RainyDay User’s Guide

Daniel B. Wright, Ph.D.

danielb.wright@wisc.edu

RainyDay User’s Guide

Overview of RainyDay:

RainyDay is a framework for generating large numbers of realistic extreme rainfall scenarios based on relatively short records of remotely-sensed precipitation fields. It is founded on a statistical resampling concept known as stochastic storm transposition (SST). These rainfall scenarios can then be used to examine the extreme rainfall statistics for a user-specified region, or to drive a hazard model (usually a hydrologic model, but the method produces output that would also be useful for landslide models). In contrast with other recent flood hazard modeling approaches, RainyDay is well suited to flood modeling in small watersheds[[1]](#footnote-1). The framework is made to be simple yet powerful and easily modified to meet specific needs, taking advantage of Python’s simple syntax and well-developed libraries. It is still a work in progress. Therefore, the contents of this guide may be out-of-date. I will attempt to keep the documentation in synch with major changes to the code. I would appreciate any feedback on this document and on RainyDay itself, so I can continue to improve both.

Please note that while this framework is relatively simple, it does require some basic understanding of hydrometeorological principles. In addition, the configuration of Python, as well as any desired alteration to the code, requires some basic familiarity with programming and scientific computing. While I will not provide unconditional technical support, I am interested in seeing people use this software and so I will provide assistance as much as possible.

It is essential that the user understand the basic principles of SST prior to running an analysis. This is because, given the degree of flexibility provided by RainyDay, it is very easy to make conceptual mistakes that will produce unrealistic results. A new reference specific to the RainyDay implementation of RainyDay will be forthcoming, but for now the user is directed to the following references on SST using rainfall remote sensing:

* Wright, D.B., J.A. Smith, M.L. Baeck. “[Flood Frequency Analysis Using Radar Rainfall Fields and Stochastic Storm Transposition,](http://onlinelibrary.wiley.com/doi/10.1002/2013WR014224/abstract)” Water Resources Research, 50 (1592–1615), 2014.
* Wright, D.B., J.A. Smith, G. Villarini, M.L. Baeck. “[Estimating the frequency of extreme rainfall using weather radar and stochastic storm transposition,](http://www.sciencedirect.com/science/article/pii/S002216941300187X)” Journal of Hydrology, 448(150-165), 2013.

RainyDay has two main steps: storm catalog creation, and storm transposition. By tweaking the parameter (“.sst”) file, the user can perform either one or both of these steps in a single instance. Please note that both steps, but in particular the storm catalog creation, can be time consuming, since the remote sensing datasets input can be very large. Runtimes of several hours are certainly possible. Runtimes for storm transposition should be approximately 1-10 minutes depending on the parameter set and whether diagnostic plots and movies are created.

The .sst file is described in more detail below. This document does not provide significant detail on the formatting of the input remote sensing data, but it is important to note that, due to the unfortunate variety of formats of various remote sensing datasets, a preprocessing step is needed to convert the original remote sensing files into a format that RainyDay will recognize. If you would like to know more about this format and preprocessing, please contact me. If feasible, I will share preprocessing scripts that I have created for different datasets, or directly share the datasets that I have prepared. This preprocessing in most cases involves selecting a certain geographic region, such as the contiguous United States, in order to reduce the size (and therefore the time that your computer has to spend loading the files) for the process.

A note about geographic projections: Earlier versions of RainyDay supported different geographic projections for some operations. This functionality was never perfected and for simplicity has been removed. RainyDay assumes that all input GIS data and coordinates are in the WSG84 geographic coordinate system (EPSG:4326, +proj=longlat +ellps=WGS84 +datum=WGS84 +no\_defs). Furthermore, input rainfall must be projected onto a regular rectangular latitude-longitude grid (though the grid spacing need not be the same north-south as east-west). All coordinates supplied by the user in the .sst file must be in latitude/longitude, with negative latitude indicating southern hemisphere and negative longitude indicating western hemisphere.

There are three main types of output:

* Diagnostic plots (and movies): These plots show different aspects of the rainfall fields and the storm catalog. These diagnostics are explained in more detail in the following sections.
* .FreqAnalysis file: This file provides the results from a rainfall frequency analysis for various return periods for an area and a rainfall duration, both of which are specified by the user. The results in the .FreqAnalysis file are roughly equivalent to an Intensity-Duration-Frequency (IDF) curve, which most civil engineers and hydrologists are familiar with. There are important differences, however. The frequency analysis from RainyDay is, by definition, a spatial average, rather than a point analysis. An IDF curve is typically based on rain gage observations, which typically measure rainfall over an area of approximately 0.1 m2. RainyDay uses remote sensing grid cells, which can cover areas of 106 to 109 m2. Therefore, caution is needed when comparing these results against conventional IDFs.
* Rainfall IDF plot: This provides a visual representation of the results provided in the .FreqAnalysis file.
* Spacetime rainfall scenarios: These are scenarios that can serve as input to a hazard model (typically a hydrologic model, but could also be a landslide model). In order to keep the size of these scenarios manageable, they do not consider the entire transposition domain but only the rainfall within a certain user-defined area (usually either a rectangular “box” or a watershed boundary).

Licensing Agreement

RainyDay is distributed under Version 3 of the GNU General Public License (<https://www.gnu.org/licenses/gpl-3.0.html>).

Disclaimer: I, Daniel Benjamin Wright, as the author of this software, will not be held responsible for the accuracy of the output and results it produces, nor for decisions taken based on said output and results. The input datasets can contain significant uncertainties that will impact the accuracy of output and results. Furthermore, the poor choice of parameters in RainyDay will negatively influence the output and results. Please consult the relevant academic literature and the RainyDay users guide prior to using this software.

How to configure and run RainyDay:

RainyDay can be obtained from the BitBucket respository: <https://bitbucket.org/danielbwright/rainyday>. For now, access is invitation-only. Please consult other sources for information on BitBucket and Git systems and syntax.

1. Install Python/IPython:

I strongly recommend using the [Anaconda distribution](https://store.continuum.io/cshop/anaconda/) from [Continuum Analytics](http://continuum.io/). Installation is easy, it comes with most of the packages that you’ll need, and has several options for code development environments (I use Spyder[[2]](#footnote-2), which comes as part of the Anaconda Launcher, but the other options might be worth examining).

The current version of RainyDay uses the following Python packages:

* StringIO
* os
* sys
* numpy
* scipy
* glob
* math
* timeit
* datetime
* time
* copy
* pylab
* pytz
* pickle
* mpl\_toolkits
* netCDF4
* Numba[[3]](#footnote-3)

2. Configure PYTHONPATH

There might be other ways to do this, but here is the one that I know (on Mac and Linux). In your home directory, create a file called “.bashrc”. This is easy to do. At a command line, simply type “vim ~/.bashrc”. Regardless of whether this opens a new or existing file, add the following to “.bashrc”:

export PYTHONPATH="${PYTHONPATH}:[proper path here]/RainyDayUtilities"

“[Proper path here]” denotes the path to wherever the RainyDay directory resides. This could be in with the rest of your Python libraries, or elsewhere.

3. Configure Python packages:

Try to run RainyDay using Python/IPython from a command line terminal:

$ python {RainyDay script name}.py

If you configured PYTHONPATH as explained in step 2, and you get an error related to not being able to find RainyDayUtilities, Type the following command at the terminal:

$source ~/.bashrc

That should set PYTHONPATH.

The script will not fully execute even if the packages are properly configured, because you haven’t supplied a parameter file (.sst file). But if any packages aren’t installed, you will get errors that will help you determine what packages haven’t been properly installed.

You can look at all packages currently installed under Anaconda using:

$ conda list

You should be able to install most packages in Anaconda using:

$ conda install {package name}

If you want to see a list of all packages that can be installed, and the correct names, use:

$ conda search

If you use the Anaconda distribution, netCDF4 and osgeo/GDAL should be the only packages that require extra steps. For these, you will need to have the requisite libraries/APIs (such as GDAL, HDF5, jpeg2000, etc., zlib) installed. Once those libraries are installed, you should be able to install both pacakges normally using “conda install {package name}.

Specifically:

For the netCDF4 package, you need to have the netCDF 4.X library, which, depending on how you configure it, may depend on other libraries such as HDF5. See [here](http://www.unidata.ucar.edu/software/netcdf/) for installation of the netCDF4.X library.

Similarly, RainyDay requires the GDAL library, which it interfaces with directly by issuing commands to the operating system. It also uses the Python osgeo/GDAL wrapper. These require the UnixImageIO framework, GEOS framework, SQLite3 framework, PROJ framework, and NumPy. Installation should be straightforward, since there are single binaries that contain all of these frameworks, however. See [here](http://trac.osgeo.org/gdal/wiki/DownloadingGdalBinaries) for information and options. To verify that GDAL is properly installed on your system, type “gdal-config” at the terminal.

As you install more packages, type “$ python {RainyDay script name}.py” at the command line. This will show you whether you’ve succeeded or if you still have more work to do to configure the packages.

4. Prepare the parameter file (.sst file)

The .sst file should contain comments which explain the different options. They are explained in more detail below. Though not thoroughly tested, I do not think the order in which these fields appear in the .sst file should matter at all.

An example .sst file is shown here (comments omitted for clarity):

#==============================================================================

# WELCOME TO THE RAINY DAY MODEL!

# USER DEFINED INFORMATION FOR STOCHASTIC STORM TRANSPOSITION

#==============================================================================

# BASIC INFO ON THE SCENARIO

MAINPATH /Users/dbwrigh3/Documents/SST/

SCENARIO SSTboxCMORPHIowaCity

RAINTYPE cmorph

RAINPATH /Volumes/DATAVAULT/DATA/CMORPH/CONUS/cmorph.\*.CONUS.nc

# SST PARAMETERS

CATALOGNAME BoxCatalogCMORPHiowacity\_24h.nc

CREATECATALOG false

DURATION 24

NSTORMS 100

NYEARS 1000

NREALIZATIONS 100

TIMESEPARATION 24

LATITUDE\_MIN 40

LATITUDE\_MAX 44

LONGITUDE\_MIN -94

LONGITUDE\_MAX -89

DIAGNOSTICS true

DIAGNOSTICMOVIES true

FREQANALYSIS true

SCENARIOS true

SPINPERIOD 5

RETURNTHRESHOLD 5

EXCLUDESTORMS none

EXCLUDEMONTHS 1,2,3,12

INCLUDEYEARS all

RESAMPLER uniform

COUNTSAMPLE poisson

SPREADTYPE ensemble

ROTATIONANGLE -15,+15,10

RETURNLEVELS 10,100,1000

POINTAREA basin

WATERSHEDSHP /Users/dbwrigh3/Documents/DATA/GIS/05412500\_TurkeyRiver\_GEOG.shp

WATERSHEDFLD 05412500\_TurkeyRiver\_GEOG

SHAPEPROJECTION geographic

WATERSHEDMASK 05412500\_TurkeyRiver\_GEOG\_cmorph.mask

BASEMAP /Users/dbwrigh3/Documents/SST/Basemaps/IowaBasemap.shp

BASEFIELD IowaBaseMap

POINTLAT 41.666667

POINTLON -91.533333

BOX\_YMIN 41.55

BOX\_YMAX 41.75

BOX\_XMIN -91.65

BOX\_XMAX -91.45

SENS\_INTENSITY false

SENS\_FREQUENCY false

INTENSDISTR 75,25,-0.15

Each parameter is explained:

MAINPATH: Directory where the RainyDay Python script is located and in which subdirectories will be created or modified.

SCENARIO: Name for the scenario. This will be the name of the subdirectory and the prefix for various output such including the frequency analysis file.

RAINTYPE: The rainfall input dataset. This defines certain properties (resolution, spatial extent, etc.) of the dataset. Currently these properties are hardcoded in the RainyDay script, but it is relatively easy to add properties of an additional dataset. There are quite a number of supported options, and I can make the associated data available upon request. The most useful option for now is “stageIVnewreproj”. You need the requisite dataset to be locally accessible.

RAINPATH: The location of the rainfall input NetCDF4 files. Only needed if you are creating a new storm catalog. Wildcards (“\*”) can and should be used.

CATALOGNAME: The name of the storm catalog. If the user specifies that a new storm catalog should be created (using the CREATECATALOG field, which is explained next), a new file will be created (and any existing one with the same name will be overwritten). If the user does not create a new catalog, the CATALOGNAME should refer to an existing storm catalog. This catalog resides at: MAINPATH/CATALOGNAME}.nc. The storm catalog will be a netCDF4 file.

DURATION: Duration of the rainfall accumulation period in hours. If you are not creating a new storm catalog but rather using one that has already been generated, this duration is compared against the duration of the storm catalog. If the specified duration is shorter than that of the storm catalog, the rainfall duration is adjusted accordingly. If the specified duration is longer than that of the storm catalog, an error message is generated. There are no simple guidelines for how to choose this duration. If you are just examining rainfall frequency analysis, then any duration can be used. If spacetime rainfall scenarios are being generated to drive a hydrologic model, then it is best to choose at least 24 hours. The duration should be longer for larger watersheds. It is not recommended to use conventional notions of response time or time of concentration as the criteria for selecting an appropriate duration. Generally speaking, this duration should exceed the time of concentration. In principle this duration could be as long as you want to facilitate modeling of very large watersheds. This is not recommended however, and for a number of reasons will not likely produce satisfactory results. A modified version of RainyDay is under development that will work better for very large watersheds.

NSTORMS: How many storms to include in the process. This number will be inherited from the existing storm catalog, if you are not creating a new storm catalog. This number will be reduced accordingly in the following situations: 1.) you are not creating a new catalog and you wish to exclude certain storms (EXCLUDESTORMS); exclude certain months that were not previously excluded in the creation of the storm catalog (EXCLUDEMONTHS), or exclude certain years via the INCLUDEYEARS option.

NYEARS: How many years of annual maxima rainfall to be synthesized.

NREALIZATIONS: How many NYEARS-long sequences to be generated.

TIMESEPARATION: The minimum separation time in hours between two storms in the storm catalog. During storm catalog creation, when a new rainfall event is being evaluated for inclusion, its timestamp is compared against those already in the catalog. If the difference in timestamps between the new storm and an existing one is less than TIMESEPARATION, the larger event (in terms of rainfall accumulation) is retained and the smaller event is discarded.

It is recommended the following values not be changed between runs unless a new storm catalog is created. The program is untested and may be unpredictable if these values are changed while an existing storm catalog is used:

LATITUDE\_MIN: Southern boundary of transposition domain.

LATITUDE\_MAX: Northern boundary of transposition domain.

LONGITUDE\_MIN: Western boundary of transposition domain.

LONGITUDE\_MAX: Eastern boundary of transposition domain.

DIAGNOSTICPLOTS: Set to “true” to produce diagnostic plots. This can be a time consuming process and is only recommended when a new storm catalog is created. There are three types of plots produced. Figure 1 shows an example of a diagnostic plot of storm total rainfall. One such plot will be produced for each storm in the storm catalog. The various components of this figure are explained in the descriptions of other parameters.

Figure 1: Sample diagnostic plot of storm total rainfall from a 48-hour duration event over eastern Iowa ending at 02:00 on April 19, 2013. Input data is the National Stage IV multisensor QPE Product (<http://www.emc.ncep.noaa.gov/mmb/ylin/pcpanl/stage4/>).

Figure 2 shows another diagnostic plot: the probability of storm occurrence. The blue circles show the locations of the storm centers that were identified through the storm catalog generation. A 2-dimensional Gaussian kernel smoother is used to estimate the spatial probability of occurrence based on these locations. The reddish background colors indicate areas higher probability. Do not be alarmed if the background colors do not extend all the way to the boundary of the transposition domain. The technique is still working properly. You will notice that, unlike Figure 1, the boundary defined by , CONTROL is not shown in Figure 2, because the storm identification and transposition only takes place within the transposition domain.

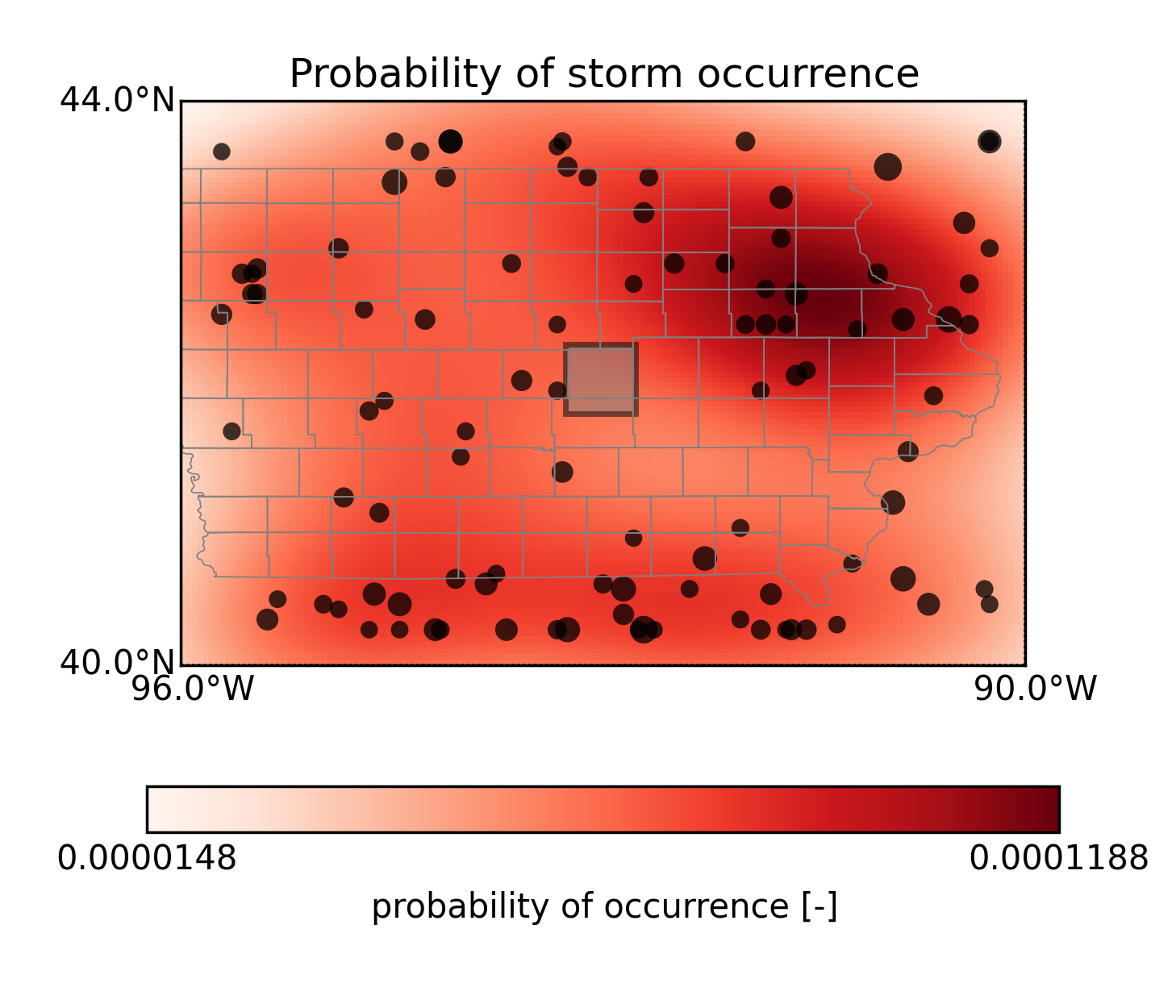


Figure 2: Spatial probability of storm occurrences (red shading) based on locations of storms in storm catalog (blue circles).

Figure 3 shows the final diagnostic plot: a map of the average storm rainfall calculated from all storms in the storm catalog. It gives a picture of the regional climatology of heavy rainfall. The map also shows the locations of the storm centers from the storm catalog.

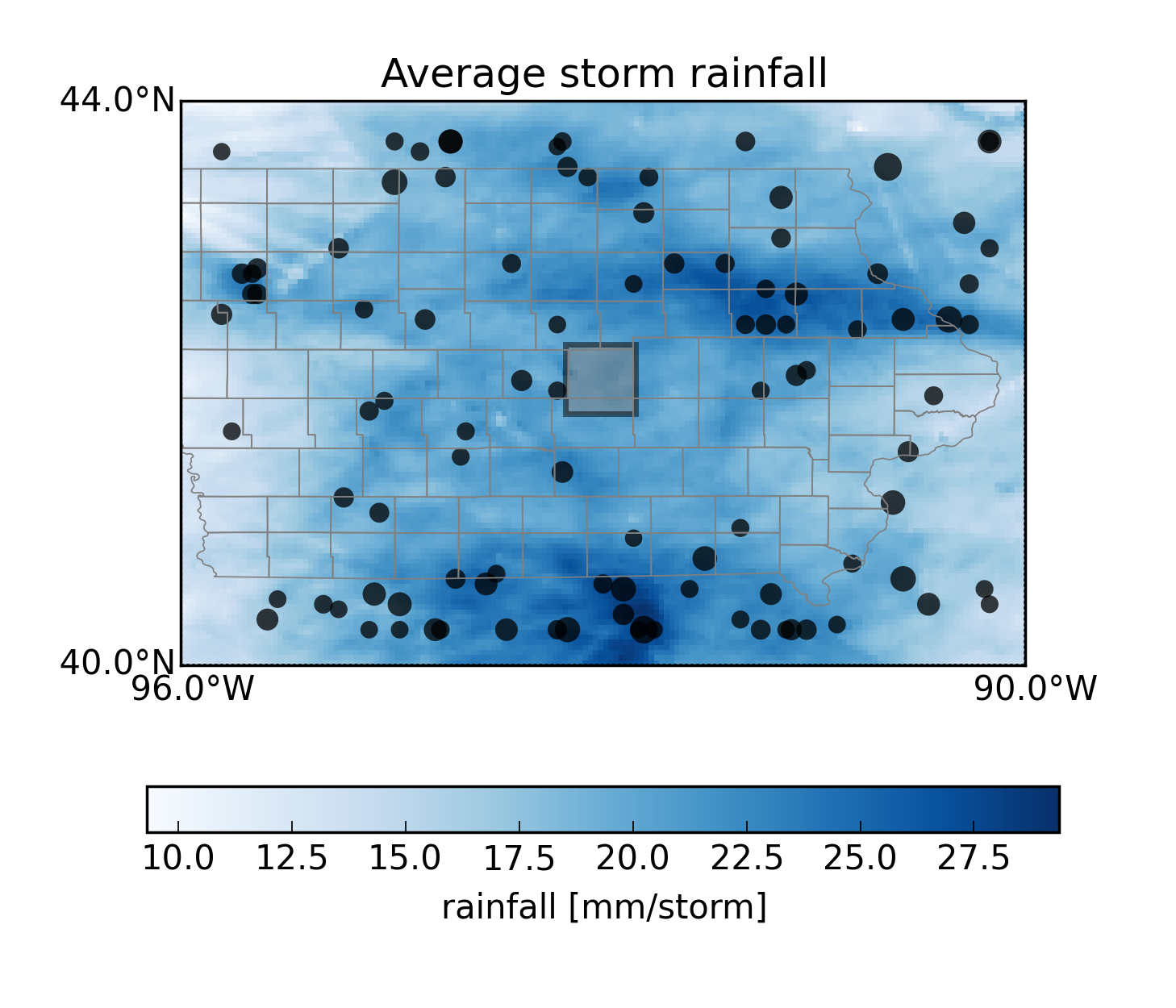


Figure 3: Average storm rainfall and location of storm centers.

DIAGNOSTICMOVIES: To create diagnostic movies or rainfall fields for each storm in the storm catalog. This is only really useful if you are creating a new storm catalog. This step can be time-consuming so should be avoided when possible. Note that movie creation can be highly platform-dependent. Adjustments of video drivers, frame rates, etc. may be necessary to avoid error messages and/or create quality movies. This functionality may not work if you run RainDay within Spyder.

FREQANALYSIS: Create a “.FreqAnalysis” file based on the rainfall annual maxima generated. The “internal variability” will be calculated for each return period between 1 and NYEARS based on the spread in NREALIZATIONS.

SCENARIOS: Create watershed-specific spacetime rainfall scenarios corresponding to each annual rainfall maxima. These scenarios can be used to drive a hydrologic model or landslide model. If you aren’t planning on doing any modeling, there is no need to generate them. The code is currently created such that these scenarios are grouped by realization, so that there are NREALIZATIONS netCDF files created, each of which contains at most NYEARS scenarios. If RETURNTHRESHOLD is greater than 1 (highly recommended) the number of scenarios will be reduced, perhaps dramatically, depending on the specified return period threshold.

SPINPERIOD: This only applies if SCENARIOS is set to “true.” It provides the ability to “pre-pend” a rainfall period to the beginning of the output scenarios. Options are either “false” or an integer number of days. So if SPINPERIOD is set to 5, then five days of rainfall will be prepended to each scenario. Currently, the code is written such that the prepended rainfall time period for each scenario is drawn randomly from the rainfall that precedes the storms in the storm catalog from a period of one month earlier to one month later. For example, if SPINPERIOD is set to 5, and if a scenario is being written that is a transposition of a storm that occurred in July, then the preceding rainfall will be randomly selected from the 5-day rainfall periods preceding all storms in the storm catalog that occurred in June, July, and August. This “spin-up period” enabled by SPINPERIOD is potentially very useful, because it allows whatever hazard model is being used to achieve a realistic initial moisture condition prior to the onset of the main storm. In the case of floods, it also allows the establishment of realistic baseflow conditions. Caution should be used, however, since for large areas, SPINPERIOD could create very large arrays and there might be memory-related crashes.

RETURNTHRESHOLD: Minimum return period for which a spacetime rainfall scenario will be written. This can dramatically reduce the number of (uninteresting) low intensity spacetime rainfall scenarios included in subsequent hazard modeling.

EXCLUDESTORMS: To exclude certain storms that are part of the storm catalog but which you do not wish to include in subsequent frequency analysis or spacetime rainfall scenarios. There are two main reasons for doing this: 1.) it allows you to examine the sensitivity of the method to the inclusion of particular storms, and 2.) it allows you to exclude storms that have unrealistic properties (easily diagnosed via the diagnostic plots). This is an important consideration when using a ground-based radar rainfall input dataset, in which radar artifacts can seriously affect the results. Multiple storms must be comma-separated. You can consult the diagnostic plots for the storm numbers.

EXCLUDEMONTHS: Months (1-2 digit numeric) to be excluded from the storm catalog creation and/or subsequent frequency analysis and spacetime rainfall scenario generation. Months must be comma-separated.

INCLUDEYEARS: Years (4-digit numeric) to be included in the storm catalog creation and/or subsequent frequency analysis and spacetime rainfall scenario generation. Years can be either specified as a range, separated by “-“ or as a comma-separated list. Examples: (2002-2005 is equivalent to 2002,2003,2004,2005).

RESAMPLER: Defines how the resampling should be done. Currently the options are “uniform” or “kernel.” With “uniform,” the resampled storms are randomly located according to a 2-D uniform distribution. With “kernel” the resampled storms are randomly located according to a 2-dimensional Gaussian kernel density estimate based on the locations of the original storms in the storm catalog. A general recommendation is only to use the kernel density method if you have a strong reason to believe it will provide better results than the uniform method. For example, if there are clear differences in storm locations based on topographic features, then the kernel option is preferable, but the estimated density is sensitive to small sample sizes from the input rainfall dataset. An example of the 2-D Gaussian kernel can be seen Figure 2.

COUNTSAMPLE: Defines the method used for generating the number of storms to be transposed for each synthetic “year.” Options for COUNTSAMPLE are either “poisson” or “empirical.” If “empirical” is selected, then the generator will be a Poisson distribution. The rate parameter for the distribution will be estimated from the number of storms in the storm catalog divided by the number of years in the dataset. If “empirical” is selected, the generator is an empirical distribution determined by the counts per year in the storm catalog. Note that a minor statistical artifact may arise in regions with extremely rare rainfall, because there is a constraint that every synthetic year must have at least one storm. But for most situations this shouldn’t be a problem. The option for the empirical distribution is useful in cases where the occurrence of extreme rainfall is over or under-dispersed. To evaluate this formally, you would compute the index of dispersion (the variance of the counts-per-year divided by the mean of the counts-per-year) and determine if it is significantly different from 1.0 via bootstrapping (or possibly some other technique).

SPREADTYPE: Defines the type of “ensemble spread” you want to calculate for inclusion in the .FreqAnalysis file and the IDF figure. If you want the full ensemble spread for any exceedance probability, put “ensemble.” If you want only the spread between the X/2 and100-X/2 quantiles, put “{X}”. “ensemble” is recommended.

ROTATIONANGLE: RainyDay allows for stochastic rotation of storms. There are a few things to note. First, this can be quite slow for large storm catalogs and for high-resolution rainfall data. Second, it should be used with caution, since in the real world, storm orientation is in many cases tightly tied to terrain such as topographic features, or tied to atmospheric phenomena such as high-pressure ridges. This means that rotation can produce storms that would be unrealistic in the real world. Nonetheless, it is a potentially useful feature when used in a restricted way. Omitting the line entirely or specifying “none” will prevent rotation. If rotation is desired, specify it in the following way “X,Y,Nangles”. X must be less than zero and Y must be greater than zero. Nangles specifies the number of discrete “angle bins” between X and Y that will be used. The speed of the rotation scheme will depend greatly on the choice of Nangles, and relatively low values are recommended. Keeping values of X and Y relatively small (for example, X=-30, y=+30) will keep rotation more physically reasonable.

RETURNLEVELS: Specify certain return periods for which you want to produce output scenarios and frequency analyses.

ACCELERATOR: Certain parts of the code, and in particular the storm catalog creation can be quite slow, particularly with high-resolution rainfall datasets. RainyDay has a few options to speed this up, using either “weave” from older versions of Scipy, or “Numba.” Numba is highly recommended. The options for ACCELERATOR are “none”, “weave”, “numba”.

POINTAREA: This defines the area that will be used in the calculation of rainfall accumulations. Options include “point” (in reality a single rainfall grid cell, rather than a point estimate), “box,” or “basin.” Point is not recommended, particularly for ground-based radar rainfall datasets, because it will be very sensitive to radar artifacts. “Basin” in principal provides the most precise estimates, but it is unlikely that it will produce very different results than those using “box” to approximate the size of the watershed. “box” is also useful for comparing results from rainfall datasets that have different input resolutions, as the procedure will aggregate the rainfall estimates to the size of the box. Note that if you wish to perform this aggregation, that it is wisest of the box is at least as large as the input resolution of the coarsest dataset.

WATERSHEDSHP: If “basin” is specified for POINTAREA, then either a “.mask” file or a basin boundary shapefile must be provided. WATERSHEDSHP provides the path and filename to the shapefile. RainyDay will convert this shapefile into a “.mask” array if necessary (see below). The rasterization is accomplished through the GDAL function “gdal\_rasterize,” which MUST be available through a command line terminal. To check, type “$ gdal\_rasterize” in a terminal. The shapefile should contain only one watershed boundary and no other feature. The method is completely untested for multiple features, will likely crash, and will certainly produce poor results.

SHAPEPROJECTION: Projection information for the shapefile. A previous implementation of the rasterization step allowed the user to specify a different projection, but for several reasons this capability was removed. All shapefiles should be in the geographic projection EPSG:4326: <+proj=longlat +ellps=WGS84 +datum=WGS84 +no\_defs>.

BASEMAP: If “false,” only a standard and relatively coarse map of land-water boundaries and national borders will be shown on the diagnostic plots and movies. If a path is provided to a shapefile, this shapefile will be loaded and displayed, assuming it is properly configured. A useful source of administrative boundaries for these basemaps is the [GADM Database](http://www.gadm.org/). You can download country-specific or a global shapefile. If you use a large shapefile (such as administrative boundaries for the globe or a large country such as the USA), however, it is strongly recommended that you export only the features you absolutely want, since the shapefile plotting is very slow in the matplotlib package. This exporting can be done using a separate GIS such as QGIS or ArcMap. One could also change the resolution of the basemap provided from matplotlib, but it is highly recommended that you not adjust the basemap resolution options in the source code, due to the very slow plotting speed of higher-resolution options. NOTE: if POINTAREA is set to “basin” and an appropriate basin boundary shapefile exists, it will also be plotted on these diagnostic plots.

BASEFIELD: If BASEMAP is used, then BASEFIELD should be set to the name of the appropriate field in the basemap shapefile.

POINTLAT: The latitude of the analysis point, if POINTAREA is set to “point.” The program will find which pixel contains this point.

POINTLON: The longitude of the analysis point, if POINTAREA is set to “point.” The program will find which pixel contains this point.

BOX\_YMIN: The southernmost boundary of the analysis box, if POINTAREA is set to “box.” The program will “snap” this box to the closest rectangular grouping of pixels.

BOX\_YMAX: The northernmost boundary of the analysis box, if POINTAREA is set to “box.” The program will “snap” this box to the closest rectangular grouping of pixels.

BOX\_XMIN: The westernmost boundary of the analysis box, if POINTAREA is set to “box.” The program will “snap” this box to the closest rectangular grouping of pixels.

BOX\_XMAX: The easternmost boundary of the analysis box, if POINTAREA is set to “box.” The program will “snap” this box to the closest rectangular grouping of pixels.

Caution is needed when comparing two different rainfall datasets with differing resolutions, since the snapping won’t necessarily produce a box with the same dimensions.

SENS\_INTENSITY: This allows easy sensitivity analysis to rainfall intensity. Options are either “false” (no alteration of intensity) or a numeric percentage change. For example, an input of 5.0 means that the intensity of rainfall will be increased by 5%.

SENS\_FREQUENCY: This allows easy sensitivity analysis to rainfall frequency. Options are either “false” (no alteration of storm rate of occurrence) or a numeric percentage change. For example, an input of -15.0 means that the rate of storm occurrences will be reduced by 15%.

INTENSDISTR: The user has the option of deriving the rainfall intensities from a GEV distribution (for example, based on a rain gage dataset). This option should be used with caution, particularly due to the spatial sampling differences between rain gages and remote sensing products. (NOTE: this option only applies the intensity sampled from the GEV distribution to the generated annual maxima. This means that it is not suitable for evaluating the space-time substitution effect via simulation. If this note makes no sense to you, don’t worry about it!).

4. Run RainDay

This is the (hopefully) easy part. At the command line, type:

$ {full path if terminal is not currently in the same directory as the script)}.py {full path to parameter file}.sst

5. Explanation of RainyDay NetCDF files

This section describes the two types of output files from RainyDay: storm catalogs, and “rainfall scenarios.”

|  |  |
| --- | --- |
| Storm Catalog | Rainfall Scenario |
| Purpose: contains all necessary rainfall information needed to conduct SST using RainyDay; basically it is a list of storms that have been identified through the current or a previous run of RainyDay, along with the full set of corresponding rainfall spacetime fields and dates/times  Dimensions: (N \* T \* Y \* X) = NSTORMS \* (DURATION ÷ input data temporal resolution) \* ([LATITUDE\_MAX - LATITUDE\_MIN] ÷ input rainfall north-south spatial resolution) \* ([LONGITUDE\_MAX - LONGITUDE\_MIN] ÷ input rainfall east-west spatial resolution)  Variables (dimensions):   * ‘rainrate’ (N \* T \* Y \* X): The spacetime rainfall fields for each of N storms that compose the storm catalog, each storm having T fields of size Y\*X * ‘time’ (N\*T): The dates and times for each rainfall field. For each of N storms, there are T entries† * ‘latitude’ (Y): Latitude•, ordered north to south, of the y-coordinate of the rainfall spacetime fields * ‘longitude’ (X): Longitude•, ordered east-to-west, of the x-coordinate of the rainfall spacetime fields * ‘gridmask’ (Y \* X): Mask showing the accumulation shape that is used to identify the storms; could be a watershed outline, a rectangle, or a single pixel * ‘ylocation’ (N): the north-south location that corresponds to the upper-left corner of the maximized rainfall of the size/shape of the gridmask for each storm * ‘xlocation’ (N): the east-west location that corresponds to the upper-left corner of the maximized rainfall of the size/shape of the gridmask for each storm * ‘basinrainfall’ (N): Average rainfall (in mm) over the size/shape of the gridmask for each storm | Purpose: contains all necessary rainfall information to drive a hazard model (hydrologic model, landslide model, etc.) with the output from the SST procedure. Note that one “realization” is written to each file, so if ‘NREALIZATIONS’ is greater than one, multiple rainfall scenario files will be produced. Each one will contain a sequence of  Dimensions: (n \* t \* y \* x) = NREALIZATIONSª \* (DURATION‡ ÷ input data temporal resolution) \* (gridmask height ÷ input rainfall north-south spatial resolution) \* ( gridmask width ÷ input rainfall east-west spatial resolution)  Variables (dimensions):   * ‘rainrate’ (n \* t \* y \* x): The spacetime rainfall fields for each of n storms that compose the storm catalog, each storm having t fields of size y\*x * ‘time’ (n\*t): The dates and times for each rainfall field. For each of n storms, there are t entries† * ‘latitude’ (y): Latitude•, ordered north to south, of the y-coordinate of the rainfall spacetime fields * ‘longitude’ (x): Longitude•, ordered east-to-west, of the x-coordinate of the rainfall spacetime fields * ‘ylocation’ (n): the randomly selected north-south location of the upper-left corner of the transposition that produces the each t\*y\*x rainfall spacetime series * ‘xlocation’ (n): the randomly selected east-west location of the upper-left corner of the transposition that produces the each t\*y\*x rainfall spacetime series * ‘basinrainfall’ (n): Average rainfall (in mm) over the size/shape of the gridmask for each transposition * ‘returnperiod’ (n): estimated rainfall return period for each transposition * ‘stormnumber’ (n): Which “member” of the storm catalog was used to generate each transposition |

‡Note that T and t need not be the same. If RainyDay is run using an existing storm catalog, the user can define any t, as long as it is less than or equal to T.

†All date/time stamps correspond to the end of the accumulation period. For example, if a rainfall dataset has hourly resolution, then the date/time stamp 5/5/2005 12:00Z corresponds to the time period from 5/5/2005 11:00Z to 5/5/2005 12:00Z.

ªIf RETURNTHRESHOLD is greater than 1, then n will be less than NREALIZATIONS. This is a practical way to reduce the number of simulations you will run, and reduce the sizes of these rainfall scenario files.

•Note that all latitude and longitude vectors relate the coordinates of the upper-left corner of the grid cell, not the lower-left or center, as some other coordinate systems do.

1. It is not possible to provide a particular watershed size above which RainyDay is not well-suited, but in general it will work better for basins in which floods are the result of singular rain events (which could last hours up to multiple days), rather than combinations of distinct rain events over an extended time period which act in concert to create the flood wave. A version of RainyDay suitable for larger basins is currently under development. Please contact me if you are interested. [↑](#footnote-ref-1)
2. **NOTE:** While attempting to run RainyDay through Spyder, I found that Python/IPython was initially not able to properly locate the RainyDayUtilities module (essentially just Python code that contains various functions called by the main RainyDay script). This problem did not occur while running Python/IPython from the command line. If you want to run RainyDay in Spyder and encounter this same problem, copy the entire RainyDayUtilities directory into your Python site packages directory (such as “/Users/{username}/anaconda/lib/python2.7/site-packages/” on a Mac). It appears that this issue can be fixed by adding the desired RainyDayUtilities path to the PYTHONPATH manager (found in the “Python” dropdown menu in Sypder). Then you’ll have to restart Sypder. It is also possible that you’ll have to try running RainyDay once from a terminal command line to activate the PYTHONPATH functionality within Spyder. This is all pretty confusing and appears to be a bug within Spyder. [↑](#footnote-ref-2)
3. Optional, but highly recommended! [↑](#footnote-ref-3)