

METEORFINDER DEMO

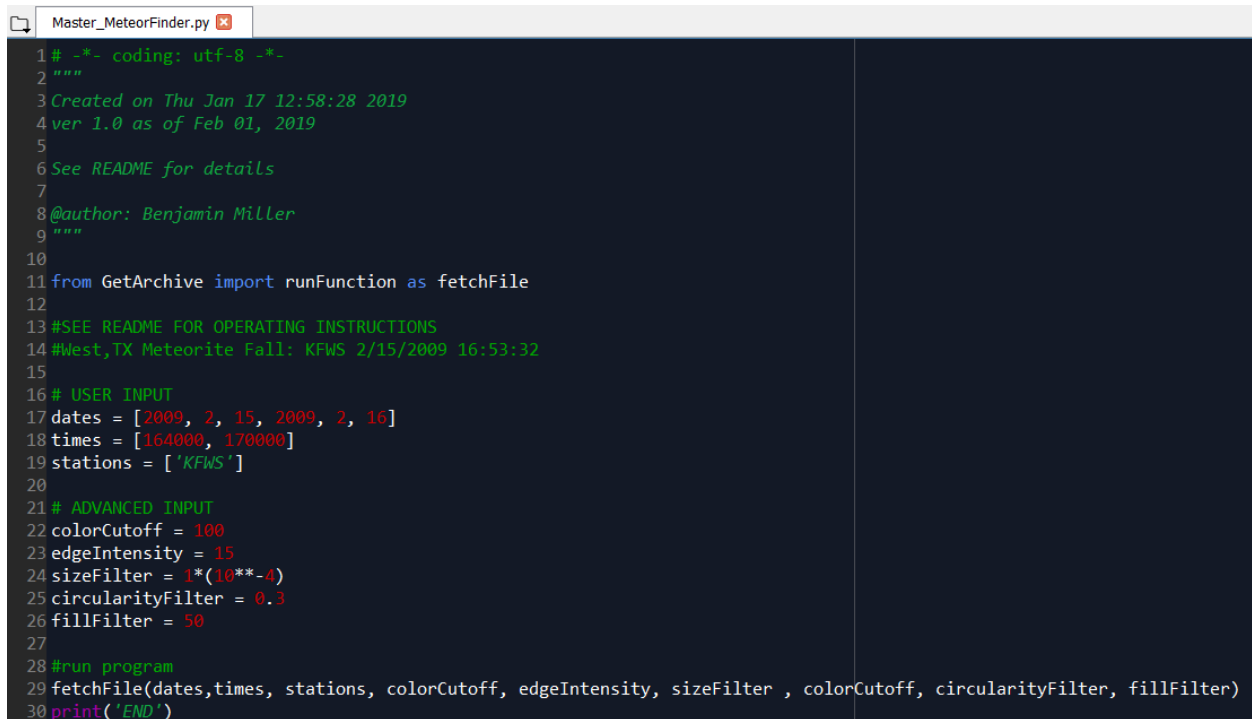
MeteorFinder ver 0.1
02/07/2019
Benjamin Miller

This file is intended to provide a visual supplement to the base README file. For a discussion of the installation requirements, processes, and variables, please see the README.

Starting the Program

The *Master_MeteorFinder.py* function serves as the primary user interface for detecting meteorite phenomena. This demonstration assumes that the user has previously installed all required packages and understands the operation of standard python IDE software such as Spyder.

After opening *Master_MeteorFinder.py*, the user will see the code presented in Figure 1. Here, the user inputs are delineated between standard and advanced variables. First, the user selects a date, time, and station range for fetching radar data. More advanced users may adjust the filters used for phenomena identification, which provides a higher level of control on narrow data ranges but is beyond the scope of this demonstration.



```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Jan 17 12:58:28 2019
4 ver 1.0 as of Feb 01, 2019
5
6 See README for details
7
8 @author: Benjamin Miller
9 """
10
11 from GetArchive import runFunction as fetchFile
12
13 #SEE README FOR OPERATING INSTRUCTIONS
14 #West,TX Meteorite Fall: KFWs 2/15/2009 16:53:32
15
16 # USER INPUT
17 dates = [2009, 2, 15, 2009, 2, 16]
18 times = [164000, 170000]
19 stations = ['KFWs']
20
21 # ADVANCED INPUT
22 colorCutoff = 100
23 edgeIntensity = 15
24 sizeFilter = 1*(10**-4)
25 circularityFilter = 0.3
26 fillFilter = 50
27
28 #run program
29 fetchFile(dates,times, stations, colorCutoff, edgeIntensity, sizeFilter , colorCutoff, circularityFilter, fillFilter)
30 print('END')
```

Figure 1: *Master_MeteorFinder.py* interface

On running the master function, the console will display output to the console as shown in Figure 2. The algorithm will process the requested data in chronological dates, processing each radar site alphabetically for the entire time range.

In the figure, which has been selected for the example radar fall event in West, TX, two time files have been processed. For the first time file, at 16:43:53, no fall events were detected. This will be the predominant output of the system, showing only the progressive altitude layers as they are unwrapped. For the time file at 16:53:32, a single fall event has been detected on the 4th altitude layer of the radar scan.

```
-----Downloading data as of 02/15/2009 -----
Downloading data from radar: "KFWS"
Time: 164354
 0 1 2 3 4 5 6
Time: 165332
 0 1 2 3 4 5 6
FALLS DETECTED AT SCAN(S): [4]
END
```

Figure 2: *MeteorFinder* console output for KFWS 02/15/2009 16:40:00-17:00:00

Analyzing Output

Whenever a fall has been detected by the program, it saves the unwrapped images of the identified phenomena, as well as the .gz file for the full radar scan. In Figure 3, a new site-specific folder has been created in which these saved files are stored. Note that if the program is canceled at any time, the program will not be able to delete temporary data, and a residual folder will remain.

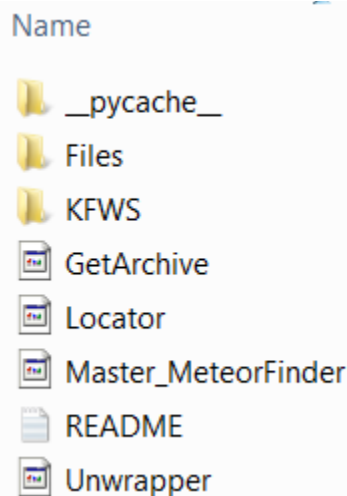


Figure 3: Folder created for a fall identified by KFWS

Within the KFWS folder, the user should see two files available. These files, beginning with a tag representing the site, data, local time, and scan format, display the identified phenomena with regards to radial velocity (Figure 4a) and spectrum width (Figure 4b). The blue

contours represent where the fall has been identified, and the other color coding represents magnitudes of the respective data. These files may be useful in confirming the validity of the meteor fall, and the advanced user may troubleshoot variable settings.

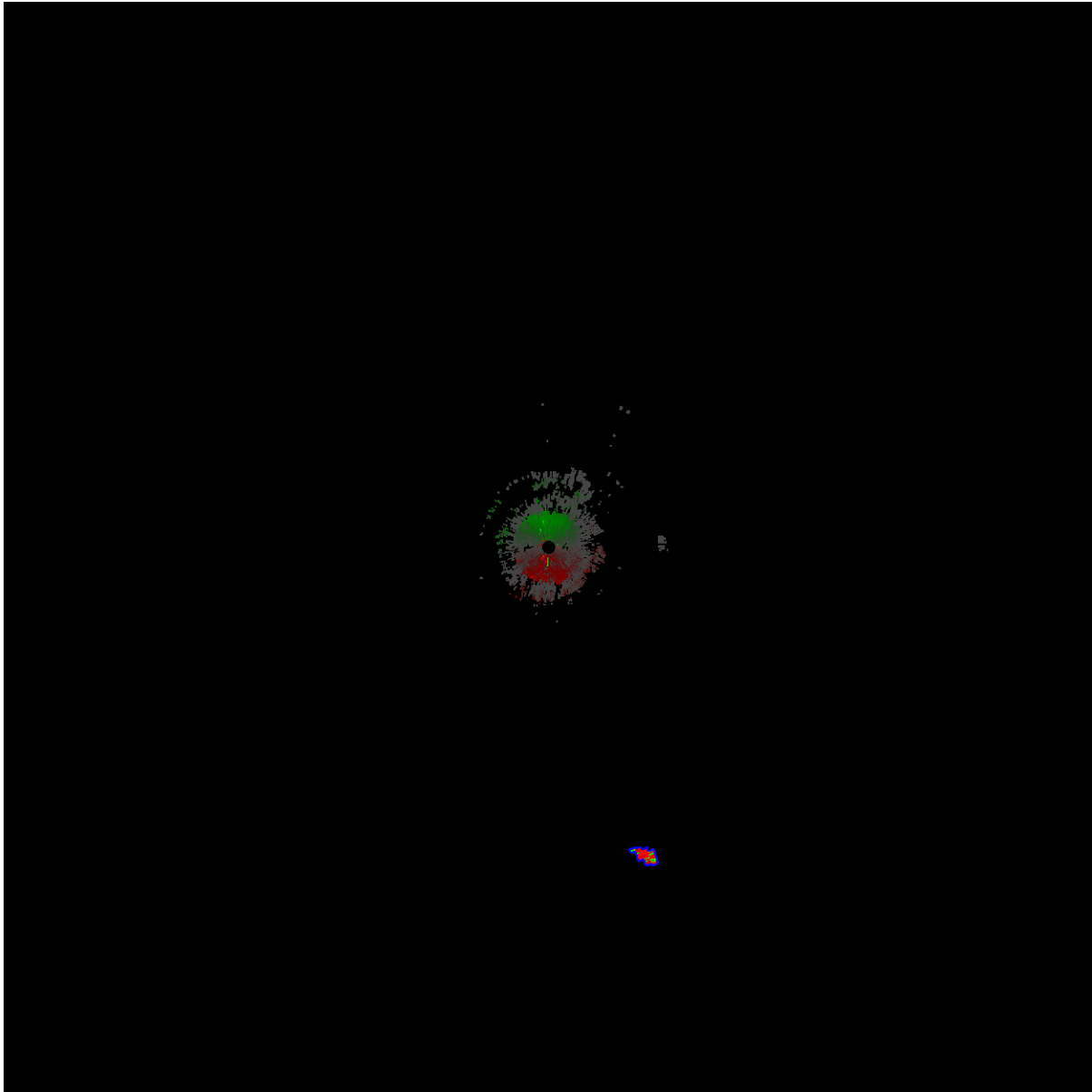


Figure 4a: *KFWS20090215_165332_V03_VEL4.png*

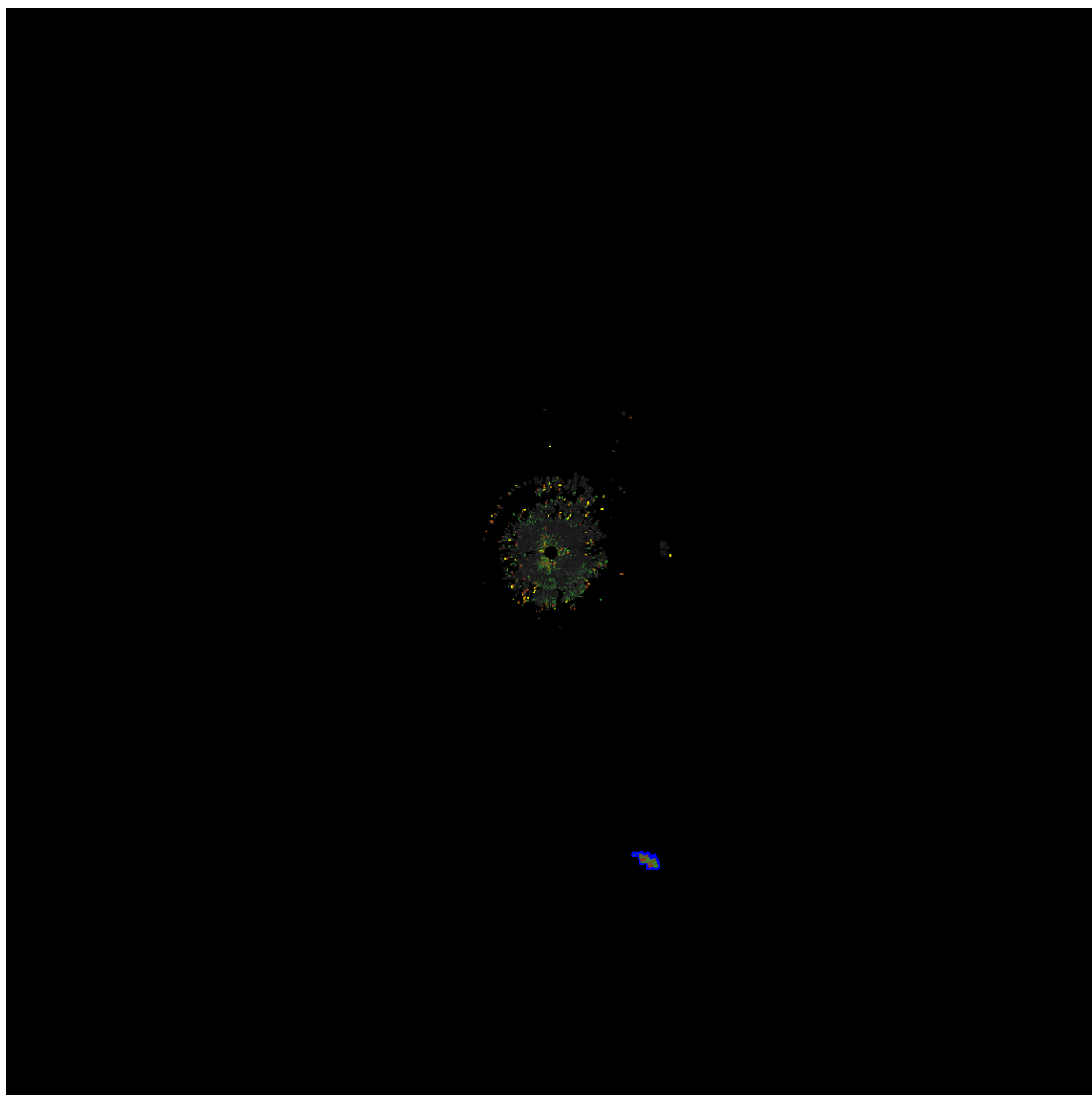


Figure 4b: *KFWS20090215_165332_V03_SPW4.png*