Probability of induced extreme precipitation due to TC locations: User Manual

Version 0.1.1

Kenneth T. Valverde Hernández
kenneth.valverdehernandez@ucr.ac.cr
February 18, 2022

Contents

1 Introduction

This document has the purposed of guiding the user to run the Python script developed to compute probabilities of induced extreme precipitation due to tropical cyclones (TCs) location.
The tool takes historical precipitation data of a station or region and fits the empirical data with a theoretical probability density function to find its extreme precipitation thresholds, then a hexagonal grid is implemented to characterized tropical cyclone locations and analyze the induced precipitation of those tropical cyclones, computing the probability of extreme precipitation according to the thresholds found earlier.
Essential Requirements are described to achieve the correct execution of the code and are mandatory if the user do not plan to change the code to its needs. Usage section is focused on how to customize the hexagonal grid and graphic representation of both the grid and the computed probabilities.

2 Essential Requirements

2.1 Software Requirements

To run the developed tool you need to have installed in your computer Python 3.7+ and some libraries, including its dependencies. This can be done either in a Windows or Linux environment. Also, PDF reader is required to show the created images.

2.1.1 Python

Python can be install easily from the official web page `https://www.python.org/`, selecting the corresponding Operative System and following the instructions. In the installation process add the program to your system variable PATH.

2.1.2 Libraries

In case you already have installed Anaconda, the Anaconda Powershell is a good option to install the libraries needed. Otherwise, using the windows PowerShell with the PIP package manager is advised, but it needs to be installed first.
Generally to install the libraries, using use the line: conda/pip install library_name in the Powershell will do the work. Next, a guide of how to install the libraries in windows using the PIP package manager in the windows PowerShell:

- Distfit: Starting with the Distfit library installation is recommended because it will install other libraries used like numpy and matplotlib as dependencies.

  » pip install distfit

- pyGMT: First, download and installation of Generic Mapping Tools (GMT) software is necessary. Follow the instructions of this repository `https://github.com/GenericMappingTools/gmt/blob/master/INSTALL.md`, all operative systems are included. Add the program to your PATH. Then, use the line

  »pip install pygmt

Once all the previous libraries are successfully installed, you should be able to run the Tool.py file.

2.2 Data Format

Data of precipitation, stations location and topical cyclones should be in the same directory along the Tool.py script, and keep the following format.

- All files should be a .txt file and columns are separated by one tab.
- Precipitation Data: Should be named "precipitation_data" where rows represent precipitation measures over a day and each column is a different station. No label should be added to the file. See Figure 1.
- Stations location data: This file should be named "stations_location" with each row representing a station and 3 columns with the next headers: "ID" to identify the station, "X" longitudinal coordinate and "Y" latitudinal coordinate. See Figure 2.
- TCs data: The name of the file should be "TCs_data" and with the next header included: "event", "day", "month", "year", "lat", "lon", "hour", "windkph", "press", "typemax" for the 10 columns correspondingly. Each row includes a different event or the same event with a new measure. See Figure 3.

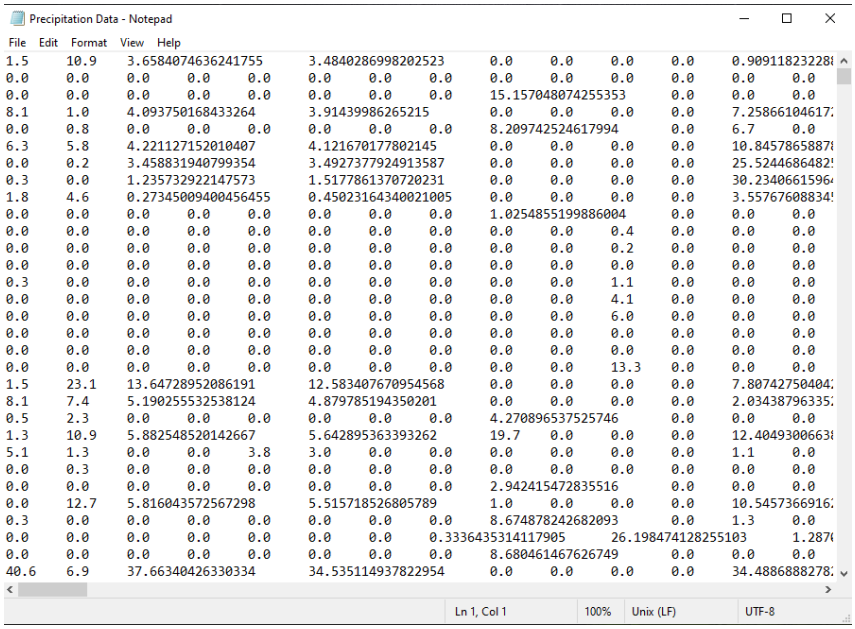

Figure 1: Format required for precipitation data.

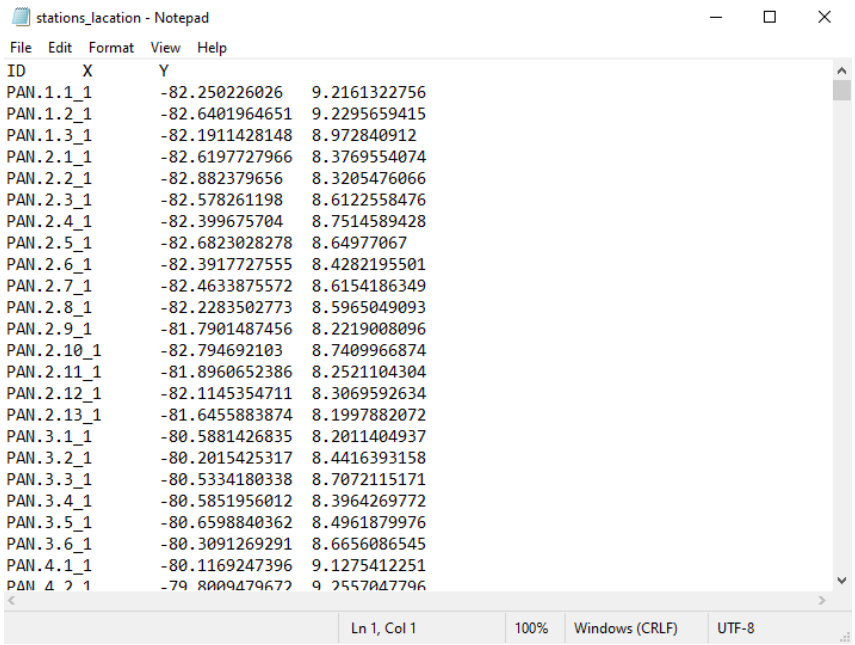

Figure 2: Format required for stations location data.

.

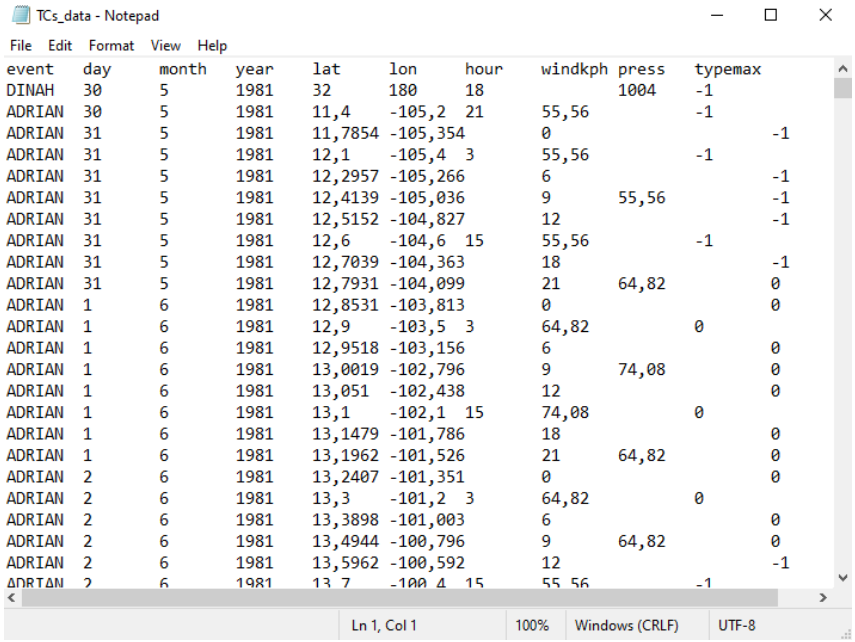

Figure 3: Format required for TCs data.

3 Usage

Open the Python IDLE, then open the tool.py script and execute it.

3.1 Getting the thresholds

To get the thresholds of each station just use the function get_thresholds() on the Python command line. This will create a .csv file in the same directory containing the ID of the stations, the theoretical PDF name, and the computed thresholds. This may take a while.

» get_thresholds()

3.2 Set Up

Before you start using the tool, you need to figure out:

1. The best region and settings for the grid plot (Grid and TCs data). This is just a region where the hexagonal grid will be placed, as you will do in the next subsection.
2. the best region and settings to show the stations (final results and most important of the two).

This can be achieved by running the demo function (see examples below) on the python command line:

» demo(region, *projection, *style) *optional arguments

- region (List of floats): min/max longitude and latitude coordinates to define the region of interest. [min longitude, max longitude, min latitude, max latitude]
- *projection (str): projection and size of the map. Default value = "M17.5c"
- *style (str): style and size of the scatter plot in the map. Default value = "c0.125c"

Style of the markers, map projection and its size can be changed. Default values are: projection is Mercator and 17.5cm image size ("M17.5c"), markers are circles of size 0.125cm ("c0.125c"). To change those check the available markers https://www.pygmt.org/latest/gallery/#symbols-and-markers and projections https://www.pygmt.org/dev/projections/index.html. Note: It is important to remember the regions and settings of your preference, you will use them later.

Example 1: Finding the best region for the stations.

» demo([-95, -75, 5, 20])

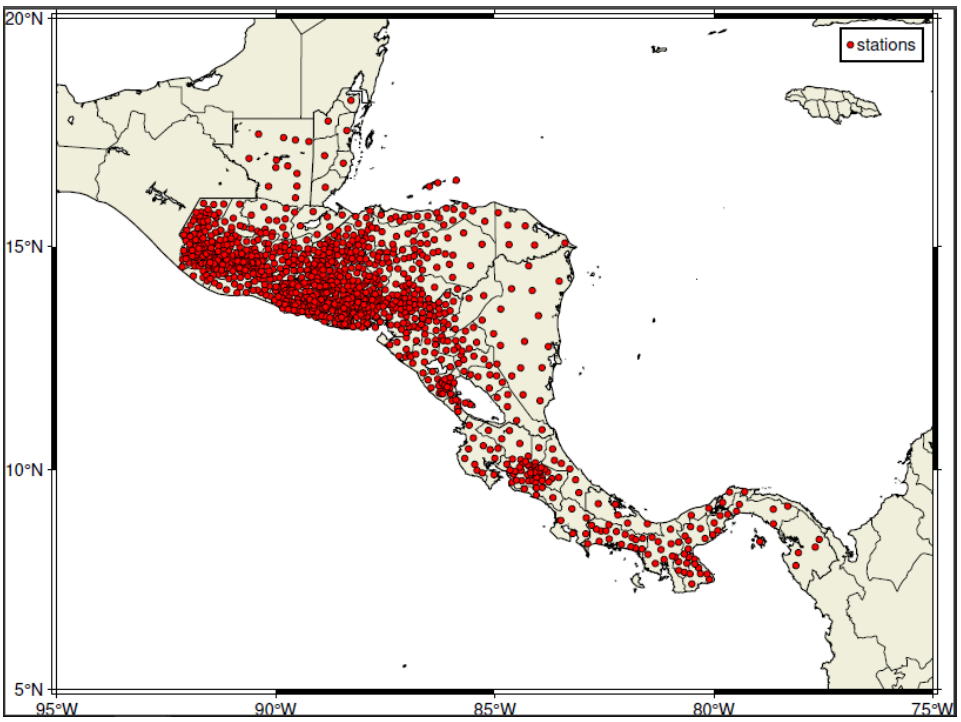

Figure 4: Define the region of stations using only the coordinates argument.

Example 2: Finding the best region for the stations and changing settings.

» demo([-95, -75, 5, 20], style = "c0.075c", projection = "Y35/30/12c")

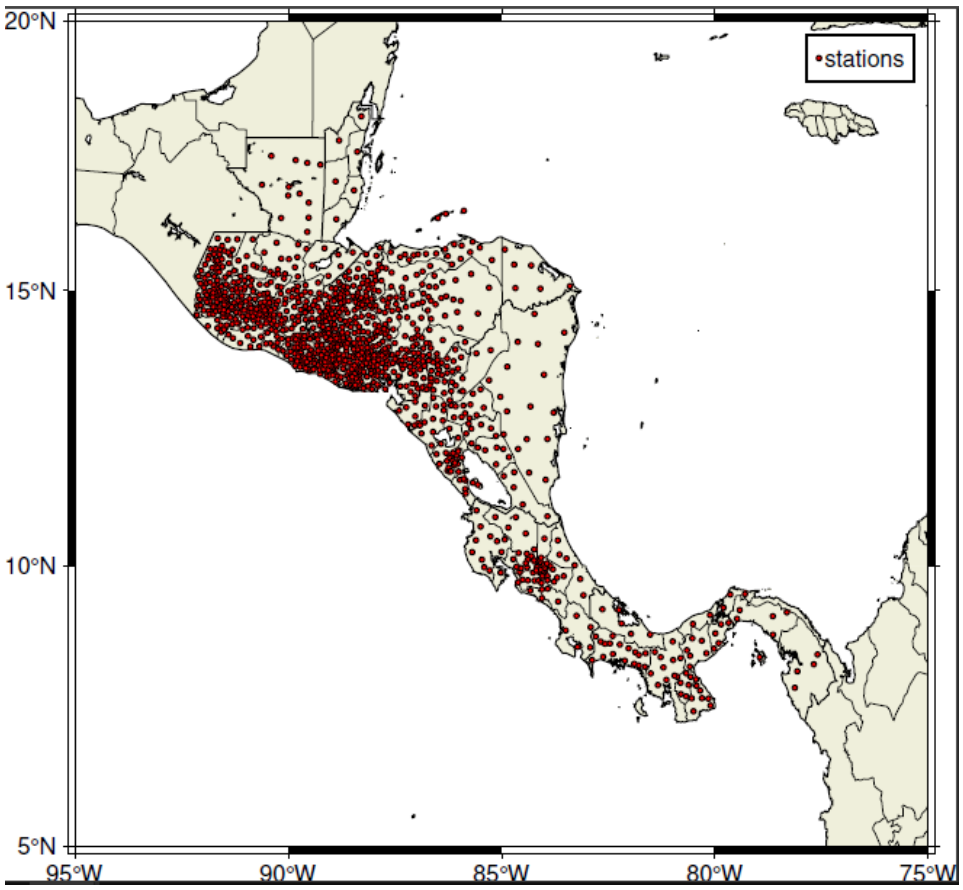

Figure 5: Same region as Figure 4, but changing the marker size to 0.075cm, projection to cylindrical equal-area and image size to 12cm.

Now that you have chosen the regions and settings (write it down) to display the 1. grid image and 2. stations image, the tool can be use in the next subsection.

3.3 Executing the tool

Run the main function in the Python command line:

» main(compute_probs, period, grid_region, stations_region, grid_location, cols, rows, apt_size, *hour_correction, *style_grid, *style_stations, *projection_grid, *projection_stations, *img_save) *optional arguments

Description of all parameters:

- compute_probs (boolean): if True the script will compute the probabilities, if False the program will end showing the hexgonal grid and TCs image, without computing probabilities. (This is used to find the appropriate hexagonal grid)
- period (List of 2 str): time period of analysis, [star, end]. The number of days should match the number of rows of the precipitation data. (see example 3).
- grid_region (List of 4 floats): same as the demo function [min longitude, max longitude, min latitude, max latitude]. Coordinates to define the region that shows the grid and TCs.
- stations_region (List of 4 floats): min/max longitude and latitude coordinates to define the region that shows the stations.
- grid_location (Tuple of 2 floats): longitude and latitude coordinates of the **lower left hexagon center of the grid.** This parameter allows you to place the hexagonal grid.
- cols (integer): number of columns of the grid.
- rows (integer): number of rows of the grid.
- apt_size (float): double the size of the hexagon apothem in degrees.
- *hour_correction (integer): hours necessary to match TCs data to the precipitation data time zone. Default value = 0
- *style_grid (str): marker and size of the scatter plot for TCs daily average position in the grid image. Default value = "c0.075c".
- *style_stations (str): marker and size of the scatter plot for the stations in the stations image. Default value = "c0.15c".
- *projection_grid (str): map projection and size of the image for the grid image. Default value = "M17.5c".
- *projection_stations (str): map projection and size of the image for the stations image. Default value = "M15c".
- *img_save (str): directory where you wish to save the final images results. Default value = "Images".

In the same way, play with this function until you find the best configuration by just modifying the grid location, columns, rows and apt_size (**setting the first argument as False**). Every time you run the function it will save the grid image. If you want to change the image settings (markers style or projection and image size) use the corresponding optional arguments.

Example 3 Creating the grid of preference. Note that the fourth argument [118, 123.5, 21, 26.5] has nothing to do with the grid customization.

» Grid = main(False, ["1/1/1981", "31/12/2020"], [115, 145, 0, 35], [118, 123.5, 21, 26.5], (127.5, 15), 7, 5, 2.5)

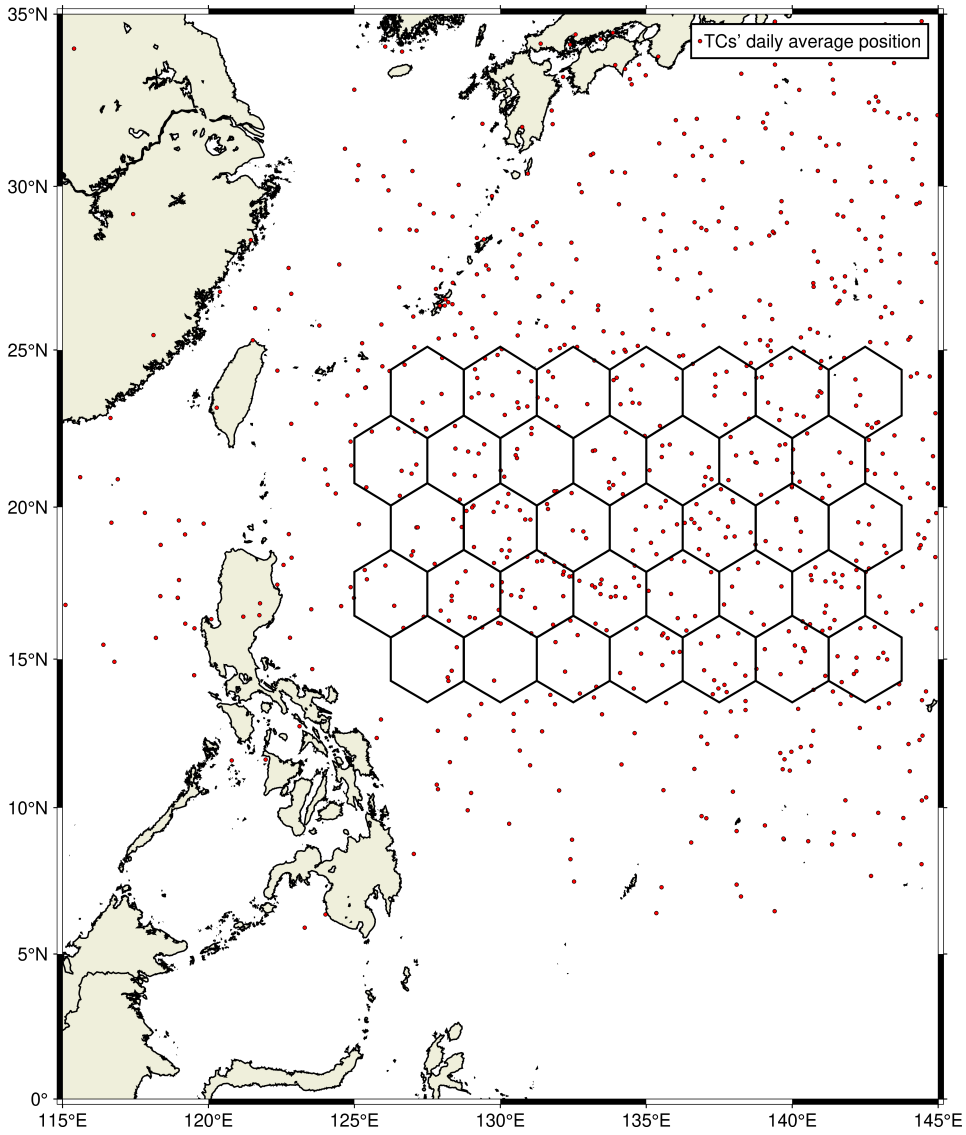

Figure 6: Define the region, markers style and projection.

You can access the data of each hexagon using the line » Grid[#row][#column] (starting from 0).

Now that the hexagonal grid is settled, just set the first argument as True, and run the main function again without modifying other arguments. Computing of probabilities will start after the image of the hexagonal grid is showed, and the results of the probabilities will be saved a .csv file and images in the Image folder. This might take hours or days depending on TCs data, stations and amount of hexagons of the grid.

Example 4 Running the tool of induced extreme precipitation in Panama due to TCs located in the Eastern Pacific Ocean in the period 1981-2021.

1. Using the demo function, the appropriate regions found are:
    - grid_region = [-100, -76, 5, 16]
    - stations_region = [-84, -76, 7, 10]
2. Using the main function with the first argument as False, the final grid configuration found is:

    » Grid = main(False, ["1/1/1981","31/12/2020"], [-100, -76, 5, 16], [-84, -76, 7, 10], (-97.5, 12), 3, 2, 1.5)

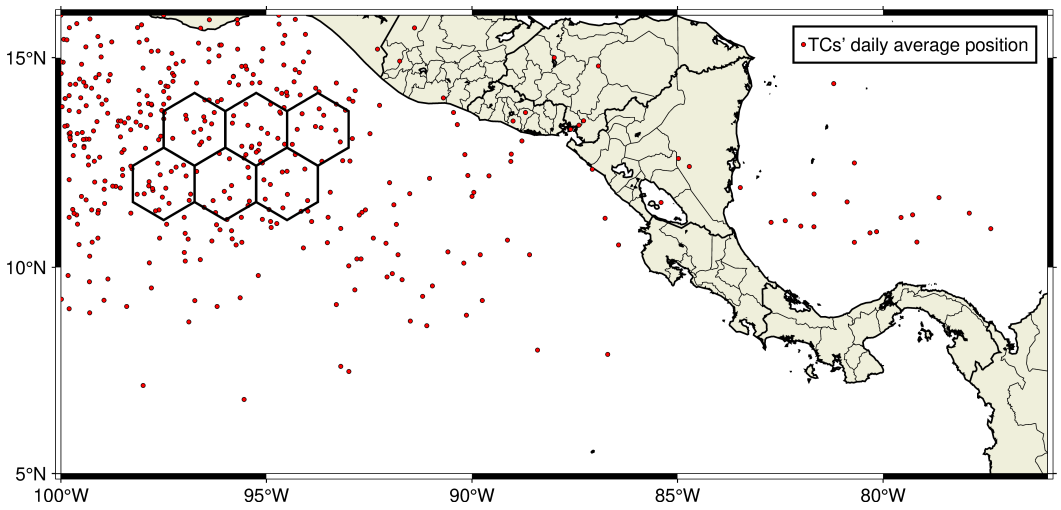

Figure 7: Image of the hexagonal grid and TCs used for the analysis.

3. Finally, setting the first argument as True, 18 images are created with computed probabilities.

   » Grid = main(True, ["1/1/1981","31/12/2020"], [-100, -76, 5, 16], [-84, -76, 7, 10], (-97.5, 12), 3, 2, 1.5)

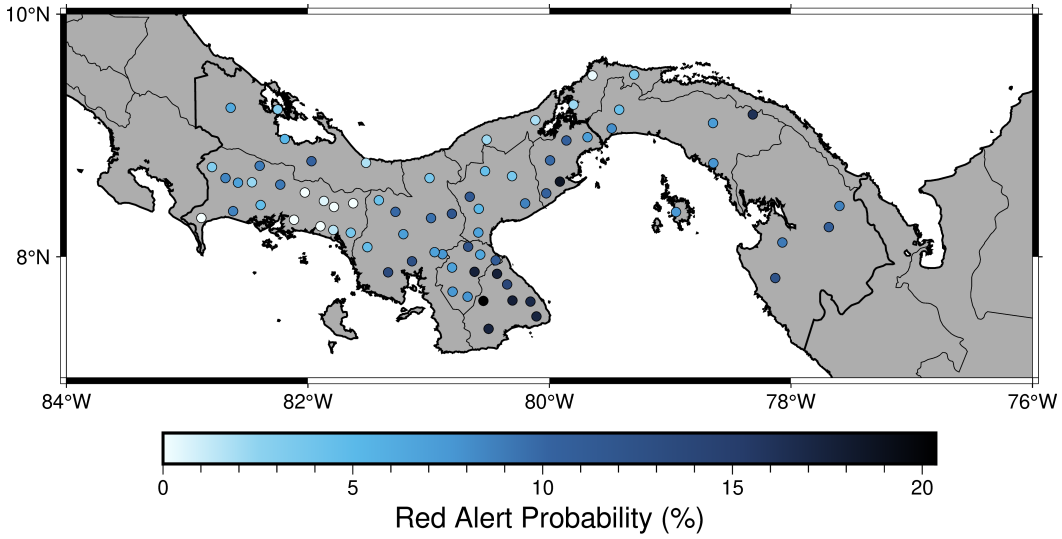

Figure 8: Example of one of the 18 total images created.