Identifying Antenna Swaps with HERA Metrics

Kevin McNulty & Adam Beardsley

November 21st, 2017

In an effort to ensure that the antenna positions are accurate, we have written a short metrics text to identify any outliers. We are particularly interested in identifying what antennas may be plugged into the wrong "box", which will return a poor modified z-score.

1. Necessary python packages

Import the required packages: `pyuvdata`, `hera_qm.ant_metrics`, and `hera_cal`.

In [1]:
```
%matplotlib notebook
import numpy as np
import matplotlib.pyplot as plt
import pyuvdata
import hera_qm.ant_metrics as ant_metrics
import hera_cal
```

2. Data file list

Import the data file list that needs to be examined, in this case we're looking at the most recent IDR1 gathered at the time of this memo. In the same cell define a `uvdata` object, the variable `aa`, and `info`.

In [2]:
```
data_file_list = ['/data6/HERA/data/2458042/zen.2458042.12552.xx.HH.uv',
                  '/data6/HERA/data/2458042/zen.2458042.12552.yy.HH.uv',
                  '/data6/HERA/data/2458042/zen.2458042.12552.yx.HH.uv',
                  '/data6/HERA/data/2458042/zen.2458042.12552.xy.HH.uv']

uvdata = pyuvdata.UVData()
uvdata.read_miriad(data_file_list)
aa = hera_cal.utils.get_aa_from_uv(uvdata)
info = hera_cal.omni.aa_to_info(aa)
```

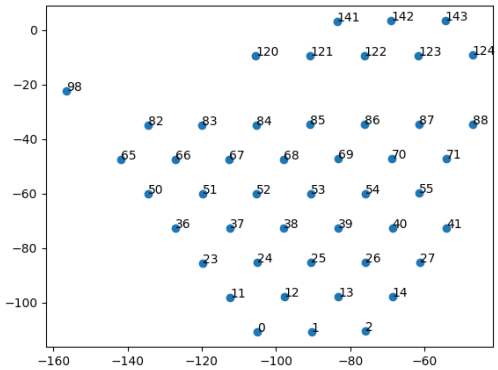
3. HERA plot

Create a plot of the current HERA antenna positions. This is useful because we can see which baselines should be correlated before running our antenna swap (e.g. antenna 0 and antenna 2).

In [3]:
```
antpos = uvdata.antenna_positions+uvdata.telescope_location
antpos = pyuvdata.uvutils.ENU_from_ECEF(antpos.T, *uvdata.telescope_location_lat_lon_alt).T
```

In [4]:
```
plt.figure("Antenna Positions")
plt.plot(antpos[:,0],antpos[:,1],'o')
for i, txt in enumerate(uvdata.antenna_numbers):
    plt.annotate(txt, antpos[i, 0:2])
```

Out [4]:



4. Antenna metrics

Define the variable `am` and print the antenna positions. We want to find out if the function "`iterative.antenna_metrics_and_flagging`" will flag corrupt data.

In [5]:
```
am = ant_metrics.Antenna_Metrics(data_file_list, reds, fileformat = 'miriad')

am.iterative_antenna_metrics_and_flagging(crossCut = 5, deadCut = 5)
```

In [6]:
```
print uvdata.antenna_positions[46]
```

Out [6]:
```
[ 26.51695001 -63.66362355   2.96685549]
```

5. Define the plot metrics

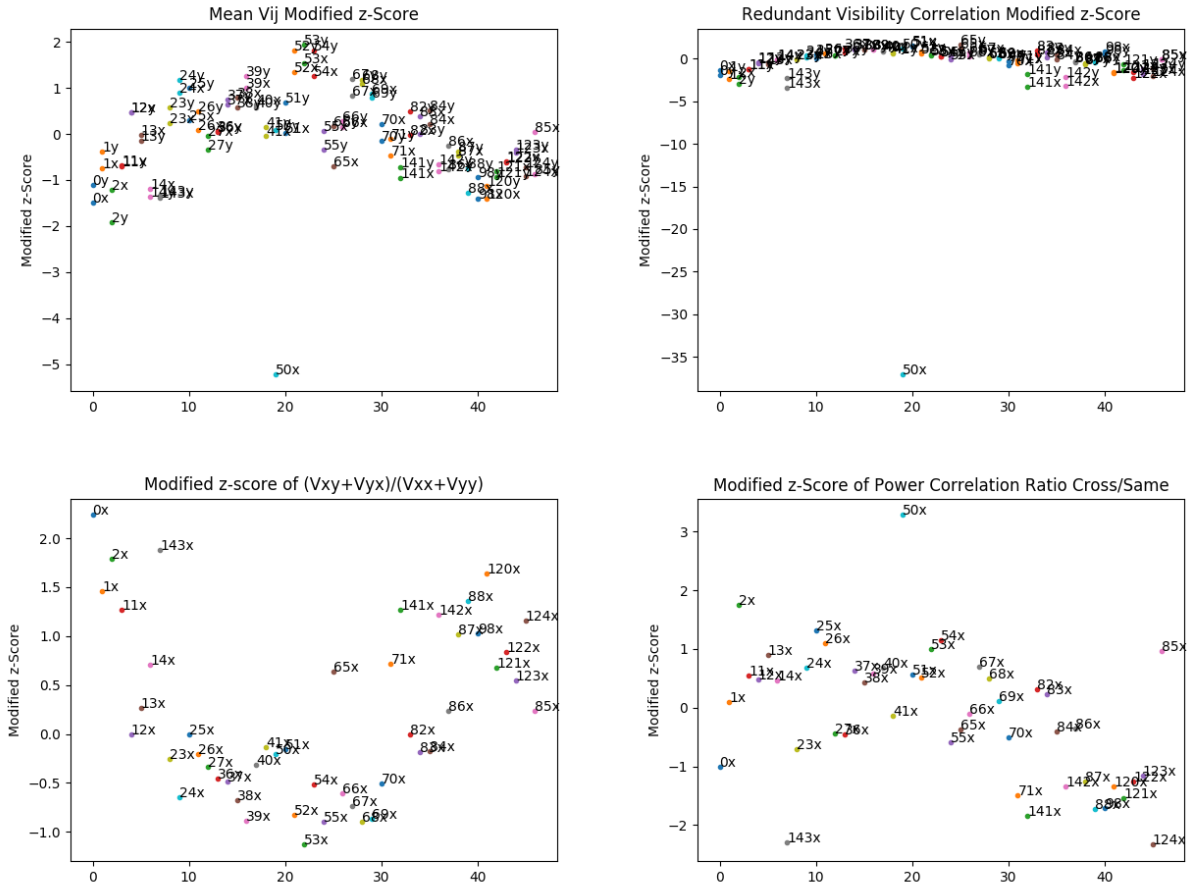
Define the plot metrics[1] and plot the "Mean Vij Modified z-Score", the "Redundant Visibility Correlation Modified z-Score", the "Modified z-Score of (Vxy+Vyx/(Vxx+Vyy)", and the "Modified z-Score of Power Correlation Ratio Cross/Same".

In [7]:
```
def plot_metric(metrics, ants=None, antpols=None, title='', ylabel='Modified z-Score',
xlabel=''):
    '''Helper function for quickly plotting an individual antenna metric.'''

    if ants is None:
        ants = list(set([key[0] for key in metrics.keys()]))
    if antpols is None:
        antpols = list(set([key[1] for key in metrics.keys()]))

    for antpol in antpols:
        for i,ant in enumerate(ants):
            metric = 0
            if metrics.has_key((ant,antpol)):
                metric = metrics[(ant,antpol)]
            plt.plot(i,metric,'.')
            plt.annotate(str(ant)+antpol,xy=(i,metrics[(ant,antpol)]))
        plt.gca().set_prop_cycle(None)
    plt.title(title)
    plt.ylabel(ylabel)
    plt.xlabel(xlabel)
```

In [8]:
```
#print "Ant Metrics for {}".format(am.metrics)

plt.figure()
plot_metric(am.finalModzScores['meanVij'],
            title = 'Mean Vij Modified z-Score')

plt.figure()
plot_metric(am.finalModzScores['redCorr'],
            title = 'Redundant Visibility Correlation Modified z-Score')

plt.figure()
plot_metric(am.finalModzScores['meanVijXPol'], antpols=['x'],
            title = 'Modified z-score of (Vxy+Vyx)/(Vxx+Vyy)')

plt.figure()
plot_metric(am.finalModzScores['redCorrXPol'], antpols=['x'],
            title = 'Modified z-Score of Power Correlation Ratio Cross/Same')
```

[1] The code for the plot metrics was written by Nick Kern and can be found at: https://github.com/HERA-Team/HERA_plots/blob/master/data_inspect_2458037.ipynb.

Out [8]:



The plots that we are particularly interested in are the "Mean Vij Modified z-Score" and the "Redundant Visibility Correlation Modified z-Score". We can clearly see that antenna 50 is an outlier, which suggests that it's "dead". For this reason, we will ignore the position of antenna 50 in further plots.

6. Switching antenna labels

Write in the original data file and the create a modified data file (e.g. `modified_data_file_list`). Define the atnennas that you want to swap. Here, we are switching antennas 0 and 2, identified by the variables `antA` and `antB`. Then, write a for loop and read in the data file and modified data file. What we hope to see is a plot that will identify antennas 0 and 2 as outliers.

In [9]:
```
#For loop
#Read in data file
#Modify data file
#Write in a new data file (modified_data_file_list)

data_file_list = ['/data6/HERA/data/2458042/zen.2458042.12552.xx.HH.uv',
                  '/data6/HERA/data/2458042/zen.2458042.12552.yy.HH.uv',
                  '/data6/HERA/data/2458042/zen.2458042.12552.yx.HH.uv',
                  '/data6/HERA/data/2458042/zen.2458042.12552.xy.HH.uv']

modified_data_file_list = ['/data4/kdmcnult/data/zen.2458042.12552.xx.HH.uv.M1',
                           '/data4/kdmcnult/data/zen.2458042.12552.yy.HH.uv.M1',
                           '/data4/kdmcnult/data/zen.2458042.12552.yx.HH.uv.M1',
                           '/data4/kdmcnult/data/zen.2458042.12552.xy.HH.uv.M1']

#variables
antA = 0
antB = 2

uvdata = pyuvdata.UVData()
```

```
for file_name, output_file_name in zip(data_file_list,modified_data_file_list):

    uvdata.read_miriad(file_name)
    indA=np.where(antA == uvdata.ant_1_array)[0]
    indB=np.where(antB == uvdata.ant_1_array)[0]
    uvdata.ant_1_array[indA]=antB
    uvdata.ant_1_array[indB]=antA
    indA=np.where(antA == uvdata.ant_2_array)[0]
    indB=np.where(antB == uvdata.ant_2_array)[0]
    uvdata.ant_2_array[indA]=antB
    uvdata.ant_2_array[indB]=antA
    uvdata.write_miriad(output_file_name,clobber=True)
```

7. Modified data

Define a new variable, `am_modified`, as the modified data for the antenna metrics.

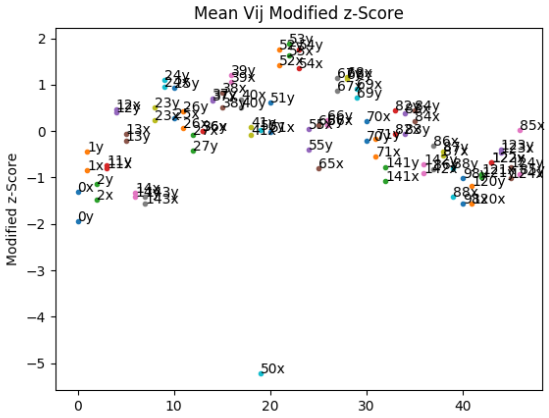
In [10]:
```
am_modified = ant_metrics.Antenna_Metrics(modified_data_file_list, reds, fileformat = 'miriad')

am_modified.iterative_antenna_metrics_and_flagging(crossCut = 5, deadCut = 5)
```

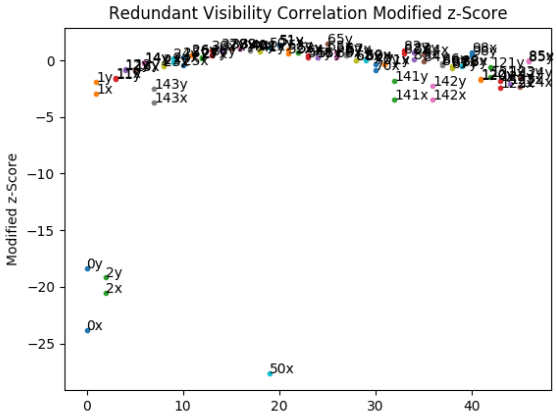
8. Plot the modified data

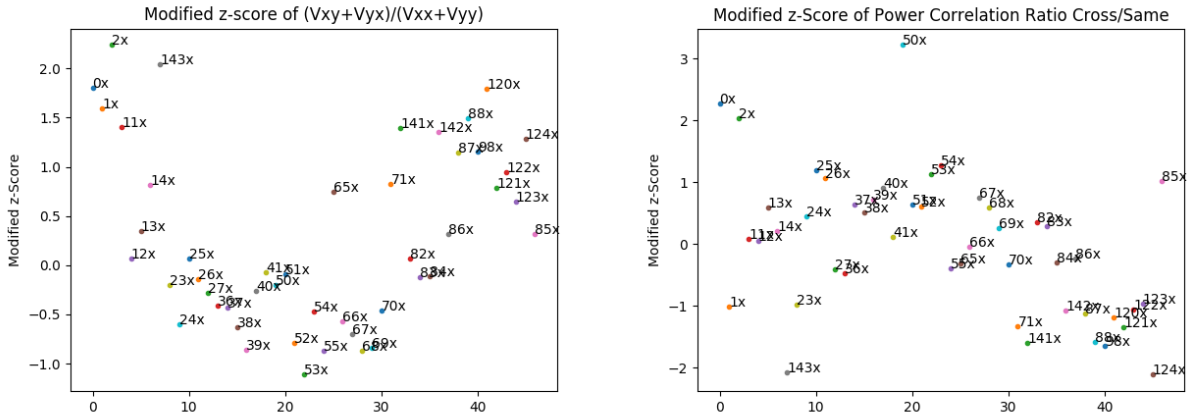
Repeat the steps in **Section 5** for the modified data, `am_modified`.

In [11]:
```
#print "Ant Metrics for {}".format(am_modified.metrics)

plt.figure()
plot_metric(am_modified.finalModzScores['meanVij'],
            title = 'Mean Vij Modified z-Score')

plt.figure()
plot_metric(am_modified.finalModzScores['redCorr'],
            title = 'Redundant Visibility Correlation Modified z-Score')

plt.figure()
plot_metric(am_modified.finalModzScores['meanVijXPol'], antpols=['x'],
            title = 'Modified z-score of (Vxy+Vyx)/(Vxx+Vyy)')

plt.figure()
plot_metric(am_modified.finalModzScores['redCorrXPol'], antpols=['x'],
            title = 'Modified z-Score of Power Correlation Ratio Cross/Same')
```

Out [11]:







From the plot of the "Redundant Visibility Correlation Modified z-Score" we can see that antennas 0 and 2 have clearly been identified as outliers. This is because the plots are measuring the correlation of baselines between antennas that are next to one another, such as 0-1, 1-2, 11-12, etc.

9. Check additional antennas

Repeat the steps in **Sections 6-9** to check the baselines of two different antennas. In this case, we are swapping antenna 71 and antenna 141, identified by the variables `antC` and `antD`. We chose these antennas because antenna 141 is an outlier and we are interested in seeing what will happen if we swap a "bad" antenna with a good one. We expect to see them both show up as outliers due to the separation between them, meaning their baselines are not correlated.

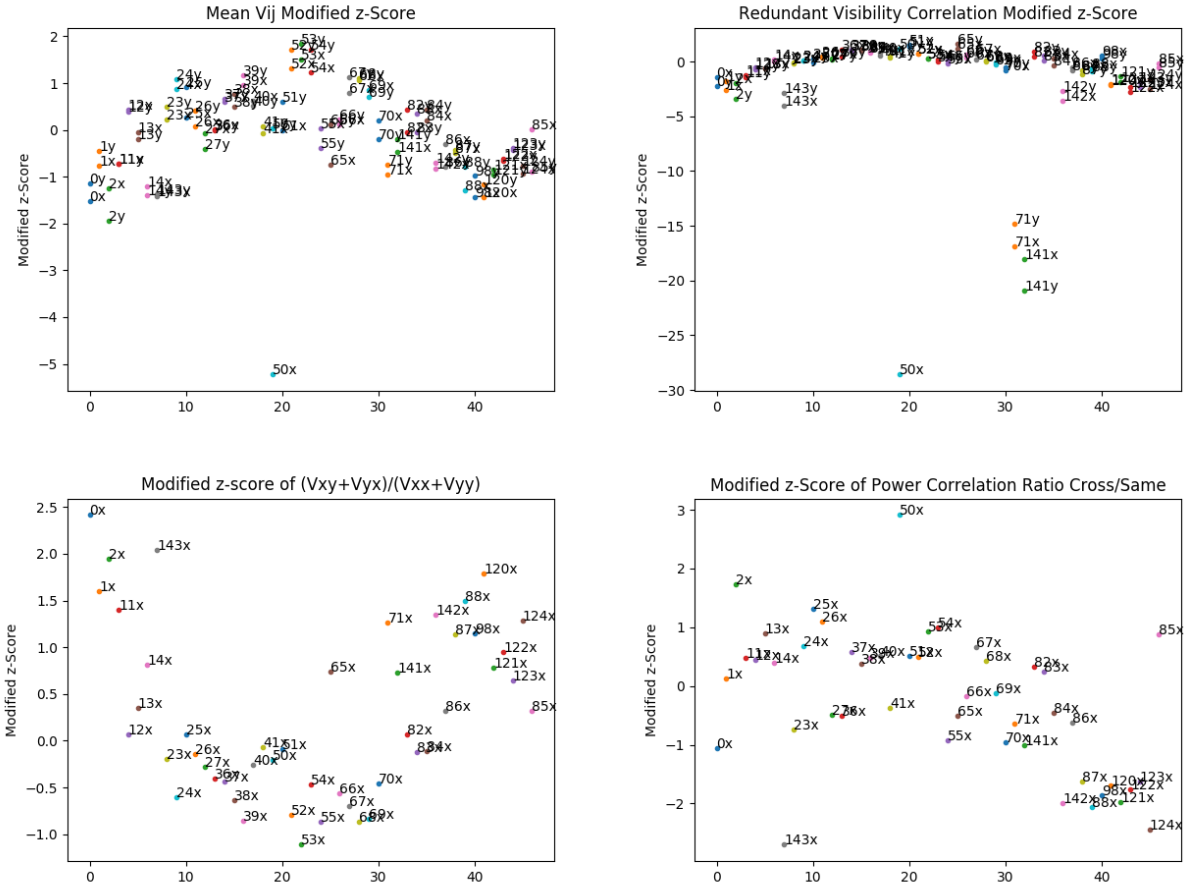
In [12]:
```
#For loop
#Read in data file
#Modify data file
#Write in a new data file (modified_data_file_list)

data_file_list = ['/data6/HERA/data/2458042/zen.2458042.12552.xx.HH.uv',
                  '/data6/HERA/data/2458042/zen.2458042.12552.yy.HH.uv',
                  '/data6/HERA/data/2458042/zen.2458042.12552.yx.HH.uv',
                  '/data6/HERA/data/2458042/zen.2458042.12552.xy.HH.uv']

modified_data_file_list = ['/data4/kdmcnult/data/zen.2458042.12552.xx.HH.uv.M2',
                           '/data4/kdmcnult/data/zen.2458042.12552.yy.HH.uv.M2',
                           '/data4/kdmcnult/data/zen.2458042.12552.yx.HH.uv.M2',
                           '/data4/kdmcnult/data/zen.2458042.12552.xy.HH.uv.M2']

#variables
antC = 71
antD = 141

uvdata = pyuvdata.UVData()

for file_name, output_file_name in zip(data_file_list,modified_data_file_list):

    uvdata.read_miriad(file_name)
    indC=np.where(antC == uvdata.ant_1_array)[0]
    indD=np.where(antD == uvdata.ant_1_array)[0]
    uvdata.ant_1_array[indC]=antD
    uvdata.ant_1_array[indD]=antC
    indC=np.where(antC == uvdata.ant_2_array)[0]
    indD=np.where(antD == uvdata.ant_2_array)[0]
    uvdata.ant_2_array[indC]=antD
    uvdata.ant_2_array[indD]=antC
    uvdata.write_miriad(output_file_name,clobber=True)
```

In [13]:
```
am_modified = ant_metrics.Antenna_Metrics(modified_data_file_list, reds, fileformat = 'miriad')

am_modified.iterative_antenna_metrics_and_flagging(crossCut = 5, deadCut = 5)
```

In [14]:
```
#print "Ant Metrics for {}".format(am_modified.metrics)
```

```
plt.figure()
plot_metric(am_modified.finalModzScores['meanVij'],
            title = 'Mean Vij Modified z-Score')

plt.figure()
plot_metric(am_modified.finalModzScores['redCorr'],
            title = 'Redundant Visibility Correlation Modified z-Score')

plt.figure()
plot_metric(am_modified.finalModzScores['meanVijXPol'], antpols=['x'],
            title = 'Modified z-score of (Vxy+Vyx)/(Vxx+Vyy)')

plt.figure()
plot_metric(am_modified.finalModzScores['redCorrXPol'], antpols=['x'],
            title = 'Modified z-Score of Power Correlation Ratio Cross/Same')
```

Out [14]:



Again, by looking at the plot "Redundant Visibility Correlation Modified z-Score" we can clearly see that antenna 71 and 141 have been identified as outliers.

10. Conclusion

Our antenna swap metric has successfully identified the baseline correlation, or lack thereof, between currently operational antennas and identified outliers as expected. Moving forward, we can use this metric in determining whether or not antennas are "dead" (such as antenna 50) and to ensure that each antenna is connected to the correct "box".