A Quick Guide to Using Maestro and Wqed
November 7, 2018

Maestro is a Matlab based photometry reduction package. The installation script can be found on the DARC website (www.physics.udel.edu). To install maestro, create a working directory and download the installation script and the file "maestrofiles.tar" to that directory. Extract the files from maestrofiles.tar. The installation script will need to be edited to set the installation path to one of your choosing. After editing the installation script, run it in the working directory. Make sure the files from maestrofiles.tar are included in that directory.

Instructions for installing the WQED tools can be found on the DARC website.

Photometry data acquired by different observers/telescopes come in many forms. The first step is to retrieve the data and determine how to extract the individual images. In the simplest case, the data comes in a tarball and can be extracted using "tar -xvf tarballname". It is best, especially if one is dealing with observations from multiple observers over multiple days, to implement some sort of directory structure to keep everything organized. When extracting images from a given night/observer, be certain to do the extraction in the appropriate directory.

Maestro has a few general commands that might be helpful:
>> maestro commands – this gives a list of commands available and a brief description. Photometry reduction will mostly use buildfield, inspect, and reduce

Maestro is run from the directory that contains the individual images and calibration files. Maestro requires 1) a list of bias images, 2) a list of dark images, 3) a list of flat images, and 4) a list of science images. The default names for these lists are biaslist, darklist, flatlist, and objlist. You can call them something else, but you will have to let Maestro know what the file names are.

If the observer has followed protocol, the appropriate keyword is contained in the individual filenames. If this is the case, the lists may be created with the following commands (in the directory containing the images):
>> ls -1 *bias*.fits > biaslist
>> ls -1 *dark*.fits > darklist
>> ls -1 *flat*.fits > flatlist
>> ls -1 *objectname*.fits > objlist (in this case, objectname should reference the observed object)

The next step is to check the calibration files to make sure they are usable. Maestro has a nice command to accomplish this:
>> maestro inspect @biaslist
>> maestro inspect @darklist
>> maestro inspect @flatlist
The output from the inspect command (printed to screen) is filename, mean, median, standard deviation. By comparing results for biases and darks, you can determine if these calibration files were properly obtained. You can also make sure that files labeled "bias" actually are biases, etc. For darks, observers sometimes send multiple series of dark frames with different integration times. It is not required that they all have the same integration times, but it is the usual routine to use only darks with the same integration times.

For flats, it is useful to check that the mean/median are within the linear response of the CCD (i.e. not saturated). If maestro raises any questions about the usefulness of the calibration files, it is best to view a sample of each using DS9.

Maestro will automatically reduce the 30 brightest stars in the science frames, in order of brightness. If you wish your variable star to appear at the top of final list of counts (this is important for WQED, which works with the output of maestro), it is necessary to build a field file that maestro will use to order the data reduction. This step needs to be done only once. The field file may be reused for subsequent observations of the same object from the same telescope. Building a field file does not require using all the images from a given night. In the interest of time, standard protocol is to use about 50 images. We have already created a list of all the science images (objlist). Edit objlist so it contains only 50 consecutive images. To create your field file:

>> maestro buildfield -w fieldfilename

The field file is stored in ~/.Maestro/fields. Edit your field file. The first line of the field file is a comment line that lists the reference image that maestro used. Open this image with DS9. The remaining lines give star number, x, y, and counts for the stars in the field. Locate your variable star in the image and note its x and y coordinates. Locate these coordinates in the field file. Copy that line and paste it below the comment line in the field file. Standard procedure is to replace "Unknown #" with the target star's name. You may then remove the original line from the field file, so maestro does not reduce the variable star twice.

We are now ready to proceed to actual data reduction. First, recreate your original objlist so it contains all of the images, not just the 50 we used for building a field.
>> ls -1 *objectname*.fits > objlist
>> maestro reduce -s fieldfilename
Maestro will look for biaslist, darklist, flatlist, and objlist. It will tell you how many files it is using of each type, check file integrity, check file size, and check fits keywords. It will build a master bias and calculate the readnoise, build a master dark, build a master flat, and calculate the gain. It will then search for stars on the images, and then proceed to the photometry reduction.

If all goes well, the output is written to a directory with a name based on the date of the observations and the field file used. For example, output from observations from PJMO on 20171024 were written to 20171024_011433_g29_pjmo2017/
>> cd 20171024_011433_g29_pjmo2017/
brings you to a directory that contains files for counts, noise and signal for multiple apertures. We are concerned here with the "counts" files. These files contain the counts by aperture for all of the stars maestro reduced.

We need to select the proper combination of aperture (in pixels) and comparison stars to produce the best signal to noise. We have multiple ways to accomplish this. First, maestro outputs its suggestion as a file called something like counts_SN_optimized10.5. Here, 10.5 is the suggested best aperture.

The next tool is a python script written by Mike Montgomery. It will look at all of the combinations of aperture and comparison stars and determine the one that produces the best signal to noise light curve. This script is located in the ~/maestro/pythonscripts/app_select_maestro.py. Copy the script into your current directory.

>>python app_select_maestro.py (might need to use python2 if that is not your default). Also, matplotlib needs to be available to run this script).

The app_select_maestro.py script does not always work, so we have a third way to choose the best aperture.
>>chooselc counts_ -a -d
The -a flag has to do with maestro output, and the -d flag will divide the first star in the counts_ files (the variable) by the second star. Chooselc will pop up a window that allows you to scroll through the light curves divided by the first comparison star for various apertures by hitting “enter” in the pgplot window. This usually allows you to select the best aperture, with input from the other tools. If there is a problem with the comparison star (it disappears from the image for some reason), you can try
>>chooselc counts_ -a
This will give undivided light curves, which are still helpful for selecting the best apertures.

When you have selected the best aperture, type “q” in the chooselc pgplot window. Your original terminal will ask you to select the aperture you liked best. Chooselc will then output a file with a *.sec” ending for the aperture you selected.

Now we need to add a header and other information to the selected file to create a file that WQED can read (more about WQED in a minute). Whiff adds this header information. The usual format is:
>> whiff firstimage.fits -l *.sec -o obscode -s starname
Explanation:
1) firstimage.fits is the first observation image. It is automatically copied to the output directory by maestro. It should already be in the directory.
2) *.sec uses a wild card to refer the the file generated by chooselc
3) obscode is the observatory code contained in whiff.dat. Whiff.dat is located in ~/.wet. Whiff.dat contains important information for each observatory, such as EXPTIME keywords, date/time keywords, etc. Take a look at whiff.dat to see the format. Observatories can be added to this file at any time.
4) starname refers to the target name listed in stars.dat. Stars.dat is also located in ~/.wet. It contains the coordinates for the target star. This is important information for determining the barycentric correction.

Whiff outputs a file with a .wq extension. This is the file that is read by WQED.

The next step is to read the .wq file with wqed.
>>wqed *.wq
Wqed allows you to manipulate the light curves (marking bad points, dividing by comparison stars, removing polynomials). A full wqed manual called help.ps is in ~/manuals. This gives a complete explanation of available commands. The usual basic procedure (all done in the pgplot window):
1) mark garbage points: {control}+g marks everything outside of a given box as garbage, which g marks everything within the box as garbage. [control}+g is great for getting all the extreme bad points, while g works will on focused areas, like clouds and edges of light curves.
2) z is also a useful command at this point. z zooms in on everything inside of the box. It is good to help pinpoint bad points.
3) divide by comparison stars. The command for this is:
    a) select the variable star light curve by typing “1” in the wqed pgplot window
    b) the command for dividing is “/”. You may choose as many stars as you wish, but it is best to use only those with higher counts than your variable star.

4) removing a polynomail – sometimes dividing by a comparison star does not completely remove long time scale trends. This is due to the fact that pulsating white dwarfs are much bluer than most stars.

a) select the variable star by typing “1” in the wqed pgplot window

b) the command for polynomial removal is “f”. Enter the polynomial order you wish to remove.

5) writing out the data

a) select the variable star by typing “1” in the wqed pgplot window

b) the command for writing out the entire data set is “w”. This will output a file with a .lc1 extension. The file will contain a header followed by times and fractional intensities.

The next step is to check your results.

>>fitlc *.lc1

Fitlc will display the FT for your light curve. You can scroll between the FT and the light curve by typing “l” in the pgplot window. This is a good check that you haven’t left any bad points in the light curve

Combining light curves from different nights is accomplished using the weld command:

>>weld filelist -o outputcurves.dat

Explanation:

filelist contains a list of files to be combined, one file per line

outputcurves.dat is the combined light curve.

You can now run fitlc on the combined light curve.