README for Data and Code for Thesis: 'Bayesian Analysis of Spatial Log-Gaussian Cox Processes'

Nadeen Khaleel

April 14, 2022

Contents

Part I
Introduction

1. Dataset Titles: THESIS_CODE.zip, THESIS_DATA.zip.

2. The code for the models and simulation studies that produced the results and plots found within the thesis can be found in several sub-directories of the two main directories, `THESIS_CODE` and `THESIS_DATA`. Each of these will have the exact same directory hierarchy so that combining each of the sub-directories for the code (`THESIS_CODE`) and the data (`THESIS_DATA`) matches the relevant input and output data sets for the `R` code found in the corresponding directory. The `README_*.txt` for each of these individual directories contains the same information for the code and the data in each of the relevant sub-directories. Some of the data is unavailable, in particular the data related to the crime and socio-economic variables for the relevant US cities. However, these can be found and accessed through the paths discussed in the file **DataAccessInformation.pdf**.
   While each sub-directory and the sub-directories within these will contain their own README files with more or similar details, Part II of this document will set out the general layout of this directory and where to find the code, data or results for a particular chapter of my thesis. Additionally, the Part III will directly link the necessary directories to the results for each chapter of my thesis.

3. Author Information:

   PhD Student: Nadeen Khaleel, n.khaleel@bath.ac.uk

   Supervisor: Dr Theresa Smith, t.r.smith@bath.ac.uk

4. Funding Resource: EPSRC (EPSRC Centre for Doctoral Training in Statistical Applied Mathematics at Bath (SAMBa))

Part II
Directories

This section contains the information for the `R` scripts and their outputs that produced the results shown and discussed in my thesis. There are four main directories, each with their own sub-directory and we discuss each in turn, and within these sub-directories there can also be found the individual `README_*.txt` files that will be reasonably similar to the text below. For each of the nested directories where it is necessary, we will try to split the files within into one of the following three labels: **Data**, **`R` code** and **Outputs**.

Note, **Data** and **Output** files are found in the `THESIS_DATA` directory, while **`R` code** files are found in the `THESIS_CODE` directory. Therefore, due to the exclusion of some data files from the archive (discussed below) and the nature of some of the sub-directories (e.g. some only containing outputs from implemented `R` code) there will be some sub-directories in either `THESIS_DATA` or `THESIS_CODE` may only contain a `README_*.txt` that describes what would be contained within the sub-directory. However, even though these sub-directories may be empty we still retain them in both `THESIS_DATA` and `THESIS_CODE` so that the two have the exact same hierarchical structure for the directories and sub-directories and, therefore, these two main directories can be combined easily if needed with all the relevant information available. Importantly, the `README_*.txt` files in the `THESIS_DATA` and `THESIS_CODE` directories contain the same information, both for the

input and output data as well as the R scripts that use and produce these data sets, respectively.

With respect to the data corresponding to the raw crime, socio-economic and shapefile data sets, we organise these into three categories:

1. RAW DATA: this refers to the original downloaded data files for crime, covariates and for the shapefiles. These will also be used to label the outputs from the initial cleaning of these data sets, which extracts the relevant subsets from the original data, such as only homicides and motor vehicle thefts rather than all crimes in the particular city. We will also label the data frame containing the socio-economic variables with the area proportions between the grids and census tracts that allow us to generate the relevant grid interpolations.

2. COLLATED DATA: this refers to the collection of the aggregated point pattern data and socio-economic variables over the census tracts, where the census tract socio-economic variables retain the original structure from the raw data.

3. AGGREGATE DATA: this refers to the final gridded count data for the cities as well as the relevant additional data produced from the shapefiles and socio-economic variables.

The relevant data will have the corresponding category placed in square brackets ([]) in blue text. The outputs from the models, simulations and any plots will not be assigned into any of these categories and should be found in the relevant directories. Full details on the access of the raw data can be found in **DataAccessInformation.pdf**.

Finally, the raw data, labelled [RAW DATA], is not archived within the relevant THESIS_DATA directories, however below we name these file as they were accessed as these are used within the relevant R code, and may be helpful for reference. However, any data sets labelled [COLLATED DATA] or [AGGREGATE DATA] should be found in the relevant directories.

1 DATA

This directory is where the the raw crime, socio-economic and shapefiles data are stored as well as the code to process these and generate count data over the census tracts and discretisation grids. We also have the code for the exploratory data analysis and some initial models for the census tract count data as well as estimating Ripley's K for the point patterns data. These can be found in the following sub-directories:

1.1 RAW_DATA

This directory contains three further sub-directories that contain the raw data as well as code for the processing of the data as described below. Any processed data is also placed in the relevant sub-directories of the DATA/PROCESSED_DATA directory.

1.1.1 SHAPEFILES

This contains two further sub-directories, one labelled BOUNDARIES which contains the shapefiles for the city boundaries and details of their access can be found in the file **DataAccessInformation.pdf** as well as Appendix F of my thesis. The second sub-directory, CENSUS_TRACTS contains the shapefiles for the states containing the cities of interest available through the TIGER/Lines shapefiles data and were accessed through the US Census Bureau. This second directory also contains an R script which allows us to use

the boundaries shapefiles to extract the census tracts which lie only within the cities of interest.

`BOUNDARIES` **Sub-directory**

This contains only three data files which contain the shapefiles information for the boundaries for each city of interest.

- **Data**
  - → **Los Angeles**: downloaded as 'City Boundaries for Los Angeles County' (`City Boundaries for Los Angeles County`). [RAW DATA]
  - → **New York City**: downloaded as 'Borough Boundaries NYC' (`Borough Boundaries NYC`). [RAW DATA]
  - → **Portland**: downloaded as 'City_Boundaries_Portland' (`City_Boundaries_Portland`). [RAW DATA]

`CENSUS_TRACTS` **Sub-directory**

- **Data**:
  - → **Los Angeles**: downloaded as `tl_2015_06_tract`. [RAW DATA]
  - → **New York City**: downloaded as `tl_2015_36_tract`. [RAW DATA]
  - → **Portland**: downloaded as `tl_2015_41_tract`. [RAW DATA]
- `R` **code**:
  - → `CensusTracts_final.R`: this is the `R` code which takes in the raw shapefiles data which is available for each state (each state can be identified through the FIPS code: 06 for California, 36 for New York and 41 for Oregon) and extracts the necessary census tracts for each city as well as the census tracts for the counties containing the cities and the neighbourhood matrices for both census tract data files.
- **Outputs**: these outputs are also moved into the directory `DATA/PROCESSED_DATA/SHAPEFILES/CENSUS_TRACTS`
  - → `*CityCT.rda`: census tracts for city, *, using a combination of the state census tracts and the city boundary shapefiles. Although at times the intersection between these is not clear cut and choices have to be made about which census tracts are included. [RAW DATA]
  - → `*CountyCT.rda`: census tracts for the counties that contain the city, *, of interest, for example Los Angeles County contains Los Angeles City. [RAW DATA]
  - → `*CityNB.rda`: this contains the neighbourhood structure of the census tracts within the county that contain the city of interest, *, for the use of interpolation of socio-economic variables, where we only use data from the counties that contain each city, or the city itself if the missing data lies surrounded by other census tracts from the city. These also contain the relevant census tracts as `sp` objects. [RAW DATA]
  - → `*CountyNB.rda`: this contains the neighbourhood structure of the census tracts within the county that contain the city of interest, *, for the use of interpolation of socio-economic variables, where we only use data from the counties that contain each city. These also contain the relevant census tracts as `sp` objects. [RAW DATA]

1.1.2 COVARIATES

This contains the files for the socio-economic variables at the census tract level for the counties that contain each city, accessed through the American Community Survey (ACS), as well as the code to extract the necessary data for the census tracts that lie only within the individual cities using two R scripts in this directory. Unlike the DATA/RAW_DATA/CRIME directory we separate the variables data across three individual sub-directories for each city of interest. The R scripts lie within the main directory and load and save the necessary data into the city sub-directories. For each city we have the necessary socio-economic data files for the census tracts in 2015. The main variables considered in this thesis were the total population and average income, however while they were not used we also accessed the data for age, sex, food stamps/SNAP and for tenure of properties (owned/rented). As mentioned above these data sets are available through the American Community Survey and can be found through the US Census Bureau, and the particular data sets downloaded for this thesis can be identified with the codes:

- Total Population: B01003;
- Average Household Income: S1902;
- Age and Sex: S0101;
- Food Stamps/SNAP: S2201;
- Tenure: B25003,

where the year of the data and the areas of interest would also need to be specified. More details can be found in the file **DataAccessInformation.pdf** as well as Appendix F of my thesis. Any R script with code that calls these additional variables has had these few lines commented out, so that the code is still present as it was originally used to create the relevant output data as was used in the thesis. However, it is commented out as these variables are not required for any of the results in the thesis. The data files generated will have been generated using this original code, however the only socio-economic variables used are the total population and average income and so the other variables are unnecessary.

- **Data**
  - Los Angeles:
    - → **2015 Data**: directories of the form ACS_15_5YR*, where * is replaced by the identifier for each table. These directories in turn contain the relevant .csv files as well as metadata files. [RAW DATA]
    - → **2014 Data**: directories of the form ACS*5Y2014.*_*, where * after the period is replaced by the identifier for each table. These directories in turn contain the relevant .csv files as well as metadata files. [RAW DATA]
  - New York:
    - → **2015 Data**: directories of the form ACS*5Y2015.*_*, where * after the period is replaced by the identifier for each table. These directories in turn contain the relevant .csv files as well as metadata files. [RAW DATA]
  - Portland:
    - → **2015 Data**: directories of the form ACS*5Y2015.*_*, where * after the period is replaced by the identifier for each table. These directories in turn contain the relevant .csv files as well as metadata files. [RAW DATA]
- **R code**:

→ `CovDataGen_final.R`: this `R` script loads the relevant cities socio-economic variable data and extracts the variables for the relevant census tracts within each city by using the processed census tract shapefiles from the `DATA/RAW_DATA/SHAPEFILES` directory. For the average income data, it is important to note that there was missing data present for some of the census tracts. In this `R` script we interpolated all of the missing data by using the average of the values for the neighbouring census tracts within the counties that contained the city. The neighbourhood matrices for the census tracts were also created within the `DATA/RAW_DATA/SHAPEFILES` directory. The income variables produced from this `R` script were only used for the Grid-Mesh Optimisation method in Chapter 4, but not for any of the final models, either for the grids or census tracts.

→ `CovDataGen_Inc_final.R`: this `R` script produces only the average income variable on the census tracts for each city. The difference between this output and the average income output from the `CovDataGen_final.R` script is that if a census tract with missing income data has a corresponding estimate of zero for the total households in the census tract, we will set the average income value to zero. The remaining missing values, not accounted for by the zero total households estimate are then interpolated, as before, using the values of the neighbouring census tracts.

– **Outputs**: these outputs are also moved into the directory `DATA/PROCESSED_DATA/CRIME/COVARIATES`

→ `*_CTPop_15_proj.rds`: total population for census tracts in city *, produced in `CovDataGen_final.R`. [RAW DATA]

→ `*_CTInc_15_imp_proj.rds`: average income for census tracts in city *, produced in `CovDataGen_final.R`, so all missing data is interpolated regardless of the corresponding estimates for the total households. [RAW DATA]

→ `*_CTInc_15_0imp_proj.rds`: average income for census tracts in city *, produced in `CovDataGen_Inc_final.R`, so any missing data with the corresponding estimates of zero for the total households is set to zero. Any of the remaining missing data are interpolated. [RAW DATA]

Importantly, note that the code for the 2015 Los Angeles covariates will be different with respect to the necessary column numbers for the relevant data within the ACS data. Instead of those currently in the code for LA, the column numbers used for New York and Portland should be the correct column numbers. Additionally, the `R` scripts contain the code to generate other raw data for socio-economic variables that were mentioned above. Additionally, for Los Angeles we also accessed the 2014 socio-economic data files (population and average income) for the minimum contrast implementation in Chapter 4.

1.1.3 `CRIME`

This contains the raw crime data for Los Angeles, New York and Portland as well as `R` scripts that extract the necessary crime data, for us this would be the homicide and motor vehicle theft crimes. We also make use of the city boundary shapefiles to exclude any points that may lie outside of the cities due to errors in the latitude and longitude locations.

– **Data**:

→ **Los Angeles Crime Data**: downloaded as "Crime Data from 2010 to Present" (`Crime_Data_from_2010_to_Present.csv`). [RAW DATA]

→ **New York Crime Data**: downloaded as "NYPD Complaint Data Historic" (`NYPD_Complaint_Data_Historic.csv`). [RAW DATA]

→ **Portland Crime Data**: downloaded as "Portland Police Bureau - Incidence" (`Portland_Open_Data_Sheet_data.csv`). [RAW DATA]

– `R` **code**:

→ `*Manip_final.R`: this is the `R` code which takes in the raw crime data for city * (which had been downloaded as .csv files) in order to extract the relevant crime data: homicide and motor vehicle theft. The outputs from these are scripts are detailed below.

– **Outputs**: these outputs are also moved into the directory `DATA/PROCESSED_DATA/CRIME/POINT_PATTERN`

→ `*_hom.rds`: this contains the crime data, locations and dates for the homicides in city *. [RAW DATA]

→ `*_hom_sf.rds`: this contains the same data as `*_hom.rds`, but saved as an object of class `sf`. [RAW DATA]

→ `*_gta.rds`: this contains the crime data, locations and dates for the motor vehicle thefts in city *. [RAW DATA]

→ `*_gta_sf.rds`: this contains the same data as `*_gta.rds`, but saved as an object of class `sf`. [RAW DATA]

1.2 PROCESSED_DATA

As for the `DATA/RAW_DATA` directory, this directory contains three sub-directories. One contains the processed socio-economic variables while the other contains the processed census tract data. The final sub-directory contains the processed crime data as well as code used to generate the count data at the census tract-level or over discretisation grids.

1.2.1 SHAPEFILES

This contains a single sub-directories, labelled `CENSUS_TRACTS`, which contains the processed shapefiles for the cities and counties of interest as well as the neighbourhood matrices for the census tracts. These were generated in the `DATA/RAW_DATA/SHAPEFILES/CENSUS_TRACTS` directory.

– **Data**:

→ `*CityCT.rda`: census tracts for city, *, using a combination of the state census tracts and the city boundary shapefiles. [RAW DATA]

→ `*CountyCT.rda`: census tracts for the counties that contain the city, *, of interest. [RAW DATA]

→ `*CityNB.rda`: this contains the neighbourhood structure of the census tracts within the county that contain the city of interest, *, for the use of interpolation of socio-economic variables, where we only use data from the counties that contain each city, or the city itself if the missing data lies surrounded by other census tracts from the city. These also contain the relevant census tracts as `sp` objects. [RAW DATA]

→ `*CountyNB.rda`: this contains the neighbourhood structure of the census tracts within the county that contain the city of interest, *, for the use of interpolation of socio-economic variables, where we only use data from the counties that contain each city. These also contain the relevant census tracts as `sp` objects. [RAW DATA]

1.2.2 COVARIATES

This contains the processed files for the socio-economic variables at the census tract level for the each city, as generated in the DATA/RAW_DATA/COVARIATES directory.
The R script also contains the code to generate the same outputs for the additional socio-economic variables, however even though these variables were contained in the finally necessary data outputs, we do not require any but the population and average income for the results in this thesis and so the relevant code to generate the necessary outputs, while maintained for reference, have been commented out.

- **Data**:
  - → *_CTPop_15_proj.rds: total population for census tracts in city *, produced in CovDataGen_final.R. [RAW DATA]
  - → *_CTInc_15_imp_proj.rds: average income for census tracts in city *, produced in CovDataGen_final.R, so all missing data is interpolated regardless of the corresponding estimates for the total households. [RAW DATA]
  - → *_CTInc_15_0imp_proj.rds: average income for census tracts in city *, produced in CovDataGen_Inc_final.R, so any missing data with the corresponding estimates of zero for the total households is set to zero. Any of the remaining missing data are interpolated. [RAW DATA]

1.2.3 CRIME

This contains four sub-directories, the first contains the processed point patterns for the two crimes and for each city of interest, the other three produce the aggregated count data for the modelling and Grid-Mesh Optimisation method, with more details below. With respect to the gridded count data, details with respect to the creation of these data sets and, in particular, the interpolation of the socio-economic data onto the grid cells from the census tracts can be found in Chapter 4 of my thesis.

- POINT_PATTERN:
  This contains the processed crime data for all cities from the DATA/RAW_DATA/CRIME directory.
  - **Data**:
    - → *_hom.rds: this contains the crime data, locations and dates for the homicides in city *. [RAW DATA]
    - → *_hom_sf.rds: this contains the same data as *_hom.rds, but saved as an object of class sf. [RAW DATA]
    - → *_gta.rds: this contains the crime data, locations and dates for the motor vehicle thefts in city *. [RAW DATA]
    - → *_gta_sf.rds: this contains the same data as *_gta.rds, but saved as an object of class sf. [RAW DATA]
- COUNT_DATA_CENSUS_TRACTS:
  This contains the code with takes the point level crime data for each city as well as the socio-economic variables at the census tract level to generate aggregated data at the census tract-level.
  The R script has commented out code for the inclusion of additional socio-economic variables as these were originally used to generate the data archived in this directory which therefore contain all of these additional variables. However, these variables are never used for any of the results within this thesis and are therefore unnecessary, and so commented out.

- `R` **code**:
  - → `CountDataGen_CT_final.R`: this is the `R` code which takes in the crime point pattern data from the `DATA/PROCESSED_DATA/CRIME/POINT_PATTERN` directory and the socio-economic variables from `DATA/PROCESSED_DATA/COVARIATES` directory to merge in to a data set with counts of incidents over the census tracts and combining them with the relevant variables values. Note that the income variable `*_CTInc_15_0imp_proj.rds` is used in the creation of these count data sets for city *.
- **Output**:
  - → `*2015CTCountData_projFinal.rda`: this contains the crime data aggregated over the census tracts (ct_homcount.df and ct_gtacount.df for the homicides and motor vehicle thefts, respectively) with the corresponding socio-economic variables for each census tracts as well as other variables such as the area of the census tracts for city *. [COLLATED DATA]
  - → `*2015CTSFCountData_projFinal.rda`: this contains the same data as `*2015CTCountData_projFinal.rda`, but saved as an object of class `sf`. [COLLATED DATA]

- `COUNT_DATA_GMO`:
  The `R` script has commented out code for the inclusion of additional socio-economic variables as these were originally used to generate the data archived in this directory which therefore contain all of these additional variables. However, these variables are never used for any of the results within this thesis and are therefore unnecessary, and so commented out.

  - `R` **code**:
    - → `CountDataGen_GMO_final.R`: this is the `R` code which takes in the Los Angeles crime point pattern data from the `DATA/PROCESSED_DATA/CRIME/POINT_PATTERN` directory and the Los Angeles socio-economic variables from `DATA/PROCESSED_DATA/COVARIATES` directory to merge in to a data set with counts of incidents over different resolution grids and combining them with interpolated the relevant variables values.
      Importantly, this produces the gridded count data for the creation of the covariates for the Grid-Mesh Optimisation method implemented on the Los Angeles window (in Chapter 4 of my thesis) for which the creation of the second covariate arises from the average income LA_CTInc_15_imp_proj.rds where all missing data for census tracts are interpolated, regardless of the total household estimates for the census tract. This is then interpolated slightly differently, using the proportion of the area of the census tract contained within the grid cells as the weights for the areal interpolation.
  - **Output**:
    - → `LA_Cov_XY_proj.rda`: these are the data frames for the intersections of the census tracts with the grid of dimension $X \times Y$. This data frame contains the labels for the intersected census tract and grid cells as well as the areas of the intersections as well as the areas of the corresponding grid cells and census tracts and the proportion of the intersection areas with respect to the total areas of the census tracts and grid cells. This data frame also contains the values of the variables at the corresponding census tracts as well as their product with both the proportion of the intersection area with respect to the total area of the corresponding census tract and the proportion of the intersection area with respect to the total area of the relevant grid cell. [COLLATED DATA]

→ `LAGridCellsXY_proj.rda`: these are the grid cells for the grid of dimension $X \times Y$, which are stored as an `sf` object. These are at the UTM projected coordinates. [AGGREGATE DATA]

→ `LA2015CTXYCountData_proj.rda`: this is the output count data for both homicides (hom_countdf) and motor vehicle thefts (gta_countdf) on a discretisation grid with dimensions $X \times Y$ over the Los Angeles study region with socio-economic variables interpolated as described and, in particular, the average income treated slightly differently in order to generate the second covariate variables for the Grid-Mesh Optimisation method that is based on the average income but not necessarily the average income variable. [AGGREGATE DATA]

→ `LA2015CTXYSFCountData_proj.rda`: this contains the same data as `LA2015CTXYCountData_proj.rda`, but saved as an object of class `sf`. [AGGREGATE DATA]

where X and Y specified to get grid cell widths of approximately: 5km, 2km, 1km and 0.5km using the following number of cells in the x and y direction:

$$X = 10, 24, 48, 95, 236$$
$$Y = 15, 36, 72, 144, 359$$

The key data set here is the one aggregated over the 236×359 grid, as the socio-economic variables interpolated over this grid are used for the data simulation in the Grid-Mesh Optimisation method for Chapter 4 which is done in the `GRID_MESH/IRREGULAR_POLYGON_LGCP/IRREGPOLLGCP_CODE` directory.

- `COUNT_DATA_FINAL`:
  This directory produces the gridded count data that will be used for fitting the models of interest to the crime data. In particular, this data is used in Section 4.5 (where the motor vehicle theft data is modelled using the INLA algorithm through the `R-INLA` package) and in Chapter 5 for the implementation of the INLA, Univariate and Multivariate INLA within MCMC algorithms. This directory contains two `R` scripts for the generation of the count data, the first generates the grids and resultant count data at the projected UTM coordinates, while the second takes these and scale and shifts the data so that the scale on the x and y axis is now 10km rather than 1m and the bounding box of the study region has the bottom-left corner at $(0, 0)$, as well as generating the meshes at the different resolutions.
  The `R` script has commented out code for the inclusion of additional socio-economic variables as these were originally used to generate the data archived in this directory which therefore contain all of these additional variables. However, these variables are never used for any of the results within this thesis and are therefore unnecessary, and so commented out.

  - `R` **code**:

    → `CountDataGen_final.R`: this is the `R` code which takes in the crime point pattern data from the `DATA/PROCESSED_DATA/CRIME/POINT_PATTERN` directory and the socio-economic variables from `DATA/PROCESSED_DATA/COVARIATES` directory to merge in to a data set with counts of incidents over the discretisation grids and combining them with the relevant variables values which are interpolated onto the grid cells from the census tracts. Note that the income variable `*_CTInc_15_0imp_proj.rds` is used in the creation of these count data sets for city *. Additionally, the interpolation of the income variable is different than that for the GMO outputs, where we use the proportion of the grid cells contained within the census tracts. This `R` script

produces the count data on different grid resolutions at the projected UTM coordinates.

→ `CountDataGen_Scale_final.R`: this `R` script takes the projected count data frames output form `CountDataGen_final.R` and shifts and scales the data so that a unit shift in the x or y direction is related to a distance shift of 10km rather than 1m, as in the UTM projected coordinates, and so that the bottom left corner of the bounding box of the city window lies at the origin, $(0, 0)$.

– **Output**:
Outputs from `CountDataGen_final.R`:

→ `*QuadXY_projFinal.rda`: these are the quadrats for the grid of dimension $X \times Y$ that are used to aggregate the point patterns for city *. These are at the UTM projected coordinates. [AGGREGATE DATA]

→ `*GridCellsXY_projFinal.rda`: these are the grid cells for the grid of dimension $X \times Y$, which are essentially the quadrats for city *, above but are stored as an `sf` object. These are at the UTM projected coordinates. [AGGREGATE DATA]

→ `*_Cov_XY_projFinal.rda`: these are the data frames for the intersections of the census tracts with the grid of dimension $X \times Y$. This data frame contains the labels for the intersected census tract and grid cells as well as the areas of the intersections as well as the areas of the corresponding grid cells and census tracts and the proportion of the intersection areas with respect to the total areas of the census tracts and grid cells. This data frame also contains the values of the variables at the corresponding census tracts as well as their product with both the proportion of the intersection area with respect to the total area of the corresponding census tract and the proportion of the intersection area with respect to the total area of the relevant grid cell. [COLLATED DATA]

→ `*2015CTXYCountData_projFinal.rda`: the aggregated points on the $X \times Y$ grid cells along with the interpolated socio-economic variables as well as the area of the intersection of the grid cells with the city window. [AGGREGATE DATA]

→ `*2015CTXYSFCountData_projFinal.rda`: the above data stored as an `sf` object. [AGGREGATE DATA]

Outputs from `CountDataGen_Scale_final.R`:

→ `*WindowProjScale.rda`: scaled and shifted window for city *, so that a unit shift in the x or y direction is related to a distance shift of 10km, and so that the bottom left corner of the bounding box of the city window lies at the origin, $(0, 0)$, and is stored as `W`. Also in this output is the unshifted and unscaled window for city *. [AGGREGATE DATA]

→ `*OrdDFXY_projFinalScale.rda`: this is a data frame with a set of indices that were created to allow us to re-order the quadrat cells of the discretisation grid with resolution $X \times Y$ for city *. This ordering ensures that we traverse down the y axis first and then incrementing across the x axis. [AGGREGATE DATA]

→ `*CoordXY_projFinalScale.rda`: this is a data frame of coordinates for the centres of the grid cells of the quadrats intersecting the window for city *. These centres are mostly the centres of the grid cells, however the intersections between the grid cells and the boundary of the city may result in odd polygon shapes. Most importantly, for these particular intersections

the 'centres' calculated may lie outside the city boundary (and therefore the polygon that defines its intersection with the grid cell), and so we re-adjust these coordinates so that they lie inside the necessary intersection of the grid cell and city polygon. These coordinates are still on the projected UTM coordinates. [AGGREGATE DATA]

→ `*2015CTXYCountData_projFinalScale.rda`: this contains the scaled and shifted count data for city * over the grid cell with resolution $X \times Y$ using the count data generated from `CountDataGen_final.R`. The coordinates included are the adjusted coordinates and then scaling and shifting with the same values for the city's window so that the coordinates will now lie within the transformed city window in `*WindowProjScale.rda`. These are also re-ordered using the indices in `*OrdDFXY_projFinalScale.rda`. (The data also still contains the original projected coordinates from the output of `CountDataGen_final.R` as $x.lab$ and $y.lab$ as well as the projected (and corrected - by ensuring they all lie within the city boundary) coordinates under $x.proj$ and $y.proj$. However, these are unused.) [AGGREGATE DATA]

→ `*2015CTXYSFCountData_projFinalScale.rda`: the same as in *2015CTXY-CountData_projFinalScale.rda but stored as an `sf` object. [AGGREGATE DATA]

→ `*MeshXY_projFinalScale.rda`: this takes in the coordinates from either the homicide or motor vehicle theft count data frames in *2015CTXY-CountData_projFinalScale.rda and produces a mesh over city * using these coordinates and matching the maximum mesh edge with the relevant grid cell size for the resolution $X \times Y$. [AGGREGATE DATA]

where X and Y specified to get grid cell widths of approximately: 5km, 2km, 1km and 0.5km (and 0.2km for interest) using the following number of cells in the x and y direction for the different cities:
- LA:

$$X = 10, 24, 48, 95\ (, 236)$$
$$Y = 15, 36, 72, 144\ (, 359)$$

- NYC:

$$X = 10, 24, 47, 94\ (, 235)$$
$$Y = 10, 24, 48, 96\ (, 239)$$

- Portland

$$X = 8, 19, 38, 76\ (, 190)$$
$$Y = 6, 13, 26, 52\ (, 129)$$

1.3 EDA

This directory contains three `R` scripts for exploring the crime data as well as the socio-economic variables. The outputs are stored in three sub-directories, one for each of the cities of interest.
The `R` scripts have code to plot the additional variables that are found in the generated data sets, but our interest lies with only the total population and average income variables. Therefore, as with the data generation code where these lines of code are commented out, we match this by commenting out the code for the corresponding plots in these `R` scripts.

- **`R` code**:
  - → `EDA_CT_final.R`: this `R` script plots the crime data for each city with respect to the census tracts, for example we plot the crimes with respect to the census tract level socio-economic variables.
  - → `EDA_GRID_final.R`: in this `R` script we more generally plot the crime data, for example number of motor vehicle thefts per year. Importantly, we also plot the maps for the crimes in each city, for all years in the data and also highlighting the year 2015. We also plot the gridded data, for example the gridded socio-economic variables for each resolution as well as the count data plotted against the grid-interpolated socio-economic variables. We also plot heat maps of the census tract-level and grid-interpolated socio-economic variables.
  - → `EDA_TRANSFORMEDGRID_final.R`: similarly to `EDA_GRID_final.R`, we plot the count crime data from the scaled gridded data against the socio-economic variables. This R script aslo produces plots of the scaled gridded socio-economic variables over a map. These should be similar to the plots from the un-scaled data, however they have been produced using the final data sets that will be used for modelling the necessary data.
  - → `AreaHistograms_final.R`: this produces plots for Los Angeles and New York census tract areas as well as the areas of the grid cells for the different resolutions in order to compare the distribution of the census tract areas against the areas of the grid cells.
- **Outputs**:
  Outputs from `EDA_CT_final.R`:
  - → ***CT.pdf**: the output plots from this `R` script take this form.

  Outputs from `EDA_GRID_final.R`:
  - → ***_final.pdf**: the output for this `R` script has this particular suffix, including the times series plots for the crime and the map plots. Those plotted on a gridded map will have the suffix `*_projSFPlot_final.pdf`.

  Outputs from `EDA_TRANSFORMEDGRID_final.R`:
  - → ***_finalscale.pdf**: the output for this `R` script has this particular suffix. Those plotted on a gridded map will have the suffix ***_projSFPlot_finalscale.pdf**.

  Outputs from `AreaHistograms_final.R`:
  - → ***CTAreaHistograms*.pdf or *CTGridAreaHistograms*.pdf**: the output for this `R` script has these particular suffixes.

1.4 MODELS

This contains two sub-directories, one for the fitting non-spatial generalised linear models to the census tract count data and for the estimation of Ripley's K function for the crime point patterns while the other contains the minimum contrast estimates for the 2014 Los Angeles crime point patterns as well as brief (unused) generalised linear models for the 2014 gridded Los Angeles count data.

1.4.1 GLMS

This contains the initial generalised linear models for the census tract level crime data for the three cities as well as the estimates of the Ripley's K function (both homogeneous and inhomogeneous) for the point patterns for each crime and city.

- **R code**:
  - → `InitialModels_final.R`: this is the `R` code which uses `rstan` and `rstanarm` to fit Poisson and Negative Binomial models to the census tract level homicide and motor vehicle theft data for Los Angeles and New York, and for only the motor vehicle theft data in Portland. Additionally, we estimate Ripley's K function for the crime point patterns where the result is plotted to visually compare the estimate form of the function against the theoretical form of the function, πr^2, assuming homogeneity in the point pattern.
- **Outputs**: these outputs are stored within individual sub-directories for each city
  - → `InitialModelsMCDF_area3Cities.rda`: this is the data set produced which contains the summary results for all three cities and both crimes in order to produce a joint plot of the results.
  - → `InitialModelsMCPlot_area3Cities.rda`: this contains the plot of the above generated data frame, with the posterior means and credible intervals for each parameter for all crimes, all cities and all GLMs.
  - → **RipleysK_*homz_proj_XY.pdf**: this contains the plots of both the homogeneous and inhomogeneous Ripley's K for the homicide point pattern for city * where the census tract socio-economic variables were given at the $X \times Y$ resolution - not interpolated but each grid cell centre from the $X \times Y$ grid was assigned the value of the variable from the census tract it lies within, to ensure that the relevant functions can assign the relevant socio-economic values at the census tract level to the incidents of crime.
  - → **RipleysK_*gtaz_proj_XY.pdf**: this contains the plots of both the homogeneous and inhomogeneous Ripley's K for the motor vehicle theft point pattern for city * where the census tract socio-economic variables were given at the $X \times Y$ resolution - not interpolated but each grid cell centre from the $X \times Y$ grid was assigned the value of the variable from the census tract it lies within, to ensure that the relevant functions can assign the relevant socio-economic values at the census tract level to the incidents of crime.
  - → `*HommcGLMS_area.rda`: the Poisson (`fit.zcov`) and Negative Binomial (`fit.nbzcov`) model results for the homicide count data for city *.
  - → `*GTAmcGLMS_area.rda`: the Poisson (`fit.zcov`) and Negative Binomial (`fit.nbzcov`) model results for the motor vehicle theft count data for city *.

  Each sub-directory for the city contains additional plots for the results of the fits of the generalised linear models such as trace plots and plots of the errors on the census tracts. More details for these plots can be found in the individual README for the GLMs directory, `README_GLMS.txt`.

1.4.2 MINIMUM_CONTRAST

This directory contains the implementation of the minimum contrast method to estimate the variance and range for the 2014 Los Angeles crime data. Additionally, we have an `R` script to quickly generate gridded count data for the 2014 Los Angeles crime data to assess

the behaviour of the estimates of the fixed effects estimates from a simple Poisson model fit, in order to help us select values for the simulation study priors and values of θ in the traditional simulation study. These latter results were aimed to supplement the results for the Poisson model fit to the 2015 in case there were any large variations.

- `R` **code**:

  → `MinimumContrast_final.R`: this `R` script takes the crime data from `DATA/PROCESSED_DATA/CRIME/POINT_PATTERN`, the census tract data from `DATA/PROCESSED_DATA/SHAPEFILES/CENSUS\_TRACTS` and the raw socio-economic variables from `DATA/RAW_DATA/COVARIATES/LA` with which we generate the population and average income at the census tract level, where the average income variable is processed as in `DATA/RAW_DATA/COVARIATES/CovDataGen_Inc_final.R` as this ensures that we have treated the variables in a similar manner to those for 2015 that are used in the final model fitting. As for Ripley's K in the `DATA/MODELS/GLMS/InitialModels_final.R` code, we use the census tract variables on an 236×359 (200m-by-200m) grid - without interpolation - in order to fit a `ppm` model that would allow us to estimate the mean field and include this in the minimum contrast method to account for some of the spatial variation that could be explained by the socio-economic variables. Although this `R` script outputs and saves the census tract level data frame, this is not used further.

  → `LA2014CountDataGenandModelRun_final.R`: this `R` script generates gridded data (mainly to generate the gridded data at the 200m-by-200m resolution) for the Los Angeles 2014 crime and socio-economic variables as for the 2015 data, although for the average income in this data we follow the methodology in `DATA/RAW_DATA/COVARIATES/CovDataGen_final.R` and the population and average income variables are interpolated onto the grid cells using the methodology in `DATA/PROCESSED_DATA/CRIME/COUNT_DATA_GMO/CountDataGen_GMO_final.R`. This is because the intention of this data set was to imitate the behaviour of the variables that would be used in the Grid-Mesh Optimisation in order to fit a simple Poisson GLM and gauge the direction and magnitude of the fixed effects in order to compare to the final parameter values for the covariates Grid-Mesh Optimisation method. Unlike the minimum contrast results, which would be important in the choice of mesh resolution for use in the Grid-Mesh Optimisation method but with the overall goal of being used for the modelling of the crime data. Note that the generation of the count data for this gridded data assumes that the population data over the census tracts has already been produced within the `MinimumContrast_final.R` code.

- **Outputs**:
  Outputs from `MinimumContrast_final.R`:

  → `LA_CTPop_14_proj.rds`: the total population from the 2014 ACS data in `DATA/RAW_DATA/COVARIATES` on the Los Angeles City census tracts. [RAW DATA]

  → `LA_CTinc_14_0imp_proj.rds`: the average income from the 2014 ACS data in `DATA/RAW_DATA/COVARIATES` on the Los Angeles City census tracts. In this we deal with the missing data in the average income variable as we did in `DATA/RAW_DATA/COVARIATES/CovDataGen_Inc_final.R`. [RAW DATA]

  → `LA2014CTCountData_proj_Final.rda`: this contains the census tract-level count data for both homicide (`ct_homcount.df`) and motor vehicle thefts (`ct_gtacount.df`) in Los Angeles in 2014 with the corresponding population and average income variables for the census tracts. [COLLATED DATA]

→ `LA2014CTSFCountData_proj_Final.rda`: this is the same as the census tract count data above for homicides (`sf_homcount`) and motor vehicle thefts (`sf_gtacount`) but stored as an `sf` object. [COLLATED DATA]

Outputs from `LA2014CountDataGenandModelRun_final.R`:

→ `LA_CTinc_14_imp_proj.rds`: the average income from the 2014 ACS data in `DATA/RAW_DATA/COVARIATES` on the Los Angeles City census tracts. In this we treat all missing data the same even if the estimated total households in the census tracts are zero, as in `DATA/RAW_DATA/COVARIATES/CovDataGen_final.R`. [RAW DATA]

→ `LA2014CT236359CountData_proj.rda`: this is the output count data for both homicides (`hom_countdf`) and motor vehicle thefts (`gta_countdf`) over the 236× 359 grid, with grid cell width of 200m, over the Los Angeles study regions with the socio-economic variables population and average income (where population variable generated in the `MinimumContrast_final.R` code) interpolated onto the grid using the same methodology as in `DATA/PROCESSED_DATA/CRIME/COUNT_DATA_GMO/CountDataGen_GMO_final.R`. [AGGREGATE DATA]

→ `LA2014CT236359SFCountData_proj.rda`: this is the output count data for both homicides (`sf_homcount`) and motor vehicle thefts (`sf_gtacount`) as above but saved as class `sf`. [AGGREGATE DATA]

Return to Table of Contents

2 GRID_MESH

This directory contains the code to implement the Grid-Mesh Optimisation method for the Gaussian Example and LGCP Example as discussed in Chapter 3 of my thesis. Additionally, we have the code for the Grid-Mesh Optimisation method applied to the Los Angeles City polygon as seen in Chapter 4. For each of these implementations we also include the necessary income data as well as the results of the Grid-Mesh Optimisation implementations. This directory is separated into three sub-directories for each individual implementation of the Grid-Mesh Optimisation method and as well as the code for running the Traditional and SBC simulation studies we also have code which produces the relevant plots in order to analyse the outputs of these simulation studies. In this directory, as well as the three sub-directories we have two `R` scripts, one that takes the outputs from the simulations split across several nodes and combines them into a single output file ready for analysing, and the second simulates from the PC prior for the range, allowing us to better understand the behaviour of the simulated variables as we alter the hyperparameters.

– **`R` code**:

→ `BalenaOutputCombined_final.R`: combines the outputs from both the traditional and SBC simulation studies into single outputs for each simulation study as the simulations are spread across multiple nodes. This takes the outputs stored in the sub-directories for each of the different Grid-Mesh Optimisation implementations and stores the combined outputs back into the same sub-directory.

→ `rangepriortest_final.R`: this just simulates the range, ρ, multiple times from the PC prior to assess its behaviour for different hyperparameter values.

2.1 GAUSSIAN

This directory contains the simulated data, meshes and code in order to implement the two simulation studies for the Grid-Mesh Optimisation method where we are interested in selecting the optimal mesh resolution where the data is Gaussian and is simulated over a regular polygon study region. The code, data, outputs and analysis are stored separately across three individual sub-directories and the output and results from this example are in Chapter 3 of my thesis.

2.1.1 GAUSSIAN_CODE

This directory contains the code and simulated covariates and mesh for the implementation of the traditional and SBC simulation studies for the Gaussian example.

- **Data**:
  - → `GridMeshSGSSCov.rda`: this contains two rasters for covariates (`cov1.ras`,`cov2.ras`) which are used to generate the data sets within the simulation studies.
  - → `MeshesRegPolSG.rda`: this contains a list of the four meshes used for the implementation of the INLA-SPDE algorithm in the two simulation studies.
- `R` **code**:
  - → `GridMeshOptimTradSG_final.R`: this `R` script contains the code to run the traditional simulation study for the Grid-Mesh Optimisation method. This file contains the code to simulate the covariates and meshes at different resolutions for the simulation study, which are commented out after their initial creation. This code is also available in the SBC simulation study `R` script but only needs to be run once.
  - → `GridMeshOptimSG_final.R`: this `R` script contains the code to run the SBC simulation study for the grid-mesh optimisation method. This file contains the code to simulate the covariates and meshes at different resolutions for the simulation study, which are commented out after their initial creation.
    Note: while there is code to calculate the ranks for the mean field, this was not considered as our interest was in the SBC results for the Gaussian field with the simulated Gaussian data. Therefore these outputs are unused and if interest lies in the mean field the code may need to be tidied/improved before further use of the mean field ranks. Additionally, for the Gaussian example (unlike the LGCP example and the LGCP implementation for the Los Angeles polygon) we output the CPO for each iteration, but these were also unused in our analysis.
  - → `PlottingSGRegularPolygonMesh_final.R`: this produces the plots of the mesh used in the Grid-Mesh Optimisation for the Gaussian example.
- **Outputs**:
  Outputs from `GridMeshOptimTradSG_final.R`: these are stored in `GRID_MESH/GAUSSIAN/GAUSSIAN_OUTPUT` directory.
  - → `GridMeshSGTradSSi.rda`: these are the outputs for the traditional simulation study, where we split the simulations in one node across separate runs for each of the 16 processors, so we had $i = 1, \ldots, 16$ output files.

  Outputs from `GridMeshOptimSG_final.R`: these are stored in `GRID_MESH/GAUSSIAN/GAUSSIAN_OUTPUT` directory

→ `GridMeshSGSBCSSi.rda`: these are the outputs for the SBC simulation study, where we split the simulations in two nodes across separate runs for each of the 32 processors, so we had $i = 1, \ldots, 32$ output files.

Outputs from `PlottingSGRegularPolygonMesh_final.R`:

→ **MeshRegPolSG*.pdf**: these are the plots of the meshes for different resolutions that are used within the simulation studies for the Gaussian example.

2.1.2 `GAUSSIAN_OUTPUT`

This directory contains the outputs from the two simulations studies, which were parallelised across the nodes, as well as the combination of the separate outputs into a single output file for each simulation study separately through the `GRID_MESH/BalenaOutputCombined_final.R` function.

– **Data**: these are the outputs from the simulation studies from the `R` scripts `GridMeshOptimTradSG_final.R` and `GridMeshOptimSG_final.R`.

→ `GridMeshSGTradSSi.rda`: these are the outputs for the traditional simulation study, where we split the simulations in one node across separate runs for each of the 16 processors, so we had $i = 1, \ldots, 16$ output files.

→ `GridMeshSGSBCSSi.rda`: these are the outputs for the SBC simulation study, where we split the simulations in two nodes across separate runs for each of the 32 processors, so we had $i = 1, \ldots, 32$ output files.

→ `GridMeshSGTradSS.rda`: the combination of the traditional simulation study outputs, `GridMeshSGTradSSi.rda`, into a single output by the `R` script `BalenaOutputCombined_final.R`.

→ `GridMeshSGSBCSS.rda`: the combination of the traditional simulation study outputs, `GridMeshSGSBCSSi.rda`, into a single output by the `R` script `BalenaOutputCombined_final.R`.

2.1.3 `GAUSSIAN_ANALYSIS`

This directory contains the code to produce the plots of the outputs results from the two simulation studies. There are R markdown files that run the `R` scripts with the input simulation study results. Note that the `.Rmd` files will be tidied up and may not be re-run after, and so the **.html** (with respect to the text and the name of `R` file sourced) file may not always match, but is kept to illustrate the output produced and this holds true for the LGCP example and the Grid-Mesh Optimisation implementation for the Los Angeles polygon. We also have some R scripts that consider the runs which produced errors during the simulation studies. We additionally have a sub-directory, `ReLabelledPlots` which contains code to re-produce the plots from the results created in this directory with small changes to the plot titles or axis labels. The functions will be discussed below, after the description of the `R` scripts and outputs for this directory.

– `R` **code**:

→ `GridMeshSGTradOutput_final.R` **and** `GridMeshSGTrad.Rmd`: this `R` script contains the code that takes in the output from the traditional simulation study and produces the necessary plots and tables to summarise the output, such as the average times for the `inla` runs or the parameter recovery. The `Rmd` file sets up the `R` script and the Gaussian outputs to produce these plots. (Additionally, there were outputs for the WAIC/DIC however, these are not used within my thesis for my decision-making process.)

→ `SBC_SG_Param_final.R` **and** `SBC_Param_SG.Rmd`: this `R` script considers the results for the SBC simulation study for the parameters only in order to produce the relevant summaries, such as plots (summary statistics, histograms) and tables. The histograms for the SBC ranks will be overlaid with the results of the GLMs for the rank frequencies, with the plots separated into two output PDFs depending on the whether there is a divergence from uniformity in the ranks or not. This `R` script will also present the errors and warnings information for the SBC simulation study, which is not considered within the `R` script for the latent field results. As for the traditional simulation study results, the `Rmd` sets up the parameters SBC results as well as the necessary `R` script in order to produce the necessary summaries. (Additionally, there were outputs for the WAIC/DIC however, these are not used within my thesis for my decision-making process.)

→ `SBC_SG_Latent_final.R` **and** `SBC_Latent_SG.Rmd`: this `R` script considers the results for the SBC simulation study for the latent Gaussian field only in order to produce the summaries such as the histogram plots, for the SBC ranks. In particular, for each Mesh the top 'worst' divergent histograms are plotted and output rather than all of the histogram plots as this is a very large number for each mesh resolution. The number of 'worst' histograms to plot and return are decided by the user in the `Rmd` file, which, as before, sets up the latent field data and runs the `R` script.

→ `TradTimeTable3sf_final.R`: this `R` script is to reproduce the results table for the traditional simulation study with three significant figures, done separately to prevent re-running the traditional simulation study `Rmd` unnecessarily.

→ `ParametersforErrors_final.R` **and** `GridMeshSGErrorMeanFieldChecks_final.R`: these `R` scripts are meant to briefly investigate the parameters and behaviour of the fixed mean of the latent field for the data that resulted in erroneous outputs.

– **Outputs**:
Outputs from `GridMeshSGTradOutput_final.R` **and** `GridMeshSGTrad.Rmd`

→ **GridMeshSGTrad.html**: this is the output html file from the `R` markdown and will contain the printed tables and some plots that are also saved.

→ **SGTrad*.pdf** and **SGCoverage.pdf**: these are the plots that are produced and saved when we run the `R` markdown file.

Outputs from `SBC_SG_Param_final.R` **and** `SBC_Param_SG.Rmd`

→ **SBC_Param_SG.html**: this contains some of the plots that are saved as well as some of the summary outputs.

→ **SGSBC*.pdf**, **sumdisttest_param.pdf**, **outsideboundstest_param.pdf**, **sbcdivergences_param.pdf**, **sbcnondivergences_param.pdf**: these are the plots that are produced and saved when we run the `R` markdown file. The plots in **SGSBC*.pdf** contain the general summaries for the simulation study while the others are the necessary plots for the SBC ranks: the summary statistics and the histograms that presented divergences or no divergence from uniformity.

Outputs from `SBC_SG_Latent_final.R` **and** `SBC_Latent_SG.Rmd`

→ **SBC_Latent_SG.html**: this contains some of the plots that are saved.

→ **sumdisttest.pdf**, **outsideboundstest.pdf**, **sbcdivergencesMesh*.pdf**: these are the plots that are produced and saved when we run the `R` markdown file. As well as the summary statistics we produce a separate file for each mesh resolution, *, which contains the 'worst' divergent histograms for the results for that particular resolution.

`ReLabelledPlots` **Sub-directory**

This contains additional `R` scripts and `R` markdowns that were essentially the same files as in the `GAUSSIAN_ANALYSIS` directory, but with some minor changes for plot titles or axis labels. Any plots that we do not alter are commented out to prevent reproducing these plots unnecessarily.

- `R` **code**:

  → `GridMeshSGTradOutput_Relabel.R` **and** `GridMeshSGTrad_Relabel.Rmd`: same as before, without the parameter recovery, credible interval coverage, error or DIC/WAIC plots output, only saving the re-plotted average time with '(s)' to denotes the time unit on the y-axis label.

  → `SBC_SG_Param_Relabel.R` **and** `SBC_Param_SG_Relabel.Rmd`: same as before with label changes for the summary statistics and title changes (remove mention of 'Grid 1' as there is only the grid used for the data generation here) for the histograms, and commenting out any plots for errors and DIC/WAIC for the SBC simulations.

  → `SBC_SG_Latent_Relabel.R` **and** `SBC_Latent_SG_Relabel.Rmd`: same as before, although as with the parameter SBC analysis we have changed the y-axis labels for the summary statistics and have also removed any mention of 'Grid 1' for the histogram titles.

- **Outputs**:
  Outputs from `GridMeshSGTradOutput_Relabel.R` **and** `GridMeshSGTrad_Relabel.Rmd`

  → **GridMeshSGTrad_Relabel.html**: this is the output html file from the `R` markdown and will contain the printed tables and some plots that are also saved.

  → **SGTradTimevMesh_Relabel.pdf**: this is the only plot that is reproduced and saved when we run the `R` markdown file.

  Outputs from `SBC_SG_Param_Relabel.R` **and** `SBC_Param_SG_Relabel.Rmd`

  → **SBC_Param_SG_Relabel.html**: this contains some of the plots that are saved as well as some of the summary outputs.

  → **sumdisttest_param_Relabel.pdf**, **outsideboundstest_param_Relabel.pdf**, **sbcdivergences_param_Relabel.pdf**, **sbcnondivergences_param_Relabel.pdf**: only the summary statistics and histogram plots are reproduced and saved plots when we run the `R` markdown file.

  Outputs from `SBC_SG_Latent_Relabel.R` **and** `SBC_Latent_SG_Relabel.Rmd`

  → **SBC_Latent_SG_Relabel.html**: this contains some of the plots that are saved.

  → **sumdisttest_Relabel.pdf**, **outsideboundstest_Relabel.pdf**, **sbcdivergencesMesh*_Relabel.pdf**: these are the reproduced plots that are saved when we run the `R` markdown file.

2.2 REGULAR_POLYGON_LGCP

This directory contains the simulated data, meshes, grids and code in order to implement the two simulation studies for the Grid-Mesh Optimisation method where we are interested in selecting the optimal mesh resolution where the LGCP data is simulated over a regular polygon study region. These are stored separately across three individual sub-directories and the output and results from this example are in Chapter 3 of my thesis.

2.2.1 REGPOLLGCP_CODE

This directory contains the code and simulated covariates, mesh and grids for the implementation of the traditional and SBC simulation studies for the LGCP example on a regular polygon. There is an additional sub-directory `REGPOLLGCP_CODE_TESTINGVALUES` within which we simulated multiple LGCPs for different prior hyperparameter values in order to select hyperparameter values that wouldn't result in extremely large and unrealistic point patterns being simulated.

- **Data**:
  - → `GridMeshRegPolSSCov.rda`: this contains two rasters for covariates (`cov1.ras`,`cov2.ras`) as well as these covariates as pixel images (`cov1.im`,`cov2.im`) and a pixel image for the intercept (`int.im` - a constant across the window of interest) which are used to generate the point patterns with the `spatstat::rLGCP` function.
  - → `MeshesRegPolLGCP.rda`: this contains a list of the four meshes used for the implementation of the INLA-SPDE algorithm in the two simulation studies.
  - → `QuadratsRegPolLGCP.rda`: this contains a list of the grids for the different data aggregation resolutions, pre-made and saved to prevent needing to re-produce these every time we simulate a new data set, which would be an unnecessary cost.
  - → `CovAggGridRegPolLGCP.rda`: this contains a list of the two covariates aggregated onto each of the above grids, pre-made to save recreating them for each simulated data set, especially for the finer grid resolutions.
  - → `CoordsRegPolLGCP.rda`: this contains three lists, one for the grid cell areas for each of the grid resolutions, the coordinates of the grid cell centres for each grid resolution and a list of data frames of the ordering of the grid cells, so that we traverse down the y axis before travelling across the x axis with respect to the grid cells.
- `R` **code**:
  - → `GridMeshOptimTrad_final.R`: this `R` script contains the code to run the traditional simulation study for the Grid-Mesh Optimisation method. This file also contains the code to simulate the covariates, grids and meshes at different resolutions for the simulation study, which are commented out after their initial creation. This code is also available in the SBC simulation study `R` script but only needs to be run once.
  - → `GridMeshOptim_final.R`: this `R` script contains the code to run the SBC simulation study for the Grid-Mesh Optimisation method. This file contains the code to simulate the covariates, grids and meshes at different resolutions for the simulation study, which are commented out after their initial creation. There was a minor error in the code when it was run, for the mean field SBC rank calculations, this typo has since been fixed with a comment discussing the alteration.

→ `GridMeshOptim_TimeErrorRuns_final.R`: this file contains the code (altered version of `GridMeshOptim_final.R`) to re-run the SBC simulation study for the simulation, iteration, grid and mesh which resulted in a TIME ERROR (did not complete in 6 hours) and run with all cores of the node available (16). This takes in the completed (with time error) simulation for a process which is stored as `GridMeshRegPolLGCPSBCSSi.rda` and, since there was only one time error in this SBC simulation study, we manually set the simulation-grid-mesh indices for the re-run and load the data which is stored with the suffix '_TIMEERROR1.rda' in `REGULAR_POLYGON_LGCP/REGPOLLGCP_OUTPUT/REGPOLLGCP_OUTPUT_ERROR/` in order to prevent the data being generated again unnecessarily. There was a minor error in the code when it was run, for the mean field SBC rank calculations, this typo has since been fixed with a comment discussing the alteration.

→ `PlottingRegularPolygonData_final.R`: this `R` script produces plots of the grids, meshes and covariates, where additional plots, targeted towards the simulations found in `SimulateRegPlot_final.R` which is found in the `REGPOLLGCP_CODE_TESTINGVALUES` sub-directory.

– **Outputs**:
Outputs from `GridMeshOptimTrad_final.R`: these are stored in `GRID_MESH/REGULAR_POLYGON_LGCP/REGPOLLGCP_OUTPUT` directory.

→ `GridMeshRegPolLGCPTradSSi.rda`: these are the outputs for the traditional simulation study, where we split the simulations in ten node across separate runs for each group of two out of the 16 processors per node, so that each simulation on the node had access to two processors, so we had $i = 1, \ldots, 80$ output files.

Outputs from `GridMeshOptim_final.R` **and** `GridMeshOptim_TimeErroRuns_final.R`: these are stored in `GRID_MESH/REGULAR_POLYGON_LGCP/REGPOLLGCP_OUTPUT` directory.

→ `GridMeshRegPolLGCPSBCSSi.rda`: these are the outputs for the SBC simulation study, where we split the simulations in fifteen nodes across separate runs for each group of two out of the 16 processors per node, so that each simulation on the node had access to two processors, so we had $i = 1, \ldots, 120$ output files. If there were any time errors, the necessary simulation-grid-mesh combination has been re-run and completed with the error message, 'TIME ERROR', removed and a warning message put in the output instead.

→ `GridMeshRegPolLGCPSBCSSi_wTE1.rda`: these outputs are the completed runs for the SBC simulation study although before the time error re-runs are implemented, so they still have a 'TIME ERROR' error message in the output. They are stored with a '_wTE1' to keep the original output from the SBC simulation study backed-up.

Outputs from `PlottingRegularPolygonData_final.R`:

→ **Quadrat*RegPolLGCP*.pdf**, **Meshes*RegPolLGCP*.pdf** and **CovariatesRegPolLGCP*.pdf**: these are the plots of the meshes, grid and covariates for different resolutions that are used within the simulation studies for the LGCP example.

`REGPOLLGCP_CODE_TESTINGVALUES` **Sub-directory**

This sub-directory contains the code to simulate different point patterns for combinations of hyperparameters in order to ensure that the values we consider do not result in extreme data sets being simulated often. We also test some possible values for the fixed parameter values in the traditional simulation study to assess whether the range of simulated points is also reasonable.

- **Data**:
  - → `GridMeshRegPolSSCov.rda`, `MeshesRegPolLGCP.rda`, `QuadratsRegPolLGCP.rda`, `CovAggGridRegPolLGCP.rda` **and** `CoordsRegPolLGCP.rda`: the data used in the simulation studies is copied into this sub-directory, although this is not entirely necessary.
- `R` **code**:
  - → `LGCPCovarianceandFixedPriorTestRegPol_final*.R`: this `R` script contains some rough code for some quick tests of the behaviour of the point pattern data generation for different priors for sigma and the fixed effects, while the prior for rho is held fixed. With *=b this involves a slightly different prior for rho which can be found in the prior simulation function.
  - → `thetatilden_final.R`: this `R` script produces some quick checks on the simulation of the point patterns for fixed values of the parameters, without any interest in the priors.
  - → `SimulateRegPlot_final.R`: this `R` script produces some plots of the simulated data sets from the fixed parameter values that will be used in the simulation studies.
- **Outputs**:
  Outputs from `LGCPCovarianceandFixedPriorTestRegPol_final*.R`:
  - → `RegPolCovFixedEffectsPriori*.rda`: these are the outputs from different combinations of priors. The suffix i indexes the simulation hyperparameter values, which are also labelled in the `R` scripts and the inclusion of *=b refers to the results of the corresponding `R` script with the different rho prior. The additional suffix '_LongTest' in the output indicates that it was run with more simulations than the original runs.

  Outputs from `SimulateRegPlot_final.R`:
  - → **RegPolSimulatedDataSeti.pdf**: these are the plots of the data simulated using the fixed values of the parameters that will be used for the traditional simulation study, where $i = 1, 2, 3$.

2.2.2 `REGPOLLGCP_OUTPUT`

This directory contains the outputs from the two simulations studies, which were parallelised across the nodes, as well as the combination of the separate outputs into a single output file for each simulation study separately through the `GRID_MESH/BalenaOutputCombined_final.R` function. Additionally, there was a time error, where one of the runs was running for more than six hours without results, and therefore we manually stopped the simulation, placed an error (done within the sub-directory `REGPOLLGCP_OUTPUT_ERROR`), 'TIME ERROR', in the results for that particular simulation, grid and mesh combination in order to continue running the remaining simulations and return to this time error where we will allow for more processors for the computations to complete (16 rather than 2), using the `R` script `GridMeshOptim_TimeErrorRuns_final.R`.

– **Data**: these are the outputs from the simulation studies from the `R` scripts `GridMeshOptimTrad_final.R`, and `GridMeshOptim_final.R`.

→ `GridMeshRegPolLGCPTradSSi.rda`: these are the outputs for the traditional simulation study, where we split the simulations in ten node across separate runs for each group of two out of the 16 processors per node, so that each simulation on the node had access to two processors, so we had $i = 1, \ldots, 80$ output files.

→ `GridMeshRegPolSBCSSi.rda`: these are the outputs for the SBC simulation study, where we split the simulations in fifteen nodes across separate runs for each group of two out of the 16 processors per node, so that each simulation on the node had access to two processors, so we had $i = 1, \ldots, 120$ output files. If there were any time errors, the necessary simulation-grid-mesh combination has been re-run and completed with the error message, 'TIME ERROR', removed and a warning message put in the output instead.

→ `GridMeshRegPolSBCSSi_wTE1.rda`: this is the output after all simulations, except the time error re-run, have been completed. This output is saved as a back-up version of the original SBC simulation study output. However, we take in `GridMeshRegPolSBCSSi.rda` along with the generated data for this simulation, `REGPOLLGCP_OUTPUT_ERROR/temp_dataisbc_TIMEERROR1.rda`, for the re-run with more processors (16) available in `GridMeshOptim_TimeErrorRuns_final.R`.

→ `GridMeshRegPolLGCPTradSS.rda`: the combination of the traditional simulation study outputs, `GridMeshRegPolLGCPTradSSi.rda`, into a single output by the `R` script `BalenaOutputCombined_final.R`.

→ `GridMeshRegPolLGCPSBCSS.rda`: the combination of the traditional simulation study outputs, `GridMeshRegPolLGCPSBCSSi.rda`, into a single output by the `R` script `BalenaOutputCombined_final.R`.

`REGPOLLGCP_OUTPUT_ERROR` **Sub-directory**

This sub-directory contains the `R` script that inserts a 'TIME ERROR' into the output of an SBC simulation study where the run took an excessively long time to run, therefore we can continue simulations without and return to this error combination of simulation, grid and mesh at the end and make use of more processors. For each 'TIME ERROR' run we have the output at the time of the error, before we included the error message, and the output after we included the error message as well as the data set that resulted in the time error (so as to not need to re-simulate when re-running) and some text files that were output from the error insertion `R` script to ensure there were no mistakes in the error inclusion that resulted in a loss of data from the output.

– **Data**:

→ `GridMeshRegPolLGCPSBCSSi_TIMEERROR1.rda`: this denotes the SBC process that for a particular simulation and grid-mesh combination, it took over 6 hours to complete and therefore needs to be re-run. This is the output when the Time Error was found and so not all the runs are completed but the 'TIME ERROR' has not been inserted yet. The suffix is included as the error insertion will overwrite the original `GridMeshRegPolLGCPSBCSSi.rda` output to return to the SBC simulation study runs.

→ `temp_dataisbc_TIMEERROR1.rda`: this is the generated data for the simulation for process i that produced a time error and it is stored so that it can be easily loaded for the re-runs in `GridMeshOptim_TimeErrorRuns_final.R`, preventing any unnecessary re-generation of the data set in the re-run.

– `R` **code**:

→ `RegPolSBCTimeErrInsertProcs_final.R`: this `R` code places an error the selected grid-mesh combination for a particular simulation and produces several text files where the old and new (pre/post-error addition) outputs can be compared to check that the code didn't make any other odd changes.

– **Outputs**:

→ `GridMeshRegPolLGCPSBCSSi.rda`: this denotes the SBC process that for a particular simulation and grid-mesh combination, it took over 6 hours to complete and therefore needs to be re-run. This is the output when the Time Error was found and so not all the runs are completed and 'TIME ERROR' was placed in the correct position for continuing the runs - this is not the completed output for process i and matched the '*_TIMEERROR1.rda' output with the only difference being the included error message.

→ `*.txt`: the text files in this sub-directory contain the printed outputs from the checks in `RegPolSBCTimeErrInsertProcs_final.R` which allow us to compare the results pre- and post- time error addition to check that no major errors were made during the addition of the error.

2.2.3 `REGPOLLGCP_ANALYSIS`

This directory contains the code to produce the plots of the outputs results from the two simulation studies. There are R markdown files that run the `R` scripts with the input simulation study results. We additionally have a sub-directory, `ReLabelledPlots` which contains code to re-produce the plots from the results created in this directory with small changes to the plot titles or axis labels. The functions will be discussed below, after the description of the `R` scripts and outputs for this directory.

– `R` **code**:

→ `GridMeshRegPolLGCPTradOutput_final.R`, `GridMeshRegPolLGCPTradbyGrid.Rmd` **and** `GridMeshRegPolLGCPTradbyMesh.Rmd`: this `R` scripts contains the code that takes in the output from the traditional simulation study and produces the necessary plots, such as the average times for the `inla` runs or the parameter recovery, and tables to summarise the output and grouped by grid or mesh, respectively. The `Rmd` file sets up the `R` script and the outputs for the LGCP on a regular polygon to produce these plots. (Additionally, there were outputs for the WAIC/DIC however, these are not used within my thesis for my decision-making process.)

→ `SBC_RegPolLGCP_Param_final.R` **and** `GridMeshRegPolLGCPSBCParam.Rmd`: this `R` script considers the results for the SBC simulation study for the parameters only in order to produce the relevant summaries, such as plots (summary statistics, histograms) and tables. The histograms for the SBC ranks will be overlaid with the results of the GLMs for the rank frequencies, with the plots separated into two output PDFs assigning the histograms to one or the other depending on the whether there is a divergence from uniformity in the ranks or not. This `R` script will also present the errors and warnings information for the SBC simulation study, which is not considered within the `R` script for the mean field results. As for the traditional simulation study results, the `Rmd` sets up the parameters SBC results as well as the necessary `R` script in order to produce the necessary summaries. The plots for the SBC rank results are grouped by grid or mesh except for the divergences or non-divergences from uniformity. (Additionally,

there were outputs for the WAIC/DIC however, these are not used within my thesis for my decision-making process.)

→ `SBC_RegPolLGCP_Latent_final.R` **and** `GridMeshRegPolLGCPSBCLatent.Rmd`: this `R` script considers the results for the SBC simulation study for the mean field only in order to produce the summaries such as the histogram plots for the SBC ranks. In particular, for each Grid the top 'worst' divergent histograms are plotted and output rather than all of the histogram plots as this is a very large number for each mesh resolution. The number of 'worst' histograms to plot and return are decided by the user in the `Rmd` file, which, as before, sets up the mean field data and runs the `R` script. The summary statistic plots are output grouped by mesh or grid, but the SBC divergences are output by grid only.

→ `ParametersforErrors_final.R` **and** `GridMeshErrorMeanFieldChecks_final.R`: these `R` scripts are meant to briefly investigate the parameters and behaviour of the fixed mean of the latent field for the data that resulted in erroneous outputs.

– **Outputs**:
Outputs from `GridMeshRegPolLGCPTradOutput_final.R`, `GridMeshRegPolLGCPTradbyGrid.Rmd` **and** `GridMeshRegPolLGCPTradbyMesh.Rmd`

→ **GridMeshRegPolLGCPTradbyGrid.html**: this is the output html file from the `R` markdown and will contain the printed tables and some plots that are also saved.

→ **GridMeshRegPolLGCPTradbyMesh.html**: this is the output html file from the `R` markdown and will contain the printed tables and some plots that are also saved.

→ **RegPolLGCPTrad*.pdf** and **RegPolLGCPCoverageby*.pdf**: these are the plots that are produced and saved when we run the `R` markdown file, where 'by*' denotes the grouping in the plots either by the Grid resolution of Mesh resolution.

Outputs from `SBC_RegPolLGCP_Param_final.R` **and** `GridMeshRegPolLGCPSBCParam.Rmd`

→ **GridMeshRegPolLGCPSBCParam.html**: this contains some of the plots that are saved as well as some of the summary outputs.

→ **RegPolLGCPSBC*.pdf**, **sumdisttest_paramwErr_by*.pdf**, **outsideboundstest_paramwErr_by*.pdf**, **sbcdivergences_paramwErr.pdf**, **sbcnondivergences_paramwErr.pdf**: these are the plots that are produced and saved when we run the `R` markdown file. The plots in **RegPolLGCPSBC*.pdf** contain the general summaries for the simulation study while the others are the necessary plots for the SBC ranks: the summary statistics and the histograms. Note that the 'wErr' denotes the inclusion of the simulations which resulted in more than 10 FFT warnings (which are discussed in Chapter 3 and Appendix C of my thesis) rather than excluding them. Not all of the output plots have this included in the filenames, however, the same data is used across all of the analysis, where we do not exclude any of the runs with > 10 warnings.

Outputs from`SBC_RegPolLGCP_Latent_final.R` **and** `GridMeshRegPolLGCPSBCLatent.Rmd`

→ **GridMeshRegPolLGCPSBCLatent.html**: this contains some of the plots that are saved.

→ **sumdisttestby*.pdf**, **outsideboundstestby*.pdf**, **sbcdivergencesGrid*.pdf**: these are the plots that are produced and saved when we run the `R` markdown file. The summary statistics are given grouped by either Grid or Mesh denoted by *, and as well as these we produce a separate file for the histograms for each Grid resolution, *, which contains the 'worst' divergent histograms for the results for that particular resolution.

Outputs from `GridMeshErrorMeanFieldChecks_final.R`

→ **MeanFieldandExpMeanFieldforErr.pdf**: these are the plots of the fixed mean and exponential of the fixed mean for the parameters where we had errors in our runs.

`ReLabelledPlots` **Sub-directory**

This contains additional `R` scripts and `R` markdowns that were the same files as in the `REGPOLLGCP_ANALYSIS` directory with some minor change to the axis labels or titles. Any plots that we do not alter are commented out to prevent reproducing these plots unnecessarily.

– `R` **code**:

→ `GridMeshRegPolLGCPTradOutput_Relabel.R`, `GridMeshRegPolLGCPTradbyGrid_Relabel.Rmd` **and** `GridMeshRegPolLGCPTradbyMesh_Relabel.Rmd`: same as before, without the parameter recovery, credible interval coverage, error or DIC/WAIC plots output, only saving the re-plotted average time with '(s)' to denotes the time unit on the y-axis label.

→ `SBC_RegPolLGCP_Param_Relabel.R` **and** `GridMeshRegPolLGCPSBCParam_Relabel.Rmd`: same as before with label changes for the summary statistics, and commenting out any plots for errors and DIC/WAIC for the SBC simulations.

→ `SBC_RegPolLGCP_Latent_Relabel.R` **and** `GridMeshRegPolLGCPSBCLatent_Relabel.Rmd`: same as before, although as with the parameter SBC analysis we have changed the y-axis labels for the summary statistics.

– **Outputs**:
Outputs from `GridMeshRegPolLGCPTradOutput_Relabel.R`, `GridMeshRegPolLGCPTradbyGrid_Relabel.Rmd` **and** `GridMeshRegPolLGCPTradbyMesh_Relabel.Rmd`

→ **GridMeshRegPolLGCPTradbyGrid_Relabel.html**: this is the output html file from the `R` markdown and will contain the printed tables and some plots that are also saved.

→ **GridMeshRegPolLGCPTradbyMesh_Relabel.html**: this is the output html file from the `R` markdown and will contain the printed tables and some plots that are also saved.

→ **RegPolLGCPTradTimev*facet**_Relabel.pdf,RegPolLGCPTradTimeby*_Relabel.pdf**: this is the only plot that is re-produced and saved when we run the `R` markdown file. Where * and ** are one of Grid or Mesh, but *≠**.

Outputs from `SBC_RegPolLGCP_Param_Relabel.R` **and** `GridMeshRegPolLGCPSBCParam_Relabel.Rmd`

→ **GridMeshRegPolLGCPSBCParam_Relabel.html**: this contains some of the plots that are saved as well as some of the summary outputs.

→ **sumdisttest_paramwErr_by*_Relabel.pdf**, **outsideboundstest_paramwErr_by*_Relabel.pdf**: the summary statistics are only re-produced and saved plots when we run the `R` markdown file. * is either Mesh or Grid.

Outputs from `SBC_RegPol_Latent_Relabel.R` **and** `GridMeshRegPolLGCPSBCLatent_Relabel.Rmd`

→ **GridMeshRegPolLGCPSBCLatent_Relabel.html**: this contains some of the plots that are saved.

→ **sumdisttestby*_Relabel.pdf**, **outsideboundstestby*_Relabel.pdf**: these are the re-produced plots that are saved when we run the `R` markdown file. * is either Mesh or Grid.

2.3 IRREGULAR_POLYGON_LGCP

This directory contains the simulated data, meshes, grids and aggregated covariates for the implementation of the two simulation studies of the Grid-Mesh Optimisation method where the LGCP data is simulated over the Los Angeles polygon. These are stored separately across three individual sub-directories and the output and results from this example are in Chapter 4 of my thesis.

2.3.1 IRREGPOLLGCP_CODE

This directory contains the code and data for the implementation of the traditional and SBC simulation studies for the LGCP example on the Los Angeles polygon including the code to re-run any TIME or SPACE errors that arose during the SBC simulation study. There is an additional sub-directory `IRREGPOLLGCP_CODE_TESTINGVALUES` within which we simulated multiple LGCPs for different prior hyperparameter values in order to select hyperparameter values that wouldn't result in extremely large and unrealistic point patterns being simulated.

– **Data**: Note that there are 5 resolutions for meshes, grids and therefore the coordinates and aggregated covariates within their respective lists. This is because they were generated initially considering the grid cell width/ maximum mesh edge lengths of $5km$, $2km$, $1km$, $0.5km$ and $0.2km$ where the latter was considered too expensive (computation-wise) to consider, especially with respect to the mesh resolution and so was excluded from the simulation studies. The `R` scripts extract the required resolution elements from each of the lists for the simulation studies.
The code to generate the data for the simulations below can be found in `GridMeshOptimIrregTrad_final.R` or `GridMeshOptimIrreg_final.R`. Once the data is produced, the code can be commented out in order to run the simulation studies.

→ `WindowsIrregPolLGCP.rda`: this contains the Los Angeles window which is scaled and shifted so that a single unit shift in the x or y direction is related to a 10km shift and the bottom-left corner of the bounding box of the window lies at the origin, $(0, 0)$. This scaled and shifted polygon window is stored under `W` in this data, with an unscaled and unshifted polygon also stored in this data. This is produced using the file `DATA/PROCESSED_DATA/SHAPEFILES/CENSUS_TRACTS/LACityCT.rda`.

→ `GridMeshIrregPolSSCov.rda`: this contains two rasters for covariates (`pop.ras`,`inc.ras`) as well as these covariates as pixel images (`popb.im`,`incb.im`) and a pixel image for the intercept (`intb.im` - a constant across the window of interest) which are used to generate the point patterns with the `spatstat:rLGCP` function. These covariates are generated from the population and average income variables in the count data for Los Angeles generated in `DATA/PROCESSED_DATA/CRIME/COUNT_DATA_GMO`, in particular `LA2015CT236359CountData_proj.rda` which is the Los Angeles data aggregated onto a grid with resolution 200m-by-200m. Additionally, note that the population variable is interpolated in the same manner as the

final data sets for modelling the crime (found in `COUNT_DATA_FINAL`), but the average income in this data varies slightly from the interpolated average income in the final data sets.

→ `MeshesIrregPolLGCP.rda`: this contains a list of the meshes used for the implementation of the INLA-SPDE algorithm in the two simulation studies.

→ `QuadratsIrregPolLGCP.rda`: this contains a list of the grids for the different data aggregation resolutions, pre-made and saved to prevent needing to reproduce these every time we simulate a new data set, which would be an unnecessary cost.

→ `CovAggGridIrregPolLGCP.rda`: this contains a list of the two covariates aggregated onto each of the above grids, pre-made to save recreating them for each simulated data set.

→ `CoordsIrregPolLGCP.rda`: this contains three lists, one for the grid cell areas for each of the grid resolutions, the coordinates of the grid cell centres for each grid resolution and a list of data frames of the ordering of the grid cells, so that we traverse down the y axis before travelling across the x axis with respect to the grid cells.

– `R` **code**:

→ `GridMeshOptimIrregTrad_final.R`: this `R` script contains the code to run the traditional simulation study for the grid-mesh optimisation method. This file contains the code to simulate the covariates, grids and meshes at different resolutions for the simulation study, which are commented out after their initial creation. This code is also available in the SBC simulation study `R` script but only needs to be run once.

→ `GridMeshOptimIrreg_final.R`: this `R` script contains the code to run the SBC simulation study for the grid-mesh optimisation method. This file contains the code to simulate the covariates, grids and meshes at different resolutions for the simulation study, which are commented out after their initial creation. There was a minor error in the code when it was run, for the mean field SBC rank calculations, this typo has since been fixed with a comment discussing the alteration.

→ `TimeErrorProcessandDataGeneration_final.R`: this code finds the Time Errors in the completed runs, and makes a note of the Simulation-Grid-Mesh indices and creates a data frame for use within `GridMeshOptimIrreg_TimeErrorRuns_final.R` for the re-runs. These are saved in a list of data frames in `TimingErrorDataFrames.rda` which are used for the Time Error re-runs on Balena. This `R` script will also take these completed outputs and save them under the same file name with '_TIMEERRORFINAL' added as a suffix to store separately and to be called by `GridMeshOptimIrreg_TimeErrorRuns_final.R`. These are stored in the `IRREGULAR_POLYGON_LGCP/IRREGPOLLGCP_OUTPUT` directory.

→ `GridMeshOptimIrreg_TimeErrorRuns_final.R`: this file contains the code (altered version of `GridMeshOptimIrreg_final.R`) which re-ran the SBC simulation study for the simulation, iteration, grid and mesh which resulted in a TIME ERROR j (did not complete in 6 hours) and run with all cores of the node available (16). The outputs loaded for the re-runs in have the suffix '_TIMEERRORFINAL' and the re-run saves the final output without any suffix added - the original output file name. There was a minor error in the code when it was run, for the mean field SBC rank calculations, this typo has since been

fixed with a comment discussing the alteration.
As well as the Grid-Mesh Optimisation outputs we also output the data frames for each individual process that were produced in a single list as `TimingErrorDataFrames.rda` from `TimeErrorProcessandDataGeneration_final.R`, and these individual data frames are saved as `ProcessiErrorDf.rda` during the runs and stored in the `IRREGULAR_POLYGON_LGCP/IRREGPOLLGCP_OUTPUT/` directory.

→ `GridMeshOptimIrreg_SpaceErrorRuns_final.R`: this file contains the code (altered version of `GridMeshOptimIrreg_final.R`) which re-ran the SBC simulation study for the simulation, iteration, grid and mesh which resulted in a space error (due to issues with the temporary directory, jobs initially failed due to a lack of space and when the error was removed and the jobs re-started some ended up not producing another error, while others still presented errors and were therefore re-run after all other simulations and TIME ERROR re-runs were completed) and run with all cores of the node available (16). The outputs loaded for the re-runs have the suffix '_SPACEERRORFINAL' and the re-run saves the final output without any suffix added - the original output file name. There was a minor error in the code when it was run, for the mean field SBC rank calculations, this typo has since been fixed with a comment discussing the alteration.
Note: if there was no problem with the temporary directory initially, then these would likely be unnecessary for the simulation study.

– **Outputs**:
Outputs from `GridMeshOptimTradIrreg_final.R`: these are stored in `GRID_MESH/IRREGULAR_POLYGON_LGCP/IRREGPOLLGCP_OUTPUT` directory.

→ `GridMeshIrregPolLGCPTradSSi.rda`: these are the outputs for the traditional simulation study, where we split the simulations in twenty nodes across separate runs for each group of eight out of the 16 processors per node, so that each simulation on the node had access to eight processors, so we had $i = 1, \ldots, 40$ output files.

Outputs from `GridMeshOptimIrreg_final.R`, `GridMeshOptimIrreg_TimeErrorRuns_final.R`,`GridMeshOptimIrreg_SpaceErrorRuns_final.R`: these are stored in `GRID_MESH/IRREGULAR_POLYGON_LGCP/IRREGPOLLGCP_OUTPUT` directory.

→ `GridMeshRegPolLGCPSBCSSi.rda`: these are the outputs for the SBC simulation study, where we split the simulations in fifteen nodes across separate runs for each group of eight out of the 16 processors per node, so that each simulation on the node had access to eight processors, so we had $i = 1, \ldots, 120$ output files.
Note that the outputs from `GridMeshOptimIrreg_final.R` without any of the error re-runs completed are also saved under `GridMeshRegPolLGCPSBCSSi_TIMEERRORFINAL.rda` when `TimeErrorProcessandDataGeneration_final.R` is run. After the time error re-runs through `GridMeshOptimIrreg_TimeErrorRuns_final.R` any of the remaining outputs that had space errors are manually saved under `GridMeshRegPolLGCPSBCSSi_SPACEERRORFINAL.rda` as well as these are loaded in by `GridMeshOptimIrreg_SpaceErrorRuns_final.R` and are also back-ups of the post-time error re-run completed SBC outputs.

→ `ProcessiErrorDf.rda`: the time error re-runs also extract the relevant process' data frame from the list in `TimingErrorDataFrames.rda` and fill in the final column as the re-runs progress and these are output when implementing the Time Error re-runs. These data frame outputs for each process, after the TIME

ERROR re-runs, should have a 1 in the final column if the run was complete, if there is a 2 then there was an error when running, usually this also occurred after 12 hours (or the error was the $>$ 12hr run), and so was not re-run. Additionally, in our case there are two with the value 0, which originally indicates that this was not re-run but in our case, the re-run took place but took over 12 hours although when we manually stopped the run, there was no 2 placed in the final column as it should have been.

Outputs from `TimeErrorProcessandDataGeneration_final.R`: these are stored in `GRID_MESH/IRREGULAR_POLYGON_LGCP/IRREGPOLLGCP_OUTPUT` directory.

→ `TimingErrorDataFrames.rda`: this contains two outputs, the first is a list of data frames, one data frame for each process with at least one time error. The data frame contains the relevant information about the indices for the time error, in particular the simulation-grid-mesh as well as the error index as there were multiple instances where there were errors, sometimes multiple errors for one process, and so we identified each instance of a group of timing errors by an increasing index j and labelled the error message with 'TIME ERROR j'. This also allows us to identify the relevant data set simulated for each particular error (stored in separate sub-directories for each time error) and so this column allows us to load the correct process and data set for each error and re-run the correct simulation and grid-mesh combinations. The final column 're-run' is initialised at zero and is intended to act as an indicator of completed re-runs or error outputs in the re-runs. The second output is a list of the processes that had at least one time error.

→ `GridMeshRegPolLGCPSBCSSi_TIMEERRORFINAL.rda`: as mentioned above the final outputs pre-time error re-runs are also saved under this file name as these are loaded in `GridMeshOptimIrreg_TimeErrorRuns_final.R` for the re-runs and also ensure we have the original outputs from the SBC simulation study backed-up.

`IRREGPOLLGCP_CODE_TESTINGVALUES` **Sub-directory**

This sub-directory contains the code to simulate different LGCPs for combinations of hyperparameters in order to ensure that the values we consider do not result in extreme data sets being simulated often. We also test some possible values for the fixed parameter values in the traditional simulation study to assess whether the range of simulated points is also reasonable.

– **Data**:

→ `GridMeshIrregPolSSCov.rda`, `MeshesIrregPolLGCP.rda`, `QuadratsIrregPolLGCP.rda`, `CovAggGridIrregPolLGCP.rda` **and** `CoordsIrregPolLGCP.rda`: the data used in the simulation studies is copied into this sub-directory, although this is not entirely necessary.

– `R` **code**:

→ `LGCPCovarianceandFixedPriorTest_final*.R`: this `R` script contains some rough code for some quick tests of the behaviour of the data generation for different priors for sigma and the fixed effects, while the prior for rho is held fixed. With *=bRho, cRho this involves a slightly different prior for rho which can be found in the prior simulation function.

→ `thetatilden_final.R`: this `R` script produces some quick checks on the simulation of the point patterns for fixed values of the parameters, without any interest in the priors.

→ `DataPlottingIrregPol_final.R`: this produces some plots (using `tmap`) of the simulated data sets from the fixed parameter values that will be used in the simulation studies.

– **Outputs**:
Outputs from `LGCPCovarianceandFixedPriorTest_final*.R`:

→ `LACovFixedEffectsPriori*.rda`: these are the outputs from different combinations of priors. The suffix i indexes the simulation hyperparameter values, which are also labelled in the `R` scripts and the inclusion of *=b or c refers to the results of the corresponding `R` script with the different rho prior. Initial simulations produced $N = 50$ point patterns for each combination, but results with 'Long_' in the filename set $N = 100$ and those with the additional suffix of '2' or 'Take2' set $N = 500$.

Outputs from `DataPlottingIrregPol_final.R`:

→ **MeshesIrregPolLGCP.pdf**, **QuadratsIrregPolLGCP.pdf**, **IrregPolLGCP-Covariates.pdf**, **IrregPolLGCPGriddedCovariates**, **IrregPolSimStudySimulatedDataSeti*.pdf** and **LA*IrregPolSimStudy.pdf**: these are the plots of the grids, meshes and covariates for the simulation studies as well as the true Los Angeles crime data, and the simulated data on two of the grid resolutions as a comparison to the true crime data. There were three simulated data sets for these plots, and so $i = 1, 2, 3$.

2.3.2 `IRREGPOLLGCP_OUTPUT`

This directory contains the outputs from the two simulations studies, which were parallelised across the nodes, as well as the combination of the separate outputs into a single output file for each simulation study separately through the `GRID_MESH/BalenaOutputCombined_final.R` function. Additionally, there were several processes with at least one time error, where a computation was running for more than six hours without completing, and therefore, we manually stopped the simulation, placed and error, 'TIME ERROR j', in the results for that particular simulation, grid and mesh combination in order to continue running the remaining simulations and return to this time error where we will allow for more processors for the computations to complete (16 rather than 2), using the `R` script `GridMeshOptimIrreg_TimeErrorRuns_final.R`. The j indexes the time errors, as there were multiple at different stages of the simulation study, and even multiple for some processes, and so these were grouped when possible (if they happened around the same time, the processes were grouped together) and for each process in this group the error message placed before continuing the simulation study was 'TIME ERROR j' (or just 'TIME ERROR' for the first such error). The data sets were also saved and stored using similar identifiers so that we could re-load the necessary data sets for the erroneous runs rather than re-simulate them.

As well as the time error, due to temporary directory issues in the initial few simulations, there were jobs (with two processes running on the same node) that stopped due to error, where the error was caused by a lack of memory space. This placed errors in both processes, and so, originally, we re-set the outputs to remove the errors and continue running the simulations. Some processes progressed past the initial error position and therefore could not have caused the issue, while some processes were caught on the particular combination of simulation-grid-mesh, and so when the job stopped due to error again the errors for these processes were kept for this process and removed permanently from the other process sharing the node. Once the issue with temporary directory on Balena

was solved, these space issues no longer occurred, and so once all the runs were completed and the time errors were also re-run, we re-ran these 'SPACE ERRORS'.

All necessary R scripts for the removal/insertion of errors are found in the IRREGPOLLGCP_OUTPUT_ERROR sub-directory and will be discussed below.

- **Data**: these are the outputs from the simulation studies from the R scripts GridMeshOptimIrregTrad_final.R, GridMeshOptimIrreg_final.R, GridMeshOptimIrreg_TimeErrorRuns_final.R and GridMeshOptimIrreg_SpaceErrorRuns_final.R.

  → GridMeshIrregPolLGCPTradSSi.rda: these are the outputs for the traditional simulation study, where we split the simulations in twenty nodes across separate runs for each group of eight out of the 16 processors per node, so that each simulation on the node had access to eight processors, so we had $i = 1, \ldots, 40$ output files.

  → GridMeshIrregPolSBCSSi.rda: these are the outputs for the SBC simulation study, where we split the simulations in twenty nodes across separate runs for each group of eight out of the 16 processors per node, so that each simulation on the node had access to eight processors, so we had $i = 1, \ldots, 40$ output files. These are also the outputs after the time-error and space-error re-runs were completed for the necessary processes.

  → GridMeshRegPolSBCSSi_TIMEERRORFINAL.rda: this is the completed outputs before the time errors were re-run with a larger number of processors. These were then copied over to Balena to load into GridMeshOptimIrreg_TimeErrorRuns_final.R for the necessary re-runs.

  → GridMeshIrregPolLGCPSBCSSi_FULL.rda: the outputs from the time error re-runs, although some of the re-runs took over 12 hours (the time limit for re-runs) and while others were manually stopped these runs had been missed. Therefore for consistency we find such runs, manually re-label them with the suffix '_FULL' and re-set those particular outputs to be missing as though they were not retrieved initially. These data sets are the outputs containing these completed runs, before the relevant output was removed and can also be found in the GRID_MESH/IRREGULAR_POLYGON_LGCP/IRREGPOLLGCP_OUTPUT/IRREGPOLLGCP_OUTPUT_ERROR directory as well, where the code to remove the output is also found and implemented.

  → GridMeshIrregPolLGCPSBCSSi_SPACEERROFINAL.rda: this is the completed output (inc. any re-runs for time errors having been completed) before the errors due to 'no space for memory allocation' were re-run. These were then copied over to Balena to load into GridMeshOptimIrreg_SpaceErrorRuns_final.R for the necessary re-runs.

  → TimingErrorDataFrames.rda: this is the output from GRID_MESH/IRREGULAR_POLYGON_LGCP/IRREGPOLLGCP_CODE/TimingErrorProcessandDataGeneration_final.R and contains a list of data frames for each process that had at least one timing error where the data frame contains information on the simulation-grid-mesh indices as well as the index for the error that will be found in the error message which is important for matching the correct simulated data (stored in the sub-directories TimeErrorData/TIMEERRORj) to load for the error re-run. Finally, there is a column in the data frame which is a numeric code, initialised to zero, to tell whether a re-run has been completed (1), re-run produced and error (2), or not re-run.

  → ProcessiErrorDf.rda: these are individual data frames from the list TimingErrorDataFrames.rda for each process with at least one error. This output is

saved for each individual process during the re-runs for easy re-loading if a run is needed to re-start due to any time limits. These are the same as the data frames in `TimingErrorDataFrames.rda` but with the final column indicating re-runs completed/errors/not started filled in as the time error re-runs occur. These data frame outputs for each process, after the TIME ERROR re-runs, should have a 1 in the final column if the run was complete, if there is a 2 then there was an error when running, usually this also occurred after 12 hours (or the error was the $>$ 12hr run) and so was not re-run. Additionally, in our case there are two i such that one of the rows of the data frame has the value 0, which originally indicates that this was not re-run but in our case, it was and took over 12 hours but there was no 2 placed as there should have been.

→ `GridMeshIrregPolLGCPTradSS.rda`: the combination of the traditional simulation study outputs, `GridMeshIrregPolLGCPTradSSi.rda`, into a single output by the `R` script `BalenaOutputCombined_final.R`.

→ `GridMeshIrregPolLGCPSBCSS.rda`: the combination of the SBC simulation study outputs (after all necessary error re-runs), `GridMeshIrregPolLGCPSBCSSi.rda`, into a single output by the `R` script `BalenaOutputCombined_final.R`.

`IRREGPOLLGCP_OUTPUT_ERROR` **Sub-directory**

As mentioned above, there were two potential error issues within this simulation study.

First, due to temporary directory issues that were later resolved, there were jobs that failed due to there being no memory left for allocation and when the jobs failed, often the resulted in errors being placed in the outputs for both of processes that were running. In order to deduce which process was having issues, we initially re-set the process outputs to remove these errors and re-run. If the job continued with no error then we let the simulations continue. If the error occurred again then we checked the simulation-grid-mesh indices for the newly placed errors in both processes. Usually one of the processes surpassed it's original error location and so we concluded that this was not the process that caused the 'space error' and thus removed the error only for this process and kept the error for the other process and let the simulations continue. Once the temporary directory issue was resolved this did not occur again, and so we resolved to re-run the processes that we kept the errors in after all remaining simulations were completed, and using all 16 processors in the node for each re-run. This sub-directory contains the necessary code for re-setting the errors in the processes and also removing the error when we found the cause of the space error. The relevant `R` scripts will be discussed below.

The second such error, as for the Regular Polygon LGCP, was the time error. This sub-directory contains the `R` script that inserts a 'TIME ERROR j' into the output of an SBC simulation study where the run took an excessively long time to run, therefore we can continue simulations without and return to this error combination of simulation, grid and mesh at the end and make use of more processors. For each 'TIME ERROR' run we have the code at the time of the error, before we included the error message, and the output after we included the error message as well as the data sets that resulted in the time error (so as to not need to re-simulate when re-running) and some text files that were output from the error insertion to ensure there were no mistakes in the error inclusion that resulted in a loss of data from the output. More discussions on the errors can be found in Appendix C of my thesis.

Note: for the naming conventions of the data in this sub-directory, the suffixes with all capitals indicate that this is the (re-labelled) output taken from Balena at the time of the error of interest and will therefore be taken as an input for one of the functions below. However, suffixes not in all capitals (i.e.`*_PostErrResetj.rda`) are usually the outputs from these functions, which are output as `GridMeshIrregPolLGCPSBCSSi.rda` and immediately returned to Balena to continue running the simulations with back-up versions

of these output files saved under file names with additional suffixes which will be discussed below.

- **Data**:

  → `GridMeshIrregPolLGCPSBCSSi_SPACEERRORj.rda`: these outputs (where j is empty or 2) were the results of the memory error and the errors were re-set with `IrregSBCErrReSetProcs_final.R` which outputs `GridMeshIrregPolLGCPSBCSSi.rda` which are returned to Balena to continue running. The j=2 distinguishes the space errors that were caught later during the simulation study run than the other processes, after the original space errors had been reset and were continuing to run.

  → `GridMeshIrregPolLGCPSBCSSi_REMERROR.rda`: these outputs were renamed from `GridMeshIrregPolLGCPSBCSSi.rda` outputs from Balena in order for `IrregSBCErrRemovalProcs_final.R` to remove the errors that were caused by the memory issue but not related to these processes in particular so they can be re-set and continue to run as normal. These files are taken in and GridMeshIrregPolLGCPSBCSSi.rda are output.

  → `GridMeshIrregPolLGCPSBCSSi_TIMEERRORj.rda`: these were the outputs where the runs took over 6 hours (or an unrealistically long time) without any result and therefore has 'TIME ERROR j' placed in the required simulation-grid-mesh error slot, with the first Time Error just denoted 'TIME ERROR'. For j=2, some errors were called `*_TIMEERROR2SKIP.rda` due to the grid-mesh combination post the TIME ERROR 2 addition being a much larger mesh resolution which could also have resulted in another time error and therefore, these errors were pre-emptively included so as not to risk waiting another 6 hours where the results could likely not be completed. The incrementing index j for the time errors were mostly due to tracking multiple time errors through the same processes and so needing to distinguish between the errors, especially for re-running. Additionally, as the time errors occurred at different times throughout the simulation study, in order to keep track it was easier to group the processes that had time errors around the same time together with the index j, and the next time processes had time errors we group them together and increment j. These outputs are taken in by `IrregSBCTimeErrInsertProcs_final.R` where `GridMeshIrregPolLGCPSBCSSi.rda` is output. (There are additional outputs for processes 5 and 14 where TIME ERROR 5 was incorrectly labelled as TIME ERROR 4, and this was also corrected in `IrregSBCTimeErrInsertProcs_final.R`).

  → `GridMeshIrregPolLGCPSBCSSi_FULL.rda`: the outputs from the time error re-runs, although maybe the re-run took over 12 hours (the time limit) and while others were manually stopped these runs had been missed. Therefore, for consistency we find such runs, manually re-label them with the suffix '_FULL' and re-set those particular outputs to be missing as though they were not retrieved initially. These data sets are the outputs containing these completed runs, before the output was removed, and they can also be found stored in the `GRID_MESH/IRREGULAR_POLYGON_LGCP/IRREGPOLLGCP_OUTPUT` directory as well but the code to reset the data is implemented in this sub-directory using the code in `TimeErrorsTimeChecks_final.R`.

  → `GridMeshIrregPolLGCPSBCSSi_PostTimeErrorRuns.rda`/`GridMeshIrregPolLGCPSBCSSi_-TIMEERRORFINAL.rda`: for i = 2 or 7, these were some initial checks for the re-runs to compare for the pre- and post- rerun outputs, especially since, initially for process 7 the time error re-run failed due to a memory issue, and so

this output saved with the original time error still in place, this was initially checked using `ComparingNewandOldProcs_final.R`. However, the time error for Process 7 was re-run successfully at a later date and so the final output that is merged with the remaining outputs, has the time error re-run and an error message moved into the warnings. The new text output from `ComparingNewandOldProcs_final.R` overwrote the text output for these initial checks.

- `R` **code**:
  - → `IrregSBCErrReSetProcs_final.R`: this code is to re-set the errors to NA in order to re-run and see which run caused the initial error. In some cases, both runs passed the original error position for simulation-grid-mesh and so were left to continue running. We would expect that this should not be required if the simulation study has no issue with the temporary directory.
  - → `IrregSBCErrRemovalProcs_final.R`: this code is to remove the errors (replace with NAs) for the runs that surpassed the previous simulation-grid-mesh settings when the original space errors occurred and therefore were not the cause of the errors. By removing these they can continue without errors that only occurred because the job failed due to the other process on the same node. We would expect that this should not be required if the simulation study has no issue with the temporary directory.
  - → `IrregSBCTimeErrInsertProcs_final.R`: this code allows us to insert TIME ERRORS in to the SBC outputs and re-save the outputs so that we can continue to run the simulation study so that the particular simulation-grid-mesh combinations that have time errors can be returned to in order to re-run with a higher number of cores available.
  - → `TimeErrorsTimeChecks_final.R`: if the results from the Time Error re-runs finished in over 12 hours, which may not have initially been caught in order to manually stop the runs, then the outputs were removed, to ensure consistency across all processes, where the 12 hour time limit was set for the additional re-runs. This `R` outputs a data frame which contains information on the particular processes that have over-run with respect to computation time and these are manually re-labelled as `GridMeshIrregPolLGCPSBCSSi_FULL.rda`. Then the remainder of this `R` script is run where we loads up any of the processes which completed in over 12 hours through the file name `GridMeshIrregPolLGCPSBCSSi_FULL.rda` in order to remove the relevant output and returns `GridMeshIrregPolLGCPSBCSSi.rda`.
  - → `ComparingNewandOldProcs_final.R`: this takes the outputs from the final pre-error re-run outputs and compares them to the outputs post-space-and-time-error re-runs to ensure there's is nothing drastically wrong with the results.
- **Outputs**: **Outputs from** `IrregSBCErrReSetProcs_final.R`:
  - → `GridMeshIrregPolLGCPSBCSSi.rda`: this output is copied over to Balena, with the error removed and ready to continue running the simulations. This will, of course, override any previous creation of this output file in this sub-directory.
  - → `GridMeshIrregPolLGCPSBCSSi_PostErrResetj.rda`: this denotes the back-up of the `GridMeshIrregPolLGCPSBCSSi.rda` output from `IrregSBCErrReSetProcs_final.R` when re-setting the errors due to lack of memory.
  - → `*.txt`: these are text files which contain the console output where we compare the old and new SBC simulation study outputs after the error message due to space issues was re-set.

Outputs from `IrregSBCErrRemovalProcs_final.R`:

→ `GridMeshIrregPolLGCPSBCSSi.rda`: this output is copied over to Balena, with the error removed and ready to continue running the simulations. This will, of course, override any previous creation of this output file in this sub-directory.

→ `GridMeshIrregPolLGCPSBCSSi_PostRemErr.rda`: this denotes the back-up of the `GridMeshIrregPolLGCPSBCSSi.rda` output from `IrregSBCErrRemovalProcs_final.R` when removing the errors due to lack of memory.

→ `*.txt`: these are text files which contain the console output where we compare the old and new SBC simulation study outputs after the error message due to space issues was removed.

Outputs from `IrregSBCTimeErrInsertProcs_final.R`:

→ `GridMeshIrregPolLGCPSBCSSi.rda`: this output is copied over to Balena, with the new error in place and ready to continue running the simulations. This will, of course, override any previous creation of this output file in this sub-directory.

→ `GridMeshIrregPolLGCPSBCSSi_PostTEj.rda`: this denotes back up of the `GridMeshIrregPolLGCPSBCSSi.rda` output from `IrregSBCTimeErrInsertProcs_final.R` when re-setting the errors due to time error j.

→ `*.txt`: these are text files which contain the console output where we compare the old and new SBC simulation study outputs after the error message 'TIME ERROR j' was included.

Outputs from `TimeErrorsTimeChecks_final.R`:

→ `longrunstimeerror.rda`: this contains the processes that, when re-run with 16 processors on a node, took over 12 hours to re-run, and so these `GridMeshIrregPolLGCPSBCSSi.rda` are manually re-labelled as `GridMeshIrregPolLGCPSBCSSi_FULL.rda` and the relevant time-consuming outputs are then removed.

→ `GridMeshIrregPolLGCPSBCSSi.rda`: for the outputs with too long a run-time in the re-runs and highlighted in `longrunstimeerror.rda` and saved as `GridMeshIrregPolLGCPSBCSSi_FULL.rda`. These are then loaded by `TimeErrorsTimeChecks_final.R` and then have the relevant simulation-grid-mesh outputs removed as though the run never completed. This will, of course, override any previous creation of this output file in this sub-directory. This is the final form of the SBC outputs for process i and will be moved into `IRREGPOLLGCP_OUTPUT` in order to be combined with the results from the other processes into one output file for the SBC simulation study.

→ `*.txt`: these are text files which contain the console output where we compare the old and new SBC simulation study outputs after the relevant outputs were removed.

Outputs from `ComparingNewandOldProcs_final.R`

→ `*.txt`: these are text files which contain the console output where we compare the original SBC outputs (before any error re-runs) and new SBC simulation study outputs after the relevant re-runs were completed.

For the `GridMeshIrregPolLGCPSBCSSi.rda`, the versions currently available in the directory, are the most recent outputs from one of the above `R` files, so they could be post final timer error insertion or error removal.

With respect to the `*.txt` files mentioned above, the individual `README*.txt` file for the directory `IRREGPOLLGCP_OUTPUT_ERROR` contains more information about the particular naming conventions for each of these text outputs.

In addition to the sub-directory `IRREGPOLLGCP_OUTPUT_ERROR` we have two further sub-directories.

- the sub-directory `TimeErrorData` contains individual sub-directories labelled `TIMEERRORj` where $j = 1, ...8$ and contain the data frames that resulted in the time error j. These were stored for the re-runs of the time errors to prevent the need to re-simulate these data sets.
- the `SBCCompletedPre-TimeErrorReruns` sub-directory contains the outputs from the SBC simulation study before any re-runs were completed, for either the time errors or space errors. In addition to the final outputs still present, we have the single combined SBC simulation study output from these pre-re-run outputs in `GridMeshIrregPolLGCPSBCSS.rda`.

For this SBC simulation study a flow diagram has been provided for the implementation of the main simulation study code and the necessary error `R` script runs in Figure 1.

2.3.3 `IRREGPOLLGCP_ANALYSIS`

This directory contains the code to produce the plots of the outputs results from the two simulation studies. There are R markdown files that run the `R` scripts with the input simulation study results. We additionally have a sub-directory, `ReLabelledPlots` which contains code to re-produce the plots from the results created in this directory with small changes to the plot titles or axis labels. The functions will be discussed below, after the description of the `R` scripts and outputs for this directory.

- `R` **code**:
  - → `GridMeshIrregPolLGCPTradOutput_final.R`, `GridMeshIrregPolLGCPTradbyGrid.Rmd` **and** `GridMeshIrregPolLGCPTradbyMesh.Rmd`: this `R` scripts contains the code that takes in the output from the traditional simulation study and produces the necessary plots, such as the average times for the `inla` runs or the parameter recovery, and tables to summarise the output and grouped by grid or mesh, respectively. The `Rmd` file sets up the `R` script and the LGCP over the regular polygon outputs to produce these plots. (Additionally, there were outputs for the WAIC/DIC however, these are not used within my thesis for my decision-making process.)
  - → `SBC_IrregPolLGCP_Param_final.R` **and** `GridMeshIrregPolLGCPSBCParamwFFT.Rmd`: this `R` script considers the results for the SBC simulation study for the parameters only in order to produce the relevant summaries, such as plots and tables. The histograms for the SBC ranks will be overlaid with the results of the GLMs for the rank frequencies, with the plots separated into two output pdfs depending on the whether there is a divergence from uniformity in the ranks or not. This R script will also present the errors and warnings information for the SBC simulation study, which is not done in the `R` script and markdown for the mean field results. As for the traditional simulation study results, the `Rmd` sets up the parameters SBC results as well as the necessary `R` script in order to produce the necessary summaries. The plots for the SBC rank results are grouped by grid or mesh except for the divergences or non-divergences from uniformity. The '*wFFT*' denotes the use of all of the output data, even if some runs contained more than 10 FFT warnings. (Additionally, there were

outputs for the WAIC/DIC however, these are not used within my thesis for my decision-making process.)

→ `SBC_IrregPolLGCP_Latent_final.R` **and** `GridMeshIrregPolLGCPSBCLatentwFFT.Rmd`: this `R` script considers the results for the SBC simulation study for the mean field only in order to produce the summaries such as the histogram plots for the SBC ranks. In particular, for each Grid the top 'worst' divergent histograms are plotted and output rather than all of the histogram plots as this is a very large number for each mesh resolution. The number of 'worst' histograms to plot and return are decided by the user in the `Rmd` file, which, as before, sets up the mean field data and runs the `R` script. The summary statistic plots are output grouped by mesh or grid, but the SBC divergences are output by grid only. The '*wFFT*' denotes the use of all of the output data, even if some runs contained more than 10 FFT warnings.

→ `IrregPolTrad_ExtraPlots_final.R`: this produces extra plots for the traditional simulation study, such as error plots for FFT warnings only.

→ `IrregPolSBCParam_ExtraPlots_final.R`: this produces extra plots for the parameter SBC simulation study results, such as individual plots for different grid or mesh resolutions rather than the plots being facetted by grid or mesh in a single figure.

→ `IrregPolSBCLatent_ExtraPlots_final.R`: this produces extra plots for the mean field SBC simulation study results, such as individual plots for different grid or mesh resolutions rather than being facetted by grid or mesh in a single figure.

→ `ParametersforErrors_final.R` **and** `IrregGridMeshErrorMeanFieldChecks_final.R`: these `R` scripts are meant to briefly investigate the parameters and behaviour of the fixed mean of the latent field for the data that resulted in erroneous outputs. The latter R script produces some plots to visualise the mean field and exponentiated mean field that resulted in data for which some errors occurred in our runs.

– **Outputs**:
Outputs from `GridMeshIrregPolLGCPTradOutput_final.R`, `GridMeshIrregPolLGCPTradbyGrid.Rmd` **and** `GridMeshIrregPolLGCPTradbyMesh.Rmd`

→ **GridMeshIrregPolLGCPTradbyGrid.html**: this is the output html file from the `R` markdown and will contain the printed tables and some plots that are also saved.

→ **GridMeshIrregPolLGCPTradbyMesh.html**: this is the output html file from the `R` markdown and will contain the printed tables and some plots that are also saved.

→ **IrregPolLGCPTrad*.pdf** and **IrregPolLGCPCoverageby*.pdf**: these are the plots that are produced and saved when we run the `R` markdown file, where 'by*' denotes the grouping in the plots either by the Grid resolution of Mesh resolution.

Outputs from `SBC_IrregPolLGCP_Param_final.R` **and** `GridMeshIrregPolLGCPSBCParamwFFT.Rmd`

→ **GridMeshIrregPolLGCPSBCParamwFFT.html**: this contains some of the plots that are saved as well as some of the summary outputs.

→ **IrregPolLGCPSBC*.pdf**, **irregpolsumdisttest_paramwErr_by*.pdf**, **irregpoloutsideboundstest_paramwErr_by*.pdf**, **irregpolsbcdivergences_paramwErr.pdf**, **irregpolsbcnondivergences_paramwErr.pdf**: these are the plots that are produced and saved when we run the `R` markdown file. The plots in **IrregPolLGCPSBC*.pdf** contain the general summaries for the simulation study while the others are the necessary plots for the SBC ranks: the summary statistics and the histograms. Note that the 'wErr' denotes the inclusion of the simulations which resulted in more than 10 FFT warnings (which are discussed in Chapter 3 and Appendix C of my thesis) rather than excluding them. Not all of the output plots have this included in the filenames, however, the same data is used across all of the analysis, where we do not exclude any of the runs with > 10 warnings. Also, 'by*' usually denotes whether the results are grouped by Grid or Mesh resolution.

Outputs from `SBC_IrregPolLGCP_Latent_final.R` **and** `GridMeshIrregPolLGCPSBCLatentwFFT.Rmd`

→ **GridMeshIrregPolLGCPSBCLatentwFFT.html**: this contains some of the plots that are saved.

→ **irregpolsumdisttestby*.pdf**, **irregpoloutsideboundstestby*.pdf**, **irregpolsbcdivergencesGrid*.pdf**: these are the plots that are produced and saved when we run the `R` markdown file. The summary statistics are given grouped by either Grid or Mesh denoted by *, and as well as these we produce a separate file for each Grid resolution, *, which contains the 'worst' divergent histograms for the results for that particular resolution.

Outputs from `IrregPolTrad_ExtraPlots_final.R`, `IrregPolSBCParam_ExtraPlots_final.R` **and** `IrregPolSBCLatent_ExtraPlots_final.R`

→ **IrregPolLGCPTradErrFFTOnlyMeanbyMesh.pdf**, **IrregPolLGCPTradErrFFTOnlyMeanbyGrid.pdf**, **irregpolsumdisttest*by*.pdf**, **irregpoloutsideboundstest*by*.pdf**: these are the plots that are produced in addition to the outputs from the `R` markdowns for both simulation studies. The indicator that these were the additional plots for the SBC are because the indexed by byMeshi or byGridi where $i = 1, 2, 3, 4$ where smaller i matches coarser grid or mesh resolutions.

Outputs from `GIrregGridMeshErrorMeanFieldChecks_final.R`

→ **IrregPolMeanFieldandExpMeanFieldforErr.pdf**, **IrregPolMeanFieldandExpMeanFieldforTimeErr.pdf**: these are the plots of the fixed mean and exponential of the fixed mean for the parameters where we had errors in our runs as well as for the parameters that resulted in Time Errors, which had been successfully re-run.

`ReLabelledPlots` **Sub-directory**

This contains additional `R` scripts and `R` markdowns that were the same files as in the `REGPOLLGCP_ANALYSIS` directory, but with most of the outputs commented out so that only a few of the output plots are saved and these have only minor changes - mostly for the axis labels.

– `R` **code**:

→ `GridMeshIrregPolLGCPTradOutput_Relabel.R`, `GridMeshIrregPolLGCPTradbyGrid_Relabel.Rmd` **and** `GridMeshIrregPolLGCPTradbyMesh_Relabel.Rmd`:

same as before, without the parameter recovery, credible interval coverage, error or DIC/WAIC plots output, only saving the re-plotted average time with '(s)' to denotes the time unit on the y-axis label.

→ `SBC_IrregPolLGCP_Param_Relabel.R` **and** `GridMeshIrregPolLGCPSBCParam_Relabel.Rmd`: same as before with label changes for the summary statistics, and commenting out any plots for errors and DIC/WAIC for the SBC simulations.

→ `SBC_IrregPolLGCP_Latent_Relabel.R` **and** `GridMeshIrregPolLGCPSBCLatent_Relabel.Rmd`: same as before, although as with the parameter SBC analysis we have changed the y-axis labels for the summary statistics.

→ `IrregPolTrad_ExtraPlots_Relabel.R`: this reproduces the additional plots with relabels as necessary.

→ `IrregPolSBCParam_ExtraPlots_Relabel.R`: this reproduces the additional summary statistic plots with relabels as necessary.

→ `IrregPolSBCLatent_ExtraPlots_Relabel.R`: this reproduces the additional summary statistic plots with relabels as necessary.

– **Outputs**:
Outputs from `GridMeshIrregPolLGCPTradOutput_Relabel.R`, `GridMeshIrregPolLGCPTradbyGrid_Relabel.Rmd` **and** `GridMeshIrregPolLGCPTradbyMesh_Relabel.Rmd`

→ **GridMeshIrregPolLGCPTradbyGrid_Relabel.html**: this is the output html file from the `R` markdown and will contain the printed tables and some plots that are also saved.

→ **GridMeshIrregPolLGCPTradbyMesh_Relabel.html**: this is the output html file from the `R` markdown and will contain the printed tables and some plots that are also saved.

→ **IrregPolLGCPTradTimev*facet**_Relabel.pdf,IrregPolLGCPTradTimeby*_Relabel.pdf**: this is the only plot that is re-produced and saved when we run the `R` markdown file. Where * and ** are one of Grid or Mesh, but *≠**.

Outputs from `SBC_IrregPolLGCP_Param_Relabel.R` **and** `GridMeshIrregPolLGCPSBCParam_Relabel.Rmd`

→ **GridMeshIrregPolLGCPSBCParam_Relabel.html**: this contains some of the plots that are saved as well as some of the summary outputs.

→ **irregpolsumdisttest_paramwErr_by*_Relabel.pdf**, **irregpoloutsideboundstest_paramwErr_by*_Relabel.pdf**: the summary statistics are only re-produced and saved plots when we run the `R` markdown file. * is either Mesh or Grid.

Outputs from `SBC_IrregPol_Latent_Relabel.R` **and** `GridMeshIrregPolLGCPSBCLatent_Relabel.Rmd`

→ **GridMeshIrregPolLGCPSBCLatent_Relabel.html**: this contains some of the plots that are saved.

→ **irregpolsumdisttestby*_Relabel.pdf**, **irregpoloutsideboundstestby*_Relabel.pdf**: these are the re-produced plots that are saved when we run the `R` markdown file. * is either Mesh or Grid.

Outputs from `IrregPolTrad_ExtraPlots_Relabel.R`, `IrregPolSBCParam_ExtraPlots_Relabel.R` **and** `IrregPolSBCLatent_ExtraPlots_Relabel.R`

→ **IrregPolLGCPTradErrFFTOnlyMeanbyMesh_Relabel.pdf**, **IrregPolLGCPTradErrFFTOnlyMeanbyGrid_Relabel.pdf**, **irregpolsumdisttest*by*_Relabel.pdf**, **irregpoloutsideboundstest*by*_Relabel.pdf**: these are the plots that are produced in addition to the outputs from the `R` markdowns for both simulation studies. The indicator that these were the additional plots for the SBC are because the indexed by byMeshi or byGridi where $i = 1, 2, 3, 4$ where smaller i matches coarser grid or mesh resolutions.

Return to Table of Contents

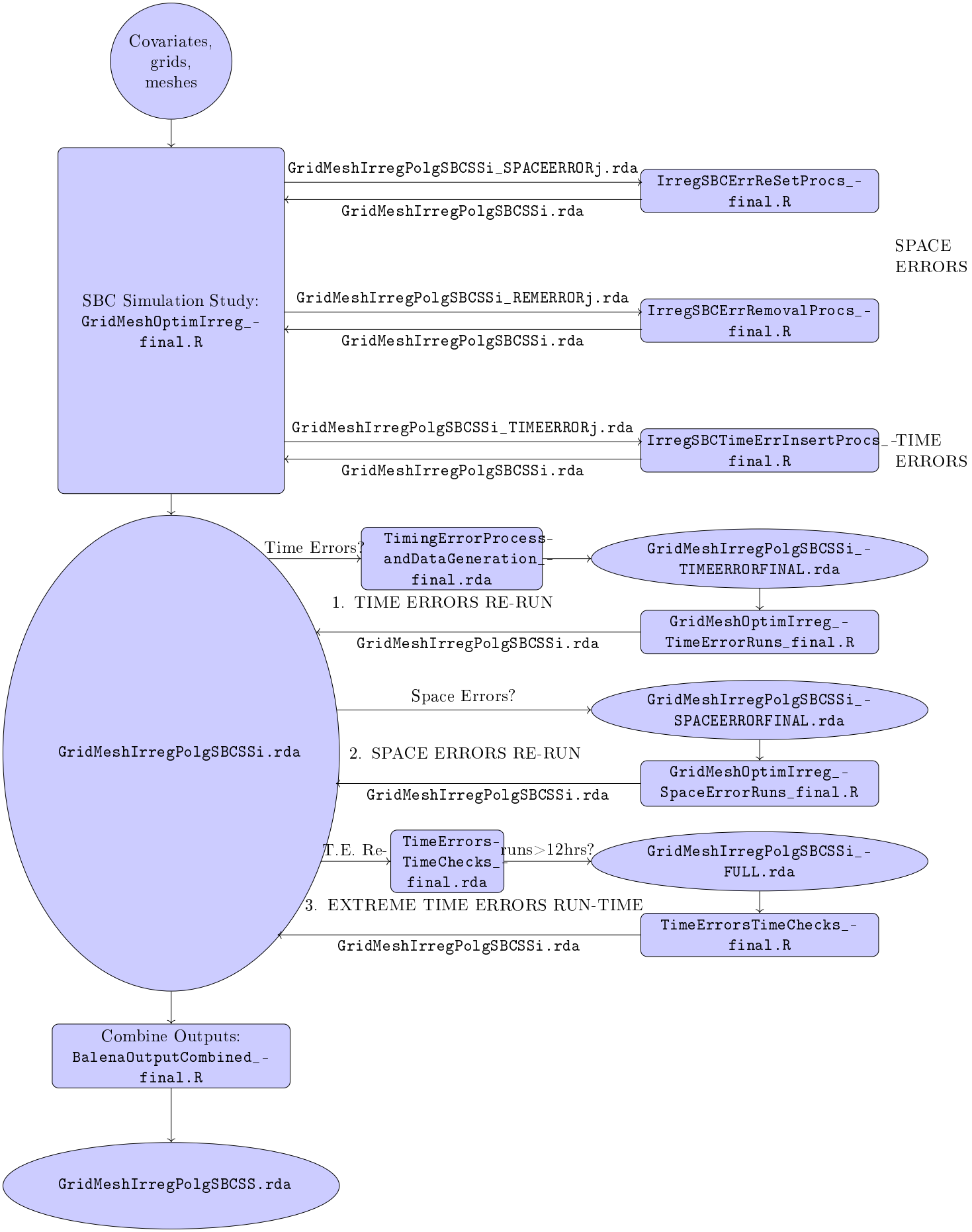

Figure 1: Flow Diagram of the Implementation of the SBC Simulation Study for the Los Angeles Polygon.

3 INLA_w_MCMC

This directory contains the code with functions for the Univariate and Multivariate INLA within MCMC algorithms as well as their implementation to the Los Angeles and New York motor vehicle thefts data. Additionally, we have the code to implement the simulation study over the two regular polygon windows as well as some code to perform quick test fr the timing of the Multivariate INLA within MCMC algorithm if we were to consider three cities (Los Angeles, New York and Portland), instead of just two cities (Los Angeles and New York).

3.1 IwMFUNCTIONS

This directory contains two `R` scripts for the implementation of the Univariate and Multivariate INLA within MCMC algorithms, respectively.

- `R` **code**:
  - → `lgcp_inla_w_mcmc_functions_final.R`: this contains the necessary sub-functions and main functions for the Metropolis-Hastings and BMA step of the Univariate INLA within MCMC algorithm.
  - → `lgcp_inla_w_mcmc_multivar_functions_final.R`: this contains the necessary sub-functions and main functions for the Metropolis-Hastings and BMA step of the Multivariate INLA within MCMC algorithm. This contains sub-functions that allows us to share data between study regions to just estimate the covariate effects, without estimating the covariate contrasts, as well as including the covariate contrasts for 'non-base' study regions. The BMA step also can combine the covariate effect samples from the MH step with the conditional posterior marginals in order to approximate the posterior marginals for the total covariate effects in the 'non-base' study regions, with details about this in Chapter 5 of my thesis.

3.2 USCITIESIwM

This directory contains the required gridded Los Angeles and New York data as well as the necessary mesh resolutions over both city polygons and the code for the implementation of the INLA, Univariate and Multivariate INLA within MCMC algorithms for these cities. This directory also contains the outputs from the implementations of these algorithms as well as an `R` script for plotting the US crime data at the resolutions for the modelling and the plots produced from this `R` script. I also include the slurm files to run the necessary R scripts and combinations, as `job*_final.slm`.

- **Data**:
  - → `LA2015CT4872CountData_projFinalScale.rda`: this is the aggregated count data for Los Angeles City crime, homicides and motor vehicle thefts, in 2015 at the 1km-by-1km grid resolution. Generated in `DATA/PROCESSED_DATA/CRIME/COUNT_DATA_FINAL`. [AGGREGATE DATA]
  - → `LAMesh4872_projFinalScale.rda`: this is the mesh over the Los Angeles City polygon, with a maximum mesh edge of 1km. Generated in `DATA/PROCESSED_DATA/CRIME/COUNT_DATA_FINAL`. [AGGREGATE DATA]
  - → `NY2015CT4748CountData_projFinalScale.rda`: this is the aggregated count data for New York City crime, homicides and motor vehicle thefts, in 2015 at the 1km-by-1km grid resolution. Generated in `DATA/PROCESSED_DATA/CRIME/COUNT_DATA_FINAL`. [AGGREGATE DATA]

→ `NYMesh4748_projFinalScale.rda`: this is the mesh over the New York City polygon, with a maximum mesh edge of 1km. Generated in `DATA/PROCESSED_DATA/CRIME/COUNT_DATA_FINAL`. [AGGREGATE DATA]

– **`R` code**:

→ `MultiCityMultiIwM_INLARuns_final.R`: this `R` script contains the code to implement the INLA algorithm to model the Los Angeles and New York motor vehicle theft count data individually.

→ `MultiCityUniIwM_final.R`: this implements the Univariate INLA within MCMC algorithm for the Los Angeles motor vehicle theft count data. First, the MH runs are implemented, with re-starts possible and then once these are complete the BMA step can be completed with options for setting the burn-in and thinning variables before the BMA is implemented.

→ `MultiCityMultiIwM_final.R`: this implements the Multivariate INLA within MCMC algorithm for the Los Angeles and New York Motor Vehicle Theft count data. As with the Univariate IwM, the Metropolis-Hastings iterations are first completed and then the BMA step is implemented separately. There are options with which we can tell the function to also approximate the covariate contrasts between study regions and we can define which study region is the 'base' study region.

→ `PlottingUSCitiesData_final.R`: this considers the Los Angeles and New York data at the resolutions of data aggregation and mesh of interest for the INLA, Univariate and Multivariate INLA within MCMC implementations and produces some simple plots of the grids, meshes, variables and crime for each city individually and for both cities combined.

– **Outputs**:
Outputs from `MultiCityMultiIwM_INLARuns_final.R`:

→ `*INLA.rda`: this denotes the INLA-only runs.

Outputs from `MultiCityUniIwM_final.R`:

→ `MultiCityUniIwM_MH_*.rda`: this denotes the output from the Metropolis-Hastings step only, where the number of runs is included in the filename as the suffix.

→ `MultiCityUniIwM_BMA_*_*.rda`: this denotes the output from the BMA implementation, the output includes the MCMC chains with the removed burn-in iterations as well as the INLA marginals for the remaining iterations and the final approximated posterior marginals for the fixed, hyperpar (internal rep., transformed internal rep. and INLA output external rep.). The total iterations of the original MH run as well as the number of iterations taken as a burn-in are given in the filename, as the suffix, so that we have `*BMA_MHiterations_-burnin.rda`.

Outputs from `MultiCityMultiIwM_final.R`:

→ `MultiCityMultiIwM_MH_*.rda`: this denotes the output from the Metropolis-Hastings step only, where the number of runs is included in the filename as the suffix.

→ `MultiCityMultiIwM_BMA_*_*.rda`: this denotes the output from the BMA implementation, the output includes the MCMC chains with the removed burn-in iterations as well as the INLA marginals for the remaining iterations and the

final approximated posterior marginals for the fixed, hyperpar (internal rep., transformed internal rep. and INLA output external rep.) as well as the combination of the MH samples with the marginals for its respective fixed effect (only for the 'non-base' study region). The total iterations of the original MH run as well as the number of iterations taken as a burn-in are given in the filename, as the suffix, so that we have `*BMA_MHiterations_burnin.rda`.

Outputs from `PlottingUSCitiesData_final.R`:

- ***.pdf**: these are the plots of the data, output from the above plotting script.

3.3 `IwMSIMSTUDY`

This directory contains the code for the data simulation, plotting and implementation of the INLA and Multivariate INLA within MCMC for the simulation study over the two regular polygon study regions as discussed in Chapter 5 of my thesis. The goal of the simulation study was to try to understand the behaviour of the Multivariate INLA within MCMC algorithm in different scenarios across the two windows.

- `R` **code**:
  - → `SimulationStudySetUp_final.R`: this is the code used to simulate the data (covariates and aggregated point patterns) and produce the required meshes for two study regions. The data is simulated using particular seeds and should, therefore, be able to reproduce the same data.
  - → `LAPortlandGMDataModelComp.txt`: this is some code and output which loaded old LA and Portland data and compared the total counts to consider a reasonable thinning percentage for the sparse data simulation. We also produced some quick models for the count data to assess the coefficients for the data. Note that the data used for these were the `DATA/PROCESSED_CRIME/CRIME/COUNT_DATA_GMO` LA data with the Portland data produced in the same way, with the average income different than those in the `DATA/PROCESSED_CRIME/CRIME/COUNT_DATA_FINAL` data sets. However, the important result from this was the proportion of counts which is a the same regardless of how the socio-economic variables are interpolated.
  - → `SimulationStudySetUpPlots_final.R`: this is the code used to produce plots for the data.
  - → `SimulationStudy_Balena_INLARuns_final.R`: this implements the INLA-only runs for the five data sets across the two study regions.
  - → `SimulationStudy_Balena_final.R`: this contains the code that runs the Multivariate INLA within MCMC algorithms to the combination of the count data in Window 1 with different count data sets in Window 2. It utilises if statements that have to be pre-selected to implement the algorithm for the correct combination of data sets. It also contains if-statements that allow for the implementation of the Univariate INLA within MCMC algorithm to the separate data sets although this was not implemented.
- **Outputs**:
  Outputs from `SimulationStudySetUp_final.R`
  - → `WindowiCovariates.rda`: the simulated covariates over each window for the simulation of the point patterns and aggregated count data, where $i = 1, 2$.

→ `Windowi*Data.rda`: simulated data, aggregated into count data for Window $i = 1, 2$ and if $i = 1$, then *=Full (Data Set 1), and if $i = 2$, then *=Full (Data Set 2 for Scenario 1), Sparse (Data Set 3 for Scenario 2), DifferentBeta2 (Data Set 4 for Scenario 3) and DifferentSignBeta2 (Data Set 5 for Scenario 4).

→ `MeshWindowi.rda`: meshes over the two windows, $i = 1, 2$.

Outputs from `SimulationStudySetUpPlots_final.R`:

→ ***.pdf**: these are the plots of the data, output from the above plotting script.

Outputs from `SimulationStudy_Balena_INLARuns_final.R`:

→ **Windowi*INLA.rda**: this denotes the INLA-only runs for both windows, $i = 1, 2$ where if $i = 1$, then *=Full (Data Set 1), and if $i = 2$, then *=Full (Data Set 2 for Scenario 1), Sparse (Data Set 3 for Scenario 2), Cov (Data Set 4 for Scenario 3) and DSCov (Data Set 5 for Scenario 4).

Outputs from `SimulationStudy_Balena_final.R`:

→ `IwMMultivarMHFullby*.rda`: this denotes the output from the Metropolis-Hastings step only, for the four different scenarios, where the * specifies the scenario (*=Full is Scenario 1, *=Sparse is Scenario 2,*=Cov is Scenario 3, *=DifferentSignCov is Scenario 4). All of these implementations were run for $10,000$ iterations.

→ `*IwM_BMA_*_*.rda`: this denotes the output from the BMA implementation, the output includes the MCMC chains with the removed burn-in iterations (500 for all) as well as the INLA marginals for the remaining iterations and the final approximated posterior marginals for the fixed, hyperpar (internal rep., transformed internal rep. and INLA output external rep.). Additionally for Scenarios 3 and 4 where we also estimated the covariate contrasts, we also output the combination of the MH samples with the marginals for its respective fixed effect (only for the non-base study region). The * specifies the scenario considered, where *=Full (Scenario 1), Sparse (Scenario 2), CovComb (Scenario 3), DifferentSignCovComb (Scenario 4), where the addition of Comb for Scenarios 3 and 4 is due to the additional approximation of the combinations of total covariate effects for the 'non-base' study regions - which is detailed in Chapter 5 of my thesis.

3.4 `IwMOUTPUTSUMMARY`

This directory contains an `R` script which has functions for creating tables and plots of the results from the Univariate and Multivariate INLA within MCMC algorithms runs for the US Cities and simulation study, where we can also consider the INLA results for the simulation study to compare to the Multivariate INLA within MCMC algorithm results. The outputs for these are then placed in one of two sub-directories, one for the INLA results and the other for the INLA within MCMC results. In addition to the summary functions, there are three `R` scripts which run the necessary summary functions to produce the required plots and tables. Two of these `R` scripts consider the INLA-only results for the two US Cities, while the other produces the outputs for the Univariate and Multivariate INLA within MCMC algorithm for the simulation studies and US Cities.

– **`R` code**:

→ `IwMSummary_Functions_final.R`: this contains the functions that produce tables and plots for the outputs from the Univariate and Multivariate INLA within

MCMC results. It creates tables for the results of the MCMC and BMA steps separately, which can be combined post-creation, as well as plots for the MH step such as trace plots and histograms of the samples where (for the latter) the INLA-only posterior approximations can be overlaid as well as the priors for the parameters. For the BMA step we can plot the approximate posteriors and also overlay the INLA-only approximations and priors. We can also overlay the true values of parameters (if they are known). Additionally, the plots of the credible intervals can be produced.

→ `IwMSummary_final.R`: this uses the functions in `IwMSummary_Functions_final.R` to generate summary tables for the simulation study outputs and plots where we overlay the true parameter values where possible. In addition to this, we also produce summary tables for the Univariate and Multivariate INLA within MCMC output for Los Angeles and New York, where we have a combined results table for all of the results from the INLA-only implementation and both INLA within MCMC implementations. As for the simulation study we can produce histogram plots for the samples with the INLA-only approximations and priors overlaid. Additionally we produce the plots for the BMA outputs for each city. Output plots and tables are stored in the `IwM_OUTPUTS` sub-directory and further separated in to the two subdirectories `IwM_OUTPUTS/REGPOLSIMSTUDY` and `IwM_OUTPUTS/USCITIES`.

→ `LA2015MVTINLASummary_final.R`: this `R` script contains the code to produce the output summaries for the INLA-only run for the 2015 Motor Vehicle Theft aggregated data in Los Angeles. Output plots and tables are stored in the `INLA_OUTPUTS` sub-directory.

→ `NYC2015MVTINLASummary_final.R`: this `R` script contains the code to produce the output summaries for the INLA-only run for the 2015 Motor Vehicle Theft aggregated data in New York. Output plots and tables are stored in the `INLA_OUTPUTS` sub-directory.

– **Outputs**:
Outputs from `IwMSummary_final.R`: these are stored in `IwM_OUTPUTS/REGPOLSIMSTUDY` for the simulation study and `IwM_OUTPUTS/USCITIES` for the US Cities results. We will list the outputs here and then more details can be found below in Section 3.4.1.

→ `IwM_OUTPUTS/REGPOLSIMSTUDY/`**RegularPolygonSimStudy*.pdf**

→ `IwM_OUTPUTS/REGPOLSIMSTUDY/RegularPolygonSimStudy*TableDF.rda`

→ `IwM_OUTPUTS/REGPOLSIMSTUDY/RegularPolygonSimStudy*TableTex.rda`

→ `IwM_OUTPUTS/REGPOLSIMSTUDY/RegularPolygonSimStudy*TableTex.tex`

→ `IwM_OUTPUTS/USCITIES/`**UnivarIwMLA*.pdf**

→ `IwM_OUTPUTS/USCITIES/UnivarIwMLATableDF.rda`

→ `IwM_OUTPUTS/USCITIES/UnivarIwMLATableTex.rda`

→ `IwM_OUTPUTS/USCITIES/UnivarIwMLATableTex.tex`

→ `IwM_OUTPUTS/USCITIES/`**MultivarIwMLANY*.pdf**

→ `IwM_OUTPUTS/USCITIES/MultivarIwMLANYTableDF.rda`

→ `IwM_OUTPUTS/USCITIES/MultivarIwMLANYTableTex*.rda`

→ `IwM_OUTPUTS/USCITIES/MultivarIwMLANYTableTex*.tex`

→ `IwM_OUTPUTS/USCITIES/AllIwMLANYTableDF.rda`

→ `IwM_OUTPUTS/USCITIES/AllIwMLANYTableTex*.rda`

→ `IwM_OUTPUTS/USCITIES/AllIwMLANYTableTex*.tex`

Outputs from `LA2015MVTINLASummary_final_final.R`: these are stored in `INLA_OUTPUTS/USCITIES`. We will list the outputs here and then more details can be found below in Section 3.4.2.

→ `INLA_OUTPUTS/USCITIES/`**LA2015INLAMVT*.pdf**

→ `INLA_OUTPUTS/USCITIES/LAMVTSummaryTable.tex`

Outputs from `NYC2015MVTINLASummary_final_final.R`: these are stored in `INLA_OUTPUTS/USCITIES`. We will list the outputs here and then more details can be found below in Section 3.4.2.

→ `INLA_OUTPUTS/USCITIES/`**NYC2015INLAMVT*.pdf**

→ `INLA_OUTPUTS/USCITIES/NYCMVTSummaryTable.tex`

3.4.1 `IwM_OUTPUTS`

This directory contains two further sub-directories for the results and plots of the Univariate and Multivariate INLA within MCMC algorithm to either the Regular Polygon Simulation Study (`IwM_OUTPUTS/REGPOLSIMSTUDY`) or the US cities crime data (`IwM_OUTPUTS/USCITIES`).

`REGPOLSIMSTUDY` **Sub-directory** This contains any plots or tables produced from the summary functions applied to the Multivariate INLA within MCMC algorithm results for the Regular Polygon Simulation Study.

– **Data**

→ **RegularPolygonSimStudy*.pdf**: plots for the different scenarios' results. Within the names of the plots we specify the following scenarios:

* 'Full' or 'FullFull': Scenario 1;
* 'Sparse' or 'FullSparse': Scenario 2;
* 'Cov' or 'FullCov': Scenario 3;
* 'DSCov' or 'FullDSCov': Scenario 4.

→ `RegularPolygonSimStudy*TableDF.rda`: this contains a list of length two containing the data frame of the MH results and BMA results for a particular scenario (as well as the INLA-only results) from which we generate the results tables to place in the .tex file. The naming conventions are as above.

→ `RegularPolygonSimStudy*TableTex.rda`: this contains the above data frames combines into a single data frame, and turned into an object of class `xtable`. The naming conventions are as above. Suffix of '_3sfTake2' involves three significant figures for the values in the table.

→ `RegularPolygonSimStudy*TableTex.tex`: this is the .tex file for the above `xtable` of results that can be simply copied into any target .tex file. The naming conventions are as above. Suffix of '_3sfTake2' involves three significant figures for the values in the table.

`USCITIES` **Sub-directory** This contains any plots or tables produced from the summary functions applied to the Univariate and Multivariate INLA within MCMC algorithm results for the Los Angeles and New York crime data.

– **Data**

→ **UnivarIwMLA*.pdf**: plots for the results from the Univariate INLA within MCMC algorithm for the Los Angeles motor vehicle theft data.

→ `UnivarIwMLATableDF.rda`: this contains a list of length two containing the data frame of the MH results and BMA results for the results from the Univariate INLA within MCMC algorithm for the Los Angeles motor vehicle theft data (as well as the INLA-only results) from which we generate the results tables to place in the .tex file.

→ `UnivarIwMLATableTex.rda`: this contains the above data frames combines into a single data frame, and turned into an object of class `xtable`. Suffix of '_3sfTake2' involves three significant figures for the values in the table.

→ `UnivarIwMLATableTex.tex`: this is the .tex file for the above `xtable` of results that can be simply copied into any target .tex file. Suffix of '_3sfTake2' involves three significant figures for the values in the table.

→ **MultivarIwMLANY*.pdf**: plots for the results from the Multivariate INLA within MCMC algorithm for the Los Angeles and New York motor vehicle theft data.

→ `MultivarIwMLANYTableDF.rda`: this contains a list of length two containing the data frame of the MH results and BMA results for the results from the Multivariate INLA within MCMC algorithm for the Los Angeles and New York motor vehicle theft data (as well as the INLA-only results) from which we generate the results tables to place in the .tex file.

→ `MultivarIwMLANYTableTex*.rda`: this contains the above data frames combines into a single data frame, and turned into an object of class `xtable`. Suffix of '_3sfTake2' involves three significant figures for the values in the table.

→ `MultivarIwMLANYTableTex*.tex`: this is the .tex file for the above `xtable` of results that can be simply copied into any target .tex file. Suffix of '_3sfTake2' involves three significant figures for the values in the table.

→ `AllIwMLANYTableDF.rda`: this contains a list of length two containing the data frame of the MH results and BMA results for the results from the Multivariate INLA within MCMC algorithm for the Los Angeles and New York motor vehicle theft data and also the results for the Univariate INLA within MCMC algorithm for the Los Angeles data (as well as the INLA-only results) from which we generate the results tables to place in the .tex file.

→ `AllIwMLANYTableTex*.rda`: this contains the above data frames combines into a single data frame, and turned into an object of class `xtable`. Suffix of '_3sfTake2' involves three significant figures for the values in the table.

→ `AllIwMLANYTableTex*.tex`: this is the .tex file for the above `xtable` of results that can be simply copied into any target .tex file. Suffix of '_3sfTake2' involves three significant figures for the values in the table.

3.4.2 `INLA_OUTPUTS`

This directory contains a further sub-directory for the results and plots of the implementation of the INLA algorithm to the US cities crime data (`INLA_OUTPUTS/USCITIES`).

`USCITIES` **Sub-directory** This contains any plots or tables produced from the summary functions applied to the INLA algorithm results for the Los Angeles and New York crime data.

– **Data**

→ **LA2015INLAMVT*.pdf**: these are the plots of Los Angeles INLA-only results for the motor vehicle theft data.

→ `LAMVTSummaryTable.tex`: table of summary of results for Los Angeles INLA-only implementation for the motor vehicle theft data.

→ **NYC2015INLAMVT*.pdf**: these are the plots of New York INLA-only results for the motor vehicle theft data.

→ `NYCMVTSummaryTable.tex`: table of summary of results for New York INLA-only implementation for the motor vehicle theft data.

3.5 `IwMMULTICITYTIMINGTEST`

This directory contains a slightly altered collection of functions for implementing the Multivariate INLA within MCMC algorithm, which allows us to test the timing of the algorithm where we vary the number of cores for the `R` function `mclapply` depending on the number of study regions. Note that when this was implemented to produce the timing and output results highlighted below, there was a slight discrepancy between the Portland count data frame at the 1km resolution, namely for the income. This data set differs from the final count data frame produced in `DATA/PROCESSED_DATA/CRIME/COUNT_DATA_FINAL` only for the income variable, where 15 cells have slightly smaller values. This is due to the use of `P_CTInc_15_imp_proj.rds` rather than `P_CTInc_15_0imp_proj.rds` in the interpolation to the grid cells where the latter assigns zeros to any of the missing data values where the estimated number of households is zero and the former imputes them. For LA the former was used for the Grid-Mesh Optimisation while the income at the census tract level of the form `*_CTInc_15_0imp_proj.rds` was used for all final crime count data sets. `README_-IwMMULTICTYTIMINGTEST.txt` contains more details on the actual differences between the count data frame used for the timing test and the final count data frame generated for Portland. However, importantly, the dimension of the count data frame and the meshes all remain the same and so the timing results should not be affected by this difference in the one of the variables.

– **Data**:

→ `*2015CT**CountData_projFinalScale.rda`: the aggregated count data for each city, *, where the resolution of the grid is roughly 1km, with the grid dimensions noted in **. Note that `P2015CT3826CountData_projFinalScale.rda` will, as mentioned above, will differ from the correct final count data for Portland found in `DATA/PROCESSED_DATA/CRIME/COUNT_DATA_FINAL/PORTLAND`.

→ `*Mesh**_projFinalScale.rda`: the mesh over the different city, *, windows with a maximum mesh edge of roughly 1km, with the matching grid dimensions given as, **.

– `R` **code**:

→ `lgcp_inla_w_mcmc_multivar_functions_wtiming_final.R`: this contains the necessary sub-functions and main functions for the Metropolis-Hastings and BMA step of the Multivariate INLA within MCMC algorithm. This contains sub-functions that allows us to share data between study regions to just estimate the covariate effects, without estimating the covariate contrasts, as well as including the covariate contrasts for 'non-base' study regions. The BMA step also can combine the covariate effect samples from the MH step with the conditional posterior marginals in order to approximate the posterior marginals for the total covariate effects in the 'non-base' study regions, with details about this in Chapter 5 of my thesis. Importantly for these functions, we test the timings

for the code with three cities, where the value of `mc.cores` for the parallelisation of a function call (in this case it is fitting the separate conditional INLA models for each city) used within the log posterior calculation varies depending on the number of cities. There is additional commented code where the value of `mc.cores` is fixed to two which can be used to compare the timings when we fix the number of cores or allow it to vary with the number of study regions.

→ `MultiCityTimingTest_Balena_final.R`: this contains the code that implements the two scenarios: running the Multivariate INLA within MCMC algorithm for two cities (Los Angeles and New York) or for three cities (Los Angeles, New York and Portland).

– **Outputs**:

→ `MultiCityTimings*Cities.rda`: this contains the output file for the implementation for *=2 or 3 cities where the MH step is run for 10 iterations, with the samples of the covariate effects and marginals are stored for each city in the object `out` and the timing of the calculation of the log-posterior for each iteration.

→ `TimeTaken*Cities10Iterations.rda`: this contains the total time taken for call to the MH function to complete with a total of 10 iterations, calculated using the `proc.time` function.

Return to Table of Contents

4 EXTRA

This directory contains any spare `R` scripts that do not necessarily belong in any of the other directories. In this case we have the `R` scripts to generate the SBC histogram plots for Chapters 2 and 3 as well as the code that implements the INLA within MCMC algorithm for the example from the Gomez-Rubio and Rue paper to illustrate the method for the BMA step of the INLA within MCMC algorithm, with the plots found in Appendix B.

– **`R` code**:

→ `Ex51Simulation_IwMBMAPlots_final.R`: this is the code that runs the INLA within MCMC algorithm for Example 5.1 from the paper "Markov chain Monte Carlo with the Integrated Nested Laplace Approximation" by Gomez-Rubio and Rue (2018) and also contains the code to generate the plots in Appendix B to illustrate the behaviour of the BMA step of the INLA within MCMC algorithm.

→ `SBCPlots_final.R`: this contains the code to generate the SBC plots in Chapters 2 and 3, which use simulations and samples from the a Normal distribution with varying means and meshes to illustrate the behaviour or divergences (Chapter 2) as well as modelling the rank frequencies and plotting the results accordingly (Chapter 3).

– **Outputs**:
Outputs from `Ex51Simulation_IwMBMAPlots_final.R`

→ **ExBMAalt_*.pdf**: these are plots for different steps of the BMA function implementation as well as the final output.

Outputs from `SBCPlots_final.R`

→ ***DistSBC.pdf**: these are histogram plots for different scenarios, either using the same distribution for simulating and sampling and also altering the mean or variance between the simulating and sampling distributions.

→ **N*N*SBC.pdf**: these are the above plots with the distributions highlighted in the title.

→ ***DistModelSBC.pdf**: these are the same histograms as above with the results from the Poisson GLMs overlaid.

Return to Table of Contents

Part III
Directories and Thesis Chapters

In this section I want to quickly highlight the links between the directories and results for my thesis by chapter, with particular details about the `R` code and related inputs and outputs to be found in the text in Part II or in the `README_*.txt` files in the specified directories. These are also laid out in Table 1 in Section 11.

5 Chapter 1 and Appendix A

- Data Plots: `DATA/EDA`
- Generalised Linear Models Plots and Results: `DATA/MODELS/GLMS`
- Ripley's K: `DATA/MODELS/GLMS`

6 Chapter 2 and Appendix B

- LA Grids: `GRID_MESH/IRREGULAR_POLYGON_LGCP/IRREGPOLLGCP_CODE/IRREGPOLLGCP_CODE_TESTINGVALUES`
- Gaussian Example Regular Polygon Mesh: `GRID_MESH/GAUSSIAN/GAUSSIAN_CODE`
- LA Mesh: `GRID_MESH/IRREGULAR_POLYGON_LGCP/IRREGPOLLGCP_CODE/IRREGPOLLGCP_CODE_TESTINGVALUES`
- SBC Histograms: `EXTRA`
- BMA Plots: `EXTRA`

7 Chapter 3 and Appendix C

- SBC Histograms with Models Plots and Results: `EXTRA`
- Gaussian Simulation Study Code: `GRID_MESH/GAUSSIAN/GAUSSIAN_CODE`
- Gaussian Example Meshes: `GRID_MESH/GAUSSIAN/GAUSSIAN_CODE`
- Gaussian Example Simulation Study Plots and Results: `GRID_MESH/GAUSSIAN/GAUSSIAN_ANALYSIS`
- LGCP Example Simulation Study Code: `GRID_MESH/REGULAR_POLYGON/REGPOLLGCP_CODE`
- LGCP Example Simulation Study Plots and Results: `GRID_MESH/REGULAR_POLYGON_LGCP/REGPOLLGCP_ANALYSIS`

8 Chapter 4 and Appendix D

- Los Angeles Grid, Mesh, Covariates and Simulated Data plots: `GRID_MESH/IRREGULAR_POLYGON_LGCP/IRREGPOLLGCP_CODE/IRREGPOLLGCP_CODE_TESTINGVALUES`
- Irregular Polygon LGCP Simulation Study Code: `GRID_MESH/IRREGULAR_POLYGON_LGCP/IRREGPOLLGCP_CODE`

- Irregular Polygon LGCP Simulation Study Plots and Results: GRID_MESH/IRREGULAR_POLYGON_LGCP/IRREGPOLLGCP_ANALYSIS
- LA Census Tract Areas Histogram: DATA/EDA
- LA Motor Vehicle Theft INLA Code and Output: INLA_w_MCMC/USCITIESIwM
- LA Motor Vehicle Theft INLA Plots and Results: INLA_w_MCMC/IwMOUTPUTSUMMARY and INLA_w_MCMC/IwMOUTPUTSUMMARY/INLA_OUTPUTS/USCITIES

9 Chapter 5 and Appendix E

- Univariate and Multivariate INLA within MCMC Code: INLA_w_MCMC/IwMFUNCTIONS
- US Cities: Data Plots: DATA/EDA and INLA_w_MCMC/USCitiesIwM
- US Cities: INLA, Univariate and Multivariate INLA within MCMC Code and Outputs: INLA_w_MCMC/USCitiesIwM
- US Cities: INLA, Univariate and Multivariate INLA within MCMC Plots and Results: INLA_w_MCMC/IwMOUTPUTSUMMARY (for code to create the results) and INLA_w_MCMC/IwMOUTPUTSUMMARY/IwM_OUTPUTS/USCITIES and INLA_w_MCMC/IwMOUTPUTSUMMARY/INLA_OUTPUTS/USCITIES
- Regular Polygon Simulation Study: Data Simulation and Plots: INLA_w_MCMC/IwMSIMSTUDY
- Regular Polygon Simulation Study: INLA and Multivariate INLA within MCMC Code and Outputs: INLA_w_MCMC/IwMSIMSTUDY
- Regular Polygon Simulation Study: Multivariate INLA within MCMC Plots and Results: INLA_w_MCMC/IwMOUTPUTSUMMARY (for code to create the results) and INLA_w_MCMC/IwMOUTPUTSUMMARY/IwM_OUTPUTS/REGPOLSIMSTUDY

10 Chapter 6

None

Return to Table of Contents

11 Table: Chapters to Directories

Thesis	Result	Directory
Chapter 1 & Appendix A	Data Plots	DATA/EDA
	Generalised Linear Models Plots and Results	DATA/MODELS/GLMS
	Ripley's K	DATA/MODELS/GLMS
Chapter 2 & Appendix B	Los Angeles Grids	GRID_MESH/IRREGULAR_POLYGON_LGCP/ IRREGPOLLGCP_CODE/IRREGPOLLGCP_CODE_ TESTINGVALUES
	Gaussian Example Regular Polygon Mesh	GRID_MESH/GAUSSIAN/GAUSSIAN_CODE
	Los Angeles Mesh	GRID_MESH/IRREGULAR_POLYGON_LGCP/ IRREGPOLLGCP_CODE/IRREGPOLLGCP_CODE_ TESTINGVALUES
	SBC Histograms	EXTRA
	BMA Plots	EXTRA
Chapter 3 & Appendix C	SBC Histograms with Models Plots and Results	EXTRA
	Gaussian Simulation Study Code	GRID_MESH/GAUSSIAN/GAUSSIAN_CODE
	Gaussian Example Meshes	
	Gaussian Example Simulation Study Plots and Results	GRID_MESH/GAUSSIAN/GAUSSIAN_ANALYSIS
	LGCP Example Simulation Study Code	GRID_MESH/REGULAR_POLYGON/REGPOLLGCP_CODE
	LGCP Example Simulation Study Plots and Results	GRID_MESH/REGULAR_POLYGON_LGCP/ REGPOLLGCP_ANALYSIS
Chapter 4 & Appendix D	Los Angeles Grid, Mesh, Covariates and Simulated Data plots	GRID_MESH/IRREGULAR_POLYGON_LGCP/ IRREGPOLLGCP_CODE/IRREGPOLLGCP_CODE_ TESTINGVALUES
	Irregular Polygon LGCP Simulation Study Code	GRID_MESH/IRREGULAR_POLYGON_LGCP/ IRREGPOLLGCP_CODE

	Irregular Polygon LGCP Simulation Study Plots and Results		GRID_MESH/IRREGULAR_POLYGON_LGCP/ IRREGPOLLGCP_ANALYSIS
	LA Census Tract Areas Histogram		DATA/EDA
	LA Motor Vehicle Theft INLA Code and Output		INLA_w_MCMC/USCITIESIwM
	LA Motor Vehicle Theft INLA Plots and Results		INLA_w_MCMC/IwMOUTPUTSUMMARY and INLA_w_MCMC/IwMOUTPUTSUMMARY/INLA_ OUTPUTS/USCITIES
Chapter 5 & Appendix E	Univariate and Multivariate INLA within MCMC Code		INLA_w_MCMC/IwMFUNCTIONS
	US Cities	Data Plots	DATA/EDA and INLA_w_MCMC/USCitiesIwM
		INLA, Univariate and Multivariate INLA within MCMC Code and Outputs	INLA_w_MCMC/USCitiesIwM
		INLA, Univariate and Multivariate INLA within MCMC Plots and Results	INLA_w_MCMC/IwMOUTPUTSUMMARY (for code to create the results) and INLA_w_MCMC/ IwMOUTPUTSUMMARY/IwM_OUTPUTS/USCITIES and INLA_w_MCMC/IwMOUTPUTSUMMARY/INLA_ OUTPUTS/USCITIES
	Regular Polygon Simulation Study	Data Simulation and Plots	INLA_w_MCMC/IwMSIMSTUDY
		INLA and Multivariate INLA within MCMC Code and Outputs	INLA_w_MCMC/IwMSIMSTUDY
		INLA and Multivariate INLA within MCMC Plots and Results	INLA_w_MCMC/IwMOUTPUTSUMMARY (for code to create the results) and INLA_ w_MCMC/IwMOUTPUTSUMMARY/IwM_OUTPUTS/ REGPOLSIMSTUDY

Table 1: Table matching results from each chapter of my thesis to the relevant directory.

Return to Table of Contents