|  |  |  |  |
| --- | --- | --- | --- |
|  | | logo_black.gif | |
|  |  |  |  |
|  | GEOG5995M: Programming for the Social Sciences – Core Skills |  |  |
|  | Assessment 2 – The Black Death |
|  |  |

|  |
| --- |
|  |
| 1. |
| 2. |
| 3. |

|  |
| --- |
|  |
|  |

|  |
| --- |
|  |
|  |

**Intention of the software**

**Issues during development**

**How these issues were overcome or not**

**General sources used**

**The thought processes going into the softward design**

**The software development process followed**

**So give context, exlain how it ended up how it is,**

**UML diagram**

The aim of this software is to calculate the total deaths per week from the Black Death in 1665 for 16 parishes using a given equation and enable the user to change the parameters of the equation to examine the impact on the deaths.

Using the guidance given (Evans, 2017), this is achieved through the development of the Jupyter Notebook that performs the following steps:

1. *Reading in the two datasets*

The first dataset is a raster map of the average number of rats caught per week by rat catchers per 100m x 100m square (parameter ‘r’). The second is a raster map of the average population density per 100m x 100m square for 16 square parishes (parameter ‘p’).

1. *Processes these and displays the maps*

This step involves creating images of both datasets that are appropriately labelled with colour schemes that make sense for the type of data they are displaying.

1. *Calculates the average number of deaths per week and displays this as a map*

The average number of deaths per week (‘d’) is calculated using the equation given by Evans (2017):

This output is displayed as another map.

1. *Displays the three maps*
2. *Saves the death map as a text file with each line in the file equally a line on the map*
3. *Calculates and displays the total deaths per week*
4. *Presents a Graphical User Interface (GUI) to the user so that they can change the parameters of the equation.*

**Development process**

Overall this was a relatively challenging piece of work for me to deliver given that I had never coded anything before this module.

The first challenge was ensuring I was confident that I had imported and displayed the raster data appropriately. I utilised the information learnt from the module about raster data (each row is an increase in the y-direction and subsequent values in a row being an increase in the x-direction) and reading such data into a 2D array to inform my code. I cross-checked this by visualising the data in a spreadsheet and by checking the example images shown in Evans (2017).

As the data was in list format I converted it into a numpy array to allow me to use floats for the equation calculation.

Once I had the data, I then displayed it as two images. I removed the default axis labels that could cause confusion to simplify the visualisation and added appropriate labelling and a colorbar. I tried different colormap schemes to identify the ones I felt conveyed the data well to the viewer. A same colour sequential colormap was used for population density whereas a different sequential colormap that enabled the viewer to clearly demark the different rat catcher patches was used for this data.

I then developed code to run the equation by creating new variables using for loops. One looped through multiplying each r value by 0.8 and the second multiplying each p value by 1.3. I then calculated the average death for each 100 x 100 square by using a numpy multiply function. These average deaths per week were then displayed as an image with appropriate labelling and a colorbar.

The next step, which proved to be very tricky, was to create an image with all 3 maps together. It was relatively easy to get three images in the same horizontal plane and close enough together but I could not manage to get them to be the same size to enable direct comparison. I tried altering the saving of the 3 individual images (by specifying image size and dpi); used various matplot options such as figure, subplot, gridspec and close(); and searched StackOverflow and Python documentation but I was unable to get them all to be equal size.

Calculating the total deaths

**Initial data vis – to help make sense of the data**

**equation**

Evans, A. 2017. *Geography Programming Courses – The Black Death.* [Online]. [Accessed 5 December 2018]. Available from: [**http://www.geog.leeds.ac.uk/courses/computing/study/core-python-phd/assessment2/death.html**](http://www.geog.leeds.ac.uk/courses/computing/study/core-python-phd/assessment2/death.html)