Mapping Challenge Project Information

Context

This document contains all information about the data science mapping challenge.

Results

There are three likely areas the researcher could be located: Westminster, Victoria, and Bethnal Green.

The area of highest probability (with around 50% of the total probability, see below) is Westminster specifically in or around whitehall. Given this area is filled with government buildings it could be that the researcher works for the national government. There are also some university buildings around that area where they could work.

This area of highest probability extends to the nearby Victoria area of London where there are also university buildings.

Looking at a wider area which contains around 90% of the probability (see below) there is also an area of reasonably high probability in the Bethnal Green area of east London which is centred around Queen Mary University of London. So it is also possible the researcher is employed there.

Or the researcher could live in the Bethnal Green area of London and work in Westminster/Victoria.

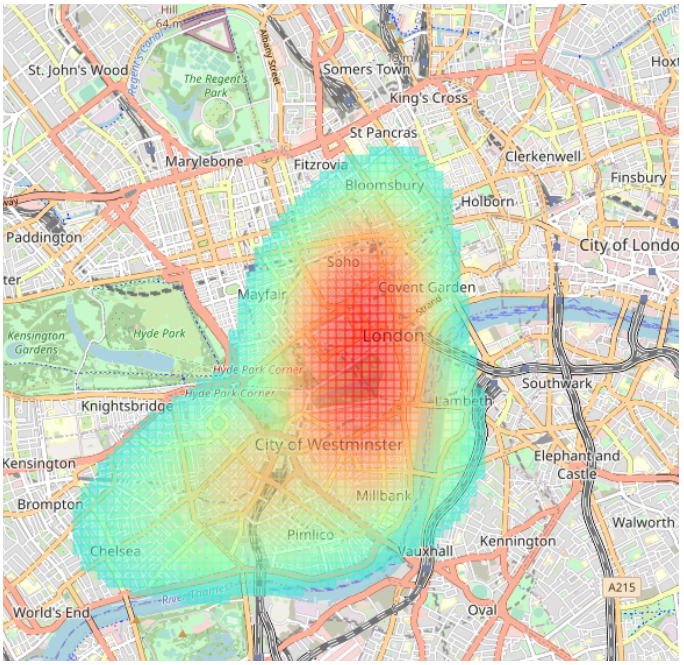

Figure showing heat map of 50% probability of researcher location, red is high probability .

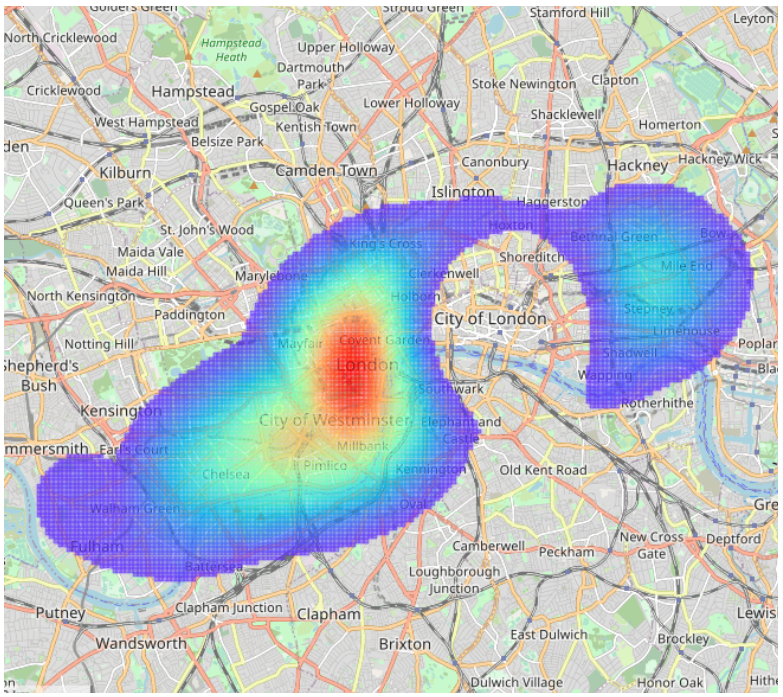

Figure showing heat map of 90% probability of researcher location, red is high probability

Approach and Assumptions

General Approach

As would be expected the approach used is to use the provided information to create probability distributions over London. Then combine these into a single probability distribution, colour the area proportional to the probability the person can be found there with red being high probability and dark blue low. Then use this to predict the probable location of the person of interest. This problem can then be viewed from a frequentist statistical view point and probabilities of success for a given area can be calculated.

Discretisation

The approach taken here is to model the probability distributions as discrete rather than continuous functions in the first iteration. The reason is twofold, firstly it simplifies the calculations and the plotting of the solution. Secondly and more importantly in a real world situation where people are searching a set area it would be valuable to the searchers to have the likely area to search subdivided into smaller zones so as to allow these areas to be divided among the searchers. This also has the added benefit that structuring the problem in this way naturally lends itself to a Bayesian search approach. See 'Future Iteration Suggestions' below for further information on this.

Probability Distributions

For simplicity the probability distributions are assumed to be independent. Details of the probability distribution parameters can be found in the ‘data_and_constants_explanation.md’ file at the following location within the repo: ‘src\artefact_creation\data_and_constants’

Projections

In order to simplify the problem, as was pointed out as a hint, the approach here is to assume the search space is locally Euclidean and to perform calculations in this space, then project the results back onto the WGS84 ellipsoid for mapping.

Given the discretisation and projections outlined above a decision needs to be made on the bounds of the probability space. In the project description document no bounds are given therefore naively it could be assumed that these probability distributions could exist over the surface of the entire earth. Given that it is stated the researcher is in London, even though technically if these probability distributions are unbounded there is a non zero chance of the researcher being in the middle of the Pacific ocean, this is obviously not realistic.

The problem then turns to what could be considered sensible boundaries. Firstly with regards to the river distribution, only a small part of that has been modelled, therefore the data points provided can give a good heuristic to the geographic bounds of the problem. In addition the satellite distribution could be interpreted as having a 95% probability within the given distance of the great circle satellite path. This is unlikely to be the intended interpretation of this distribution. Therefore the assumption will be that this distribution is only true between the given points describing this path. These points provide another good heuristic of problem bounds. Similarly the distribution described around the Bank of England would quickly vanish to approximately zero.

The below figures shows the probability distribution described by the Bank of England distribution (encoded such that the more red the region the higher the probability), the satellite path is shown in red, and the river path is shown in blue.

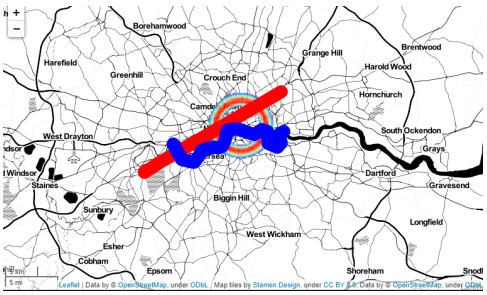

Figure showing a wide view of Bank of England distribution, satellite path, and river path.

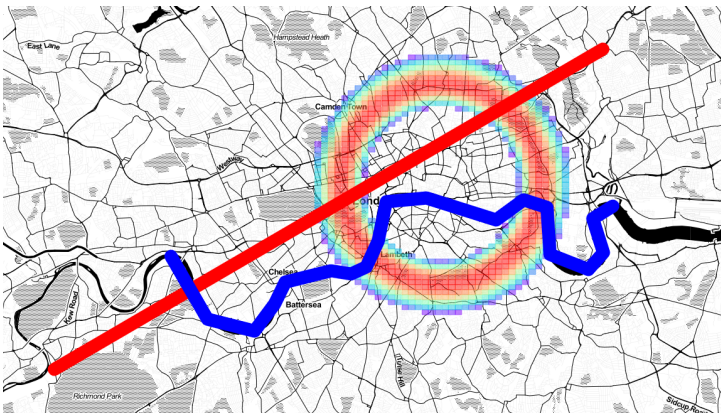

Figure showing close view of Bank of England distribution, satellite path, and river path.

Given the above figures the following bounds of the problem will be used: max and min longitude of the river for longitudinal bounds, max and min latitude of the satellite path for the latitudinal bounds. Specifically these are:

Table outlining boundaries of search space	
Minimum latitude	51.451
Maximum latitude	51.560
Minimum longitude	-0.236313
Maximum longitude	0.005536

Given these bounds and the WGS84 ellipsoidal distance between points calculated using the python library geopy's 'distance.distance' function the difference in distance between the top edge vs the bottom edge of a polygon with the above points as vertices would be approximately 40 metres, therefore the top edge would be approximately 0.2% shorter than the bottom edge.

Due to this very small difference in shape, this seems sufficiently similar to model on a Euclidean plane. The lat long space will be converted to a Euclidean plane, then discretised into points linearly spaced. These points will form the centre of (approximately) squares (known more generally as polygons), the vertices of these polygons will be projected back into the lat long space. Therefore calculations can be performed on these polygons in the euclidean space and then any output can be displayed on the map by the paired polygon in the lat long space.

Solution Architecture

The code is effectively split into two parts: analysis, and deployment. The analysis code performs all projections, probability calculations, and data manipulation. The data is then saved. The deployment code uses a folium front end map delivered in a 'streamlit' data app

which is deployed in a docker container, and the solution can then be viewed in the host machine's browser on port 8501 of localhost.

Without being able to converse with end users of the tool it is not known exactly how the final results should be presented. The end user is assumed to be non-technical and therefore the solution avoids numbers and mathematics and focuses on an intuitive map interface allowing the user to easily understand where to go on a map to likely find the person of interest. Fast feedback is vital for iteratively building a system, therefore delivering this approach to the end user and receiving feedback would be ideal.

Note, more unit tests are needed on this codebase however were not completed due to time constraints. Also some docstrings are missing from function also due to time constraints.

Running Solution

The analysis code is pre-run and the data is already saved so the dock container need only deploy the solution. To run the solution docker should be installed in the host machine and browser software should be available. The dockerfile to build the app can be found in the 'src\app' directory. Then the following command can be run from the terminal of the host machine to build the image from a location where the dockerfile is situated: *docker build -t mapping_app:0.1 .* This will create the docker image locally. Once built the image can be run using the following command: *docker run -p 8501:8501 mapping_app:0.1*.
The solution can then be viewed as outlined above.

Future Iterations

Bayesian Search

This problem has been posed and initially approached as a frequentist statistical problem. Specifically in this case meaning that the probability distributions are assumed to be accurate and static depictions of the real world probability of finding a person in a given location. The reality of search problems however is that even if a person is in a given location there is not a 100% chance of that person being discovered even if a searcher searches the given location. In the real world the person could be hidden under thick undergrowth, or hiding in an attic or basement.

Given this scenario it is likely more optimal to view the provided probability distributions as prior probabilities which can be updated after searching a given location. Since it may be better, if the person hasn't been found in the expected area, to search some of that area again, rather than (as would happen with a frequentist approach) search new areas of the space with relatively small probabilities of success.

Another future iteration could be to include the population density/time into the model. This could be useful in this particular case in that the Westminster area of London is likely more

populated during the day than Bethnal Green so could have a large effect on the probability of finding someone in those locations.

Also the river Thames in this initial iteration is assumed to have zero width. In reality however it is relatively wide and several polygons are over the river.