions
COMMENT   FITS (Flexible Image Transport System) format is defined in 'Astronomy
COMMENT   and Astrophysics', volume 376, page 359; bibcode: 2001A&A...376..359H
BZERO   =                32768 / offset data range to that of unsigned short
BSCALE  =                    1 / default scaling factor
EXTNAME = 'MEGS_IMAGE'         / Extension Name
SOD     =                86345 / Seconds in day
DOY     =              2010120 / Year - Day of year
TAI_TIME=           1651363179 / tai time
INT_TIME=                    1 / Integration time
RAM_BANK=                    0 / Ram bank
VALID   =                    1 / Validity flag
HW_TEST =                    0 / Test pattern
SW_TEST =                    0 / Test pattern
REV_CLK =                    0 / Reverse clock
HIERARCH tlm_filename = 'VC03_2010_120_23_58_45_0006a842cf0_07068_00.tlm' / TLM
HISTORY VC03_2010_120_23_58_45_0006a842cf0_07068_00.tlm
```

Read and display the image (HDU #0)

In [ ]:
```python
import os
import matplotlib.pyplot as plt
from skimage import exposure
from astropy.io import fits

with fits.open(megsa_url) as hdul:
    image = hdul[0].data    # the image is the first HDU

# histogram equalization is helpful for viewing the features in the image
image_eq = exposure.equalize_hist(image)
plt.figure(figsize=(12,6))
plt.imshow(image_eq, cmap='gist_heat', origin='lower')
plt.title(os.path.basename(megsa_url), fontsize=16)
plt.show()
```

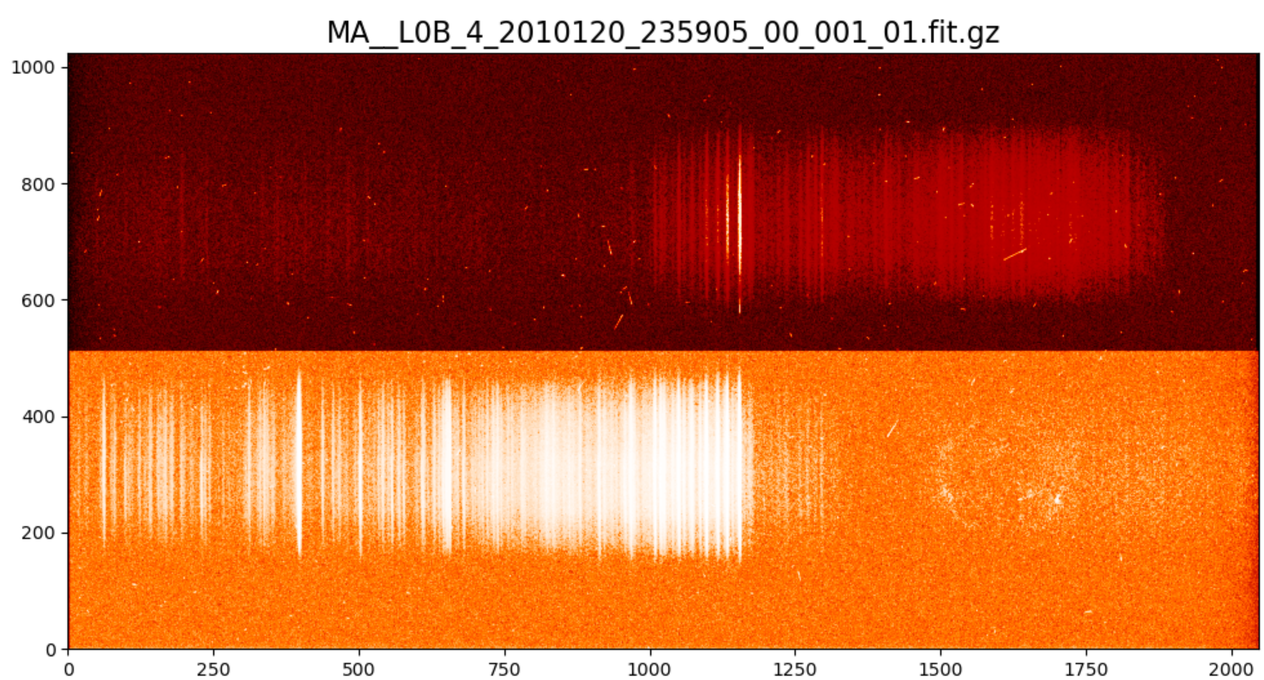

Figure 6: A single histogram equalized MEGS-A image with 10-second integration time. The slit 1 spectrum is dispersed across the top with short wavelengths on the right side. The bright Fe IX line at 17.1 nm is the brightest line in slit 1. Slit 2 also shows 17.1 and all of the longer wavelengths to the left. The SAM pinhole camera is in the lower right. Particle spikes and streaks are scattered across the detector.

Read multiple files

In [ ]:
```
import numpy as np
import os
from urllib.error import HTTPError

# this gives us five minutes of MEGS-A L0B files
base_url = 'https://lasp.colorado.edu/eve/data_access/eve_data/products/level0b/megs_a'
files = [f'{base_url}/2010/120/MA__L0B_4_2010120_235{x}{y}5_00_001_01.fit.gz' for x in range(5) for y in ra

image_sum = np.zeros((1024,2048))       # accumulate the running sum of each image

# loop over the files and add each image to the running total
# if the file isn't found on the web, skip it
for this_file in files:
    try:
        with fits.open(this_file) as hdul:
            image_sum += hdul[0].data     # the image is the first HDU
    except HTTPError:
        print(f"File not found: {os.path.basename(this_file)}")
        continue

# display the histogram equalized sum of images
image_sum_eq = exposure.equalize_hist(image_sum)
plt.figure(figsize=(12,6))
plt.imshow(image_sum_eq, cmap='gist_heat', origin='lower')
plt.title("Level 0B MEGS-A 5-minute image", fontsize=16)
plt.show()
```

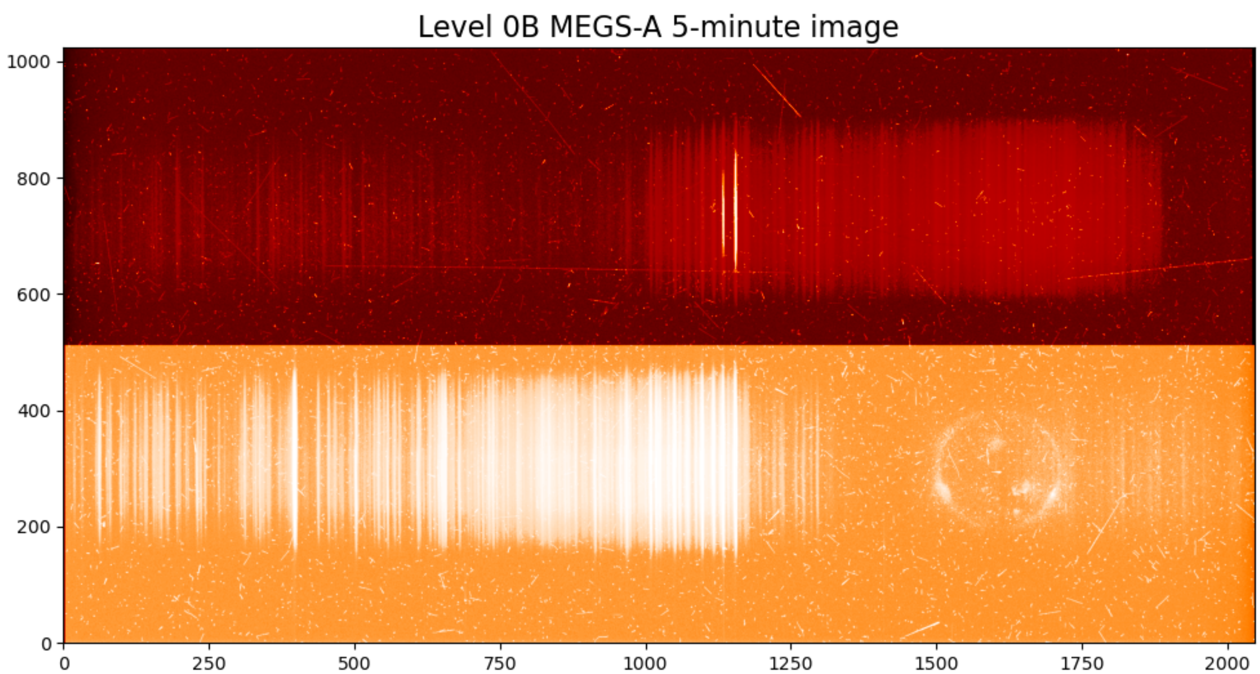

Figure 7: A 5-minute sum of 10-second integrations with histogram equalization makes it easier to see the SAM image and lines transmitted through slit 2. Larger particle spikes and streaks are easily observed when viewing multiple images.

Plot uncalibrated spectrum

Use the 5-minute image from the last step to plot an uncalibrated spectrum. As noted in the IDL section, the wavelengths are not uniformly distributed and are slighly curved.

In [ ]:
```
fig, axes = plt.subplots(2, 1, figsize=(12,8))

axes[0].plot(np.arange(1024)+1024, np.median(image_sum[800:808,1024:], axis=0))
axes[1].plot(np.arange(1200), np.median(image_sum[300:308,:1200], axis=0))

for i, ax in enumerate(axes):
    ax.grid(linestyle='dotted', zorder=0, color='grey')
```

```
    ax.set_title(f'Slit {i+1}')
    ax.set_ylabel('Arbitrary')
    ax.autoscale(enable=True, axis='x', tight=True)
axes[1].set_xlabel('Pixel')

plt.show()
```

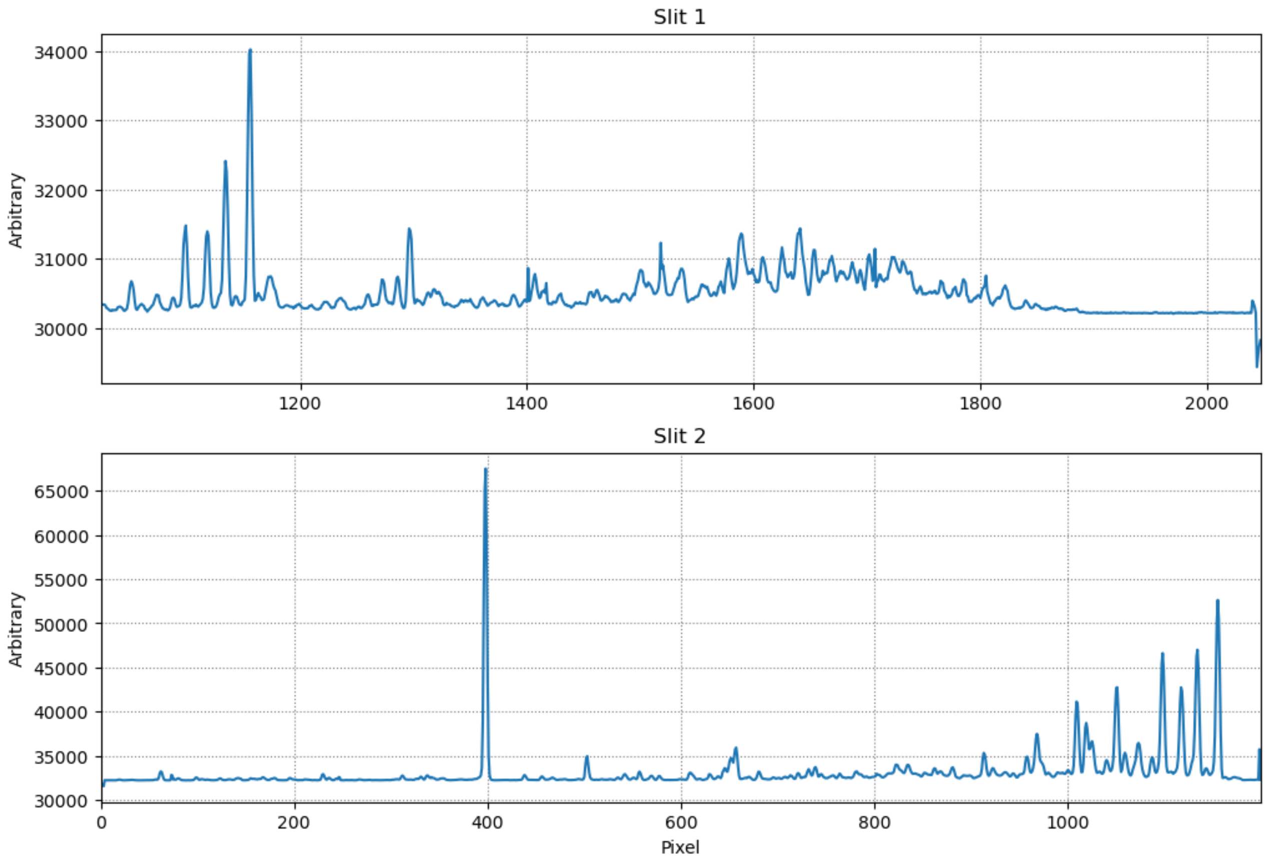

Figure 8: Uncalibrated spectra from MEGS-A slit 1 and 2 near the centers of each slit. The vertical axis has arbitrary units, and the horizontal axis is a reversed non-linear function of wavelength. Median filtering was applied in cross-dispersion to reduce the effect of particle strikes.

Read a MEGS-B file and display the image

In [ ]:
```
mb_file_url = 'https://lasp.colorado.edu/eve/data_access/eve_data/products/level0b/megs_b/2010/123/MB__L0B_

with fits.open(mb_file_url) as hdul:
    image = hdul[0].data

image_eq = exposure.equalize_hist(image)
plt.figure(figsize=(12,6))
plt.imshow(image_eq, cmap='Blues_r', origin='lower')
plt.title(os.path.basename(mb_file_url), fontsize=16)
plt.show()
```

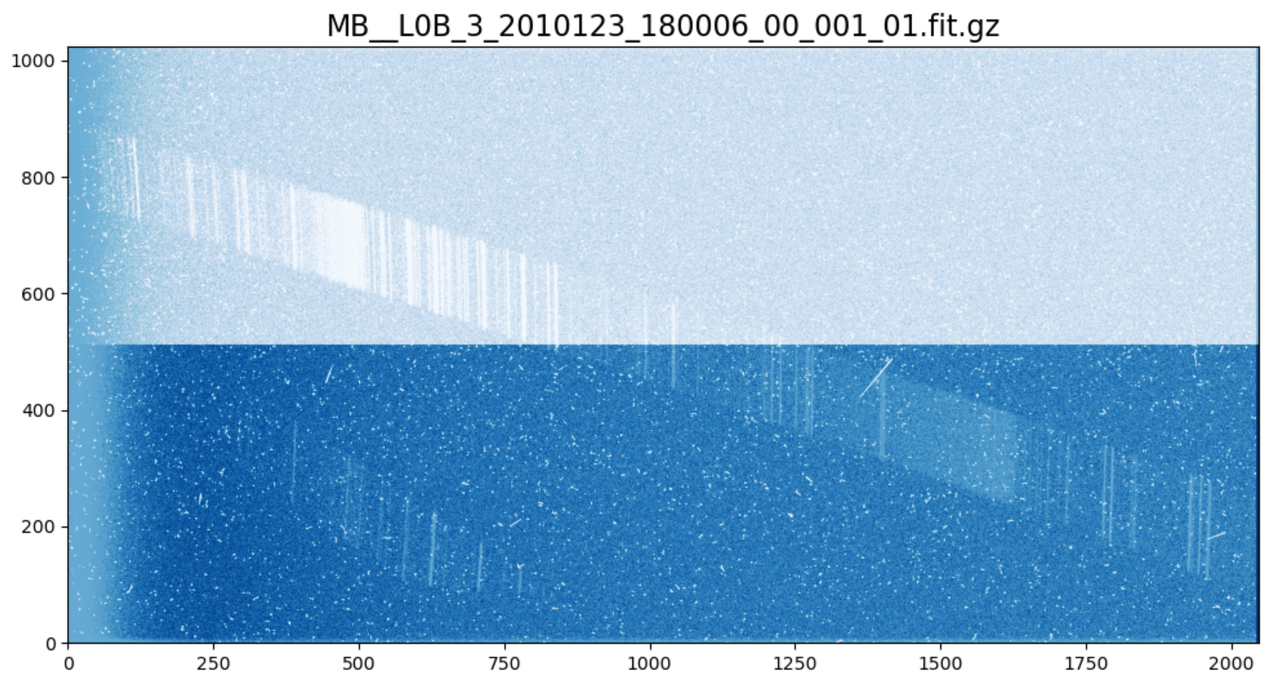

Figure 9: MEGS-B 10-second solar spectrum on the first rocket day, 2010 day 123, near 18:00 UTC.

The next example shows the same image with the median for each of the top and bottom halves subtracted to approximate the dark correction.

In [ ]:

```
cutoff = 512    # divide the image into top and bottom halves
flatmb = np.array(image, dtype='float')
flatmb[:cutoff,:] -= np.median(image[:cutoff,:])
flatmb[cutoff:,:] -= np.median(image[cutoff:,:])

flatmb_eq = exposure.equalize_hist(flatmb)
plt.figure(figsize=(12,6))
plt.imshow(flatmb_eq, cmap='Blues_r', origin='lower')
plt.title(f'{os.path.basename(mb_file_url)} median subtracted', fontsize=16)
plt.show()
```

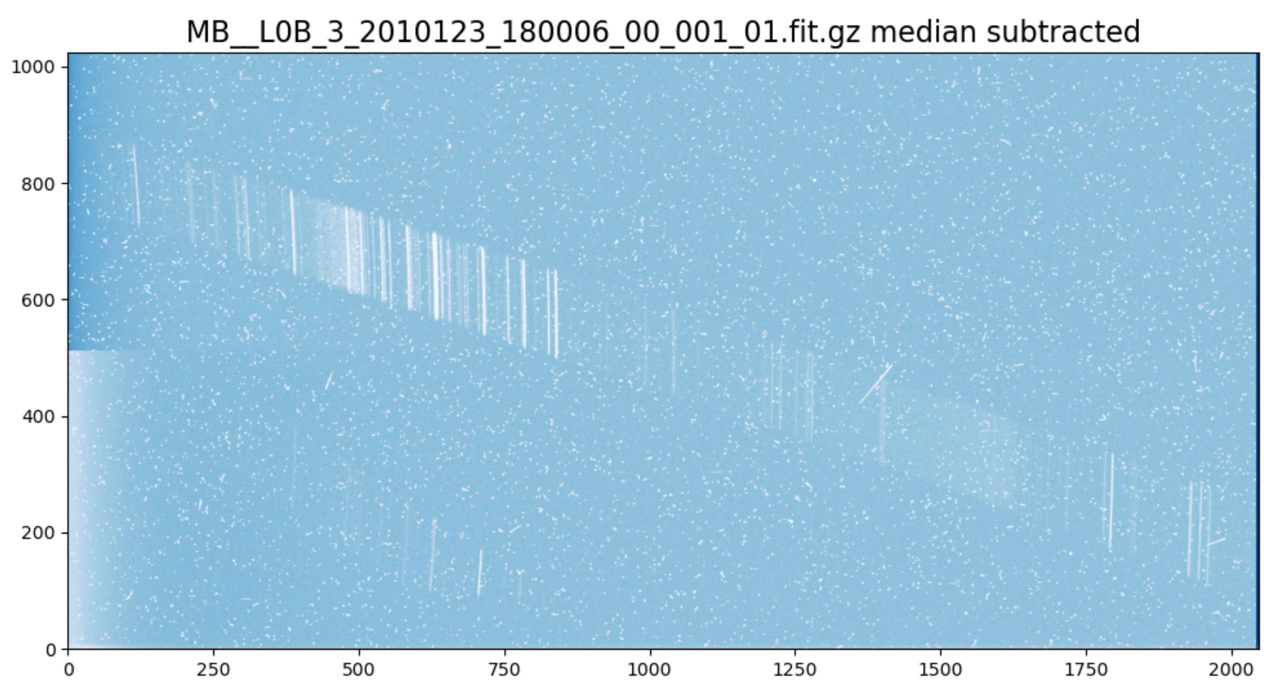

Figure 10: The same MEGS-B image data with median subtraction from each amplifier and histogram equalization reveals the solar spectrum spread diagonally across the detector with non-uniform background, magnification, line curvature, and

wavelength scale, plus particle streaks/spikes and a higher order spectrum in the lower left.

More detailed information about the images and features is in the IDL section of the Level 0B readme.