MRR Perusal Webpage Creation

General Summary:

The overall task of running the Perusal Webpage functions can be divided into 5 major modules:

1. Storm Detection
    a. MRR radar data is loaded into Matlab from either cached SIMP files or from reprocessed Ukoln files.
    b. Storms are detected in the radar data and saved in a Matlab struct.
2. Graphing
    a. Surface observation data is converted to a predefined Matlab struct for use in graphing statistics (e.g. pressure, temperature, wind speed, etc).
    b. Graphs for statistics are generated and saved.
    c. Graphs of MRR radar data are generated and saved.
3. HTML Creation
    a. The front page HTML file displaying the table of storms is created, with links to storm page HTML files in the "Day" column of the table.
    b. Storm page HTML files are created, with links to graphs of MRR radar data and statistics. These HTML files also include a table of statistical data (Maxima, Minima, and Averages).
4. Updating Statistics and Functionality
    a. Averages of statistics for all storms in the front page HTML file are calculated and put into a table at the top of the page.
    b. Buttons in storm page HTML files are activated.
    c. User fills in missing table values by analyzing graphs in the storm page HTML files.
5. Additional Functionality
    a. Ability to create HTML files for different types of storms.

Limitations:

- *Storm Detection* – The detection algorithm uses a column averaging method to find both where storms are, and detect the edges of each storm. In some cases, storms may overlap and cause the algorithm to return storms that may last for long periods of time (i.e. >24 hours). The algorithm will display a message when a storm was saved with a duration of >24 hours. It will be up to you to decide if you want to change the date range in the variable returned, or proceed with the modules. If you decide to leave long storms the way they are, your MRR radar graphs will have a decreased temporal resolution.

- *Graphing* – When loading surface observation data into the predefined Matlab struct, the function you will use is called **gen_metar2struct**. This function primarily accepts only ONE type of txt file. In order to use the function as it is, you must obtain an ASCII txt file of simplified observations from NOAA's NCDC website, found here: <insert link>. If you are receiving surface observations whose txt file is not of the same format as a simplified ASCII txt file from NOAA NCDC, then you will have to modify the code in the function to parse your own txt file. If you have no access to surface observations near your MRR, or you cannot find any observations, you can still make the webpage, but you will only have the links to the MRR images.

  IMPORTANT:

  The meteorological statistics gathered from the surface observations are based off of what NCDC hourly observations typically include. The statistics supported throughout my functions are: temperature, pressure, relative humidity, wind speed and direction, and precipitation. If you are using surface observations that DO NOT include one of these statistics, then you will have to send me an email. These functions currently do not support choosing which statistics to graph and show in the tables. Until I have built in this support, please email me if you are unable to find surface observations which have all of the statistics listed above.

Updates:

Any updates made to this suite of software will be noted here. Major or important changes are in bold. Please let me know via email if function(s) do not work following an update. It is always best to remain up to date with the development.

Modification Date	Functions Altered	Description
01/15/2014	MRR_Add_Storms caller_MRR_read_ukoln_nc	- **MRR_Add_Storms now supports creating a webpage without any surface observations.** - Read_ukoln caller function now displays progress more effectively.

Instructions

Make sure that you have read all of the limitations before you proceed with the instructions. All the functions that you will need are included in the MRR_Perusal.zip file at the bottom. You MUST save all of the functions in the zip file to your MATLAB path.

Due to the complexity of the functions and the length of execution, no wrapper function was made to run everything at once. Alternately, a sample script has been supplied at the bottom of the page. This script includes a similar step-by-step procedure to what you will read below, but not quite as in-depth. The script is merely for you to get an idea of how your function calls should look throughout the process.

Step 1:

Start Matlab. Decide which working directory you will use. It is preferred that you save the functions from the zip file to the working directory you are using. You can keep all the functions in a folder if you like, just make sure that folder is included in the Matlab path (File -> Set Path).

Place the frontpage.html and stormpage.html files in your working directory.

Open frontpage.html in a text editor of your choosing. Please read the header message about editing the html file. You will need to fill in and/or change some of the information that will be specific to your MRR/location. **Warning:** If you change anything that I have not outlined as available to change, you run the risk of one of the Matlab functions not working properly. Please do not change anything outside of what I have instructed.

Create a folder in your working directory called “storms”. This folder will hold all of the storm page html files as well as all of the graph images.

Step 2:

This step covers loading the MRR data into a Matlab struct. You should have access to either cached SIMP files or reprocessed Ukoln files from your MRR radar. If you do not have either of these file types, see our article: MRR Processing.

If you have SIMP files, then you will use the **caller_MRR_simp2matrix** function. If you have Ukoln files, you will use the **caller_MRR_read_ukoln_nc** function. If you have access to both file types and you’re unsure of which to use, I recommend using SIMP files because they are much smaller and can be loaded much more quickly than the Ukoln files.

The caller functions allow you to read multiple MRR files into one Matlab struct. The naming convention of your MRR files should be uniform, but they must be in the correct form. For SIMP files, the convention is “yyyymmdd.MRR.simp.asc” (e.g. 20091225.MRR.simp.asc would be the file for Dec 25, 2009). For Ukoln files, the convention is “prefix_yyyymmdd.ukoln.mrr.nc” (e.g. miami_20091225.ukoln.mrr.nc would be the file for Dec 25, 2009 in Miami). The input parameters on both the SIMP caller and Ukoln caller are identical and listed here:

- *directory*: The absolute path to the folder that contains the files you are loading.
- *prefix*: If anything comes before the date portion of the file name, specify it here.
- *datestart*: The date of the first file you are loading.
- *dateend*: The date of the last file you are loading.
- *suffix*: Anything that comes after the date portion of the file name, including the extension.
- *rows_per_day*: Specify the number of records given per day in the MRR file. Typically, this number is 1440.

The output for this function is the Matlab struct that was created with the MRR data.

Example:

```
MRR = caller_MRR_simp2matrix(directory, prefix, datestart,
dateend, suffix, 1440);
```

Step 3:

Now that you have loaded the MRR data, you can run the storm detection function: **MRR_Storm_Detection**. Before you do so, you must declare a settings struct which will hold some important values. The fields that you must include are specified here:

- *Z_threshhold*: The reflectivity (dBZ) threshold value for detecting a storm in the MRR data. This can be whatever value that you deem reasonable for the types of storms usually recorded at the location of your MRR radar. I usually set this value to around 10 dBZ.
- *Z_threshhold_2*: The dBZ threshold value for detecting the edge of a storm in the MRR data. I usually set this value to around 3 dBZ.
- *rows_per_day*: This number will be the same as the one you use in the caller function (i.e. 1440).

To call the function, you must input the MRR struct you loaded in the previous step, as well as the settings struct. This function will return a filtered struct of the MRR data as well as an array of structs that holds the date ranges of the storms that were detected.

Example:

```
[MRR_filtered dates] = MRR_Storm_Detection(MRR, settings);
```

Step 4:

As of the 01/15/2014 Update, you may skip this step if you do not have any surface observations. Otherwise, you will now need to load your txt file of surface observations into a Matlab struct using the **gen_metar2struct** function provided (See *Graphing* limitation if you have not already read it).

Assuming you have either a NOAA NCDC txt file, or have configured the function to your particular txt file, the function call is fairly straight forward, and requires the following input:

- *metarfilepath*: The txt file you are loading. If the txt file is not on your Matlab path, make it so.
- *savefilepath*: What you want to save the Matlab struct as. Something simple will do (e.g. mymetar_struct.mat). The .mat file extension is how Matlab stores its variables in a file. It is recommended that you include .mat in this input. *Note that if you set this input to zero, the function will return the Matlab struct of surface observations, instead of saving it to a .mat file.
- *metartype*: Set this to zero, unless you are using a different portion of the code as your txt file parser.
- *minuteinterval*: REQUIRED for txt files from NCDC. This is the minute interval that the hourly surface observations are recorded on at the beginning of the txt file. Look at the txt file to find out what the minute interval is.

There are no outputs for this function.

Example:

```
gen_metar2struct('miami_surfaceobs.txt', 'miami_metar.mat', 0,
'55');
```

<u>Step 5:</u>

In this step, you will call one function to complete both the *Graphing* and *HTML Creation* module. The function you must call is **MRR_Add_Storms**. Before you call the function, there are a few fields you must add to the settings struct you declared in Step 3. You should still have settings in your variable workspace. Here are the fields you need to add:

- *frontpage_orig*: File name of the frontpage html file to which you are adding storms (e.g. 'frontpage.html').
- *frontpage_saveas*: File name of the frontpage html file after the new storms have been added. This should be not the same as frontpage_orig, so you can save the old html file the way it was as a backup (e.g. 'myfrontpage.html').
- *metarfile_mat*: File name of the Matlab struct of surface observations. Include the .mat extension.
- *empty_stormpage*: File name of the template stormpage.html file.

To call the function, you must input the MRR struct and dates struct that were returned by **MRR_Storm_Detection** in Step 3, along with the settings struct. There are no outputs for this function.

Example:

```
MRR_Add_Storms(MRR_filtered, dates, settings)
```

Step 6:

At the top of your frontpage html file will be a button that says “Show Statistics”. After Step 5, when you click on this button to view the averages for the entire table, there will still be dashes for each statistic. In order to get the averages, you must call the **calculate_avgs** function. Here are the inputs you will need to supply:

- *htmlfilepath*: File name of your frontpage html file.
- *returndata*: For your purposes, always set this input to zero. This controls whether or not the averages will be returned (used by another function).
- *numstorms*: The number of storms in your frontpage html file. It is very important that this number is correct, otherwise, the averages will be incorrect.
- *is_cat*: For your purposes, always set this input to zero. This tells the function what type of html file it is reading (used by another function).

There are no ouputs for this function. Note that running this function on your frontpage html file will overwrite it. It is a good idea to save your html file somewhere else, just in case.

Example:

```
calculate_avgs(‘myfrontpage.html’, 0, length(dates), 0)
```

Step 7:

In each storm’s html file, there are four navigation buttons at the top. In order to enable these buttons, you must run the **add_next_prev** function. Here are the inputs you must supply to this function:

- *htmlfile*: File name of your frontpage html file.
- *homelocation*: Relative path you want to set for the “Home” button in each storm. I always make the “Home” button go back to the index of the directory the front page html file is in, so my relative path is “../” because each stormpage html file is in the storms folder. You are free to make your own Home page though.
- *stormsummarypg*: The relative path to your frontpage html file from the storms folder.
- *is_cat*: For your purposes, always set this input to an empty string. This input is used by another function to determine what type of html file is being read.

There are no outputs for this function. Similar to the last step, this function will overwrite your frontpage html file. Again, it is a good idea to make a backup of your frontpage html file somewhere else.

Example:

```
add_next_prev(‘myfrontpage.html’, ‘../’, ‘../myfrontpage.html’, ‘’)
```

Step 8:

At this point, there are still columns of missing data in the table of storms. Unfortunately, I have not been able to develop an efficient way of automatically determining a storm's type, AGL (Above-Ground Level), continuity, and statistical trends. To fill in these columns, you must analyze the graphs of the MRR and statistical data that have been generated in each storm's html file.

For convenience, I have created a runnable JAR file that makes it very easy to change the html code in your frontpage html file. The JAR file is included at the bottom of this page. When you run it, it will open a window with several fields for you to fill. First, select your frontpage html file using the file chooser at the top of the window. Each storm has a unique date. Fill in the date fields for the storm you want to change. If you leave a text field blank or a checkbox in its default selection (e.g. Select trend), then the program will not change that value in the html file. The rest should be pretty self-explanatory. Once you click Update, you should see a confirmation message that the storm was successfully updated. Refresh the html file in your browser to make sure that everything looks correct.

Once again, and I cannot stress this enough: please keep backups of your frontpage html file as you work through these steps. Otherwise, you may have to start all over if something goes wrong.

Step 9:

Depending on the amount of storms you have to analyze, Step 8 may take a little bit of time. However, once you have finished Step 8, you will be able to complete the last step. This step will produce separate html files for each type of storm in your frontpage, as well as html files for Continuous and Scattered storms. Doing this makes it more efficient to examine similar storms. I would not recommend doing this step if you have more storms to add to the current frontpage you are working on later. Meaning, only do this step if the season or year that you are doing it on is completely finished. If you do this before a season or year is complete, you will have to do it all over again later.

The function that you will call in Matlab to do this is **create_categories**. Before you can call this function, you must create some folders and copy some files:

1. In the same folder as the "storms" folder and your frontpage html file, create a folder called "categories".
2. In this "categories" folder, create two folders: "storms_type" and "storms_continuity".
3. Copy all of the HTML files in the "storms" folder into the "storms_type" folder AND the "storms_continuity" folder.
4. Copy the "storms" folder and paste it one directory UP from where it currently is.

Here is an example of how your folder hierarchy should look:

```
C:\
   User\
      MATLAB\
         myfrontpage.html
         storms\
            (stormpages and images)
         categories\
            storms_type\
               (stormpages only)
            storms_continuity\
               (stormpages only)
      storms\
         (stormpages and images)
```

Once your file structure is set up similar to this, you may run the **create_categories** function. Here are the inputs you must supply to run this function:

- *htmlfilename*: Name of your frontpage html file (DO NOT INCLUDE .html EXTENSION).
- *numstorms*: Number of storms in the table of your frontpage.
- *numyears*: This corresponds to the number of seasons you have for your MRR radar. It's also the number of choices in the drop down menu at the top of your frontpage that says "Choose a season to view…" For the first year of perusal pages, this would just be 1.

There are no outputs for this function. Keep a backup of your frontpage somewhere else, just in case.

Example:

```
create_categories('myfrontpage.html', length(dates), 1)
```

End Instructions

Contact Information:

Spencer Rhodes
Undergraduate Research Assistant
Clouds and Precipitation Processes and Patterns Group
Department of Marine, Earth, and Atmospheric Sciences, NCSU
Personal email (preferred): spencer.rhodes2@gmail.com
School email: srrhodes@ncsu.edu