Utrecht University

GreenExp: An open-source R package for modelling multidimensional greenspace exposure

Martijn Koster
6234119

Supervised by:
Dr. S.M. Labib (Utrecht University)

A dissertation submitted in partial fulfilment of the requirements for the degree of
Master of Science in Applied Data Science
Utrecht University
June 2023

Acknowledgements

I express my deepest gratitude to my supervisor, Dr S.M. Labib, for his guidance, support, and valuable insights throughout the research process. His expertise and dedication have been instrumental in shaping this thesis.

I would also like to thank my girlfriend, friends and family for their support and encouragement. Their belief in my abilities has been a constant source of motivation.

Finally, I am also grateful to the participants of Spatial-Data-Science-and-GEO-AI-Lab for their willingness to contribute their time and insights; without them, this research would not have been possible.

Abstract

Numerous studies have shown the positive impact of greenspace exposure on physical and mental health. However, concerns arise regarding the quality and comparability of studies in this field. To address these issues, this article introduces GreenExp, an open-source R package for modelling multidimensional greenspace exposure. GreenExp follows a logical flow, offering comprehensive functionalities related to greenspace availability, accessibility, and visibility. It adheres to FAIR(4RS) principles, promoting transparency and enhancing research quality in the field.

GreenExp is a software package developed using various open and free software tools. The software architecture of GreenExp consists of five components: Data Acquisition, Data Processing, Modelling, Visualisation, and Interface. Each component plays a specific role in analyzing green exposure. GreenExp offers eight functionalities related to the availability, accessibility, and visibility of greenspaces.

To provide a comprehensive understanding of the functions in GreenExp, illustrative examples show code snippets, figures, and excerpts from the interface.

GreenExp enhances transparency in greenspace studies by providing detailed information and insights into the modelling processes. It introduces visibility as a component of green space exposure, along with availability and accessibility. GreenExp incorporates open source and global data, overcoming the limitations of proprietary software. However, it has limitations related to data capacity, comprehensive accessibility analysis, availability of required models, and computation speed. Overall, GreenExp contributes to spatial analysis but requires further development.

Contents

List of Abbreviations

The following table describes the significance of various abbreviations and acronyms used throughout the thesis—the page on which each one is defined or first used is also given.

Abbreviation	**Meaning**	**Page**
CRS	Coordinate Reference System	7
CRAN	Comprehensive R Archive Network	6
DSM	Digital Surface Model	12
DTM	Digital Terrain Model	12
EPSG	European Petroleum Survey Group	7
ESRI	Environmental Systems Research Institute	5
FAIR	Findable, Accessible, Interoperable, Reusable	5
FAIR4RS	FAIR for researchers	5
GEE	Google Earth Engine	10
GIS	Graphical Information System	5
GVI	Greenness Visibility Index	5
GUI	Graphical User Interface	5
GreenExp	green exposure	6
KNN	K-Nearest Neighbours	11
NDVI	Normalized Difference Vegetation Index	7
OSM	OpenStreetMap	7
PC	Planetary Computer	7
POI	Point of Interest	7
STAC	Spatio Temporal Asset Catalog	7
VGVI	Viewshed Greenness Visibility Index	13
WHO	World Health Organization	11
WGS 84	World Geodetic System 1984	7

1 Motivation and significance

Numerous studies have recognised the connection between health and natural surroundings. Researchers such as Remme et al., 2021 and Frumkin et al., 2017 have emphasised this relationship. Several systematic reviews have indicated that contact with nature, especially exposure to green and blue areas (together referred to as "greenspaces"), have a beneficial effect on physical and mental health and well-being (Bowler et al., 2010; Gianfredi et al., 2021; Haluza et al., 2014; Lee & Maheswaran, 2011; Markevych et al., 2017).

Existing literature commonly categorises objectively measured greenspace exposure in three core principles; availability, accessibility, and visibility (Dadvand & Nieuwenhuijsen, 2019; Labib et al., 2020; Zhang et al., 2021). Availability exposure refers to the physical extent of greenspace, like the total area of tree coverage within a specified zone around homes, schools, or other relevant locations of interest (Bratman et al., 2019; Dadvand & Nieuwenhuijsen, 2019; Zhang et al., 2021). An objective measure of accessibility focuses on the spatial proximity of greenspace to areas of interest (Ekkel & de Vries, 2017). It examines how easily one physically reaches and engages greenspaces. The third principle, visibility, assesses the visual exposure to greenspace from a particular location of interest. It measures the amount of greenery within streetscapes and is commonly quantified using the greenness visibility index (GVI) or greenness visibility measures (Labib et al., 2021; Zhang et al., 2021).

Although the significance of greenspace exposure is widely acknowledged, concerns arise regarding the quality and comparability of studies within this field (Cardinali et al., 2023; Labib et al., 2020; Twohig-Bennett & Jones, 2018). The research landscape exhibits considerable diversity in study designs and exposure assessment methods. The study by Taylor and Hochuli, 2017 revealed that published research must clearly define greenspace. When researchers provide a definition, there is a variation in the report. Moreover, there is a lack of consistency and methodology in defining and studying greenspace and health relations, which hinders the ability to compare findings across studies. Establishing a meaningful definition of greenspace that accurately qualifies and quantifies what the term means will make the research more consistent (Matsler et al., 2021; Morpurgo et al., 2023). Furthermore, there is a need for improved transparency in greenspace studies regarding geospatial and analytical methods. Increased transparency can be achieved through the rigorous definition and reporting of indicators and contextual variables, which will facilitate understanding and enhance the interdisciplinary nature of this field (Browning et al., 2022; Collins et al., 2020; Markevych et al., 2017).

Such issues of the inconsistency of using different methods and data also resulted in a lack of adherence to the FAIR and FAIR4RS (FAIR for Research Software) framework in greenspace and health studies, which emphasises the importance of making data and software: findable (F), accessible (A), interoperable (I), and reusable (R). These frameworks provide guidelines and best practices to ensure that research data and software are correctly managed, documented, and shared, fostering transparency, reproducibility, and collaboration in scientific endeavours (Barker et al., 2022; Wilkinson et al., 2016). In the case of greenspace research, some issues prevent adherence to these principles. Firstly, inadequate reporting of retrieval methods in greenspace studies comprises research reproducibility and usability. Transparent reporting is crucial for research reproducibility and facilitating other researchers' reuse of research data. When vital information about data retrieval methods is missing, the accessibility and usability of the research are compromised. Secondly, the greenspace research landscape shows considerable diversity regarding study designs and techniques to assess greenspace exposure. The majority of the existing research uses graphical user interface (GUI) based proprietary geospatial software such as ESRI's ArcGIS products or free, open-source software such as QGIS to objectively analyse greenspace exposure (Cardinali et al., 2023; Labib et al., 2020). While GUI software is user-friendly, using such software result

in difficulty in reproducing the same results again, as exact information about the steps the analyst followed during analyses are often not reported extensively. Moreover, suppose proprietary software is used for analysis. In that case, the analytical process remains a black box within the software code, and the files generated are often not interoperable with other software. This lack of consistency and interoperability makes it challenging to compare and combine the results of different studies.

To make significant progress in the field, it is imperative to promote data consistency, methodological transparency, and collaboration among researchers, ultimately leading to evidence-based policies and interventions that optimise the health benefits of greenspaces. Therefore this study introduces the Green Exposure (GreenExp) package, an open-source R tool for conducting multidimensional spatial analysis in modelling diverse greenspace exposure variables usually used in public health research. Next, *GreenExp* incorporated several flexibilities to improve greenspace exposure assessment in multiple ways. Firstly, *GreenExp* will include spatial data like higher resolution images, which can provide a more accurate representation of greenspaces (Labib et al., 2020; Markevych et al., 2017). In addition, the package will enable researchers to make informed decisions regarding the selection of buffer distances and scales of analysis. This feature is essential because varying scales and buffer distances can lead to varying research outcomes.

2 Software description

The *GreenExp* package is developed to facilitate robust and transparent analyses in greenspace and health research. It provides researchers with a collection of functionalities that can be applied across multiple countries, enabling comprehensive spatial analysis and enhancing the accuracy of results. In the following subsections, we will provide a detailed description of *GreenExp*, highlighting its key features and architecture.

2.1 Software architecture

To develop *GreenExp*, we have used multiple open and free software, like R packages such as devtools, terra and sf (Hijmans, 2023; Pebesma, 2018; Wickham et al., 2022), among others. Table 1 provides an overview of all packages that have been instrumental in developing *GreenExp*.

Installation

Since the package is not published in the Comprehensive R Archive Network (CRAN), it can be installed by executing code snippet 1.

```
devtools::install_github("Spatial-Data-Science-and-GEO-AI-Lab/GreenExp_R")
```

Code Snippet 1: Installing GreenExp

For users encountering any difficulties during the installation process, additional guidance and support can be found in our GitHub repository. Once the installation has succeeded, *GreenExp* can be loaded into an R session using the following code:

```
library(GreenExp)
```

Code Snippet 2: Load GreenExp

Following these steps, will install the necessary components and leverage the capabilities of *GreenExp* for green exposure modelling.

Table 1: Packages used in GreenExp

Package	Description	Source
sf	Simple features for R	(Pebesma, 2018)
terra	Creating, reading, manipulating, and writing raster data	(Hijmans, 2023)
sfnetworks	Tidy Geospatial Networks in R	(van der Meer et al., 2023)
osmdata	Provides access to the vector data underlying OpenStreetMap	(Mark Padgham et al., 2017)
osmextract	Download and import OpenStreetMap Data Extracts	(Gilardi & Lovelace, 2022)
dplyr	Data manipulation	(Wickham, François, et al., 2023)
magrittr	Forward-Pipe Operator	(Bache & Wickham, 2022)
rgee	Calling Google Earth Engine API	(Aybar, 2023)
rstac	Search and download spacetime earth observation data via Spatio Temporal Asset Catalog (STAC)	(Simoes et al., 2021)
FNN	Fast Nearest Neighbor Search Algorithms	(Beygelzimer et al., 2023)
tidygraph	A tidy API for graph/network manipulation	(Pedersen, 2023)
tidyr	Changing the shape and hierarchy of a dataset	(Wickham, Vaughan, et al., 2023)
Rcpp	R and C++ integration	(Eddelbuettel & Balamuta, 2018)
progress	Make a progress bar in loops	(Csárdi & FitzJohn, 2019)

Architecture Flow

The software architecture of the functionalities in *GreenExp* follows a logical flow encompassing five key components: Data Acquisition, Data Processing, Modelling, Visualisation, and Interface. The steps of the architecture are also shown in Figure 1

Data Acquisition. In the initial step, the user provides Points of Interest (POIs) with projected coordinates. These POIs, representing addresses, neighbourhoods, or other areas of interest, are essential inputs for all functionalities. In addition, supplementary data are required in the function to model the greenspace exposure. Within our package, users can provide data they have obtained independently or utilise data extracted from external sources like OpenStreetMap (OSM) or Planetary Computer (PC). This functionality expands the options available to users when working with our package.

Data Processing. In the data processing stage, the integrity and compatibility of the provided data are ensured. All data used in the analysis must have the exact Coordinate Reference System (CRS). In cases where the user did not provide the data in projected coordinates, the functions will automatically convert it to WGS 84 / World Mercator, which corresponds to European Petroleum Survey Group (EPSG) 3395. Consistency in projected coordinates is critical for accurate calculations because mismatched projected coordinates lead to erroneous results. In addition, data will be cropped into a specific area of interest to optimise computational efficiency.

Modelling. The modelling stage involves calculations based on the processed data. Various algorithms and methods are employed to estimate and quantify green exposure, given the POIs. For example, the Normalized Difference Vegetation Index (NDVI) in the PC is calculated based

on land surface reflection of visible (red) and near-infrared parts of the spectrum (Ekkel & de Vries, 2017). Furthermore, in accessibility functions, distances will be calculated from POIs to greenspaces. The software functionality section will give a more comprehensive description of the calculations used.

Visualisation. Visualisation options will be included in the functionalities' arguments to improve the results' understanding and interpretation. We designed these visualisations to be user-friendly and intuitive. In the functionalities, the user can either download the converted data points used in the analysis or plot the data used in the analysis (e.g., satellite images). The software functionalities subsection will discuss how to download or plot the data.

Interface. The interface offers a wide range of features and functionalities, allowing users to input parameters and settings related to the modelling process, like buffer distances, buffer types, and spatial reference. These features ensure flexibility and customisation in the analysis. Additionally, the interface ensures transparency by clearly indicating the sources of information used in the modelling process.

These components form the key architecture for developing the functionalities, which we will describe in the following subsection.

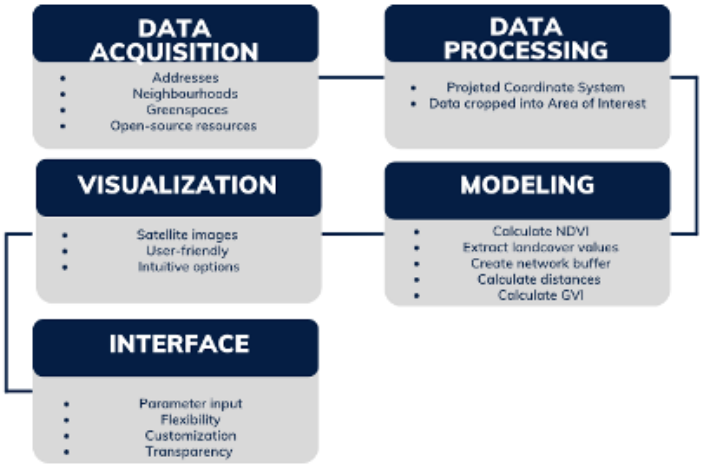


Figure 1: Architecture flow of GreenExp

2.2 Software functionalities

The *GreenExp* package offers a comprehensive set of functionalities focused on three major subjects: Availability, Accessibility, and Visibility. Each subject encompasses a range of specific functionalities that contribute to analysing greenspaces. To provide an overview, Table 2 outlines the functionalities corresponding to each subject to give an overview. The following sections will delve into each functionality, describing its purpose and arguments.

2.2.1 Availability

Table 2 shows that *GreenExp* assesses availability using four functions. Each availability function will yield a standard R simple feature data frame (Pebesma, 2018), including the specific values requested within a defined buffer. By default, the availability functionalities will create a Euclidean buffer around the POIs. However, it can be modified to utilise a network buffer. Euclidean buffers are generated by drawing a straight line from a specific location and using that line as the radius of a circle (Berke et al., 2007), where network buffers are irregular shapes formed by drawing line segments from a given location along a road network, including a specified distance along the

Table 2: Functionalities in the GreenExp package

Subject	Functionality
Availability	Calc_ndvi Land_ct Canopy_pct
Accessibility	Greenspace_access
Visibility	Viewshed VGVI_from_sf VGVI_from_address

network from that location (Forsyth et al., 2012). Figure 2 illustrates the distinction between the two types of buffers. In this instance, the Euclidean buffer (presented in blue) has a fixed radius of 1000 meters, while the network buffer (depicted in red) is calculated based on a speed of 5 km/h over 10 minutes. If the user wants to use a network buffer without a network file, osmextract will download the network file (Gilardi & Lovelace, 2022). This package matches, downloads, converts, and imports OSM data hosted by providers such as geofabrik GmbH and bike (Ibarra-Espinosa, 2020; Schneider, n.d.). Next to the Euclidean and network buffer, the user can use the input data as a buffer; this is only possible when the user provides a polygon layer.

Figure 2: Different Buffers for availability

Before diving into the function-specific arguments, these arguments are available for all availability functionalities:

- **address_location:** A spatial object representing the location of interest, the location should be in projected coordinates.
- **buffer_distance:** A distance in meters to create a buffer or isochrone around the address location. If missing, it is optional to calculate the buffer distance with speed and time.
- **network_buffer:** A logical value. The default is an Euclidean buffer, but a network buffer will be used when set to TRUE.
- **network_file:** An optional sf network object representing a road network; road network will be created if missing.
- **address_location_neighborhood:** A logical, indicating whether to calculate with an address point or a neighbourhood. Default is FALSE
- **epsg_code:** Optional: An EPSG code to get a preferred Projected CRS in the final output.

- **speed:** A numeric value representing the speed in km/h to calculate the buffer distance (required if time is provided)
- **time:** A numeric value representing the travel time in minutes to calculate the buffer distance (required if speed is provided)

Next, the function-specific arguments will be discussed.

calc_ndvi(). The function calc_ndvi() determines the mean NDVI within a given distance for one or more specified locations. The NDVI indicates greenness based on land surface reflection of the spectrum's visible (red) and near-infrared parts. Its values range between 1 and +1, with higher values indicating more greenness (Ekkel & de Vries, 2017). The user has the option to provide a raster file that contains NDVI values. If the user provides no raster file, the function will retrieve a raster file from either PC or Google Earth Engine (GEE) (Aybar, 2023; Gorelick et al., 2017; Simoes et al., 2021; Source et al., 2022). Next to the average NDVI, the user can opt to calculate the standard deviation of the NDVI within the buffer. The following arguments are specific to this function:

- **raster:** A raster file with NDVI values,
- **add_sd:** Returns the standard deviation of the NDVI values within the selected buffer.
- **folder_path_ndvi:** Folder path to where the retrieved NDVI values should be saved.
- **plot_NDVI:** If TRUE, an image of the raster will be plotted.
- **engine:** Which engine the user want to retrieve the images from, PC or GEE.

land_cover(). This function calculates the percentage of land cover types within a specified distance for a given location(s). As in the NDVI function, this function requires a raster file representing the land cover or land use. If the user did not provide a land cover raster, the ESA WorldCover dataset from PC would automatically retrieve land cover data (Zanaga et al., 2021). The arguments that follow are specifically relevant to this function.

- **raster:** A raster file with land cover values,
- **folder_path_land_cover:** Folder path to where the retrieved land cover values should be saved.
- **plot_landcover:** If TRUE, an image of the raster will be plotted.

canopy_pct(). The Canopy percentage determines the extent of the canopy within a specified buffer. The user must provide an alternative layer, the **canopy_layer**, to utilise this function. This additional argument should ideally be a spatial object, preferably in the exact projected coordinates as the POI(s).

greenspace_pct(). The greenspace percentage function calculates the percentage of greenspace within a given buffer from a vector data layer. This function requires input from a greenspace vector layer (e.g., ESRI shape file, GeoJSON, GeoPackage). If the user does not provide a greenspace layer, the function automatically retrieves greenspace data using the osmdata package (Mark Padgham et al., 2017). The retrieved features from OSM are selected from the leisure, nature, and land use categories.

To ensure the relevance of the extracted greenspace features, they need to adhere to specific criteria, as outlined in Breekveldt and Labib, 2023. These criteria include the following requirements: the feature represents an area, the area is outdoors, the site is (semi-)publicly accessible, the area is likely to contain trees, grass, or other greenery, and the place is suitable for walking or recreational activities. Table 3 provides an overview of the extracted OSM features that meet the criteria above and are considered representative of greenspaces.

The following arguments are explicitly used in the greenspace percentage function:

- **greenspace:** A file with greenspaces,
- **folder_path_greenspace:** Folder path to where the retrieved greenspaces should be saved.

Table 3: Values used for greenspaces in OSM

Key	**Value**
Landuse	Allotments Forest Greenfield Village green
Leisure	Garden Fitness-station Nature reserve Park Playground
Natural	Grassland

2.2.2 Accessibility

The accessibility feature of *GreenExp* focuses on determining the distance to the nearest greenspaces from given address locations using a greenspace vector layer.

By default, the function searches greenspaces within a Euclidean buffer around the address locations and calculates the shortest distance to the centroid of the greenspaces. The K-nearest Neighbours (KNN) algorithm calculates the Euclidean distance between the address location and the greenspaces (Beygelzimer et al., 2023).

Besides the Euclidean buffer, the software offers the option to utilize a network buffer. When the user decides to use the network buffer, the function performs a distance calculation using the sfnetworks package, which leverages road networks to calculate distances between points (van der Meer et al., 2023).

Additionally, pseudo-entry points can be employed to calculate the distance to the greenspaces. These pseudo entrances are created by generating a 10-meter buffer around the greenspace polygons and intersecting them with the network nodes obtained from the intersection of the network points with the greenspaces.

Furthermore, the accessibility function allows users to select a minimum size for green spaces. Adapting the size is implemented because different organizations have varying guidelines for determining the minimum size of urban greenspaces. For instance, the World Health Organization (WHO) recommends that residents access at least 0.5-1 hectare of green space within a 300-meter radius of their homes (Organization, 2017). On the other hand, the European Urban Atlas guidelines propose a minimum area of 0.25 hectares for urban green spaces (Europea, 2011). Additionally, Natural England has established the Accessible Natural Greenspace Standard for England, which provides guidelines based on the distance from one's home address (England, 2010). By leaving the size of the greenspace open, users have the flexibility to determine the minimum size required for their analysis.

The greenspace_access() function introduces the following unique arguments:

- **address_location:** A spatial object representing the location of interest. The location should be in projected coordinates.
- **greenspace:** A spatial object representing greenspace.
- **buffer_distance:** A distance in meters to create a buffer or isochrone around the address location. The buffer distance can be calculated based on speed and time if not provided.
- **network_file:** A sfnetwork object representing a road network. If not provided, the road network will be created.
- **epsg_code:** An EPSG code to obtain a preferred Projected CRS in the final output.
- **speed:** A numeric value representing the speed in km/h to calculate the buffer distance (required if time is provided).
- **time:** A numeric value representing the travel time in minutes to calculate the buffer distance (required if speed is provided).
- **euclidean:** Determines whether the distance between the pseudo-entry points and the address location is Euclidean or calculated with the network.
- **pseudo_entrance:** Allows the distance calculation to create pseudo entrances.
- **minimum_greenspace_size:** The minimum size of the greenspace (in square meters).
- **folder_path_network:** Folder path to where the retrieved network should be saved.
- **folder_path_greenspace:** Folder path to where the retrieved greenspaces should be saved.
- **folder_path_lines:** Folder path to where the shortest distance lines should be saved.

Using the greenspace_access() function with the appropriate arguments, users can assess the accessibility of greenspaces concerning specific address locations, either through Euclidean or network distances.

2.2.3 Visibility

GreenExp also provides functions to model eye-level greenness visibility for any given location. The visibility of greenspace refers to the amount of greenness that can be seen visually from a particular place of interest. As was shown in Table 2, viewshed, vgvi_from_sf, and vgvi_from_address are the functions that will assess the visibility from certain POIs.

The GVI is calculated using a Digital Surface Model (DSM), Digital Terrain Model (DTM) and Greenness Raster. GVI is written in C++ to provide fast and light weighted functionality. Most of the code in the visibility is based on the algorithm of Labib et al., 2021 and code written in the GVI package (Brinkmann & Labib, 2022).

Before diving into the function-specific arguments, these arguments are available for all availability functionalities:

- **dsm_rast:** object of class Rast;
- **dtm_rast:** object of class Rast
- **max_distance:** Buffer distance to calculate the viewshed
- **observer_height:** Height of observer in meters

Viewshed. Viewshed calculates a binary geographical area visible from a location by analysing a DSM raster from a specific point. It applies a radial buffer around the observer's position and determines visibility using Bresenham's line algorithm (J. E. Bresenham, 1965; J. Bresenham, 1977), implemented in C++. The algorithm decides which areas are visible and which are not. The output of the viewshed function is a radial raster where 0 represents non-visible areas, and 1 illustrates visible areas. The main component of the function is implemented in the GVI package Brinkmann and Labib, 2022, with slight modifications made to enhance the robustness of data usage.

The following arguments are used in viewshed:

- **observer:** object of class sf with one point
- **plot:** Plot the intersection of the buffer around the observer location and the DSM.

vgvi_from_sf() The Viewshed Greenness Visibility Index (VGVI) is a metric that represents the proportion of visible greenness to the total visible area based on the viewshed. The estimated VGVI values range between zero and one, where zero means no green cells, and one indicates that all visible cells are green. The VGVI values are based on a viewshed and a binary greenness surface raster. The viewshed classifies points as visible or not visible, whereas the greenness surface raster classifies these points as green or not green. Both classify all visible points as visible green and visible no-green. A decay function is applied to summarise the values to accurately represent the visual prominence of objects in space as they move farther away from the observer. Two types of decay functions are currently supported: a logistic function and an exponential function. These functions account for the diminishing visual impact of objects with increasing distance. As viewshed (), vgvi_from_sf() is based on the algorithm of Labib et al., 2021, and the code is based on the GVI package (Brinkmann & Labib, 2022), with alterations in the robustness of the usage of the data.

The following arguments are used in vgvi_from_sf():

- **observer:** An object of class sf with observer location(s)
- **greenspace_rast** Object of class Rast
- **mode:** whether The decay weight should be logit or exponential
- **folder_path:** Optional; folder path to where the VGVI should be saved

vgvi_from_address() The VGVI from the address function employs a broader approach. It samples multiple points around the address location within a defined buffer. This buffer represents a circular area around the address. The function collects data from various points within this buffer and calculates the VGVI by taking the mean of the collected values. Incorporating multiple sample points offers a more comprehensive representation of the VGVI within the vicinity of the address.

The VGVI from sf function analyses the VGVI at a specific observer point. In contrast, the VGVI from the address function expands the analysis by sampling multiple points around the address location. The latter approach captures the GVI within a defined buffer and provides a more averaged assessment of the VGVI.

The following arguments are explicitly used in VGVI from the address:

- **address:** Object of class sf with address location(s)
- **greenspace_rast** Object of class rast

- **buffer_distance** A distance in meters to create a buffer around the address location
- **sample_points** The number of points that should be sampled in the buffer created around the address to calculate the mean VGVI.
- **mode:** Whether the decay weight should be logit or exponential
- **folder_path** Optional; Folder path to where the mean VGVI should be saved
- **folder_path_random_points** Optional; folder path to where the VGVI of all (random) points should be saved

3 Illustrative examples

To provide a comprehensive understanding of the functions in *GreenExp*, this section presents illustrative examples through code snippets, figures, and excerpts from the interface. A more detailed overview of the function outputs and figures can be found in the README file in our GitHub repository. We have derived examples from various neighbourhoods in Amsterdam, namely Rapenburg, Uilenburg, Valkenburg, Marine-Etablissement, Kadijken, Plantage, Kattenburg, Wittenburg, and Oostenburg. The data for these neighbourhoods are extracted from open geo data in Amsterdam (Gemeente Amsterdam, 2023), which is also available in *GreenExp* as 'Ams_Neighborhoods'. In code snippet 3, we show how the data for the examples are created. Note that we have used the centroid of the neighbourhood polygon for the address locations.

```
library(magrittr)
neighbourhoods <- c('Rapenburg', 'Uilenburg', 'Valkenburg',
            'Marine-Etablissement', 'Kadijken', 'Plantage',
            'Kattenburg', 'Wittenburg', 'Oostenburg')

# Filter the neighbourhoods
df <- Ams_Neighborhoods %>%
  dplyr::filter(Buurt %in% neighbourhoods)

# Create data points
df_points <- sf::st_centroid(df)
```

Code Snippet 3: Creating examples data

3.1 Availability

First, we will delve into the Availability functionalities. The maps of the results, which we made with tmap and mapview (Appelhans et al., 2022; Tennekes, 2018), can be found in Figure 3a.

3.1.1 NDVI

Calc_ndvi() focuses on calculating NDVI. In this scenario, we have provided the calculation of NDVI for neighbourhood buffers. Code snippet 4 shows how the mean NDVI is calculated with the neighbourhood buffer.

```
# Neighborhoods
calc_ndvi(df, address_location_neighborhood = TRUE)
```

Code Snippet 4: Calculate NDVI

The initial printed message informs the user about the buffer type utilised upon running the function. Furthermore, if a raster layer is not supplied, as in this instance, the function provides

essential information regarding the data source used for retrieving the NDVI values. This includes details such as the ID of the selected image, the date the photo was captured, and the extent of cloud cover present in the picture. An excerpt of the interface that corresponds with this can be found in the chunk below;

```
You are using the provided area as a buffer to extract the NDVI scores
Sentinel-2-l2a data is used to retrieve the ndvi values.
 The ID of the selected image is:
    S2B_MSIL2A_20201118T104329_R008_T31UFU_20201119T222019
 The date of the picture that was taken is: 2020-11-18T10:43:29.024000Z
 The cloud cover of this day was 0.118422%
```

After the informative messages, the interface will provide the resulting output, which is, in this instance, nine neighbourhoods in Amsterdam, including the mean NDVI.

```
Simple feature collection with 9 features and 4 fields
Geometry type: POLYGON
Dimension:     XY
Bounding box:  xmin: 121895.4 ymin: 486266.6 xmax: 123998.4 ymax: 487863.7
Projected CRS: Amersfoort / RD New
  UID  mean_NDVI Buurtcode               Buurt                           geom
1   1 0.06167251      AF04            Rapenburg POLYGON ((122254.7 487241.8...
2   2 0.19874419      AF06            Uilenburg POLYGON ((121895.4 486971.2...
3   3 0.21607993      AF07           Valkenburg POLYGON ((122088.4 486839.2...
4   4 0.36863100      AJ02             Plantage POLYGON ((122233.5 486651, ...
5   5 0.23472975      AK01 Marine-Etablissement POLYGON ((122584.8 487858.4...
6   6 0.34115585      AK02          Kattenburg POLYGON ((122804.3 487108.2...
7   7 0.26490680      AK03          Wittenburg POLYGON ((122978.2 486996.9...
8   8 0.16019079      AK04          Oostenburg POLYGON ((123194.2 486865.7...
9   9 0.19806864      AK07            Kadijken POLYGON ((122546.8 486975.5...
```

In addition, Figure 3a exhibits the Sentinel-2-l2a image incorporating the NDVI values. This map visualises the resulting NDVI scores and the neighbourhoods used in the analysis.

3.1.2 Land cover

The land_cover() function extracts the land cover values within a specified area. This function calculates the percentage of our site covered by each land cover class.

To illustrate the functionality of land_cover, we have considered an example involving a network buffer. To get the land cover data, ESA WorldCover 10 m 2020 v100 data (Zanaga et al., 2021) is obtained from PC (Source et al., 2022). This dataset is a default source when no specific land cover data is provided. Additionally, a network file is extracted from OSM (Gilardi & Lovelace, 2022) to calculate the buffer. Code chunk 5 shows how the land cover is calculated for a network buffer around the address locations.

```
# Neighborhoods
land_cover(df_points, buffer_distance=300, network_buffer=TRUE)
```

Code Snippet 5: Calculate Land Cover

The code output starts with a transparent message about what buffer is created for the calculations. Because we used a network buffer this time, it also warns that it can be computationally extensive when utilising a network buffer for large files. Furthermore, it provides feedback on the download of the network file. Additionally, it prints information about the land cover data used for calculating the land cover percentage. A fragment of the output messages in the interface can be found below.

```
You will use a network to create a buffer around the address location(s),
    Keep in mind that for large files it can take a while to run the function.

You did not provide a network file, osm will be used to create a network file.
If a city is missing, it will take more time to run the function
The input place was matched with Noord-Holland.

The data is retrieved from the ESA WorldCover product at a 10m resolution
 The product is based on sentinel-1, sentinel-2
 The data is retrieved in the year 2021
```

When all informative messages are provided, the interface shows a simple features data frame, including the land cover percentages within the calculated buffer. Lastly, figure 3b illustrates the map corresponding to the results.

```
Simple feature collection with 9 features and 8 fields
Geometry type: POINT
Dimension:     XY
Bounding box:  xmin: 122168.8 ymin: 486602.6 xmax: 123603.6 ymax: 487497.6
Projected CRS: Amersfoort / RD New
  UID tree_cover grassland built.up bare_vegetation perm_water_bodies
1   1       0.04      0.00     0.73               0              0.23
2   2       0.11      0.00     0.83               0              0.05
3   3       0.09      0.00     0.87               0              0.04
4   4       0.39      0.00     0.61               0              0.00
5   5       0.25      0.04     0.55               0              0.16
6   6       0.44      0.01     0.53               0              0.03
7   7       0.00      0.00     0.69               0              0.31
8   8       0.16      0.00     0.83               0              0.00
9   9       0.15      0.00     0.72               0              0.13
  Buurtcode                Buurt                       geom
1      AF04            Rapenburg POINT (122550.8 487284.1)
2      AF06            Uilenburg POINT (122168.8 487033.6)
3      AF07           Valkenburg POINT (122341.7 486895.6)
4      AJ02             Plantage POINT (122767.5 486602.6)
5      AK01 Marine-Etablissement POINT (122906.4 487497.6)
6      AK02           Kattenburg   POINT (123179.1 487316)
7      AK03           Wittenburg POINT (123344.6 487201.2)
8      AK04           Oostenburg POINT (123603.6 487073.4)
9      AK07             Kadijken   POINT (123035 486830.7)
```

3.1.3 Canopy percentage

For the canopy percentage, we will provide an example of Marine-Etablissement, a district in Amsterdam. In the code below, we are calculating the canopy percentage.

```r
# load canopy data
canopy_layer <- sf::st_read('path/to/canopy_layer.gpkg')

# Select Marine-Etablissement
df_point_canopy <- df_points[df$Buurt=='Marine-Etablissement',]

canopy_pct(df_point_canopy, canopy_layer = canopy_layer, buffer_distance=200)
```

Code Snippet 6: Calculate Canopy percentage

The interface tells us which buffer is used for the calculation;

```
Euclidean distance will be used to calculate the buffers around the address
    that is given.
```

Furthermore, the function returns the provided data, including the canopy percentage that is calculated within a Euclidean buffer of 200 meters;

```
Simple feature collection with 1 feature and 4 fields
Geometry type: POINT
Dimension:     XY
Bounding box:  xmin: 122906.4 ymin: 487497.6 xmax: 122906.4 ymax: 487497.6
Projected CRS: Amersfoort / RD New
  UID canopy_pct Buurtcode                Buurt                       geom
1   1   13.93482      AK01 Marine-Etablissement POINT (122906.4 487497.6)
```

The canopy layer, calculation area (i.e. buffer around the address) and address corresponding to the output are shown in a map in Figure 3c.

3.1.4 Greenspace percentage

Code snippet 7 calculates the greenspace percentage for the nine-point addresses in Amsterdam.

```
1 greenspace_pct(df_points, buffer_distance=300)
```

Code Snippet 7: Calculate Green space

The interface first describes the buffer that is used to make the calculations. Afterwards, it says something about the greenspace layer. This time, the greenspace layer was not provided, thus osmdata was used to retrieve the greenspaces;

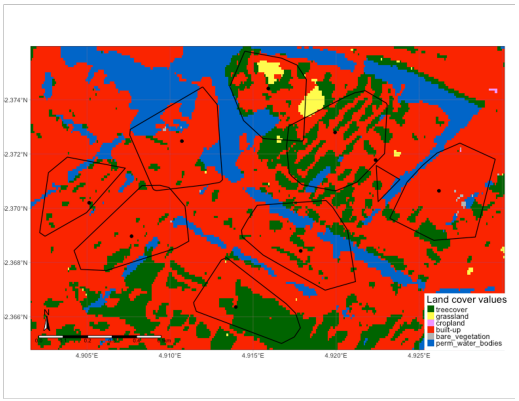
```
Euclidean distance will be used to calculate the buffers around the
    address location that is given
You did not provide a greenspace layer, osmdata will be used to find greenspaces
```

After the informative message, the output is given, which is a simple features dataset with the provided input and the greenspace percentage;

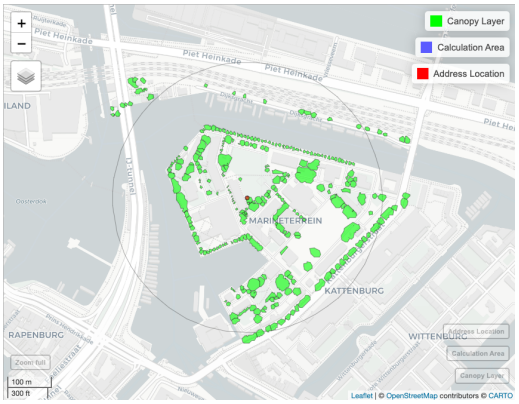
```
Simple feature collection with 9 features and 4 fields
Geometry type: POINT
Dimension:     XY
Bounding box:  xmin: 122168.8 ymin: 486602.6 xmax: 123603.6 ymax: 487497.6
Projected CRS: Amersfoort / RD New
  UID greenspace_pct Buurtcode                Buurt                       geom
1   1      0.2726892      AF04             Rapenburg POINT (122550.8 487284.1)
2   2      2.8255676      AF06             Uilenburg POINT (122168.8 487033.6)
3   3     11.1759821      AF07            Valkenburg POINT (122341.7 486895.6)
4   4      5.5207141      AJ02              Plantage POINT (122767.5 486602.6)
5   5      2.8631906      AK01 Marine-Etablissement POINT (122906.4 487497.6)
6   6      7.3827057      AK02            Kattenburg   POINT (123179.1 487316)
7   7      7.2886344      AK03            Wittenburg POINT (123344.6 487201.2)
8   8      5.5762805      AK04            Oostenburg POINT (123603.6 487073.4)
9   9      3.2033758      AK07              Kadijken   POINT (123035 486830.7)
```

Figure 3d shows the map with the greenspaces, buffer, and addresses that match the function results.

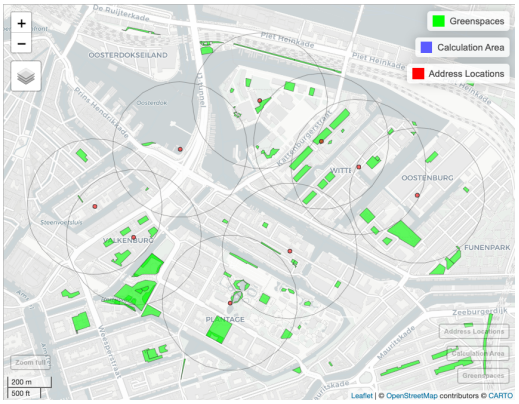
(a) NDVI in neighbourhoods



(b) Land cover in a network buffer



(c) Canopy in a Euclidean buffer



(d) Greenspace in a Euclidean buffer

Figure 3: Visualisation of the output of the functionalities in the availability functions

3.2 Accessibility

In the accessibility examples, we will use the address point in the neighbourhood *Oostenburg*. The following subsections will explain different approaches to using the greenspace_access() function.

```
df_point_access <- df_points[df$Buurt=='Oostenburg', ]
```

Code Snippet 8: Create point for accessbility examples

3.2.1 Euclidean distance calculation

In the first example, we applied the default settings of greenspace_access(). This involves calculating the Euclidean distance from the address location to the nearest greenspace centroid. Code snippet 9 demonstrates how to run this example.

```
greenspace_access(df_point_access, buffer_distance = 300)
```

Code Snippet 9: Calculate accessiblity default

Upon running the code, the interface displays the results. The first message notifies the user that no greenspace data was provided and that OSM data will be utilised to identify the greenspaces (Mark Padgham et al., 2017). Furthermore, it states that the Euclidean distance will be used to calculate the distance from the address location to the centroid of the greenspace. The duration of the function execution is also presented in the interface;

```
You did not provide a greenspace layer, osmdata will be used to find greenspaces
To calculate the distance from the address to the greenspace,
    Euclidean distance to the centroid of the greenspaces is used.
Time difference of 9.944231 secs
```

The final output of the function is a simple feature data frame that includes the input data, the distance to the nearest greenspace centroid, and whether the centroid falls within the specified buffer or not;

```
Simple feature collection with 1 feature and 5 fields
Geometry type: POINT
Dimension:     XY
Bounding box:  xmin: 123603.6 ymin: 487073.4 xmax: 123603.6 ymax: 487073.4
Projected CRS: Amersfoort / RD New
  UID closest_greenspace greenspace_in_300m_buffer Buurtcode      Buurt
1   1           178.7477                      TRUE      AK04 Oostenburg
                       geom
1 POINT (123603.6 487073.4)
```

The map in Figure 4a illustrates the example; green polygons represent the greenspace, while the blue lines indicate the Euclidean distance from the address location to the nearest centroid. The greenspace centroids are depicted as black points, while a red point denotes the address location.

3.2.2 Network Distance Calculation

In this example, the accessibility function utilizes network distance to compute the distance from the address location to the nearest greenspace centroid. Code snippet 10 shows how we run this code for a buffer distance of 300 meters.

```
greenspace_access(df_point_access, buffer_distance = 300, euclidean=FALSE)
```

Code Snippet 10: Calculate accessiblity greenspace centroid and network distance

After running the code above, the interface displays several messages and outputs. The initial message indicates no network file was provided and informs the user that one will be created using OSM. As the network file is being downloaded, the interface shows the region to which the input data is matched and provides the progress bar of the download. After the download, the interface displays the simple feature collection of the network, which is in geographic coordinates. To ensure compatibility with the input data, the downloaded network data is transformed into the same CRS as the input data within the function. After acquiring the network data, the interface retrieves the greenspaces from OSM since no greenspace file was provided. After acquiring and processing all the data, a message is displayed indicating the calculation of the distance, which in this case is the distance from the address location to the centroid of the network using the network. Additionally, the interface provides the duration of the function before showing the final output. The chunk below shows excerpts of the interface upon running the code;

```
You did not provide a network file, a network will be created using osm.
If a city is missing, it will take more time to run the function
The input place was matched with Noord-Holland.

You did not provide a greenspace layer, osmdata will be used to find greenspaces.

To calculate the distance from the address to the greenspace,
    the network distance to the centroid of the greenspaces is used.

Time difference of 9.73464 secs
```

The results displayed include the input data, the closest distance to the greenspace centroid in meters, and whether it falls within the specified buffer or not;

```
Simple feature collection with 1 feature and 5 fields
Geometry type: POINT
Dimension:     XY
Bounding box:  xmin: 123603.6 ymin: 487073.4 xmax: 123603.6 ymax: 487073.4
Projected CRS: Amersfoort / RD New
  UID closest_greenspace greenspace_in_300m_buffer Buurtcode      Buurt
1   1           263.8876                      TRUE      AK04 Oostenburg
                       geom
1 POINT (123603.6 487073.4)
```

Figure 4b shows the map made with the function results. Green polygons represent the greenspaces. However, since the lines are retrieved from an OSM network file, the greenspace centroids and address centroids may not align precisely with the network lines. As a result, you may notice that the lines do not intersect with the points in the plot. The greenspace centroids are depicted as black points.

3.2.3 Network Distance to Pseudo Entrances

In the last example, we have calculated the network distance from the address location to the pseudo entrances of the greenspaces. Code snippet 11 shows how this distance can be calculated in *GreenExp*.

```
greenspace_access(df_point_access, buffer_distance = 300,
                      euclidean=FALSE, pseudo_entrance=TRUE)
```

Code Snippet 11: Calculate accessiblity network distance and pseudo entrances

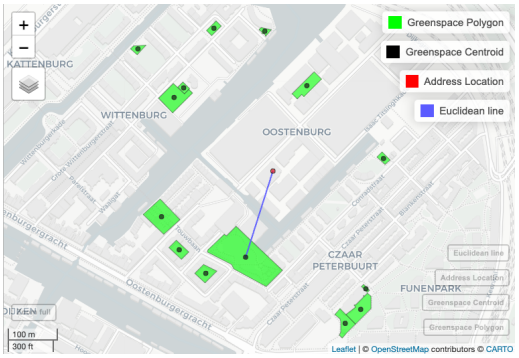
Running the function will return an interface with information about how the data is acquired and the resulting output. As in the Network Distance calculation, the initial message indicates that no network file is provided and that OSM will be used to create one. Next to the network file, a greenspace layer will be extracted from OSM. After acquiring and processing the data, a message is displayed with the information on the distance calculation;

```
You did not provide a network file, a network will be created using osm.
If a city is missing, it will take more time to run the function
The input place was matched with Noord-Holland.

You did not provide a greenspace layer, osmdata will be used to find greenspaces.

To calculate the distance from the address to the greenspace,
    the network distance to the pseudo entrances of the greenspaces is used.

Time difference of 11.53716 secs
```

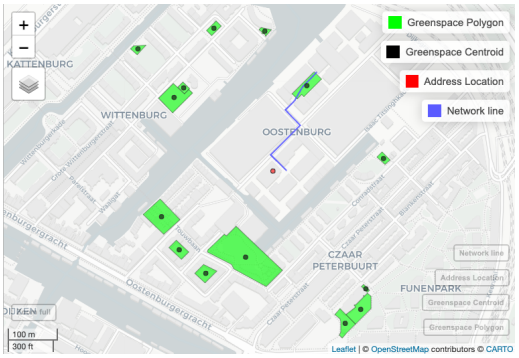
The resulting output comprises the following information; the input layer, including all their fields, the distance from the POI to the nearest pseudo entrance, and whether the nearest pseudo entrance lies within a Euclidean buffer of 300 meters;

```
Simple feature collection with 1 feature and 5 fields
Geometry type: POINT
Dimension:     XY
Bounding box:  xmin: 123603.6 ymin: 487073.4 xmax: 123603.6 ymax: 487073.4
Projected CRS: Amersfoort / RD New
  UID closest_greenspace greenspace_in_300m_buffer Buurtcode      Buurt
1   1            230.747                      TRUE      AK04 Oostenburg
                       geom
1 POINT (123603.6 487073.4)
```

Furthermore, Figure 4c depicts a map where the network path to the nearest pseudo entrance point is drawn. The parks are shown as green polygons. The blue lines indicate the Euclidean distance from the address location to the nearest pseudo entrance. The park centroids are depicted as black points, and a red point represents the address location. Additionally, you may observe multiple pseudo entrances within the parks, as roads passing through the parks can also serve as potential entry points.



(a) Euclidean distance to greenspace centroids



(b) Network distance to greenspace centroids

(c) Network distance to pseudo-entrances of greenspaces

Figure 4: Visualisation of the output of the accessibility function

3.3 Visibility

In the following subsections, we will present the output of the visibility functions. You will need the Digital Terrain Model(DTM), the Digital Surface Model(DSM), and the Green Space Raster to run these functions successfully. Code snippet 12 demonstrates how to read and load these models into the system for further analysis.

```
# Read the greenspace
GS <- terra::rast('data/GS_AMS.tif')
# Read the digital surface model
DSM <- terra::rast('data/DSM_AMS.tif')
```

```
# Read the digital terrain model
DTM<- terra::rast('data/DTM_AMS.tif')
```

Code Snippet 12: Load Data for visibility functions

3.3.1 Viewshed

For starters, we have the viewshed function. The code snippet below will retrieve the visible area in a 200-meter radius.

```
viewshed(observer = df_points[1,], dsm_rast = DSM, dtm_rast = DTM,
                          max_distance = 200, observer_height = 1.7, plot = TRUE)
```

Code Snippet 13: Viewshed function

Running this function will return a SpatRaster and a plot of the viewshed. Upon running the function, the interface shows a message that the code is retrieved from the GVI function, along with the GitHub link where additional information can be found (Brinkmann & Labib, 2022);

```
This code is retrieved from the GVI function; for more information, look at
    https://github.com/STBrinkmann/GVI
```

In Figure 5a, the left plot represents the DSM, whereas the right plot represents the viewshed, where green is the visible area, and grey is not visible.

3.3.2 VGVI from sf

The Viewshed Greenness Visibility Index (VGVI) represents the proportion of visible greenness to the total visible area based on the viewshed. Code snippet 14 shows how to calculate the GVI for all address points.

```
vgvi_from_sf(observer = df_points,
                       dsm_rast = DSM, dtm_rast = DTM, greenspace_rast = GS,
                       max_distance = 200, observer_height = 1.7,
                       m = 0.5, b = 8, mode = "logit")
```

Code Snippet 14: VGVI from sf function

As can be read in the interface, the code for running the VGVI function is based on the GVI package.

```
This code is retrieved from the GVI function; for more information, look at
    https://github.com/STBrinkmann/GVI
```

A simple feature collection is also returned, containing the input data set and the VGVI scores per address location.

```
Simple feature collection with 9 features and 4 fields
Geometry type: POINT
Dimension:     XY
Bounding box:  xmin: 122168.8 ymin: 486602.6 xmax: 123603.6 ymax: 487497.6
Projected CRS: Amersfoort / RD New
# A tibble: 9 × 5
     id   VGVI Buurtcode Buurt                              geom
  <int>  <dbl> <chr>     <chr>                          <POINT [m]>
1     1 0.176  AF04      Rapenburg            (122550.8 487284.1)
2     2 0.165  AF06      Uilenburg            (122168.8 487033.6)
3     3 0.200  AF07      Valkenburg           (122341.7 486895.6)
4     4 0.530  AJ02      Plantage             (122767.5 486602.6)
5     5 0.324  AK01      Marine-Etablissement (122906.4 487497.6)
6     6 0.200  AK02      Kattenburg             (123179.1 487316)
7     7 0.0896 AK03      Wittenburg           (123344.6 487201.2)
8     8 0      AK04      Oostenburg           (123603.6 487073.4)
9     9 0.175  AK07      Kadijken               (123035 486830.7)
```

Since this function does not provide visualisations, we created a visual representation of the datasets used to calculate the VGVI in QGIS (see Figure 5b). The dots depicted in the figure represent the observer points. These points correspond to the addresses used. Each observer point serves as a starting location for measuring the VGVI. The numbers assigned to the address locations in the figure correspond to the ID numbers in the interface results. These ID numbers uniquely identify each address and allow for easy referencing and analysis in the code. The green shades in the plot represent the greenspace raster. It indicates the areas covered by vegetation, such as trees or parks, and is a crucial factor in determining the VGVI. The figure also includes the DTM and DSM. These models provide information about the elevation of the terrain and structures present in the area. The combination of DTM and DSM helps understand the region's topography.

3.3.3 VGVI from address

The last function we will elaborate on is VGVI_from_address(), which collects data from various points within this buffer and calculates the VGVI by taking the mean of the collected values. Code snippet 15 shows how to run the function.

```
vgvi_from_sf(observer = df_address,
                      dsm_rast = DSM, dtm_rast = DTM, greenspace_rast = GS,
                      max_distance = 200, observer_height = 1.7,
                      m = 0.5, b = 8, mode = "logit")
```

Code Snippet 15: VGVI from address

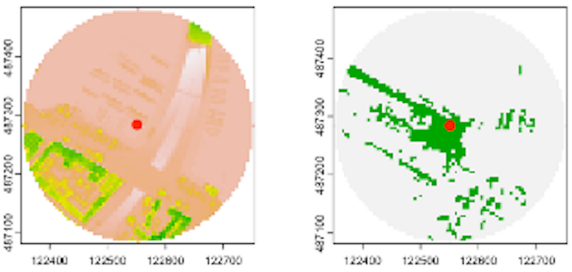
The initial message in the interface tells how the mean VGVI will be calculated. In this instance, 30 sample points are created within a Euclidean buffer of 50 meters around the address locations. The second message in the interface tells again that the function is based on the code of the GVI function;

```
The mean VGVI will be calculated using the mean of 30 sample points within
    a Euclidean buffer of 50 meters around the address location

This code is retrieved from the GVI function; for more information, look at
    https://github.com/STBrinkmann/GVI
```

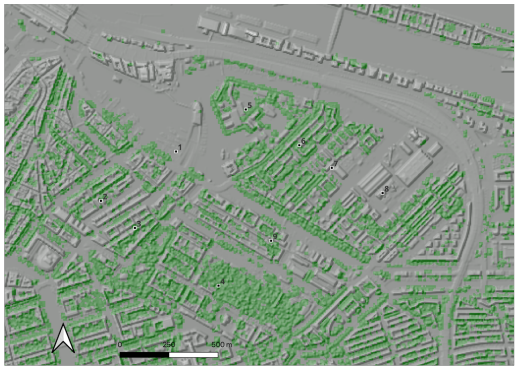
Finally, the output of the function will be given, which is the data that was used as input and the mean VGVI;

```
Simple feature collection with 9 features and 4 fields
Geometry type: POINT
Dimension:     XY
Bounding box:  xmin: 122168.8 ymin: 486602.6 xmax: 123603.6 ymax: 487497.6
Projected CRS: Amersfoort / RD New
  UID  mean_VGVI Buurtcode                Buurt                       geom
1   1 0.11437754      AF04            Rapenburg POINT (122550.8 487284.1)
2   2 0.12715923      AF06            Uilenburg POINT (122168.8 487033.6)
3   3 0.08686229      AF07           Valkenburg POINT (122341.7 486895.6)
4   4 0.31866944      AJ02             Plantage POINT (122767.5 486602.6)
5   5 0.18546960      AK01 Marine-Etablissement POINT (122906.4 487497.6)
6   6 0.14670017      AK02           Kattenburg   POINT (123179.1 487316)
7   7 0.06806295      AK03           Wittenburg POINT (123344.6 487201.2)
8   8 0.01157957      AK04           Oostenburg POINT (123603.6 487073.4)
9   9 0.13389976      AK07             Kadijken   POINT (123035 486830.7)
```

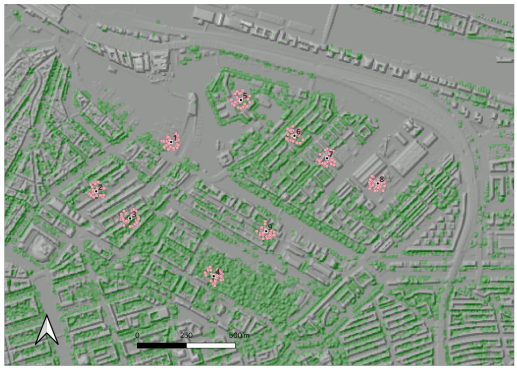
Figure 5c shows the exact visualisation as Figure 5b, only we added the random points used for the calculation in the map. These points are depicted in pink.



(a) Viewshed



(b) VGVI from sf



(c) VGVI from address

Figure 5: Visibility Figures

4 Discussion and Conclusion

4.1 Impact

This paper presents 'Greenexp', a free and open-source R package for conducting multidimensional spatial analysis in modelling diverse green space exposure variables. Our package contains functions that model green exposure from three core principles; availability, accessibility, and visibility. This section will discuss the impact of *GreenExp*.

First, the field of green space studies requires enhanced transparency when it comes to using geospatial and analytical methods. *GreenExp* addresses this need by offering detailed information

in its interface. Running the functions will give the users insights into the processes used for modelling green space exposure variables.

Second, a significant contribution of *GreenExp* is its ability to measure visibility as a component of green space exposure. While availability and accessibility have been widely explored, green pace studies often do not include visibility. Users can adopt a more comprehensive approach when assessing green space exposure by including visibility.

Third, our proposed package incorporates open sources and global data. Using open source data ensures that *GreenExp* is accessible to a wide range of users. Open source data can also accurately represent green spaces (Labib et al., 2020; Markevych et al., 2017). In addition, once the projected coordinates of the point of interest are obtained, the package can be analysed at any desired location.

Fourth, *GreenExp* offers a solution to overcome limitations posed by proprietary software in data analysis. Because R is an open-source platform, it allows for examining code, which fosters trust and accountability in data analysis. Due to R's flexible nature, it will enable the integration of various software and ensures consistent results across different studies. Sharing *GreenExp* with the research community promotes consistency, comparability, and collaboration among researchers.

Lastly, the package is wider applicable than just for greenspace studies. The package's functions can model various areas of interest, such as the visibility of blue spaces, the distance to schools, and the availability of public toilets, among other things.

4.2 Limitations

While we acknowledge the value of the initial version of *GreenExp* for future greenspace research, it is essential to note that the package is still in the developmental stage, and certain limitations warrant consideration.

First, the availability functions to retrieve the NDVI and land cover in *GreenExp* rely on the planetary computer API if no raster files are provided. This setup works well for smaller regions. However, for significantly more areas of interest, the data may exceed the capacity of the provided tile in the planetary computer. One potential solution could involve utilising composite images, but it is important to note that *GreenExp* does not currently support this feature.

Second, the current implementation of the accessibility function in *GreenExp* solely considers the distance between the address and the greenspace points. However, for a more comprehensive accessibility analysis, it is imperative to incorporate the supply and demand aspects of the greenspaces, as suggested by Breekveldt and Labib, 2023.

Third, using the visibility functions in *GreenExp* depends on the availability of a DSM and a DTM. Consequently, this means that the visibility functions may not be accessible for researchers who do not have access to such models. The Mapillary API offers street-view images. These images can be subjected to analysis using image segmentation. It would be beneficial for *GreenExp* to implement a function that uses Mapillary so that everyone can measure the visibility.

Finally, there is room for improvement in the speed of availability and accessibility functions. These functions tend to have long computation times when dealing with large datasets and network files. It is vital to address the speed of these functions. One potential solution is to implement the algorithms in C++, similar to what has been done for the visibility function, as C++ offers significantly faster computational performance compared to R (Wickham, 2019).

4.3 Conclusion

In conclusion, *GreenExp* presents a valuable contribution to spatial analysis for modelling green space exposure. The package showcases strengths in transparency, the inclusion of visibility analysis and the utilisation of open-source data. The package also has limitations that should be

considered for future development and improvement. We have successfully developed a robust, transparent package that adheres to the FAIR(4RS) principles. We think *GreenExp* is a solid foundation for enhancing the quality of research in this field.

References

Appelhans, T., Detsch, F., Reudenbach, C., & Woellauer, S. (2022). *Mapview: Interactive viewing of spatial data in r* [R package version 2.11.0]. https://CRAN.R-project.org/package=mapview

Aybar, C. (2023). *Rgee: R bindings for calling the 'earth engine' api* [https://github.com/r-spatial/rgee/, https://r-spatial.github.io/rgee/, https://github.com/google/earthengine-api/].

Bache, S. M., & Wickham, H. (2022). *Magrittr: A forward-pipe operator for r* [R package version 2.0.3]. https://CRAN.R-project.org/package=magrittr

Barker, M., Chue Hong, N. P., Katz, D. S., Lamprecht, A.-L., Martinez-Ortiz, C., Psomopoulos, F., Harrow, J., Castro, L. J., Gruenpeter, M., Martinez, P. A., et al. (2022). Introducing the fair principles for research software. *Scientific Data*, *9*(1), 622. https://doi.org/10.1038/s41597-022-01710-x

Berke, E. M., Koepsell, T. D., Moudon, A. V., Hoskins, R. E., & Larson, E. B. (2007). Association of the built environment with physical activity and obesity in older persons. *American journal of public health*, *97*(3), 486–492.

Beygelzimer, A., Kakadet, S., Langford, J., Arya, S., Mount, D., & Li, S. (2023). *Fnn: Fast nearest neighbor search algorithms and applications* [R package version 1.1.3.2]. https://CRAN.R-project.org/package=FNN

Bowler, D. E., Buyung-Ali, L. M., Knight, T. M., & Pullin, A. S. (2010). A systematic review of evidence for the added benefits to health of exposure to natural environments. *BMC public health*, *10*(1), 1–10. https://doi.org/10.1186/1471-2458-10-456

Bratman, G. N., Anderson, C. B., Berman, M. G., Cochran, B., De Vries, S., Flanders, J., Folke, C., Frumkin, H., Gross, J. J., Hartig, T., et al. (2019). Nature and mental health: An ecosystem service perspective. *Science advances*, *5*(7), eaax0903. https://doi.org/10.1126/sciadv.aax0903

Breekveldt, B., & Labib, S. (2023). *Modelling greenspace accessibility at multi- spatial contexts: a pilot study of comparing the E2SFCA and Gravity models*. Zenodo. https://doi.org/10.5281/zenodo.7823447

Bresenham, J. E. (1965). Algorithm for computer control of a digital plotter. *IBM Systems Journal*, *4*(1), 25–30. https://doi.org/10.1147/sj.41.0025

Bresenham, J. (1977). A linear algorithm for incremental digital display of circular arcs. *Commun. ACM*, *20*(2), 100–106. https://doi.org/10.1145/359423.359432

Brinkmann, S., & Labib, S. (2022). *Gvi: R package for computing vgvi for spatial raster (v1.1)*. 10.5281/zenodo.7057132

Browning, M. H., Rigolon, A., McAnirlin, O., & Yoon, H. ((2022). Where greenspace matters most: A systematic review of urbanicity, greenspace, and physical health. *Landscape and Urban Planning*, *217*, 104233. https://doi.org/10.1016/j.landurbplan.2021.104233

Cardinali, M., Beenackers, M. A., van Timmeren, A., & Pottgiesser, U. (2023). Preferred reporting items in green space health research. guiding principles for an interdisciplinary field. *Environmental Research*, 115893. https://doi.org/10.1016/j.envres.2023.115893

Collins, R. M., Spake, R., Brown, K. A., Ogutu, B. O., Smith, D., & Eigenbrod, F. (2020). A systematic map of research exploring the effect of greenspace on mental health. *Landscape and Urban Planning*, *201*, 103823. https://doi.org/10.1016/j.landurbplan.2020.103823

Csárdi, G., & FitzJohn, R. (2019). *Progress: Terminal progress bars* [R package version 1.2.2]. https://CRAN.R-project.org/package=progress

Dadvand, P., & Nieuwenhuijsen, M. (2019). Green space and health. *Integrating human health into urban and transport planning: A framework*, 409–423.

Eddelbuettel, D., & Balamuta, J. J. (2018). Extending extitR with extitC++: A Brief Introduction to extitRcpp. *The American Statistician*, *72*(1), 28–36. https://doi.org/10.1080/00031305.2017.1375990

Ekkel, E. D., & de Vries, S. (2017). Nearby green space and human health: Evaluating accessibility metrics. *Landscape and urban planning*, *157*, 214–220. https://doi.org/10.1016/j.landurbplan.2016.06.008

England, N. (2010). *Nature nearby: Accessible natural greenspace guidance*.

Europea, U. (2011). Mapping guide for a european urban atlas. *Copenhague: Agencia Ambiental Europea*.

Forsyth, A., Van Riper, D., Larson, N., Wall, M., & Neumark-Sztainer, D. (2012). Creating a replicable, valid cross-platform buffering technique: The sausage network buffer for measuring food and physical activity built environments. *International journal of health geographics*, *11*(1), 1–9.

Frumkin, H., Bratman, G. N., Breslow, S. J., Cochran, B., Kahn Jr, P. H., Lawler, J. J., Levin, P. S., Tandon, P. S., Varanasi, U., Wolf, K. L., et al. (2017). Nature contact and human health: A research agenda. *Environmental health perspectives*, *125*(7), 075001. https://doi.org/10.1289/EHP1663

Gemeente Amsterdam. (2023). Amsterdam open geodata [Accessed: May 12, 2023].

Gianfredi, V., Buffoli, M., Rebecchi, A., Croci, R., Oradini-Alacreu, A., Stirparo, G., Marino, A., Odone, A., Capolongo, S., & Signorelli, C. (2021). Association between urban greenspace and health: A systematic review of literature. *International Journal of Environmental Research and Public Health*, *18*(10), 5137. https://doi.org/10.3390/ijerph18105137

Gilardi, A., & Lovelace, R. (2022). *Osmextract: Download and import open street map data extracts* [R package version 0.4.1]. https://CRAN.R-project.org/package=osmextract

Gorelick, N., Hancher, M., Dixon, M., Ilyushchenko, S., Thau, D., & Moore, R. (2017). Google earth engine: Planetary-scale geospatial analysis for everyone. *Remote Sensing of Environment*. https://doi.org/10.1016/j.rse.2017.06.031

Haluza, D., Schönbauer, R., & Cervinka, R. (2014). Green perspectives for public health: A narrative review on the physiological effects of experiencing outdoor nature. *International journal of environmental research and public health*, *11*(5), 5445–5461. https://doi.org/10.3390/ijerph110505445

Hijmans, R. J. (2023). *Terra: Spatial data analysis* [R package version 1.7-29]. https://CRAN.R-project.org/package=terra

Ibarra-Espinosa, S. (2020). *Geofabrik: Downloading open street map data* [R package version 0.1.0]. https://CRAN.R-project.org/package=geofabrik

Labib, S., Huck, J. J., & Lindley, S. (2021). Modelling and mapping eye-level greenness visibility exposure using multi-source data at high spatial resolutions. *Science of the Total Environment*, *755*, 143050. https://doi.org/10.1016/j.scitotenv.2020.143050

Labib, S., Lindley, S., & Huck, J. J. (2020). Spatial dimensions of the influence of urban green-blue spaces on human health: A systematic review. *Environmental research*, *180*, 108869. https://doi.org/10.1016/j.envres.2019.108869

Lee, A. C., & Maheswaran, R. (2011). The health benefits of urban green spaces: A review of the evidence. *Journal of public health*, *33*(2), 212–222. https://doi.org/10.1093/pubmed/fdq068

Mark Padgham, Bob Rudis, Robin Lovelace, & Maëlle Salmon. (2017). Osmdata. *Journal of Open Source Software*, *2*(14), 305. https://doi.org/10.21105/joss.00305

Markevych, I., Schoierer, J., Hartig, T., Chudnovsky, A., Hystad, P., Dzhambov, A. M., De Vries, S., Triguero-Mas, M., Brauer, M., Nieuwenhuijsen, M. J., et al. (2017). Exploring pathways

linking greenspace to health: Theoretical and methodological guidance. *Environmental research*, *158*, 301–317. https://doi.org/10.1016/j.envres.2017.06.028

Matsler, A. M., Meerow, S., Mell, I. C., & Pavao-Zuckerman, M. A. (2021). A 'green' chameleon: Exploring the many disciplinary definitions, goals, and forms of "green infrastructure". *Landscape and Urban Planning*, *214*, 104145. https://doi.org/10.1016/j.landurbplan.2021.104145

Morpurgo, J., Remme, R. P., & Van Bodegom, P. M. (2023). Cugic: The consolidated urban green infrastructure classification for assessing ecosystem services and biodiversity. *Landscape and Urban Planning*, *234*, 104726. https://doi.org/10.1016/j.landurbplan.2023.104726

Organization, W. H. (2017). Urban green spaces: A brief for action [Accessed on 20th June 2023].

Pebesma, E. (2018). Simple Features for R: Standardized Support for Spatial Vector Data. *The R Journal*, *10*(1), 439–446. https://doi.org/10.32614/RJ-2018-009

Pedersen, T. L. (2023). *Tidygraph: A tidy api for graph manipulation* [R package version 1.2.3]. https://CRAN.R-project.org/package=tidygraph

Remme, R. P., Frumkin, H., Guerry, A. D., King, A. C., Mandle, L., Sarabu, C., Bratman, G. N., Giles-Corti, B., Hamel, P., Han, B., et al. (2021). An ecosystem service perspective on urban nature, physical activity, and health. *Proceedings of the National Academy of Sciences*, *118*(22), e2018472118. https://doi.org/10.1073/pnas.2018472118

Schneider, W. (n.d.). BBBike OpenStreetMap Extracts.

Simoes, R., Souza, F., Zaglia, M., Queiroz, G. R., Santos, R., & Ferreira, K. (2021). Rstac: An r package to access spatiotemporal asset catalog satellite imagery. *2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS*, 7674–7677. https://doi.org/10.1109/IGARSS47720.2021.9553518

Source, M. O., McFarland, M., Emanuele, R., Morris, D., & Augspurger, T. (2022). *Microsoft/-planetarycomputer: October 2022* (Version 2022.10.28). Zenodo. https://doi.org/10.5281/zenodo.7261897

Taylor, L., & Hochuli, D. F. (2017). Defining greenspace: Multiple uses across multiple disciplines. *Landscape and urban planning*, *158*, 25–38. https://doi.org/10.1016/j.landurbplan.2016.09.024

Tennekes, M. (2018). tmap: Thematic maps in R. *Journal of Statistical Software*, *84*(6), 1–39. https://doi.org/10.18637/jss.v084.i06

Twohig-Bennett, C., & Jones, A. (2018). The health benefits of the great outdoors: A systematic review and meta-analysis of greenspace exposure and health outcomes. *Environmental Research*, *166*, 628–637. https://doi.org/10.1016/j.envres.2018.06.030

van der Meer, L., Abad, L., Gilardi, A., & Lovelace, R. (2023). *Sfnetworks: Tidy geospatial networks* [R package version 0.6.3]. https://CRAN.R-project.org/package=sfnetworks

Wickham, H. (2019). *Advanced r*. CRC press.

Wickham, H., François, R., Henry, L., Müller, K., & Vaughan, D. (2023). *Dplyr: A grammar of data manipulation* [R package version 1.1.2]. https://CRAN.R-project.org/package=dplyr

Wickham, H., Hester, J., Chang, W., & Bryan, J. (2022). *Devtools: Tools to make developing r packages easier* [R package version 2.4.5]. https://CRAN.R-project.org/package=devtools

Wickham, H., Vaughan, D., & Girlich, M. (2023). *Tidyr: Tidy messy data* [R package version 1.3.0]. https://CRAN.R-project.org/package=tidyr

Wilkinson, M. D., Dumontier, M., Aalbersberg, I. J., Appleton, G., Axton, M., Baak, A., Blomberg, N., Boiten, J.-W., da Silva Santos, L. B., Bourne, P. E., et al. (2016). The fair guiding principles for scientific data management and stewardship. *Scientific data*, *3*(1), 1–9. https://doi.org/10.1038/sdata.2016.18

Zanaga, D., Van De Kerchove, R., De Keersmaecker, W., Souverijns, N., Brockmann, C., Quast, R., Wevers, J., Grosu, A., Paccini, A., Vergnaud, S., Cartus, O., Santoro, M., Fritz, S.,

Georgieva, I., Lesiv, M., Carter, S., Herold, M., Li, L., Tsendbazar, N.-E., ... Arino, O. (2021). Esa worldcover 10 m 2020 v100. https://doi.org/10.5281/zenodo.5571936

Zhang, Y., Wu, Q., Wu, L., & Li, Y. (2021). Measuring community green inequity: A fine-scale assessment of beijing urban area. *Land*, *10*(11), 1197. https://doi.org/10.3390/land10111197