Geospatial Analysis on Green Spaces and Transport Networks

Taylor Chu, Kaitlyn Louth, Jöel Tatang & Rowan Turner

Supervisors: Dr. Stuart King & Dr. Jonas Latz
Tim Hurst & Anne Braae (Brainnwave)

MAXWELL INSTITUTE FOR MATHEMATICAL SCIENCES

HERIOT WATT UNIVERSITY

Brainnwave
Data. Intelligence.

Maxwell Institute Graduate School, The University of Edinburgh and Heriot-Watt University in collaboration with Brainnwave

Abstract

This report explores the accessibility of green spaces in urban environments, focusing on the City of Edinburgh as a test case. Accessible green spaces are important due to their role in mitigating pollution, and reducing the urban heat island effect, as well as their links to socio-economic factors and health outcomes. We discuss the need for green space accessibility metrics in an attempt to help in decision making on which areas require more green spaces in the City of Edinburgh. The aim of the project is to explore and compare potential new metrics to measure the accessibility of green spaces, as well as determine how their placement influences an urban environment. The report outlines the methodology for implementing various green space accessibility models, namely the shortest path algorithm, commute distances, and diffusion models. The report also considers how the size of green spaces may impact accessibility, by implementing a diffusion model where green spaces are sources of 'happy influence' across the city. Finally, we present our results of the various algorithms and simulations, comparing and discussing the different metrics, and outlining the potential applications of green space accessibility metrics in urban policy-making and optimisation.

Acknowledgements

Thank you to our supervisors Stuart King and Jonas Latz for their excellent insight, advice and support throughout this project. Thank you to Tim Hurst and Anne Braae at Brainnwave for motivating the project and their helpful guidance throughout.

Author Contributions

Taylor Chu worked on acquisition of open source data, construction of the graph from such data and implementations of diffusion and drift-diffusion with alternative graph Laplacian. Kaitlyn Louth worked on the motivations behind the research and understanding of the graph theory that underpins the accessibility models, as well as a focus on the theory, implementation and computation of the forward Kolmogorov diffusion model, and a copywriting role throughout. Jöel attempted to implement diffusion model with Fokker-Planck equation which was in the end not used in the project. Rowan Turner worked on implementations of shortest path, commute distance, happiness diffusion and the drift-diffusion with the alternative graph Laplacian.

Contents

1 Introduction **3**
- 1.1 Why is this Research Important? . . . 3
- 1.2 Project Aims and Objectives . . . 3

2 What is a Graph? **4**
- 2.1 The Graph Laplacian . . . 6

3 Edinburgh as a Graph **7**
- 3.1 Data Acquisition . . . 7
- 3.2 Data Pre-processing and Graph Construction . . . 7

4 Green Space Accessibility Metrics **9**
- 4.1 Shortest Path . . . 9
- 4.2 Commute Distance . . . 9
- 4.3 Diffusion Equation on a Graph . . . 11
  - 4.3.1 Diffusion with Alternative Graph Laplacian (Drift) . . . 14
  - 4.3.2 Adding Sinks & Sources: The Happiness Metric . . . 14

5 Results **15**
- 5.1 Shortest Path . . . 16
- 5.2 Commute Distance . . . 17
- 5.3 Diffusion Across the Graph . . . 20
  - 5.3.1 Sources & Sinks Diffusion: Happiness Metric . . . 26

6 Discussion & Conclusions **27**
- 6.1 Metrics . . . 27
  - 6.1.1 Shortest Path and Commute Distance . . . 27
  - 6.1.2 Diffusion . . . 27
  - 6.1.3 Happiness Metric . . . 28
- 6.2 Computational Constraints and Scalability of the Metrics . . . 28
- 6.3 Error in Data . . . 30
- 6.4 Conclusion . . . 31

References **32**

A Detailed Description on Data Pre-processing and Graph Construction **34**

B Code **35**

1 Introduction

1.1 Why is this Research Important?

By 2050, 68% of the global population will live in cities [1]. Therefore, it is vital to ensure that our urban environments are sustainable and healthy places to live. Green spaces in cities mitigate the effects of pollution and can reduce a phenomenon known as the urban heat island effect, which refers to heat trapped in built-up areas. Additionally, there are socio-economic and health reasons as to why accessible green spaces in urban environments are necessary. For example, those without access to green spaces tend to have higher incidences of poverty [2], because low-income earners typically occupy the urban core and/or low-income inner ring suburbs, where green space is either scarce or poorly maintained. Furthermore, it has been found that those without access to green space suffer poorer health, with researchers estimating that 9 million people die every year as a direct result of air pollution [3].

In 2018 the UN stated a target that by 2030, access to safe, inclusive and accessible, green and public spaces should be universal, in particular for women and children, older persons and persons with disabilities [4]. These targets have been adopted into the UK government's own Environmental Improvement Plan (EIP), which put forward a requirement that all households should be within 15 minutes of green space by 2030 [5]. Such targets require the ability to measure the access and influence of green spaces, and these metrics impact how future policy is developed to increase access.

Since green spaces mitigate pollution, the City of Edinburgh in particular is aiming to become a one million tree city by 2030 in order to fulfil climate emergency commitments [6]. It is estimated that Edinburgh needs around 250,000 more trees to be planted in the next ten years [6]. Therefore, it is important to be able to measure how green an urban environment is, and determine which regions would benefit from more green spaces due to a current lack of accessibility.

For this project, we partnered with Brainnwave – a decision intelligence firm that utilises artificial intelligence and machine learning to tackle business problems. These involve finding ways to protect the environment, redesigning supply chains, and generally adjusting to a new turbulent world. They are currently working with industrial partners in urban solutions, who want to understand how to develop city areas to improve the well-being of their residents.

1.2 Project Aims and Objectives

Green space accessibility metrics may be used to compare regions of a city, for instance to inform policy makers which neighborhoods are most underserved. They may also be used as tools in optimisation, such as seeking where to place new green spaces to best maximise societal benefit. The commonly used shortest path metric, described in Section 4 measures one dimension of access – namely how close any part of the city is to a green space by a direct path – but potential applications of green space accessibility metrics could require metrics that capture different elements of a city. For instance, the shortest path metric assumes access to green space is relevant when people are

travelling directly to those green spaces. However it may also be relevant to examine how likely people are to incidentally walk through parks on their way to other destinations, such as work or school. The difference between direct access and incidental access could be understood to quantify access in leisure time, versus access during everyday necessary tasks. Quality or size of green spaces, in combination with how close they are may also be another relevant dimension.

The primary aim of this project is as a preliminary exploration to identify and compare potential new metrics to measure how accessible green spaces are, and how their placement influences an urban environment, using the City of Edinburgh as our test case.

In order to achieve this, we want to firstly transform geospatial data into a graph. Our project will focus on data obtained on the City of Edinburgh, with the aim that our green space accessibility models can be generalised to other cities. Once we initialise Edinburgh as a graph, we will implement multiple models which measure green space accessibility in various ways. Namely, we will implement the shortest path algorithm and calculate the commute distances. These find the shortest paths between a green space and all other regions of the city, and calculate the expected value of a random walk, round trip, between city regions and green spaces, respectively. Additionally, we will simulate people moving from areas of the city towards green space using different diffusion models. Finally, we will extend a diffusion model such that we account for the effects of different sized green spaces on the city. In this case, rather than measuring green space accessibility, we are measuring how such green spaces hold influence over the city. Therefore, we must consider what 'access' means, and what a useful metric looks like in order to compare our different models.

The structure of this report is as follows. Section 2 defines what a graph is and introduces the mathematics that underpins the graph theory required for this project. In particular, this section establishes what the graph Laplacian is and how it is calculated. Section 3 details the types of data sources which were required to initialise Edinburgh into a graph, explains the process of how this was achieved, and describes how the features of a graph (its edges and nodes) are defined across a city. Section 4 outlines the mathematics and methodologies behind the various green space accessibility models. In Section 5 and Section 6, we present the results of our algorithms and simulations. These sections also include a discussion and comparison between our various metrics, and a comparison within each metric on different regions within the city.

2 What is a Graph?

Graph theory is a branch of mathematics that deals with the study of graphs, which are mathematical structures used to model pairwise relationships between objects. Graph theory is widely used in various fields, and its importance today can be seen in applications such as social networks, transportation networks, biological networks and epidemiology. Our focus will be on its application to transportation networks. In this section, we will define the mathematical objects that underpin the study of networks represented as graphs.

Definition 2.1. A graph G is given by a pair (V,E), where V is a set of vertices (or nodes in the context of a network), and $E \subset V \times V$ is a set of pairs of vertices called edges (or interactions in the

context of a network) [7]. The graph G would then be visually represented by nodes connected by edges.

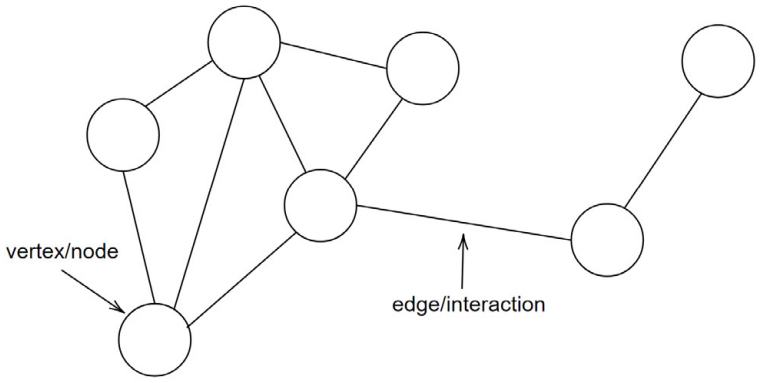


Figure 1: Visual representation of a basic graph, labelling the nodes and edges.

Depending on the network we want to initialise and the problem we want to solve, a graph can be directed, undirected or a mix of undirected and directed. If a graph is directed, this means that at least one edge in our graph can only be travelled across in one direction. On the other hand, an undirected graph means that all edges in our graph are bidirectional. Additionally, a weighted graph is a graph in which a weight or cost is assigned to each edge, and because we will be considering a transportation network, the weight of each edge in the graph will represent the distance between two nodes (see Section 3). This connectivity of a graph is encoded by a weighted adjacency matrix $\boldsymbol{W}$.

Definition 2.2. The weighted adjacency matrix of an undirected graph G is the square matrix $\boldsymbol{W} = (w_{ij})_{i,j=1,\ldots,n}$, where each edge between two vertices v_i and v_j carries a non-negative weight $w_{ij} \geq 0$. If $w_{ij} = 0$, this means that v_i and v_j are not connected.

For an undirected graph, the weighted adjacency matrix is symmetric i.e. $w_{ij} = w_{ji}$, however this fails for a directed graph. The degree of a vertex $u_i \in V$ is defined as

$$d_i = \sum_{j=1}^{n} w_{ij},$$

and describes the number of edges connected to a vertex u_i.

Definition 2.3. The degree matrix $\boldsymbol{D}$ is a diagonal matrix such that $\boldsymbol{D} = \text{diag}\{d_1, \ldots, d_n\}$, where d_i is the degree of vertex u_i.

Alternatively, for a directed graph we have separate in-degree and out-degree matrices. The in-degree matrix is a diagonal matrix that contains the number of incoming edges to a vertex (the in-degrees), and the out-degree matrix is a diagonal matrix containing the number of outgoing edges from a vertex (out-degrees).

2.1 The Graph Laplacian

Graph Laplacian matrices are one of the main tools required for spectral clustering, and spectral graph theory is the field of study dedicated to these matrices [8]. In this section we want to define the unnormalised graph Laplacian, and highlight some of its important properties. We will later utilise the graph Laplacian in calculations for our green space accessibility models (Section 4).

The graph Laplacian $\boldsymbol{L}$ has several important properties. One of the most important properties is that it is always a positive semi-definite matrix. This means that the scalars $\boldsymbol{z}^T\boldsymbol{L}\boldsymbol{z}$ and $\boldsymbol{z}^*\boldsymbol{M}\boldsymbol{z}$ are positive or zero, for any non-zero real column vector $\boldsymbol{z}$, where $\boldsymbol{z}^*$ is its conjugate transpose. The eigenvalues and eigenvectors of the graph Laplacian are used to study various properties of the graph. For example, the number of zero eigenvalues of the graph Laplacian is equal to the number of connected components in the graph. The eigenvectors corresponding to the non-zero eigenvalues of the graph Laplacian can be used to partition the graph into clusters, a technique known as spectral clustering. We define the unnormalised graph Laplacian as follows.

Definition 2.4. For an undirected graph, the graph Laplacian $\boldsymbol{L}$ is defined by $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W}$, where $\boldsymbol{D}$ and $\boldsymbol{W}$ are respectively the degree matrix and the weighted adjacency matrix. That is

$$L_{ij} = \begin{cases} d_i = \sum_{i=1}^{n} w_{ij} & \text{if } i = j \\ -w_{ij} & \text{if } i \neq j \text{ and there is an edge} \\ 0 & \text{if } i \neq j \text{ and there is not an edge} \end{cases}. \tag{2.1}$$

The following proposition outlines important properties of the graph Laplacian for an undirected graph [7]:

Proposition 1. (Properties of the graph Laplacian $\boldsymbol{L}$). The graph Laplacian $\boldsymbol{L}$ satisfies the following properties:

1. For every vector $f \in \mathbb{R}^n$ we have
$$f'\boldsymbol{L}f = \frac{1}{2}\sum_{i,j=1}^{n} w_{ij}(f_i - f_j)^2.$$
2. $\boldsymbol{L}$ is symmetric and positive semi-definite.
3. The smallest eigenvalue of $\boldsymbol{L}$ is 0, and the corresponding eigenvector is the constant one vector $\mathbf{1}$.
4. $\boldsymbol{L}$ has n non-negative, real-valued eigenvalues $0 = \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_n$.

For a directed graph, we need to consider the in-degree and out-degree matrices ($\boldsymbol{D}_{in}$, $\boldsymbol{D}_{out}$) when calculating the unnormalised graph Laplacian such that

$$\boldsymbol{L} = \begin{cases} \boldsymbol{D}_{in} - \boldsymbol{W} & \text{for the in-degree matrix} \\ \boldsymbol{D}_{out} - \boldsymbol{W}^T & \text{for the out-degree matrix} \end{cases}, \tag{2.2}$$

where $\boldsymbol{W}^T$ is the transpose of the weighted adjacency matrix.

There are other versions of the graph Laplacian such as types of normalised graph Laplacians, which we did not use in our project. These are discussed in Von Luxburg (2007) [7]. We will see the unnormalised graph Laplacian being utilised in our green space accessibility models, both for directed and undirected graphs.

3 Edinburgh as a Graph

To calculate distance-based metrics regarding accessibility of open green spaces, as well as simulate diffusion of population between different areas, we constructed a weighted graph from the street network of Edinburgh. In this section we will explain the process of data acquisition, and the steps to obtain the final graph used in our project.

3.1 Data Acquisition

All data used in this project was obtained from free and open sources, as shown in Table 1. The majority of the data we used is of GeoJSON format, which encodes various attributes of geographic data structure, including points, lines and polygons. Using the Python package GeoPandas, the data in GeoJSON files can be imported as a Pandas dataframe.

Data	Data file type	Year	Source
Edinburgh street network	GeoJSON	2023	OpenStreetMap [9]
Open green space geometry and location	GeoJSON	2022	Ordnance Survey Open Green Space [10]
Edinburgh local authority boundary	GeoJSON	2019	SpatialData.gov.scot [11]
Data zone boundaries	GeoJSON	2021	SpatialData.gov.scot [11]
Population Estimate per Data Zone	xlsx	2022	Statistics.gov.scot [12]

Table 1: Table displaying the data file types, the year the data was published, and the data sources.

3.2 Data Pre-processing and Graph Construction

With our data acquired from different sources, here we describe intuitively the process of constructing the graphs we used in the project. A step-by-step description of the process can be found in Appendix A.

A street networks naturally forms a graph with edges being streets or street sections, and nodes being the intersections of streets. Since in this project we only consider distance travelled on foot, the resulting graph is undirected because there are no restrictions to the direction one could take when travelling. In the same sense, we removed any edges representing streets that are by OpenStreetMap's definition not walkable, such as highways, by filtering out such types of edges. Up to this point, the only information we have are the coordinates of the nodes and lengths and shapes of the edges. To encode green space, data zones and population data, we utilised the coordinates and geometries describing each green space and data zone, and assigned attributes to nodes belonging

to such areas with unique identifiers. For example, nodes that are in green spaces were assigned to be 'green space nodes', and gained an attribute corresponding to the unique identifier of the green space they belonged to. Now that we are able to distinguish between green space and non-green space nodes, within each data zone we uniformly distributed the population of the data zone to its non-green space nodes, resulting in the full graph shown in Fig. 2. The graph consists of $72,008$ nodes, of which $2,498$ belong to green spaces, and $89,166$ edges, of which $7,730$ are connected to green spaces.

The total area of all open green spaces combined is $28.90km^2$, which is just above 10% of the city's area of $264km^2$. Residents of Edinburgh might be curious why the west and southwest parts of the graph do not show as many green spaces as we could physically see. We note that the green space data is obtained from The Ordnance Survey of **Open** Green Spaces. This means that any privately owned areas, or places categorised by Ordnance Survey's definition of not an open green space, are not shown. We also note that although The Pentlands is an open green space, it is actually outside of the local authority boundary of the City of Edinburgh, hence also not present in the graph. This authority boundary between the City of Edinburgh and The Pentlands creates an artificial border that affects the green space metrics, as further discussed in Section 6.

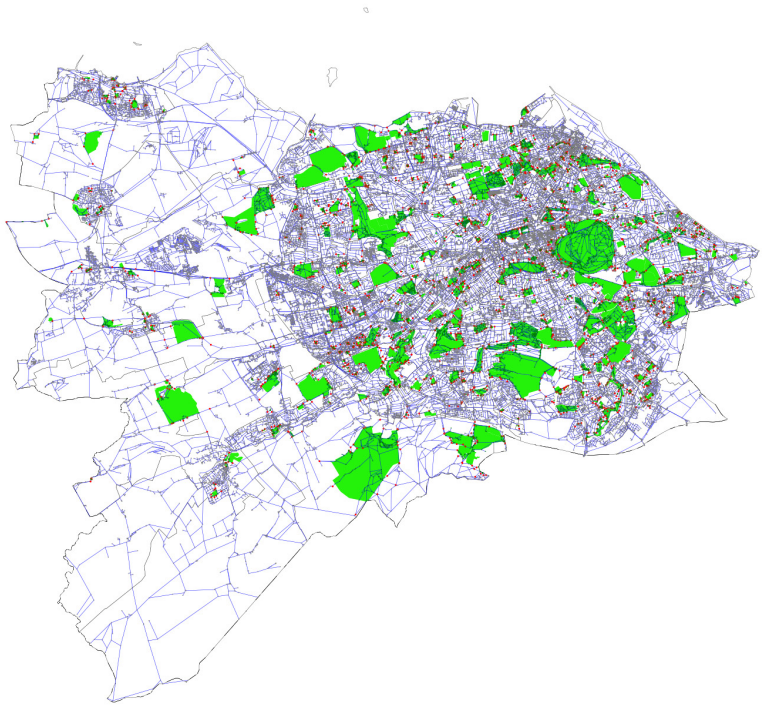


Figure 2: Graph of Edinburgh where green polygons represent green spaces, grey nodes represent city nodes and red nodes represent green space nodes.

4 Green Space Accessibility Metrics

4.1 Shortest Path

The current industry standard for accessibility metrics applied to transport networks is the shortest path algorithm. Generally this is because it is relatively simple to implement in comparison to other metrics. The idea is to find a path between two nodes on a graph, such that the sum of the weights on its constituent edges is minimised [13]. One of the most commonly used shortest path algorithms is Dijkstra's algorithm, which was designed to solve the single-source shortest path problem for a static graph. It works by starting from the source node and calculating the shortest path from that node to each other node in the network. We can describe this algorithm as follows:

Algorithm 1 Dijkstra's algorithm [14]

1: Create sets of visited and unvisited nodes.
2: Initialise all nodes with a distance of ∞, except for the starting node which has a distance of 0.
3: Select the node with the smallest distance and visit all of its neighbours.
4: For each neighbour, calculate the distance to that neighbour through the current node and update the neighbour's distance if it is smaller than its current distance.
5: Move the current node to the visited set.
6: Repeat for other unvisited nodes, until the unvisited set is empty.

We implement Dijkstra's shortest path algorithm across our graph by setting each green space node as a source and finding its shortest path to each city node. For each city node, we then have a list of the lengths of its shortest paths to each green space node by symmetry. We simply take the minimum of the list to obtain the path length from the city node to its nearest green space.

As a model, the output of the shortest path metric is as expected: at every city node it gives the value of the distance to the nearest green space.

4.2 Commute Distance

An alternative to the shortest path algorithm is to compute the commute distance, which gives the expected length of a random walk from node u_i to node u_j and back. In the context of graph theory, a random walk can be described as a process where a 'walker' starts at a given node in a graph, and randomly chooses one of the neighboring nodes to move to at each step. This process is repeated until the walker reaches a destination node or a stopping condition is met. We explore the commute distance as an alternative to the shortest path metric, to consider the difference in access to green spaces between directly walking to one and walking upon one randomly.

Intuitively, the commute distance between two nodes decreases if there are many different, short paths to travel between node u_i and u_j. In our case, this would correspond to many different walking routes from a city node to a green space. We expect this metric to encode not only closeness to green spaces, but also the general connectedness of a node to green spaces.

If we consider an undirected and unweighted graph G, then let u_k be the node visited by a random walk after k steps, and p_{ij} be the probability of moving from node u_i to node u_j in one step of the random walk. We can now write the probability of being at node u_j after k steps as

$$\mathbb{P}(u_k = u_j) = \sum_{i=1}^{n} \mathbb{P}(u_k = u_j | u_0 = u_i)\mathbb{P}(u_0 = u_i), \tag{4.1}$$

where $\mathbb{P}(u_k = u_j | u_0 = u_i)$ is the conditional probability of arriving at node u_j after k steps given that we started at node u_i, and $\mathbb{P}(u_0 = u_i)$ is the probability of starting at node u_i.

The expected number of steps to travel to node u_j from node u_i is then given by

$$\mathbb{E}[T_j] = \sum_{k=0}^{\infty} k\mathbb{P}(u_k = u_j | u_0 = u_i), \tag{4.2}$$

where T_j is the number of steps taken to reach node u_j from u_i. Similarly, the expected number of steps to return to node u_i is given by

$$\mathbb{E}[T_i] = \sum_{k=0}^{\infty} k\mathbb{P}(u_k = u_i | u_0 = u_j), \tag{4.3}$$

where T_i is the number of steps taken to get back to u_i from u_j. Since our graph is connected and has no absorbing states (i.e. all bidirectional edges), we can define the commute distance to be the expected absolute difference between the number of steps to reach node u_j from u_i, and the number of steps to return back:

$$c_{ij} = \mathbb{E}[|T_j - T_i|]. \tag{4.4}$$

For our weighted graph of Edinburgh, the commute distance can be calculated using the pseudo-inverse $\boldsymbol{L}^{\dagger}$ of the graph Laplacian [7]. From Proposition 1 we have that the graph Laplacian can be decomposed as $\boldsymbol{L} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^T$, where $\boldsymbol{U}$ is the matrix containing all eigenvalues as columns and $\boldsymbol{\Lambda}$ is the diagonal matrix containing eigenvalues $\lambda_1, ..., \lambda_n$. We can then define the pseudo-inverse to be $\boldsymbol{L}^{\dagger} = \boldsymbol{U}\boldsymbol{\Lambda}^{\boldsymbol{\dagger}}\boldsymbol{U}^{\boldsymbol{T}}$, where $\boldsymbol{\Lambda}^{\boldsymbol{\dagger}}$ is the diagonal matrix containing entries $1/\lambda_i$ or 0 if $\lambda_i = 0$. Given this definition of the pseudo-inverse of our graph Laplacian, we can calculate the commute distance as [7]

$$\begin{aligned} c_{ij} &= \mathrm{vol}(V)(l^{\dagger}_{ii} - 2l^{\dagger}_{ij} + l^{\dagger}_{jj}) && (4.5)\\ &= \mathrm{vol}(V)(e_i - e_j)^T L^{\dagger}(e_i - e_j) && (4.6)\\ &= \mathrm{vol}(V)(e_i - e_j)^T U \Lambda^{\dagger} U^T (e_i - e_j), && (4.7) \end{aligned}$$

where we are considering a connected, undirected graph $G = (V, E)$. This is equivalent to Eq. (4.4) because we can express the expected absolute difference in arrival times T_i and T_j as

$$\begin{aligned} \mathbb{E}[|T_j - T_i|] &= \int_0^{\infty} \mathbb{P}(T_j - T_i > t)\mathrm{d}t + \int_0^{\infty} \mathbb{P}(T_j - T_i < -t)\mathrm{d}t \\ &= 2\int_0^{\infty} \mathbb{P}(T_j - T_i > t)\mathrm{d}t. \end{aligned} \tag{4.8}$$

Then we can express $\mathbb{P}(T_j - T_i > t)$ in terms of the transition probabilities between nodes u_i and u_j. Let $p_{ij}(t)$ be the probability that the random walk starting from node u_i reaches u_j in t steps. Then, we have

$$\mathbb{P}(T_j - T_i > t) = \sum_{k=t+1}^{\infty} p_{ij}(k). \tag{4.9}$$

Substituting this expression into Eq. (4.8) and using $p_{ij}(t) = [\exp(t\boldsymbol{L})]_{ij}$, we obtain

$$\begin{aligned}
\mathbb{E}\left[|T_j - T_i|\right] &= 2\int_0^{\infty} \sum_{k=t+1}^{\infty} \exp(t\boldsymbol{L})]_{ij} \mathrm{d}t \\
&= 2\int_0^{\infty} [\exp(t\boldsymbol{L})]_{ij} \mathrm{d}t \\
&= 2[\boldsymbol{L}^{\dagger}]_{ij} \\
&= \mathrm{vol}(V)(e_i - e_j)^T U \Lambda^{\dagger} U^T (e_i - e_j),
\end{aligned} \tag{4.10}$$

where the (i, j)-th entry of $\exp(t\boldsymbol{L})$ over $[0, \infty]$ is equal to the (i, j)-th entry of the pseudoinverse of $\boldsymbol{L}$ [15].

In our implementation, we calculate the commute distance from every green space node to every city node, and then as in the shortest path metric, we consider the list of commutes for each city node and take the minimum for that node as the commute value.

4.3 Diffusion Equation on a Graph

While shortest path and commute distance are both static metrics, we might like to have some measurement that looks at how people are able to access green space through time. Thus we model people walking through the city, using a difussion process. We do this using the Kolmogorov forward equation

$$\frac{\partial p(x,t)}{\partial t} = -\frac{\partial}{\partial x}\left(\mu(x,t)p(x,t)\right) + \frac{1}{2}\frac{\partial^2}{\partial x^2}\left(\sigma(x,t)^2 p(x,t)\right). \tag{4.11}$$

Here, $\mu(x,t)$ is a drift term which represents a systematic change or bias in the behavior of a system over time, and we will let $\eta = \frac{\sigma(x,t)^2}{2}$ be our diffusion coefficient. Typically, we think of this drift-diffusion equation describing the time evolution of the probability density function of the velocity of a particle, under the influence of drift and random forces. With regard to diffusion across a graph, we can think of such particles as people randomly walking throughout the city, while drift is people walking in a deterministic way towards green spaces. The aim is to simulate the distribution of the population through time, and track when people reach green spaces.

To derive the forward Kolmogorov equation for a discrete-state stochastic process, we can use the transition probabilities to approximate the continuous-time evolution of the process by a sequence of discrete-time steps. In the limit as the time step goes to zero, this sequence of discrete-time steps approaches the continuous-time evolution, and the discrete-space probability density function approaches the continuous-space probability density function.

In our first model, we set the drift term to zero and consider a pure diffusion process, which is equivalent to people walking randomly through the city.

If the diffusing substance (in our case, people) is moving across edges of a graph from node to node, the domain is discrete and we can consider p people at our nodes moving across an edge at a diffusion rate η as shown in Fig. 3. Then the number of people that move from node u_i to node u_j over a time period $\mathrm{d}t$ is $\eta(p_i - p_j)\mathrm{d}t$ and vice versa. Therefore, we can write

$$\begin{aligned} \frac{\mathrm{d}p_i}{\mathrm{d}t} &= \eta(p_j - p_i), \\ \frac{\mathrm{d}p_j}{\mathrm{d}t} &= \eta(p_i - p_j). \end{aligned} \tag{4.12}$$

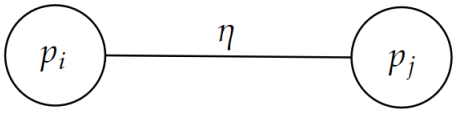


Figure 3: Simple visualisation of how diffusion is formulated on a graph, where p_i is the number of people at node u_i and η is the diffusion rate across an edge.

Diffusion to and from node u_i must take into consideration all other nodes in our graph. We know that the connectivity of a graph is defined in the weighted adjacency matrix $\boldsymbol{W}$, so we can write

$$\begin{aligned} \frac{\mathrm{d}p_i}{\mathrm{d}t} &= \eta W_{i1}(p_1 - p_i) + \eta W_{i2}(p_2 - p_i) + \ldots + \eta W_{in}(p_n - p_i) \\ &= \eta \sum_{j=1}^{n} W_{ij}(p_j - p_i). \end{aligned} \tag{4.13}$$

Re-writing this expression we obtain,

$$\begin{aligned} \frac{\mathrm{d}p_i}{\mathrm{d}t} &= \eta \sum_{j=1}^{n} W_{ij} p_j - \eta p_i \overbrace{\sum_{j=1}^{n} W_{ij}}^{\substack{\text{degree of} \\ \text{node } i,\, d_i}} \\ &= \eta \sum_{j=1}^{n} W_{ij} p_j - \eta p_i d_i. \end{aligned} \tag{4.14}$$

Note that

$$\eta p_i d_i = \eta \sum_{j=1}^{n} \delta_{ij} p_j d_j.$$

Substituting this into Eq. (4.14) we have

$$\frac{\mathrm{d}p_i}{\mathrm{d}t} = \eta \sum_{j=1}^{n} W_{ij} p_j - \eta \sum_{j=1}^{n} \delta_{ij} p_j d_j. \tag{4.15}$$

Then using the n-dimensional vector $\boldsymbol{p}$, and the $n \times n$ degree matrix $\boldsymbol{D}$, we can write the inner product of row i of $\boldsymbol{W}$ with $\boldsymbol{p}$ and the inner product of row i of $\boldsymbol{D}$ with $\boldsymbol{p}$ i.e.

$$\begin{aligned} \eta \sum_{j=1}^{n} W_{ij} p_j &= \eta \left[\boldsymbol{W}\boldsymbol{p}\right]_i \\ \eta \sum_{j=1}^{n} \delta_{ij} p_j d_j &= \eta \left[\boldsymbol{D}\boldsymbol{p}\right]_i . \end{aligned} \tag{4.16}$$

Therefore Eq. (4.15) becomes

$$\begin{aligned} \frac{\mathrm{d}\boldsymbol{p}}{\mathrm{d}t} &= \eta \boldsymbol{W}\boldsymbol{p} - \eta \boldsymbol{D}\boldsymbol{p} \\ &= \eta (\boldsymbol{W} - \boldsymbol{D})\boldsymbol{p} \\ &= -\eta (\boldsymbol{D} - \boldsymbol{W})\boldsymbol{p}. \end{aligned} \tag{4.17}$$

We can now make use of our definition of the graph Laplacian $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W}$ (Definition 2.4) such that

$$\frac{\mathrm{d}\boldsymbol{p}}{\mathrm{d}t} = -\eta \boldsymbol{L}\boldsymbol{p}. \tag{4.18}$$

To implement this type of diffusion model, we use a directed graph whereby any edge connected to a green space node can only be travelled across in one direction, such that once someone enters a green space, they will remain there. This means that for long times, we would expect everyone to end up in a green space. We know from Section 2.1, Eq. (2.2), that for a directed graph we need to consider the out-degree matrix when calculating our graph Laplacian, hence Eq. (4.18) becomes

$$\frac{\mathrm{d}\boldsymbol{p}}{\mathrm{d}t} = -\eta \boldsymbol{L}^T \boldsymbol{p}. \tag{4.19}$$

Additionally, we will set our diffusion constant to be equivalent to an average walking speed of $1.4 m/s$. This means that the ODE which we want to solve is

$$\frac{\mathrm{d}\boldsymbol{p}}{\mathrm{d}t} = -1.4 \boldsymbol{L}^T \boldsymbol{p}. \tag{4.20}$$

We defined the weights of our graph to be the inverse of the edge length for this model. This is because for shorter edges, we want to simulate more diffusion of people across that edge than people diffusing across a longer edge in the same time frame. Finally, we set the initial distribution of people throughout the city based on the population estimates per data zone (Section 3.1). The population of a data zone is uniformly distributed amongst the nodes within each data zone of the graph as described in Appendix A.

We numerically solve Eq. (4.20) in Python (Appendix B) using the backwards differentiation method via the SciPy's function `scipy.integrate.solve_ivp`. This numerical method approximates the solution of an ODE,

$$p_{n+1} = p_n + \gamma_n f(t_{n+1}, p_{n+1}) \tag{4.21}$$

where p_n is the solution at the previous time step, γ_n is the time step size, $f(t_n, p_n)$ is the derivative of the solution, and p_{n+1} is the approximation of the solution at the current time step. It is an implicit method, meaning the solution at the current time step is expressed in terms of the solution at the previous time step, and it is based on the Taylor series expansion of the solution at the current time step. By truncating the Taylor series after the first term and using the approximation for the derivative at the current time step, the solution at the next time step can be approximated.

4.3.1 Diffusion with Alternative Graph Laplacian (Drift)

The diffusion model using the subsequent graph Laplacian simulates random walks where there is a higher probability that an individual would choose a path of shorter length, when this individual does not have a specific destination. This is because we used the inverse of the edge lengths as weights for the graph. To simulate more realistic movement on the graph, we could construct a graph Laplacian in an alternative way to include other information. Particularly, we look at including the shortest path to the closest green space as a decision factor when choosing the next edge in the diffusion process.

To construct this alternative graph Laplacian $\boldsymbol{L}_a$, we first need to construct an alternative weight matrix $\boldsymbol{W}_a$. To encourage movement towards the closest green spaces from any node, we construct $\boldsymbol{W}_a$ such that the only other edge connecting node u_i in $\boldsymbol{W}_a$ is the next node in the shortest path towards the closest green space from u_i. Again we take the inverse of the edge lengths as the entries of $\boldsymbol{W}_a$, then the resulting $\boldsymbol{L}_a$ is obtained by $\boldsymbol{L}_a = \boldsymbol{D}_a - \boldsymbol{W}_a$.

The balance between pure diffusion and shortest path influence is controlled by constants α and β, such that $\alpha + \beta = 1$. The resultant ODE to solve is therefore

$$\frac{\mathrm{d}\boldsymbol{p}}{\mathrm{d}t} = -1.4(\alpha \boldsymbol{L}^T + \beta \boldsymbol{L}_a^T)\boldsymbol{p}, \tag{4.22}$$

where $\boldsymbol{L}$ represents incidental access to green spaces, while $\boldsymbol{L}_a$ represents direct access to green spaces. This is also solved numerically using the backwards differentiation method in Python (Eq. (4.21)). When $\beta = 0$, Eq. (4.22) is a pure diffusion process, and is exactly the same as Eq. (4.20). In the same sense, when $\alpha = 0$, the process is a pure deterministic drift process and is the same as the population at each node moving directly to their closest green space, according to their shortest paths.

4.3.2 Adding Sinks & Sources: The Happiness Metric

The final metric we examined attempts to encode the area of the green spaces into a metric, while considering a similar diffusion process as in Section 4.3. Intuitively, we consider that being close to green spaces is beneficial and that larger green spaces have greater benefit. In this abstracted metric, we consider green space nodes to be sources of 'happiness', while city nodes are 'sinks' to happiness. The happiness diffuses from the green spaces into the rest of the graph.

We consider how the happiness flows through the graph: $\frac{d\boldsymbol{h}}{dt}$, and seek the equilibrium solution to

$$\frac{d\boldsymbol{h}}{dt} = -\boldsymbol{L}\boldsymbol{h} + \boldsymbol{s}, \tag{4.23}$$

where $\boldsymbol{L}$ is the undirected graph Laplacian, $\boldsymbol{h}$ is the happiness at each node, and $\boldsymbol{s}$ is a vector containing the sources and sinks. We require our sources and sinks to be balanced i.e.

$$\sum_{i=0}^{n} \boldsymbol{s}_i = 0, \tag{4.24}$$

in order to have an equilibrium in Eq. (4.23). To balance sources and sinks, we consider each city node to be a sink proportional to the population data given at that node, as described in Section 3.2. This gives a sink vector $\boldsymbol{s}_-$. For the green spaces, we divide the area of each green space by the sum of the area of all the green spaces, which gives a source share to each green space. We then divide this source share equally amongst the nodes belonging to that green space, generating the source vector $\boldsymbol{s}_+$. Lastly, we scale $\boldsymbol{s}_+$ such that $\sum_{i=0}^{n} \boldsymbol{s}_- = -\sum_{i=0}^{n} \boldsymbol{s}_+$. This gives us our the vector $\boldsymbol{s} = \boldsymbol{s}_+ + \boldsymbol{s}_-$.

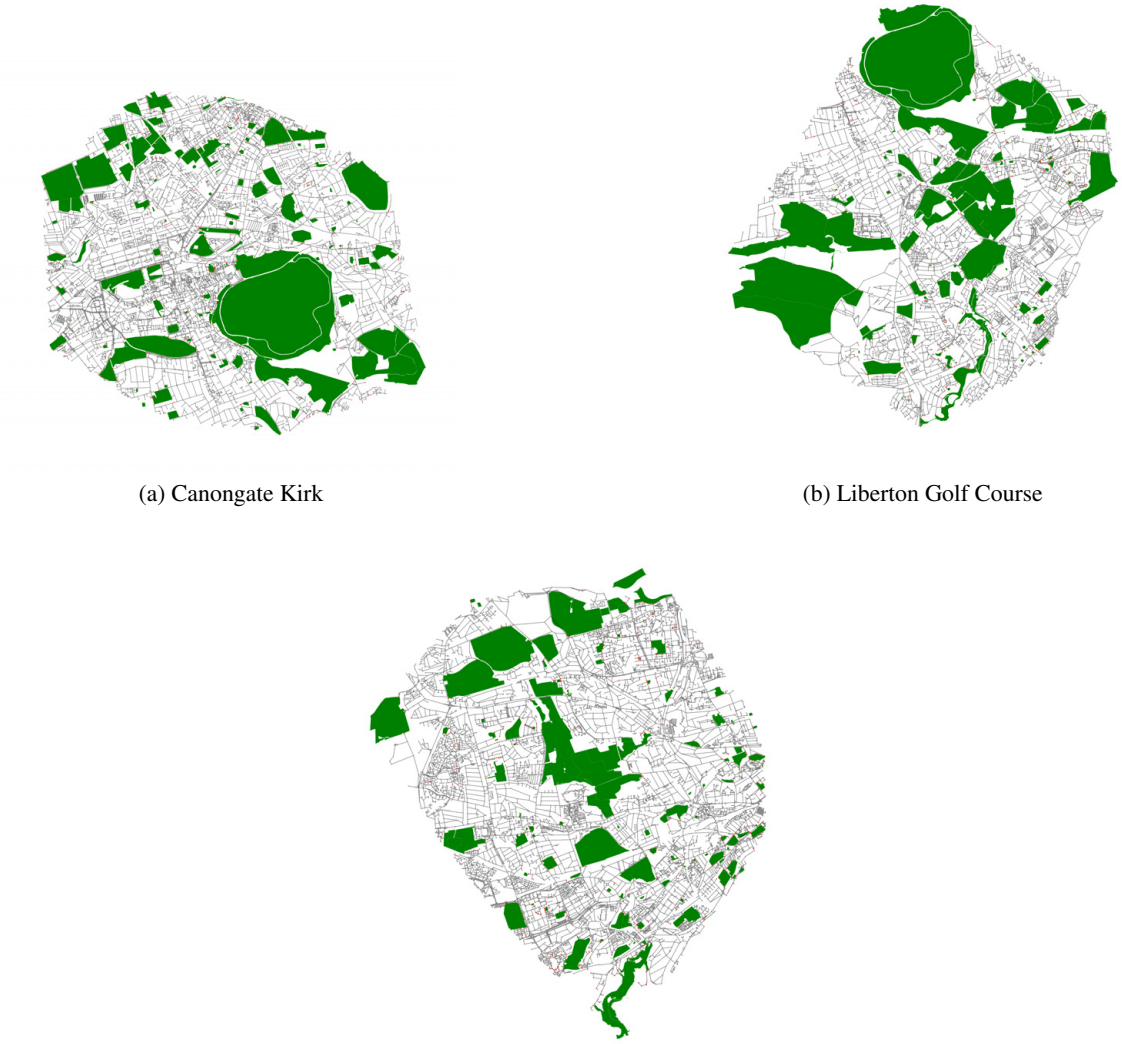
To solve

$$0 = -\boldsymbol{L}\boldsymbol{h} + \boldsymbol{s}, \tag{4.25}$$

we use the least squares function in Python `numpy.linalg.lstsq`. We note, however, that $\mathbf{L}$ is never invertible, because a graph Laplacian always has at least one zero eigenvalue (and the number is given by the number of connected components of the graph). Thus, while our method finds some solution, the solution is not guaranteed to be unique.

5 Results

Due to computational constraint, instead of computing the metrics for the entire city, we decided to compare the metrics at different areas of the city by generating subgraphs centred at three chosen green spaces. Within these subgraphs we included all green spaces, nodes and edges within a radius of $2,500m$ from the centre green space, as well as nodes and edges within $200m$ of all green spaces. To ensure we have a fully connected graph, we removed all nodes not connected to the largest connected component from the subgraphs. The green spaces, namely Canongate Kirk, Liberton Golf Course and Corstophine Hill Local Nature Reserve, are chosen to represent respectively the Central and North East, the South, and the West parts of the city. The resulting subgraphs are displayed in Fig. 14.



(a) Canongate Kirk

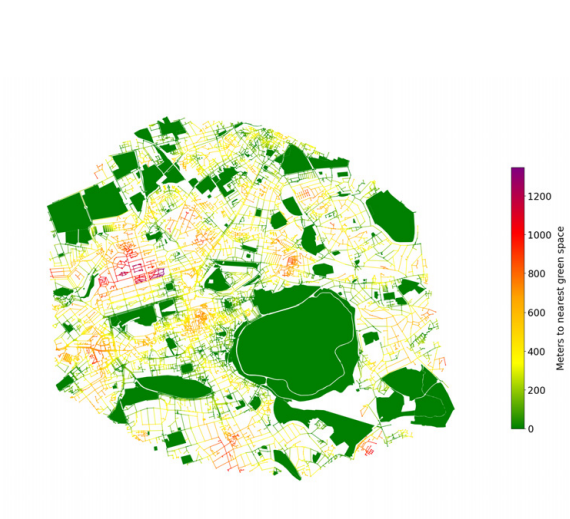
(b) Liberton Golf Course

(c) Corstophine Hill Local Nature Reserve

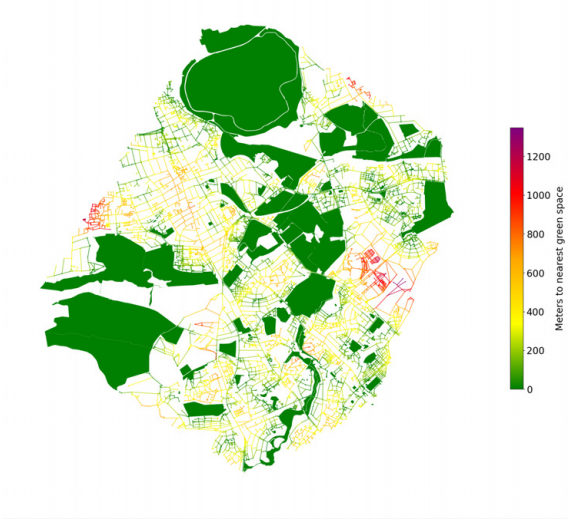
Figure 4: Subgraphs of Edinburgh considered for each metric, centred at three different green spaces. Green polygons represent green spaces, grey nodes represent the city nodes and red nodes represent the green space nodes.

5.1 Shortest Path

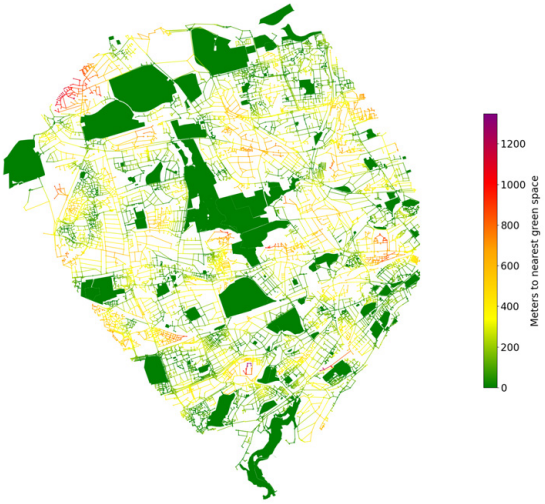
Visualisations of the results of the shortest path metric on the three subgraphs are given in Fig. 5. Average path distance, normalised by population numbers at each node, and corresponding variance is given in Table 2.



(a) Canongate Kirk



(b) Liberton Golf Course



(c) Corstophine Hill Local Nature Reserve

Figure 5: Implementation of the shortest path metrics on each subgraph. Node values give the distance in metres to the nearest green space.

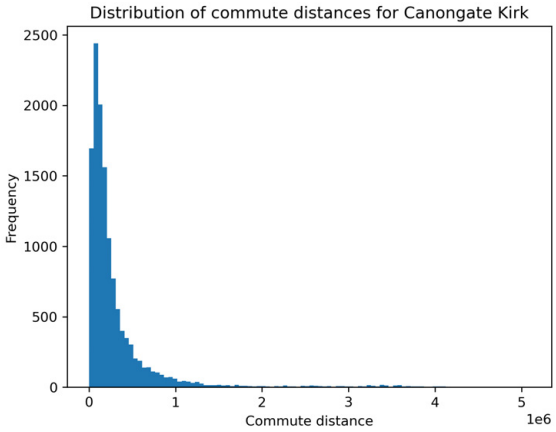
5.2 Commute Distance

Visualisations of the results of the commute distance metric on the three subgraphs are given in Fig. 7. Average path distance, normalised by population numbers at each node, and corresponding

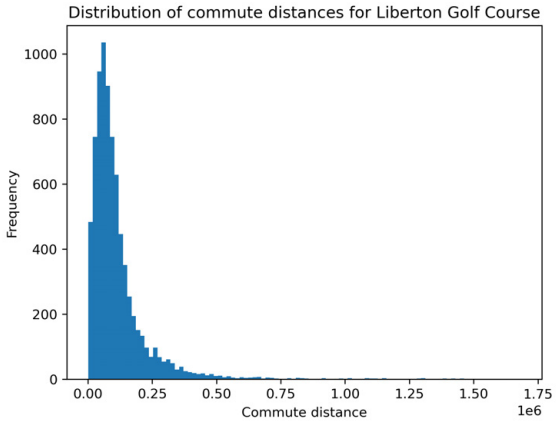
Subgraph	Short path mean	Short path var.	Comm. dist. mean	Comm. dist. var.
Canongate Kirk	192.6m	16368.8m^2	205854.8m	1.04×10^{11}m^2
Liberton Golf Course	206.8m	21841.7m^2	208812.0m	9.94×10^{10}m^2
Corstophine Hill	240.3m	29747.0m^2	107129.5m	1.92×10^{10}m^2

Table 2: Mean and variance, adjusted for population per node, of shortest paths and commute distances to green spaces in each subgraph.

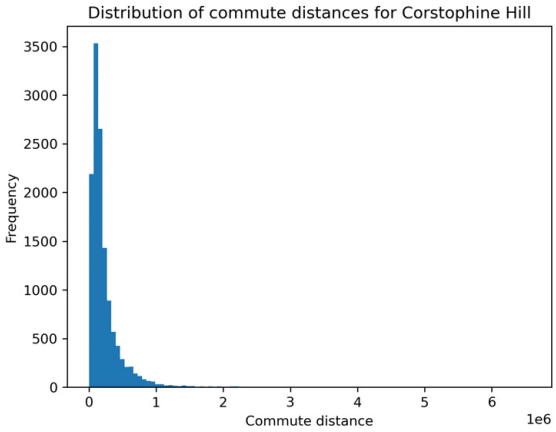
variance is given in Table 2. The distribution of the commute distances across the three subgraphs are given in Fig. 6.



(a) Canongate Kirk

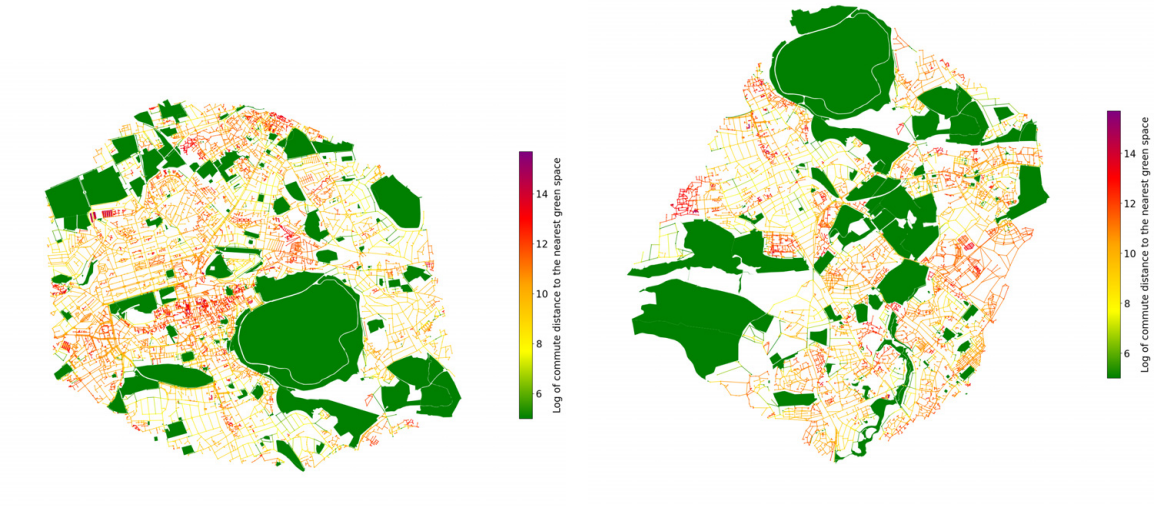


(b) Liberton Golf Course



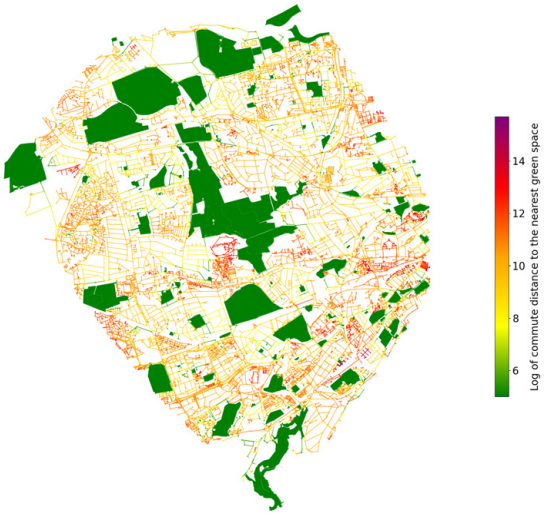
(c) Corstophine Hill Local Nature Reserve

Figure 6: Distribution of commute distances by node.



(a) Canongate Kirk

(b) Liberton Golf Course



(c) Corstophine Hill Local Nature Reserve

Figure 7: Implementation of the commute distance metric on each subgraph. Node values give the log of the minimum of the expected values of random walks from that node to each green space node and back.

5.3 Diffusion Across the Graph

Solving the ODEs in Eqs. (4.20) and (4.22) with $10,000$ time steps by varying values of α and β, we display the distribution of population at each green space at the final time step, for each combination of α and β in Figs. 8 to 10. Additionally, we present the change in total population between city nodes and green spaces for each subgraph in Figs. 11 to 13, respectively. We have highlighted the time steps when 25%, 50% and 75% of the population within the subgraphs have ended up in a green space.

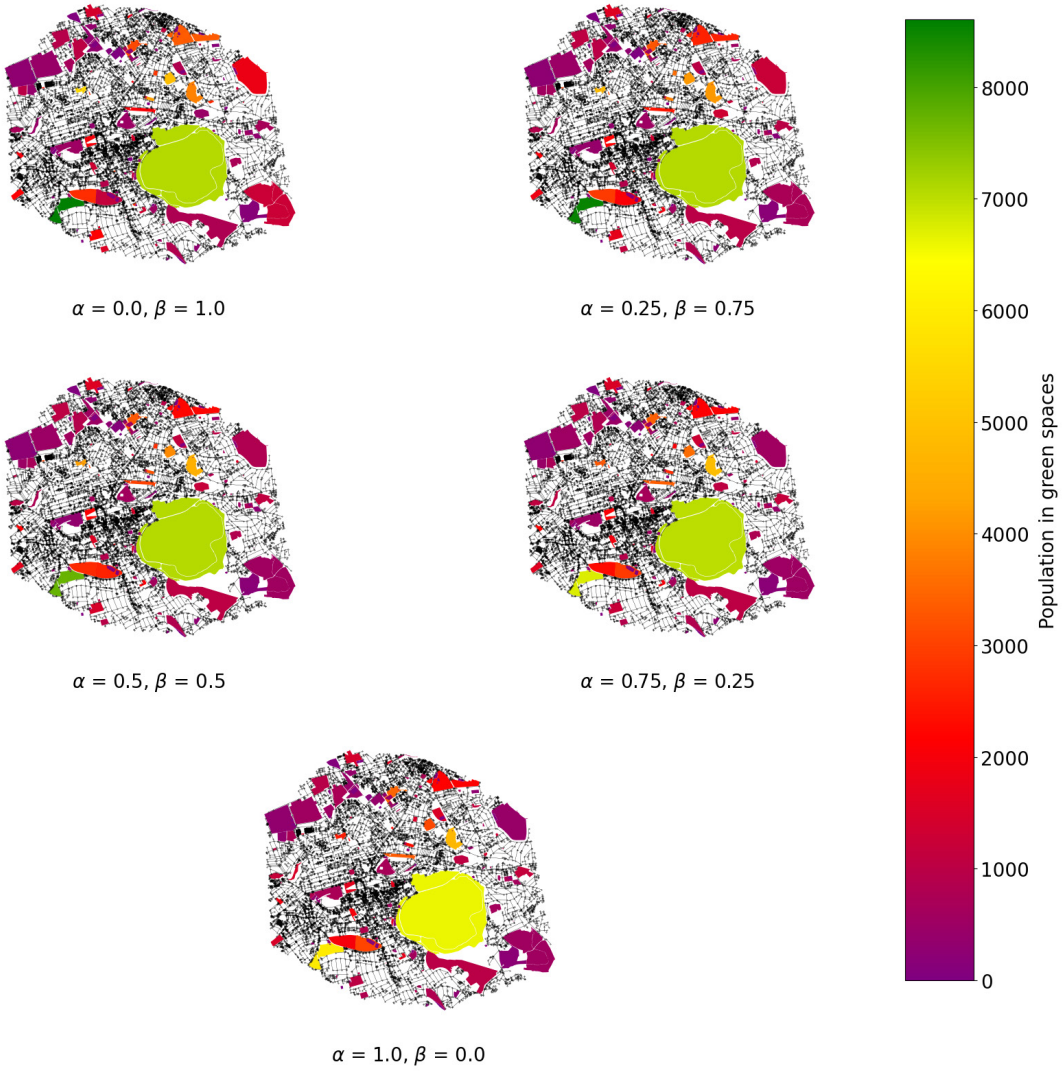


Figure 8: Distribution of population among green spaces at final time step for subgraph centred at Canongate Kirk.

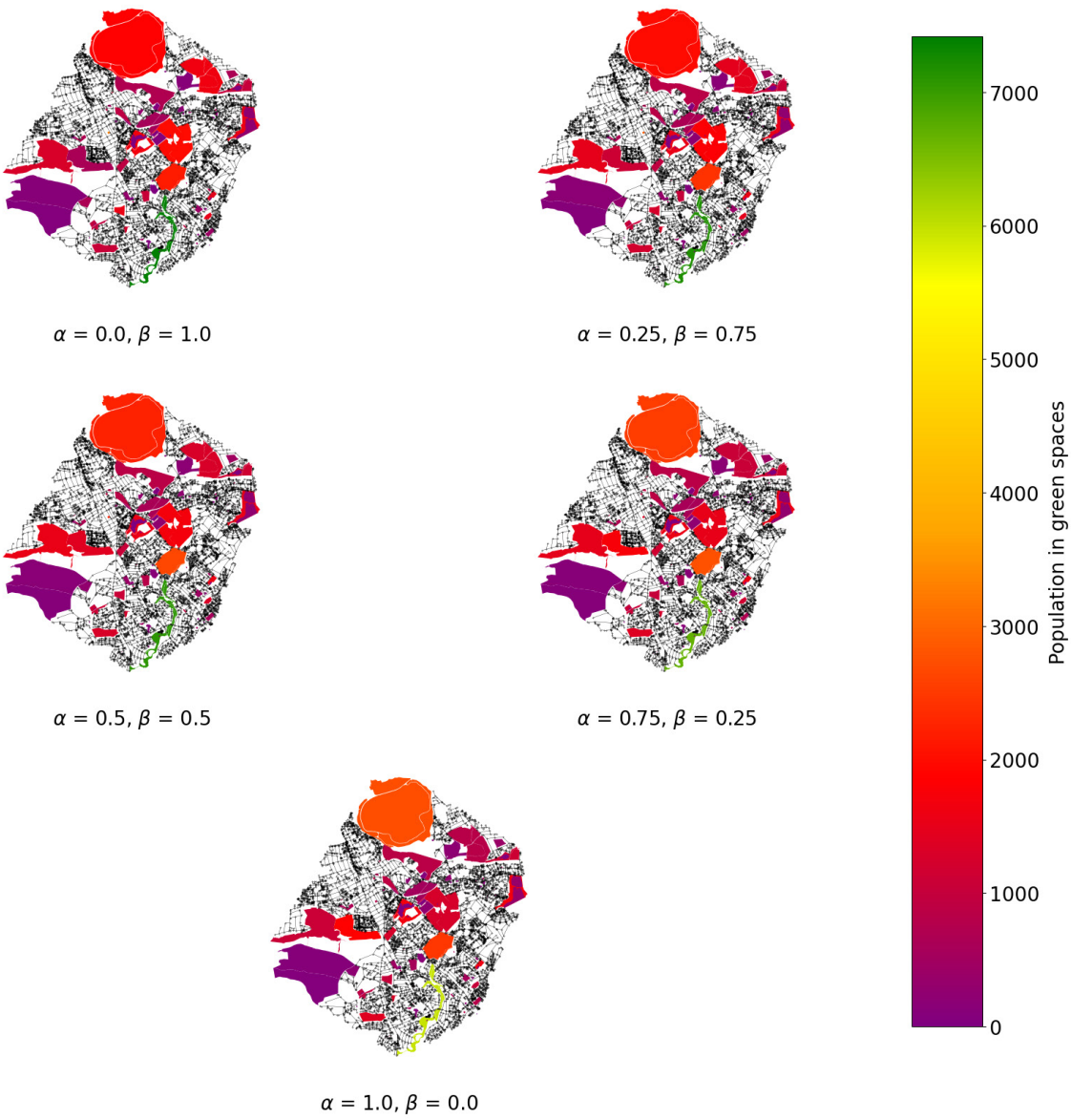


Figure 9: Distribution of population among green spaces at final time step for subgraph centred at Liberton Golf Course.

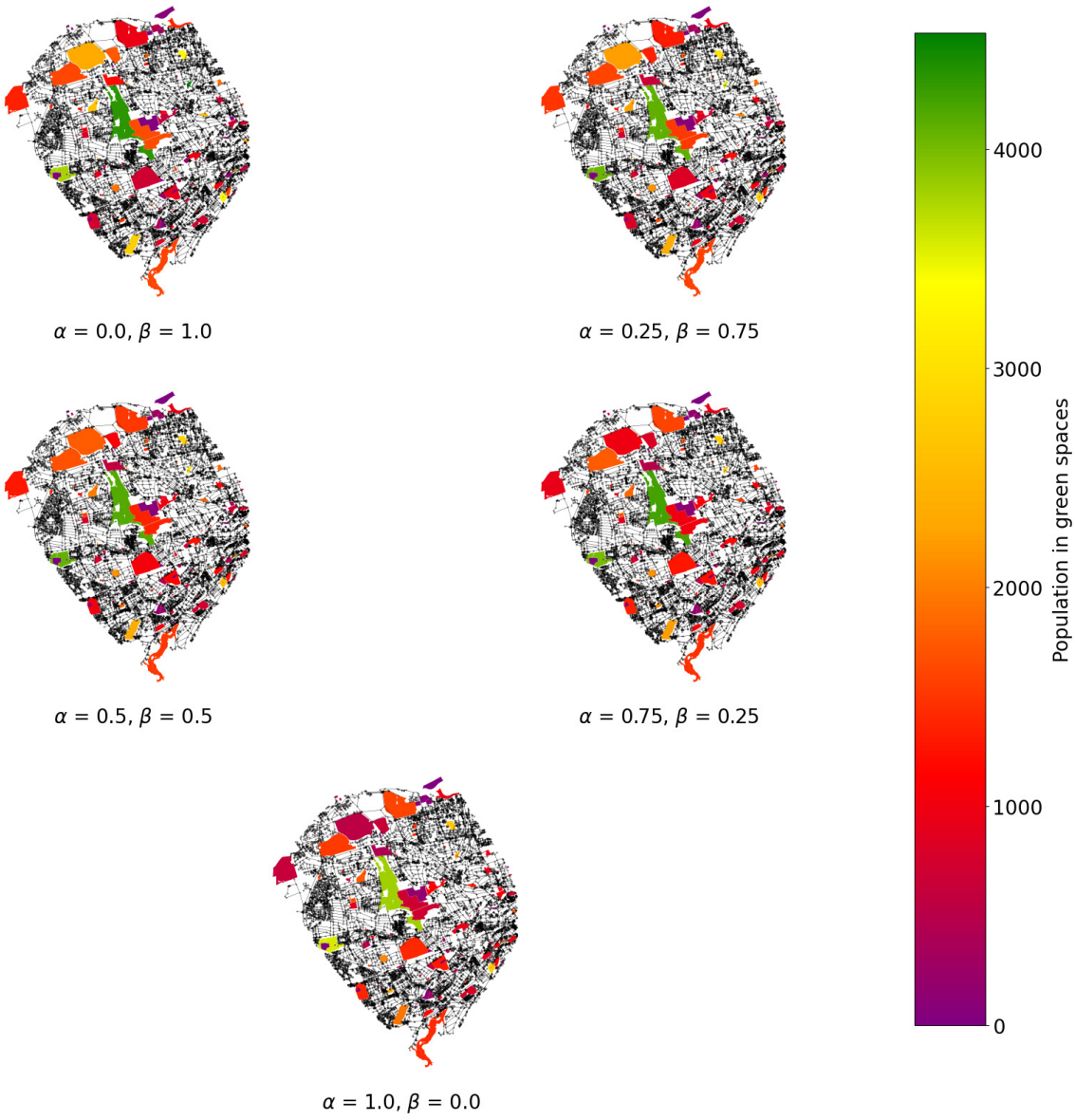
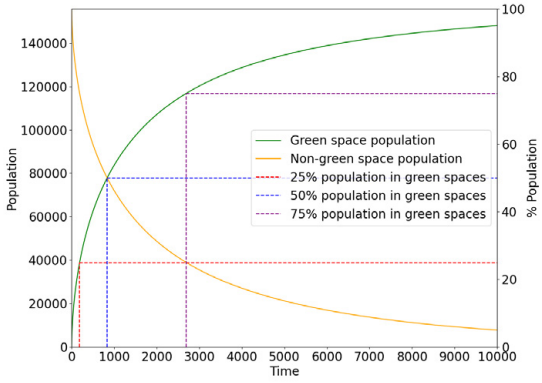
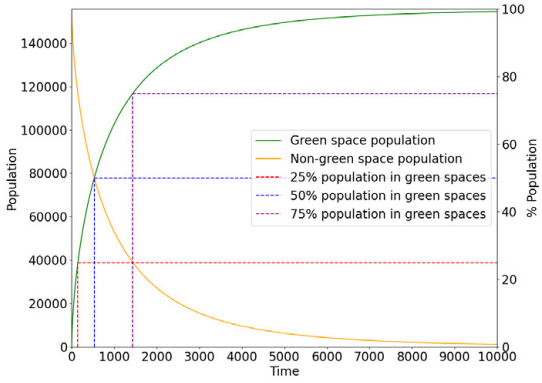


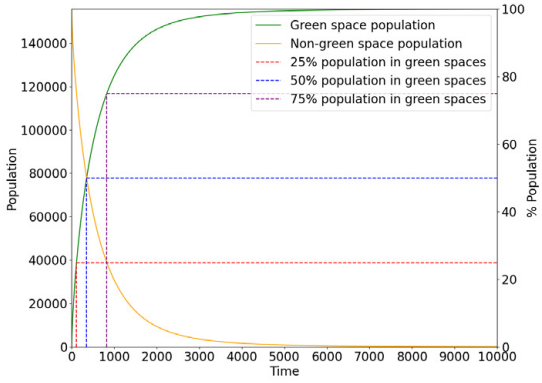
Figure 10: Distribution of population among green spaces at final time step for subgraph centred at Corstophine Hill Local Nature Reserve.



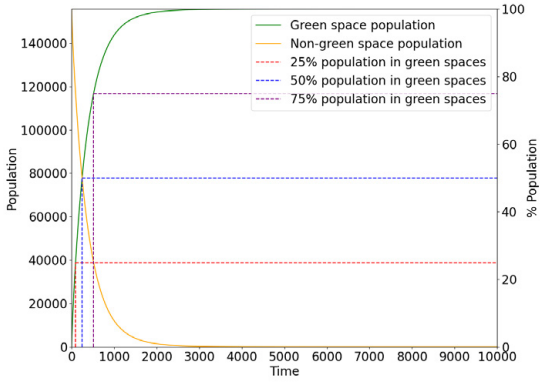
(a) $\alpha = 1,\ \beta = 0$



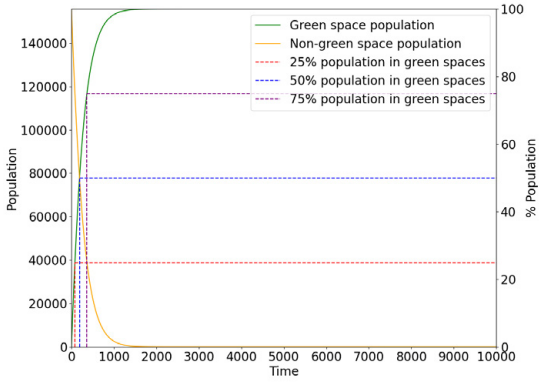
(b) $\alpha = 0.75,\ \beta = 0.25$



(c) $\alpha = 0.5,\ \beta = 0.5$

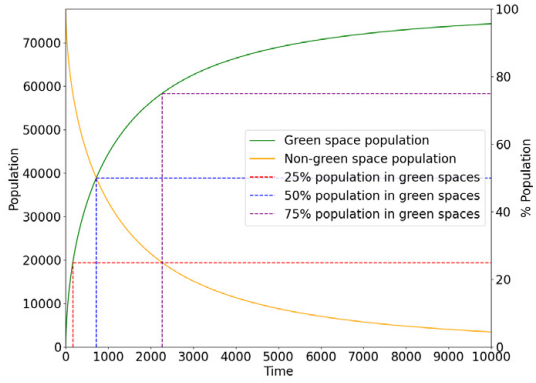


(d) $\alpha = 0.25,\ \beta = 0.75$

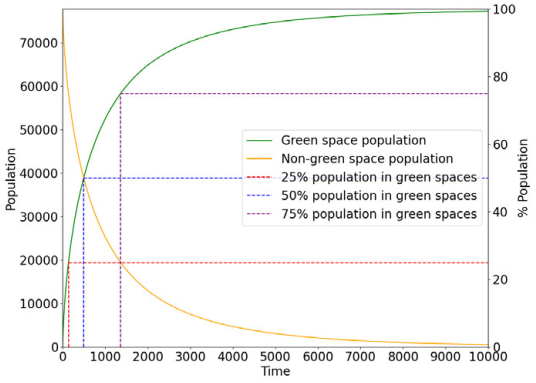


(e) $\alpha = 0,\ \beta = 1$

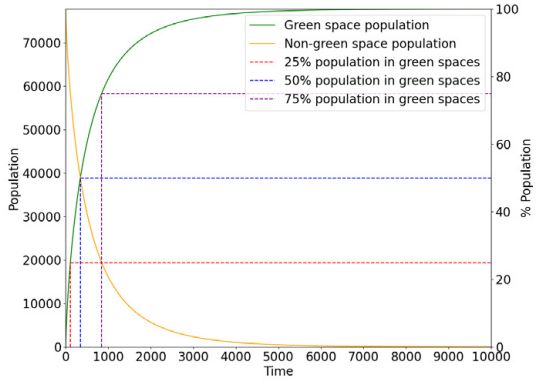
Figure 11: Diffusion of population with subgraph centred at Canongate Kirk.



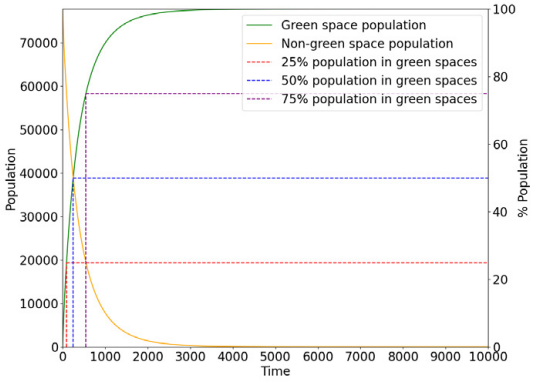
(a) $\alpha = 1.0, \beta = 0.0$



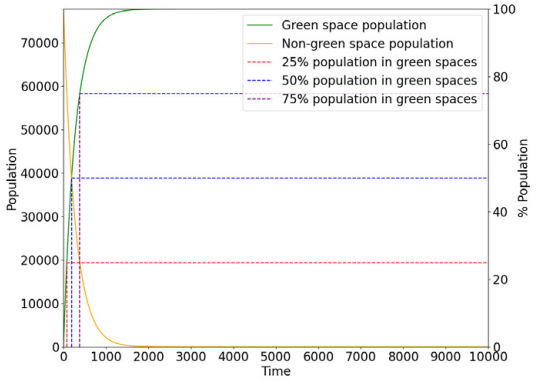
(b) $\alpha = 0.75, \beta = 0.25$



(c) $\alpha = 0.50, \beta = 0.50$

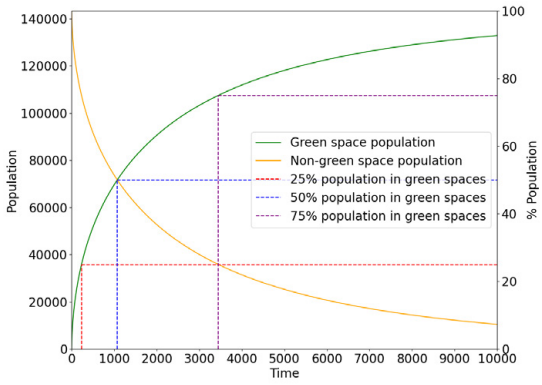


(d) $\alpha = 0.25, \beta = 0.75$

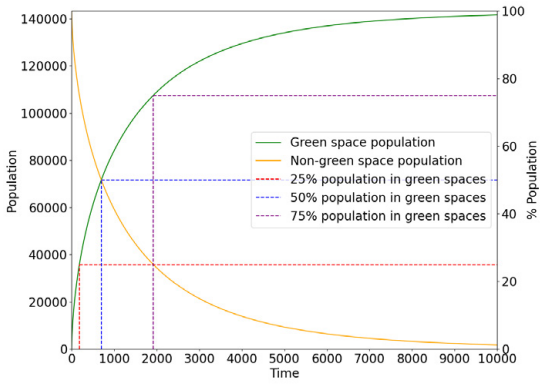


(e) $\alpha = 0.0, \beta = 1.0$

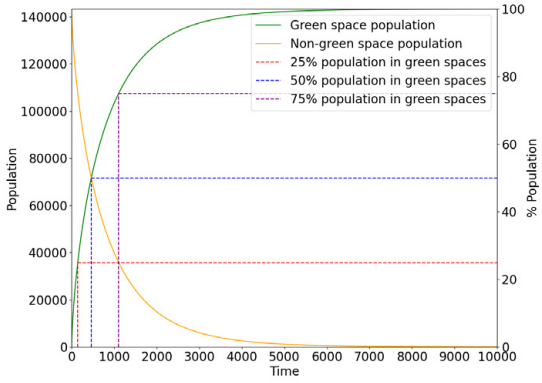
Figure 12: Diffusion of population with subgraph centred at Liberton Golf Course.



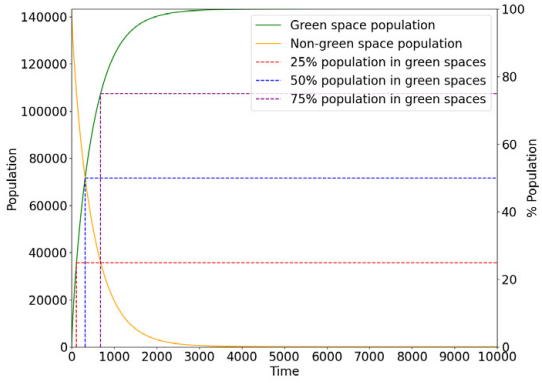
(a) $\alpha = 1.0, \beta = 0.0$



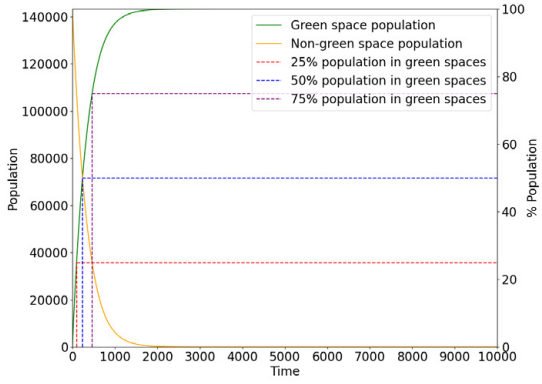
(b) $\alpha = 0.75, \beta = 0.25$



(c) $\alpha = 0.50, \beta = 0.50$



(d) $\alpha = 0.25, \beta = 0.75$

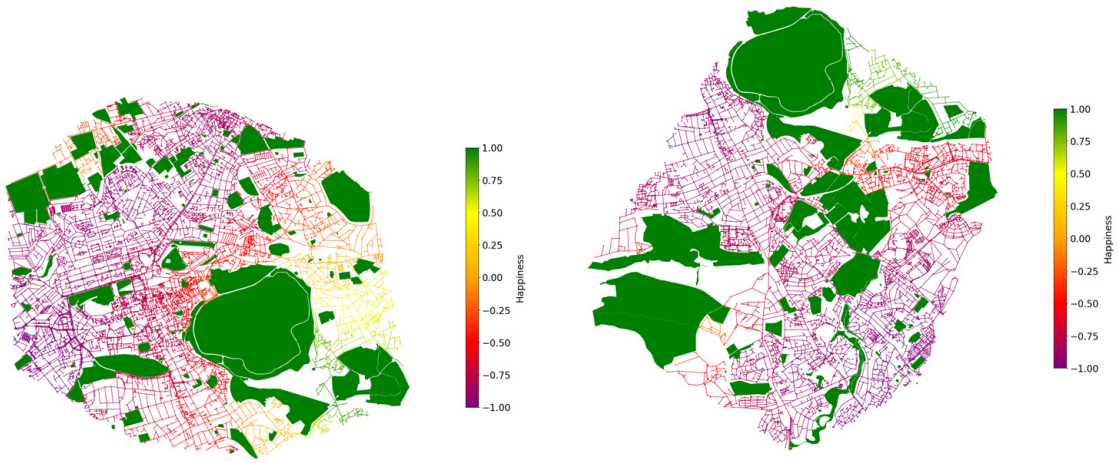


(e) $\alpha = 0.0, \beta = 1.0$

Figure 13: Diffusion of population with subgraph centred at Corstophine Hill Local Nature Reserve.

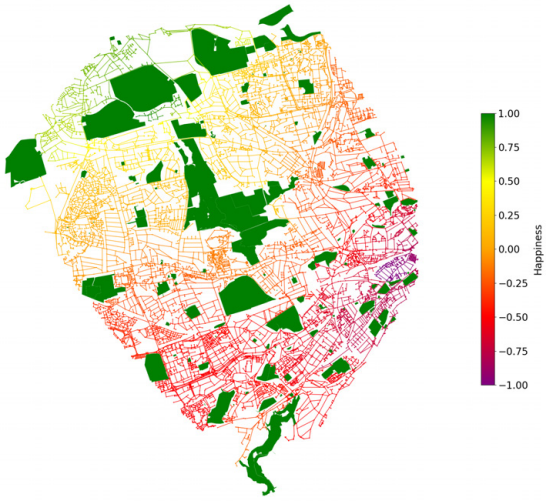
5.3.1 Sources & Sinks Diffusion: Happiness Metric

Visualisation of the results of the happiness metric on the three subgraphs is given in Fig. 14.



(a) Canongate Kirk

(b) Liberton Golf Course



(c) Corstophine Hill Local Nature Reserve

Figure 14: Implementation of the happiness metric on each subgraph. Node values give the normalised happiness value at equilibrium.

6 Discussion & Conclusions

6.1 Metrics

6.1.1 Shortest Path and Commute Distance

The shortest path implementation shown in Fig. 5 serves as a benchmark for our other metrics. As the most commonly used method in geospatial analysis, we can compare our other potential metrics to it to see if they provide more information about the structure of the city. The shortest path metric has two clear outputs: the visualisation of path lengths as a heatmap of the region, and a mean and variance of path length for the population living in that region. As far as we know, the overlay of the population data on the shortest path implementation in calculating mean and variance is not common practice, and we suggest it as a method to improve the accuracy of interpretation of shortest paths.

In particular, Canongate Kirk visually looks like it may have the worst access, as measured by the shortest path, as it has the most regions of concentrated red areas. However, on average, people living in the Canongate Kirk region have shorter paths to their nearest green spaces than those living in the Corstophine Hill area. Of note, the mean access in all three regions is less than 300m, while extreme values are above $1,200$m. This highlights the necessity of looking at both average access and worst case access in an area.

The commute distance metric, with its comparatively simple implementation and simple interpretation, seemed like a good candidate for use in conjunction with shortest path, as it reveals a different dimension of the structure of the city – namely how connected nodes are to green spaces by a multitude of paths. However, the results suggest it may be unclear how to use it. Of note, as the commute distances model random walks, the range of their values is large, and the distributions have long tails as seen in Fig. 6. The metric is likely also highly sensitive to the size of the subgraph. Liberton Golf Course appears to far out perform the other two regions, however this subgraph only has $8,260$ nodes and $10,315$ edges, compared to $13,551$ nodes and $16,094$ edges for Canongate Kirk and $13,686$ nodes and $16,747$ edges for Corstophine Hill. As such, it is hard to say if the area has better access in a random walk model, or if the fewer nodes and edges from the localisation mean that people are less able to 'become lost' for long periods.

6.1.2 Diffusion

As shown in Figs. 11 to 13, we were able to obtain an understanding of population exchange between city nodes and green spaces through diffusion with different values of α and β. In particular, it can be observed that the rate of increase in green space population is inversely proportional to β, meaning the higher the value of β, the faster the green space population increases. This is as expected since larger values of β represent larger portions of the population taking the shortest path to their closest green space from their original city nodes. Without explicitly comparing numerically, we could deduce that when $\beta = 0$, the population at each green space is proportional to the sum of the population of the city nodes having the shortest path to the corresponding green spaces.

By comparing the populations in green spaces for different values of α and β, we can also see how different green spaces end up with more or fewer people in them depending on the balance of the terms. This indicates which green spaces are more incidentally accessed versus intentionally accessed via a direct walk. Additionally, in all three subgraphs we can single out green spaces which end up with the most people, across all values of α and β. This could be an important feature to pull out of the metric, as it perhaps indicates a well placed or well connected green space. Such a green space, for example, might be valuable to expand.

Being able to predict how long it takes for certain portions of the population to come across or end up in a green space could be helpful in understanding the need of additional, or modification of current green spaces in different parts of the city. To simulate citizens' behaviours even more realistically, additional drift terms could be added to Eq. (4.22) to include other information of green spaces that could affect citizens' decision to head towards or enter them. For example, the type and size of each green space. Adding drift terms that incorporate information about common destinations, such as city centres, university campuses etc. would add another dimension to the analysis, by considering how people come across green spaces on their way to other places.

We have assumed a constant walking speed of $1.4m/s$ for the entire population. However, this could differ depending on factors including age, fitness level and incline of a street. To more accurately simulate the diffusion process, one could analyse the age distribution of each data zone and assign an average walking speed to nodes per data zone accordingly. The weight of the graph could also be modified, such that any incline is accounted for.

6.1.3 Happiness Metric

The results of the happiness metric tidily demonstrate the influence of the area of a green space on this metric. The areas around the largest park (Holyrood Park) in the Canongate Kirk and Liberton Golf course subgraphs both see the highest happiness values, with smooth gradients away from high value areas. In Corstophine Hill, we see the cluster of green spaces at the top left of the graph are associated with the highest happiness areas. This metric looks very different from either the shortest path or commute distance metrics, as it contains information not just about a single green space, but each node's relation to the many green spaces around it, and also the other city nodes around it. However, in this implementation, the influence of the area of the green spaces seems to dominate any other more local influence of adjacency to parks. We do not see, for example, clearly higher values right next to green spaces of any size, in addition to the general gradient of happiness radiating from the large areas. Fine tuning of the parameters, such as the flow rate of happiness through the edges, and scaling the source and sink vector might lead to more nuanced results.

6.2 Computational Constraints and Scalability of the Metrics

A significant bottleneck in computing our metrics was memory constraint. Let $G(V,E)$ be the full graph of Edinburgh as shown in Fig. 2, consisting of $N = 72,008$ nodes within an area of $264km^2$. Let N_g and N_{ng} denote the number of nodes belonging and not belonging to green spaces respectively, such that $N_g + N_{ng} = N$. $\boldsymbol{W}$ of G is sparse since the majority of its entries are 0, in

fact the average number of connected edges per node, k, is only 2.5. The corresponding $\boldsymbol{D}$ and $\boldsymbol{L}$ matrices are hence also sparse. We show the memory required to store the $N \times N$ matrix in Table 3. The memory required to store an $N \times N$ SciPy sparse array is given by summing the memory of the data and index of the N_{nz} non-zero entries, and index pointer of size N. The entries in a matrix are of precision float64, each occupying 8 bytes of memory, while the index and index pointer are integers, each of size 4 bytes. On the other hand, the memory required to store an $N \times N$ NumPy dense array is drastically higher. With that in mind, along with the computational complexity of the algorithms and functions we used in each metric (shown in Table 4), we discuss the problems we encountered with regards to computation of each metric.

Array size	**Array type**	**Memory Required for our Graph**	
		(Bytes Formula)	**(Memory Size for Edinburgh)**
$N \times N$	SciPy sparse array	$(8+4)N_{nz}+4N$	2.43MB
$N \times N$	NumPy dense array	$8N^2$	41.48GB

Table 3: Memory requirements

Algorithm/ function	**Metric**	**Computational complexity**
Djikstra's algorithm with Fibonacci heaps	Shortest path	$\mathcal{O}(N(N*k+N*log(N)))$
SVD of $\boldsymbol{L}$	Commute distance	$\mathcal{O}(N^3)$
Backward-differentiation formula	Diffusion family	$\mathcal{O}(N^2)$

Table 4: Computational complexity of methods used

Exploiting SciPy's shortest path function, `scipy.sparse.csgraph.shortest_path`, which is optimised for a sparse weight matrix of a graph, and specifying the use of Dijkstra's algorithm with Fibonacci heaps, the computational complexity is significantly lower than the Floyd-Warshall algorithm which costs $\mathcal{O}(N^3)$. In fact, shortest path was the only metric we were able to generate a result for using the full graph.

The computational complexity of commute distance is dominated by the computation of the pseudo-inverse of $\boldsymbol{L}$, $\boldsymbol{L}^\dagger$. Computing $\boldsymbol{L}^\dagger$ requires the singular-value decomposition (SVD) of $\boldsymbol{L}$ into $\boldsymbol{U}, \boldsymbol{\Lambda}$ and $\boldsymbol{U^T}$, and computation of $\boldsymbol{U\Lambda^\dagger U^T}$, both costing $\mathcal{O}(N^3)$. Even though we only consider the commute distance from any non-green space nodes to any green space nodes, computing the corresponding entries of $\boldsymbol{L}^\dagger$ by $l^\dagger_{ij} = \sum_{k=2}^{n} \frac{1}{\lambda_k} u_{ik} u_{jk}$ [7] is still of $\mathcal{O}(N_{ng} N_g N)$. Here, we are letting i and j be all non-green space and all green space nodes respectively. Although this greatly improves from $\mathcal{O}(N^3)$, we note that this is still lower bounded by $\mathcal{O}(N_{ng}^2)$ in the case that the graph contains only one green space node, which is unrealistic. Since all eigenvalues and eigenvectors are required, computation of commute distance is in fact dominated by the SVD of $\boldsymbol{L}$. On top of its expensive cost, computation of commute distance also suffers from memory issue since $\boldsymbol{L}^\dagger$ is not sparse. We note that NumPy's `numpy.linalg.pinv` also requires the SVD of the input matrix, and hence suffers from the same

problem when applied in our case.

Our diffusion model, regardless of the values of α and β in Eq. (4.22), are computed by the backwards differentiation method Eq. (4.21), which is dominated by the computation of $f(t_{n+1}, p_{n+1})$ by Eq. (4.20), with cost of $\mathcal{O}(N^2)$. On top of this, SciPy's `scipy.integrate.solve_ivp` does not allow for the use of sparse arrays, therefore the sparse $\boldsymbol{L}$ has be to converted to a NumPy dense array, skyrocketing the memory requirement.

An idea to compute the metrics at localised subgraphs was proposed. In general, localisation causes issues near boundaries for metrics, as boundaries artificially 'appear' to only have access to green spaces in one direction. Additionally, for metrics that have non-local properties, such as diffusion and random walks, the effect of the localisation is across the whole of the graph. Shortest path is least impacted, as we only consider the minimum values from each city node to any green space, and so it is a truly local metric. Shortest path does still suffer however at boundaries, as a boundary node may be closer to a green space that has been cut out of the localised graph than any green space in the graph. Thus, overlapping of subgraphs is likely necessary to compensate for green spaces located at the boundary of the subgraphs. For commute distance and diffusion, it is difficult to 'stitch' the results back together, as we essentially restricted the available edges people could randomly walk towards. Further investigation is required to address the localisation issue for these metrics. Another possibility for commute distance is to obtain $\boldsymbol{L}^{\dagger}$ on the whole graph by solving a least-squares problem, although implementation of least-squares using NumPy's `numpy.linalg.lstsq()` still has high memory requirements for large graphs, and may not be scalable to larger cities.

6.3 Error in Data

Since the data source of the street network used, OpenStreetMap, is provided and updated by multiple contributors using different methods, there always exists some level of error in terms of node positions and edge lengths. Besides, the current state of the city might not be reflected correctly by such a network due to road closures or re-purposing of some parts of the city. One could obtain more accurate and up to date geospatial data with satellite images, which could be expensive and restrictive in terms of accessing the data. Therefore, we propose a method to estimate the effects of such errors and temporary changes by computing the metrics with perturbed graphs. More specifically, to estimate the effect of geospatial error from noisy data sources, the graph could be perturbed by adjusting the lengths of all edges by a Gaussian random variable.

On the other hand, the graph could be perturbed by random removal and addition of nodes and edges to simulate future city planning actions. The metrics should be computed with multiple versions of perturbation of the original graph to allow comparison of the results obtained from the original graph. This would not only show the uncertainties that might be present in our metrics, but also serve as a way to predict changes to accessibility of green spaces with modification of city structure.

Different metrics may be more or less sensitive to such perturbations, which would be an important dimension to consider when choosing a metric. For example, if a metric gives very different results due to small changes in the graph structure, it should be discarded as there is certainly inaccuracy

in the model of the city used. Analysis of the robustness of each metric to uncertainties in the graph is an area of further investigation.

Similarly, the structure of the city itself may not be robust to small changes – for example, there may be a single road that connects many people to green spaces, and a road closure there would lead to dramatic changes in access. The robustness of a city to small changes to things like road closures and diversions is another dimension we would like to include in further development of green space accessibility metrics.

6.4 Conclusion

Having constructed a weighted undirected graph, where the edges are walkable streets of the City of Edinburgh and nodes are the intersections of streets, we computed distance-based and diffusion-based metrics in an attempt to understand the accessibility of green spaces within the city.

The shortest path algorithm suggested that there was an average of just under 300m to access each green spaces, however we we were also able to identify clearly extreme values that have the worst case access. We found this to be in Canongate Kirk. In comparison with the commute distance, here we found Corstophine Hill to have the worst green space accessibility. However, we found this may have differed because Corstophine Hill was the largest subgraph and therefore random walkers are more likely to get 'lost'.

By altering the graph Laplacian used in diffusion to simulate movement of people across the city, we were able to predict the time steps required for certain amounts of the population to have reached a green space. Specifically, we compared the effect of a mixture of diffusion and diffusion with shortest path influence.

Additionally, we extended our diffusion models to measure a more abstract green space accessibility metric, whereby 'happiness' diffuses from green space nodes and we interpret the influence these green spaces have across the city. We found that the largest green spaces have the highest happiness values. However, the area of the green spaces hold much weight in this model, such that the largest parks dominate over more local influence of adjacency to parks.

Following the discussion of computational constraints when computing our metrics, we proposed methods such as localisation to share the cost between multiple subgraphs. We stress that Edinburgh is a relatively small city. For comparison, New York City and London, two of the most densely populated cities in the world, are 3 times and 6 times larger in area than Edinburgh, respectively. Although their graphs can be easily generated by following the steps described in Appendix A, their number of nodes would be drastically more, highlighting the need to remediate the expensive computational cost required to compute our metrics.

References

[1] Callum Mair. *Why we need green spaces in cities*. https://www.nhm.ac.uk/discover/why-we-need-green-spaces-in-cities.html. Natural History Museum. 2018.

[2] Jennifer R Wolch, Jason Byrne, and Joshua P Newell. "Urban green space, public health, and environmental justice: The challenge of making cities 'just green enough'". In: *Landscape and urban planning* 125 (2014), pp. 234–244.

[3] Richard Burnett et al. "Global estimates of mortality associated with long-term exposure to outdoor fine particulate matter". In: *Proceedings of the National Academy of Sciences* 115.38 (2018), pp. 9592–9597. DOI: 10.1073/pnas.1803222115. eprint: https://www.pnas.org/doi/pdf/10.1073/pnas.1803222115. URL: https://www.pnas.org/doi/abs/10.1073/pnas.1803222115.

[4] UN-HABITAT and SDG. "Indicator 11.7.1 Training Module: Public Space. United Nations Human Settlement Programme". In: *United Nations Human Settlement Programme* (2018), p. 39.

[5] *Environmental Improvement Plan 2023*. https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/1133967/environmental-improvement-plan-2023.pdf.

[6] *One Million Tree City - The City of Edinburgh Council*. https://www.edinburgh.gov.uk/parks-greenspaces/one-million-tree-city.

[7] Ulrike Von Luxburg. "A tutorial on spectral clustering". In: *Statistics and computing* 17.4 (2007), pp. 395–416.

[8] Fan RK Chung. *Spectral graph theory*. Vol. 92. American Mathematical Soc., 1997.

[9] *OpenStreetMap*. https://www.openstreetmap.org. Accessed: 2023-03-10.

[10] *Ordnance Survey Open Greenspace*. https://osdatahub.os.uk/downloads/open/OpenGreenspace. Accessed: 2023-03-10. 2022.

[11] *Local Authority boundaries - Scotland*. https://spatialdata.gov.scot/geonetwork/srv/eng/catalog.search;jsessionid=432BF03B3EFC7DA676D4DB9ABE47D3A6#/metadata/890afb85-01d2-443e-be33-f3c490c02a18. Accessed: 2023-03-10.

[12] *Population Estimates Summary (Current Geographic Boundaries)*. https://statistics.gov.scot/resource?uri=http%3A%2F%2Fstatistics.gov.scot%2Fdata%2Fpopulation-estimates-2011-datazone-linked-dataset. Accessed: 2023-03-10. 2021.

[13] Zhen Zhang, Wu Jigang, and Xinming Duan. "Practical algorithm for shortest paths on transportation network". In: *2010 International Conference on Computer and Information Application*. 2010, pp. 48–51. DOI: 10.1109/ICCIA.2010.6141534.

[14] Jogamohan Medak and Partha Pratim Gogoi. "Review and Analysis of Single-Source Shortest Path Problem Using Dijkstra's Algorithm". In: *IOSR Journal of Computer Engineering* 20.2 (2018), pp. 10–15.

[15] Fan Chung and S-T Yau. "Discrete Green's functions". In: *Journal of Combinatorial Theory, Series A* 91.1-2 (2000), pp. 191–214.

[16] Mihai Cucuringu et al. “Hermitian matrices for clustering directed graphs”. In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2020, pp. 983–992.

[17] Venu Satuluri and Srinivasan Parthasarathy. “Symmetrizations for clustering directed graphs”. In: *Proceedings of the 14th International Conference on Extending Database Technology*. 2011, pp. 343–354.

[18] Jonathan J Crofts et al. “Mapping directed networks”. In: *Electronic Transactions in Numerical Analysis* 37 (2010), pp. 337–350.

[19] Zhenqi Lu, Johan Wahlström, and Arye Nehorai. “Community detection in complex networks via clique conductance”. In: *Scientific reports* 8.1 (2018), pp. 1–16.

[20] *Data Zone Boundaries 2011*. `https://www.data.gov.uk/dataset/ab9f1f20-3b7f-4efa-9bd2-239acf63b540/data-zone-boundaries-2011`. Accessed: 2023-03-10. 2021.

A Detailed Description on Data Pre-processing and Graph Construction

1. Query street network data from OpenStreetMap and import into Python as a GeoPandas dataframe.
2. Filter out any streets not within the boundary of the city from the dataframe.
3. Add a buffer of 1m to each street geometry to ensure intersections between streets.
4. Remove any streets deemed not walkable by OpenStreetMap's definition.
5. Check if the last coordinates for each street geometry are the same as its first, and remove the last coordinates if this is true.
6. Construct a NetworkX undirected graph from the GeoPandas dataframe using Momepy, where each node represents intersections of any two streets, and each edge represents a street or part of a street between intersections.
7. Initialise attributes 'gs_bool' as False, 'gs_id' as an empty list and 'gs_entrance' as False for each node. These attributes represent whether the node belongs to any green space, a list of each green space's ID the node belongs to, and if the node is considered an entrance to a green space, respectively.
8. For each green space, add new nodes to the graph at intersections of any edge and the green space's boundary if they do not already exist.
9. Assign 'gs_bool' as True, and pass this to the green space's ID. This attribute labels each node as a green space node or as a city node otherwise.
10. Check iteratively if each node's coordinates are within any green space boundary, and if so change 'gs_bool' to True and append the green space's ID to the 'gs_id' list.
11. For green spaces without any nodes within them, a buffer that increased in increments of 0.5 from 0.5m up to 5m was introduced to extend their boundary, and the previous step was repeated.
12. For any green space nodes connected to city nodes, change their 'gs_entrance' attributes to True.
13. Assign each node to a data zone according to their coordinates and the boundary of the data zone.
14. Uniformly distribute the population of each data zone to city nodes within them.

B Code

The code used for this project is available here `https://github.com/Taylor-Chu/geospatial_analysis`.