Journal of Transport Geography 123 (2025) 104130

Contents lists available at ScienceDirect

Journal of Transport Geography

journal homepage: www.elsevier.com/locate/jtrangeo

Madina Python package: Scalable urban network analysis for modeling pedestrian and bicycle trips in cities

Andres Sevtsuk *, Abdulaziz Alhassan

City Form Lab, Department of Urban Studies and Planning, Massachusetts Institute of Technology, Cambridge, MA 02139, USA

ARTICLE INFO

Keywords:
Madina
Python package
Urban network analysis
New York City
Urban science

ABSTRACT

There is growing interest around sustainable mobility in cities, particularly pedestrian mobility, but methodological limitations and scarcity of software tools to analyze the dynamics between pedestrians and urban land uses have limited both research on and policy-relevant planning applications of pedestrian modeling. To address these challenges, we introduce *Madina*, a new *Python* package for modeling pedestrian and cycling trips along spatial networks in urban environments. The package enables managing and visualizing spatial network datasets and implements a set of *Urban Network Analysis* (UNA) tools for measuring pedestrian accessibility to given destination facilities, for identifying critical walking routes between origin-destination types, and for estimating pedestrian flows over network segments. While some of the methods we use for modeling pedestrian trips along networks have been previously implemented in desktop software plugin-ins, such as *ArcGIS* or *Rhinoceros 3D*, *Madina* introduces three new capabilities to researchers and practitioners. First, it incorporates innovative algorithms that enable computationally expensive pedestrian routing assignments to be scaled to larger geographic areas than previously possible the UNA Rhinoceros environment. Second, an implementation in *Python* offers new and powerful opportunities to automate various types of pedestrian and bic modeling steps through scripts and allows outputs to be connected with other types of analytic tasks, not available in desktop software applications. Enabling complex spatial analysis procedures to be replicated in automated ways contributes to verifiability, reproducibility, and the development of more robust urban science. And third, it presents the first end-to-end implementation of urban network analysis methods in an open-source Python environment, where no proprietary software is needed. We demonstrate the application of Madina tools in New York City, one of the largest pedestrian networks in the world.

1. Introduction

With the transportation sector responsible for over a third of all CO2e emissions in industrialized nations (US EPA, 2024), inactivity-related diseases burdening public health budgets (Grasser et al., 2013), and employment centers competing over labor with quality of life offerings (Glaeser, 2011), cities around the world are seeking solutions to increase pedestrian mode share, along with public transport ridership (New York City, 2014; City of Los Angels, 2019; City of Boston, 2022). Urban travel choices are known to represent "derived" demand–demand that results from land use patterns, configurations of urban form, street characteristics, climate conditions as well as place-based demographics, combinations of which can make travel on some modes more likely than others (Handy et al., 2002; Cervero and K., 1997). Understanding how existing urban environments generate demand for pedestrian mobility, and how today's land use, infrastructure and urban design decisions could influence mobility choices of tomorrow, requires frameworks and models that capture built environment and pedestrian interactions at a high spatial resolution.

In light of the cross-sectoral importance that walkability has begun to hold in urban planning, still relatively few analytic frameworks enable researchers and practitioners to estimate pedestrian trip distribution in urban environments or anticipate the impacts of spatial development changes on pedestrian mobility. Most pedestrian modeling tools that are suitable for urban-scale analyses have been developed as desktop applications or plugins for other software platforms, such as GIS, Rhinoceros 3D or CAD software (Cooper and Chiaradia, 2020; Dogan et al., 2018; Sevtsuk, 2018; Sevtsuk and Mekonnen, 2012). Others require combining different software environments for different modeling steps (Zhang et al., 2023). Albeit powerful, such applications can require the

* Corresponding author.
E-mail address: asevtsuk@mit.edu (A. Sevtsuk).

https://doi.org/10.1016/j.jtrangeo.2025.104130
Received 12 March 2024; Received in revised form 7 November 2024; Accepted 17 January 2025
Available online 8 February 2025

use of proprietary software licenses and face limitations imposed by the underlying software architecture.

This paper introduces Madina (Arabic for "city"): a new, open-source Python package designed for modeling pedestrian trips in urban environments. The package enables managing and visualizing spatial network datasets needed to measure and model pedestrian mobility, and offers a set of Urban Network Analysis (UNA) tools (Sevtsuk and Kalvo, 2024) in the Python environment to measure pedestrian accessibility to given facilities (e.g. from homes to public transit stations); to identify critical routes between given origin-destination types (e.g. critical walking routes from homes to schools); and to estimate pedestrian volumes for sidewalk and crosswalk segments, which can constitute an important input for public investment into the pedestrian realm, and a critical denominator for various hazard assessments (e.g. vehicle pedestrian crashes). While some of the methods we use for modeling pedestrian activity have been previously implemented in desktop software plugin-ins, such as the Rhinoceros 3D UNA plugin (Sevtsuk, 2018; Sevtsuk and Kalvo, 2024, Madina introduces three additional contributions. First, Madina presents an implementation of these methods in an open-source Python environment, where no proprietary software is needed. Second, an implementation in Python offers new and powerful opportunities to automate various types of pedestrian modeling steps through scripts and allows outputs to be connected with other types of analytic tasks, not available in desktop software applications. Madina relies on several open-source libraries that handle geometry, spatial processing, network operations and visualization, and builds on their functionality to offer unique methods for analyzing pedestrian activity in cities. Third, Madina incorporates innovative algorithms that enable computationally expensive pedestrian routing assignments to be scaled to larger geographic areas than previously possible in UNA Desktop applications in Rhinoceros3D and ArcGIS. This enables pedestrian or cycling travel demand models to be run at the scale of entire cities using a free and open-source framework. While the package aims to simplify and streamline active mobility simulations, its data structures and visualization system is also designed to enable extensions to other types of urban analysis workflows in the future, including optimization and generative design models (Krish, 2011).

1.1. Literature review on pedestrian modeling

Developing models of active mobility as part of land-use transportation interactions has gained traction with several research groups across the world over the last couple of decades. OSMNx (Boeing, 2017) is a Python package to download, model, analyze, and visualize street networks from OpenStreetMap, allowing users to visualize walking, driving, or biking networks with a single line of code. Many other specialized Python packages for social or spatial network analysis functions have been developed, but none, to our knowledge, have focused specifically on modeling pedestrian or bicycle trips over networks.

In transportation research, several frameworks and desktop software tools have been developed for pedestrian modeling. Two modeling approaches can be broadly distinguished. First, pedestrian movement has been spatially analyzed in agent-based simulations that animate movement of pedestrian agents between origin–destination pairs, and a set of behavioral assumptions (Batty, 2003; PTV, 2012). Such simulations can offer high resolution results where each individual trip is spatially visualized. However, a high degree of specificity has also limited the applicability of agent-based pedestrian models to fairly small sites, such as airports, transit stations, or a few downtown blocks (Puusepp et al., 2018). Such models typically prioritize capturing interactions between agents (e.g. pedestrian congestion inside transit stations), which are less relevant to outdoor streets and largescale pedestrian models we discuss below. More importantly, popular microscopic agent-based platforms, such as VisSim and Viswalk use highly simplified assumptions about pedestrian destination and route choices, not accounting for significant behavioral literature on pedestrian mobility (Friis and Svensson, 2013). MatSim (Horni et al., 2016), an agent based framework discussed below, and MoPed (Zhang et al., 2022; Zhang et al., 2023), a pedestrian-specific travel demand model that works in conjunction with MatSim, constitute exceptions and have been applied at the urban scale.

Second, large-scale pedestrian flow prediction has emerged in network-based models, such as Urban Network Analysis (Sevtsuk and Kalvo, 2024) and sDNA (Cooper and Chiaradia, 2020), which also estimate trip level pedestrian trajectories but do so mathematically, without visualizing individual agents' journeys. Instead, the number of trips traversing each network link is estimated using graph theory methods, computing results faster and for significantly larger study areas than agent-based approaches. This makes network analysis an attractive option for planning and transportation scholars and practitioners, who tend to work in environments of significant complexity at the neighborhood or city scale (Crucitti et al., 2006; Kazerani and Winter, 2009; Ye et al., 2016).

Table 1 compares Madina's Urban Network Analysis framework with the MoPed and sDNA frameworks. MoPed is a four-step travel demand model that is designed to predict daily pedestrian trips and the selection of travel modes within designated "Pedestrian Analysis Zones" consisting of 80 m by 80 m grid cells (Clifton et al., 2016; Zhang et al., 2023). Core components of MoPed include pedestrian trip generation, distribution and mode choice. Route assignment is not handled within MoPed and can instead be added with other platforms, such as MatSim. MoPed operates at a finer resolution than traditional traffic analysis zones and depends on household survey data to create a synthetic population of agents. MoPed incorporates a Logit-based destination choice model, where trips from each origin PAZ are assigned a super-PAZ destination first (a group of destination PAZs), and then split between individual PAZ within the super-PAZ. Mode choice is determined according to a binomial choice model (Walk versus vehicle trip) with trip distance and area-specific independent variables as determinants. When it comes to route choice, MoPed assumes shortest routes and does not consider route attributes besides distance.

The sDNA model, which offers metrics of pedestrian accessibility and estimates pedestrian trip volumes over networks using Betweenness indices, is available as a plugin for Autocad, ArcGIS and QGIS (Cooper and Chiaradia, 2020). sDNA only models pedestrian trips without an explicit mode choice model, though similar to UNA, pedestrian-specific trip generation levels can be derived in a model calibration phase, based on observed pedestrian counts. Destinations can be represented at the addresses or building level of granularity, with a simplified destination choice model where all destinations of a given type, available within a given access radius, are considered as equal destinations with no probability assignment. In terms of route choice, sDNA can find the shortest, most direct, or a hybrid (shortest and most direct) route in 2D and 3D networks. Like MoPed, each trip between an origin-destination pair is assigned one lowest cost route.

The UNA modeling framework shares similarities with both the sDNA and MoPed approaches, but it introduces four key distinguishing features.

First, when multiple competing pedestrian destinations are available for a trip, the UNA framework splits the trip into fractions and allocates portions of the journey to all competing destinations (not only a single destination) within a defined search radius at the same time, using the Huff destination choice model (Huff, 1963a; Sevtsuk and Kalvo, 2018). This approach addresses the idiosyncratic destination preferences of different users, leading to a more realistic aggregate pedestrian flow distribution.

Second, for each origin-destination (O—D) pair, most existing models, including MoPed and sDNA, identify a single, lowest cost route. In contrast, the UNA framework not only identifies the lowest-cost route, but all "plausible" routes that are up to a specified percentage ("detour ratio") longer than the shortest route. Research on pedestrian route choice shows that people often select routes that are longer than the

Table 1
Comparison of pedestrian travel-demand frameworks in UNA and two comparable models: MoPed and SDNA.

	Model:	MoPed	SDNA	UNA
	Type of trip origins	Land uses	Land uses	Land uses
	Geographical resolution	80x80m grid cells	Address-level point locations	Address-level point locations
	Origin weights	Any grid cell attributes	Any origin point attributes	Any origin point attributes
	Elasticity with respect to destination availability?	No	No	Yes, according to Gravity accessibility to each destination type.
Trip Generation	Destination choice model?	Yes, multinomial logit pedestrian destination choice models; first choose a super PAZ then PAZ.	All destinations equally weighted within a specific radius.	Huff destination choice model, based on gravity accessibility to competing destinations.
	Simultaneous allocation to multiple destinations	No	Yes	Yes
Trip Distribution	Activity schedules?	No	No	No
	Modes considered	binary choice (walk vs. vehicular modes).	Pedestrian only	Pedestrian only
Mode Choice	Mode choice model?	Binomial choice model with trip distance and area-specific independent variables.	Pedestrian trip generation levels are calibrated with respect to observed counts.	Pedestrian trip generation levels are calibrated with respect to observed counts.
Route Choice	Route characteristics that can be included?	Shortest path only	Distance, angularity and slope.	Distance, turns, slope and any segment characteristics converted into "perceived length" units.
	Simultaneous allocation along multiple routes?	No, one path assignment for each O—D only	No, one path assignment for each O—D only	All paths that are up to a "detour ratio" longer than the shortest route are allocated for each O—D pair.
Applications/ Case Studies	Where has the model been applied?	Portland, OR, USA (Clifton et al. 2016; Zhang et al. 2022); Munich, Germany (Zhang et al. 2023)	Cardiff, Wales, UK (Cooper et al. 2019; Cooper & Chiaradia 2020)	Seoul, Korea (Kang 2015; Woo 2021); Beijing, China (Sun et al. 2016); Mexico City, (Escamilla 2016); Boston, MA, USA (Sevtsuk 2021); Melbourne, AUS (Sevtsuk et al. 2021).

shortest route (Li and Tsukaguchi, 2005; Zacharias, 2001). This feature allows the UNA framework to capture a wider variety of user preferences and provides a more realistic representation of how pedestrians, on aggregate, navigate networks (Sevtsuk and Kalvo, 2021).

Third, the UNA framework allows users to incorporate the effects of pedestrian route characteristics—such as sidewalk dimensions, the presence of ground-floor amenities, and traffic conditions—into a"perceived" length measure for each segment, rather than relying on geometric distance (Sevtsuk et al., 2024). These perceived segment lengths can serve as generalized costs and used in route assignment (see Section 3). The generalized travel costs can also affect accessibility estimates: more pleasant pedestrian routes are perceived as shorter, increasing the accessibility of destinations and potentially boosting trip generation. To our knowledge, no other pedestrian models have accounted for such generalized street qualities in route assignment.

Finally, the UNA framework in Madina offers the first free and open-source Python environment for users to set up networks with origins and destinations, and their attributes, and to execute pedestrian analysis workflows in an end-to-end framework. This eliminates the need for proprietary software, promoting transparency and accessibility in the modeling process.

In this paper, we present an initial version of the Madina Python package, which is integrated with other open-source Python libraries to visualize input and output data on networks, to enable editing network attributes, and to provide a range of different pedestrian and bicycle modeling tools we describe below. We chose the Python environment for implementation due to its increasing popularity among both developers and data analysts, particularly in geospatial research and application fields.

2. Madina capabilities

The Madina Python package architecture brings geometric spatial data manipulation, network representation, active mobility analysis tools, visualization, and handling of data inputs and outputs into a unified package in the Python programming language (Fig. 1). The package does so by integrating its tools with different open-source libraries in a modular way that allows for expansion or replacement of dependency parts. The package is intended to provide a workspace of layers, networks and visual representation options, mimicking the experience of GIS and CAD software in a scripting environment that enables users to spend less time on working on interoperability and data restructuring, and more on substantive analysis workflows.

At a high level, Madina allows users to import spatial network files, origin and destination points, along with their attributes and geographic projections into a visual Python environment and provides users with a set of analytic tools to model pedestrian and cycling mobility over the networks. The types of analytic tools currently available in Madina are described in section 3.

Madina represents spatial data using the Geopandas (den Bossche et al., 2024) and Shapely (Gillies et al., 2023) Python packages. Shapely supports the representation of individual geometric shapes like Points, LineStrings, and Polygons, while GeoPandas builds on top of Shapely to work with collections of geometries as part of GeoDataFrames. Madina creates spatial networks through the NetworkX package (Hagberg et al., 2008), along with origin and destination objects added through custom algorithms; runs urban network analysis routines via a scalable implementation of UNA tools, and visualizes results through an integration with the Deck.gl package (Urban Computing Foundation, 2016). The formatting, restructuring and data exchange between these packages is handled through a `Zonal()` object– Madina's equivalent of a project workspace. The Madina package is composed of two main parts, depicted in Fig. 1:

- ***Workspace Management:*** A set of objects, algorithms and package integrations to streamline file reading/writing, layers, visualizations and network creation procedures through a `Zonal()` workspace object.
- ***Analysis Tools and Workflows:*** Analysis tools that take a `Zonal()` object populated with spatial input data layers. Integrated tools, such as UNA, are provided as modules, with each module containing a list of functions. Each function takes as input a Zonal object, and tool-specific parameters. During execution, the function can retrieve spatial data and networks from the Zonal object, and passes output back to layers and networks in the Zonal object.

2.1. Zonal: A workplace to coordinate data and tools

Each workflow in Madina starts with creating a Zonal object that

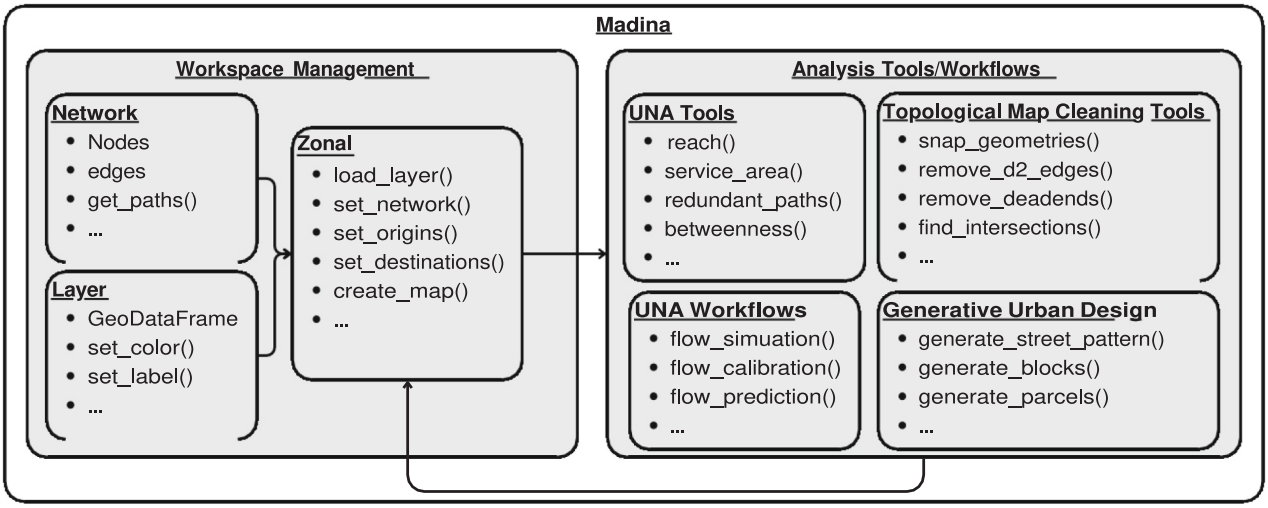

Fig. 1. Madina package architecture.

serves as a convenient workspace interface for analysis tasks. The Zonal object is populated with data layers and parameters for establishing networks and visualizations. Once a Zonal object is created and populated, it can be passed to analysis tools, where each tool is able to reference the data it needs, and store analysis results to the appropriate layers and networks. The user is expected to ensure that all input geometry layers share the same geographic projection system. The following describes a typical workflow:

```
import madina as md
import madina.una.tools as una

# creating a workspace
cambridge = md.Zonal()

''' --- create layers and networks --- '''

# passing a workspace to analysis functions
una.accessibility(cambridge, *args)

''' --- carry out more analysis --- '''

# creating visualizations of outputs.
cambridge.create_map()
```

2.2. Layer management

For spatial data management, Madina uses a layer system that allows users to import various datasets used for analysis (e.g. input networks, trip origins, destinations, etc.). Madina's layer representation uses the Geopandas' GeoDataFrame, which is stored as an attribute inside Madina's "layer" object that also maintains metadata, visual styling instructions, analysis tools, and workflow results for a particular layer. Geopandas, in turn, relies on Pandas for DataFrame representation and relational data querying, Numpy [1] for vector data representation, Shapely[2] for geometry representation and manipulation, Fiona[3] and Pyproj[4] for spatial inputs-output handling and coordinate reference system transformations, and Rtree[5] for spatial indexes for improved performance of spatial queries. Each GeoDataFrame contains a "geometry" column of Shapely objects. This allows for geometry manipulations at a fine scale and enables algorithmic manipulations of individual geometries in a manner similar to what is offered by CAD or GIS software. The Zonal workspace handles coordination between layers and visualizations. Calling `create_map()` passes inputs to Deck.GL for visualization, with specifies styling instructions for individual geometry objects inside a layer. As the layer carries generic styling instructions, it is possible to integrate with different visualization packages in the future.

2.3. Network management

To analyze pedestrian or bicycle activity over networks, a topologically inter-connected network with appropriately joined origins and destinations is needed. In addition to spatial networks composed of edges representing links, and nodes representing intersections, Madina uses Origin and Destination nodes, which serve as sources or sinks for trip generation and trip distribution. Origins and Destinations can be imported from a layer consisting of point objects with attribute data. Layers can be loaded from GeoJSONor SHP files, then inserted into network edges at appropriate locations where each point splits the nearest available edge.

The `create_street_network()` function in Madina takes a source layer of lines or polylines and represents it as a network of topologically interconnected edges. Madina does not offer network editing tools, rather the imported networks are assumed to be topologically work-ready. Networks can be edited outside of Madina, in CAD software, GIS or, other editing software. Madina does support non-planar networks—i.e. network, where some edges that traverse each other do not share endpoints, which allows users to represent overpasses and underpasses. The `create_street_network()` function extracts all unique geometry endpoints to create nodes, and creates edges between these nodes based on the line geometry to form a topologically interconnected and routable network, using a node snapping tolerance input value to handle input line endpoints that may be slightly misaligned (assuming units in the projection system of the input GeoJSON file). These nodes and edges are stored in GeoDataFrames (e.g. `cambridge.network.nodes`, `cambridge.network.edges`) that retains information about topological connectivity and geometric representation and attributes like edge length, "weight", or other attributes of nodes and edges. A network is set up as follows:

[1] https://numpy.org
[2] https://shapely.readthedocs.io/en/stable/
[3] https://pypi.org/project/fiona/
[4] https://pypi.org/project/pyproj/
[5] https://pypi.org/project/Rtree/

```
import madina as md
cambridge = md.Zonal()
cambridge.load_layer(
    '            sidewalks                ',
    'Cities/Cambridge/Data/sidewalks.geojson'
)
cambridge.create_street_network(
    source_layer="sidewalks",
    node_snapping_tolerance=0.1
)
```

Once a topologically connected network is created, point objects from the Origin and Destination layers can be inserted into the network, splitting network edges at the appropriate locations. After a layer is loaded to the Zonal object, the function `insert_node()` enables inserting point objects from a layer as Origin or Destination nodes into the network:

```
cambridge.load_layer(
    'subway',
    'Cities/Cambridge/Data/subway.geojson'
)
cambridge.insert_node(
    label='origin',
    layer_name="subway"
)
cambridge.load_layer(
    'buildings',
    'Cities/Cambridge/Data/building_entrances.
    geojson'
)
cambridge.insert_node(
    label='destination',
    layer_name="buildings"
)
cambridge.create_graph()
```

Origin nodes and destination nodes are added to the `cambridge.network.nodes` GeoDataFrame with the identifier of the nearest network edge, and the appropriate edge splitting location, as well as any Origin or Destination specific data attributes (e.g. origin type, size etc.), which can inform how many trips are generated from or distributed to different locations. The Origin and Destination dataframes carry intermediate analysis results before they are joined back to their source layers' dataframes. As the Origin and Destination GeoJSON point files provided by the user typically include point objects that are not exactly coincident with the network segments or nodes, Madina automatically maps each point to its closest network edge. In subsequent analysis functions, it is these re-mapped locations on the network that are used as trip Origins and Destinations. The `create_graph()` function creates three different NetworkX Graph objects, each of which is used separately to enhance computational efficiency, depending on the analysis function:

- `cambridge.network.light_graph`: Only contains the network nodes and edges. This is used as a baseline graph, into which origins and destinations are added. This graph can be used to access all NetworkX's functionalities and algorithms.

- `cambridge.network.d_graph`: Contains the network nodes, edges, and only Destination nodes. This graph provides performance enhancements by storing minimal information needed for a single Origin for cases, where the behavior of each Origin is independent of other Origins. This graph is used internally to implement a custom alternative path finding algorithm needed in UNA tools using a 'detour ratio' (see Section 2.4 and Table 3).
- `cambridge.network.od_graph`: A NetworkX Graph object that contains all network intersection, Origin, and Destination nodes. This graph is useful for NetworkX procedures, where all Origins and Destinations are used concurrently.

The three different NetworkX objects are provided for use by developers, a typical end user does not need to interact with them. As all Origin and Destination nodes are retained in `cambridge.network.nodes`, the function `cambridge.network.update_light_graph()` can dynamically add and remove Origin and Destination nodes from a NetworkX graph as needed to optimize the performance of particular Madina functions described below.

2.4. Implementation of UNA tools and workflows

The Madina implementation of UNA tools offers more flexibility for automation and provides an end-to-end replacement of GIS or Rhinoceros3D Desktop environments. Madina also improves computational efficiency, compared to Rhinoceros 3D and GIS plugins by offering multiprocessing and more efficient network algorithms for analyzing pedestrian trips over large networks. Madina offers specific efficiency gains and extensions for previous implementations of UNA tools in Desktop software, such as the path generator algorithm discussed below.

Similar to the UNA functionality in Rhinoceros3D, Madina allows users to assign custom"perceived" costs to the input network segments, which can differentiate the attractiveness or quality of different street segments for pedestrians. Pedestrian route choice literature has shown that more pleasant and attractive street segments have a lower perceived cost than their geometric length (e.g. a 100-m segment can feel like 85 m), while negative street attributes, such as higher traffic volumes and speeds, narrower sidewalks, uphill walking, or lack of shade can increase perceived route lengths (Olszewski and Wibowo, 2005; Lue and Miller, 2019). These custom segment costs must be included in the input network datasets as an attribute field for each segment, and the attribute field name is passed as parameter in the `create_street_network()`.

```
cambridge.create_street_network(
    source_layer='sidewalks',
    weight_attribute='perceived_distance'
)
```

For example, custom segment costs could include penalties or walking delays at both signalized and unsignalized crossings, which we ignore below for simplicity, but which have been demonstrated in a recent walkability study in Beirut, Lebanon (Sevtsuk et al., 2024). Like the Rhinoceros3D implementation of UNA tools, Madina allows turn penalties as part of impedance costs—a customized implementation of a single-source Dijkstra algorithm that allows imposing path-specific penalties for turns (defined as a minimum angle between consecutive network segments on at trip).

3. Madina algorithms

3.1. Path generation

Madina includes an innovative `path_generator()` algorithm to identify all possible paths between an Origin and a Destination up to a given 'detour ratio' longer than the shortest path, avoiding node repetition in paths. This is used in alternative path generation in the Madina betweenness tool, which estimates pedestrian flows over networks. The `path_generator()` finds all possible paths between an Origin and a Destination point according to a specified detour ratio (e.g. 1.2 times longer than the shortest available path). Though an analogous approach is implemented in our Rhino UNA toolbox, to our knowledge, there is no widely recognized algorithm to accomplish this efficiently. NetworkX's `shortest_simple_paths()` finds the K shortest paths between an origin and a destination (Yen, 1971), which could be used sequentially until a path exceeds the detour ratio. However, in practice, this operation is computationally prohibitively expensive. Madina's `path_generator()` instead uses a three step approach: a) Destination Discovery, which generates paths and distances to all reachable nodes from an

Origin in a given radius, b) Subgraph Generation, which uses a multi-source Dijkstra algorithm from all Destinations to the Origin to identify all eligible segments that form a smaller subgraph and a distance matrix heuristic, and c) Alternative Path Generation, which uses depth-first multi-target search from the Origin to all reachable Destinations in the smaller subgraph, with a termination condition based on the distance matrix heuristic. Use of the lighter `d_graph` enables Madina to use parallel computing and to assign individual cores to process one Origin at a time, substantially improving calculation times in large networks.

3.2. Pedestrian accessibility indices in Madina

Madina can be used to estimate pedestrian accessibility between a set of Origin-Destination pairs (e.g. from residential blocks to public transport stations). It can also be used to estimate how a number of different Destination types can be bundled and weighted to achieve combined pedestrian accessibility estimates, similar to the widely-used, but proprietary WalkScore metric available for purchase in many U.S. cities (WalkScore, 2021).

There are three different accessibility metrics available in Madina for a given set of origin and destination points:

- **Reach**. The Reach index (Sevtsuk, 2010), also known as a "cumulative opportunities accessibility index (Bhat et al., 2000; Geurs and van Wee, 2004) captures the number of Destinations that can be reached from each Origin within a given network radius, optionally weighted by numeric destination attributes. The Reach of an Origin *i* in a graph *G* describes the sum of Destination weights W[j] in *G* that are reachable from *i* at a lowest-cost path distance of at most *r*. It is defined as follows:

$$\text{Reach } [i]^r = \sum_{j \in G-\{i\}, d[i,j] \le r} W[j] \quad (1)$$

where *W[j]* represents numeric weights of destinations *[*j], and *d[i,j]* is the lowest-cost path distance between Origin *i* and Destination *j* in network *G*. The Reach index' elegant simplicity—it describes how many bus stops, playgrounds or grocery stores can be reached within a 10 min walk within a given network radius–enables intuitive interpretation. This ease of interpretability is a key reason for its use in planning and policy making (Jaber et al., 2017).

- **Gravity**. Whereas the Reach index simply counts the number of Destinations or Destination weights within a given search radius around each Origin, the Gravity index additionally accounts for travel costs required to reach each of the Destinations (Hansen, 1959). It therefore offers a more precise definition of accessibility, where each spatially separated Origin has unique accessibility to surrounding Destinations due to differences in travel costs (Bhat et al., 2000). Gravity index assumes that accessibility at Origin *i* is proportional to the attractiveness (weight) of Destinations *j*, and inversely proportional to the distance or travel cost between *i* and *j*. The Gravity index at Origin *i* within graph *G* at Search Radius *r*, is defined as follows:

$$\text{Gravity } [i]^r = \sum_{j \in G-\{i\}, d[i,j] \le r} \frac{W[j]}{e^{\beta * d[i,j] - \text{plateau}}} \quad (2)$$

where W[j] is the weight of Destination *j*, and the function of distance between *i* and j in the denominator represents an exponential distance decay rate that is controlled by parameter beta. In Madina, we have added an optional plateau radius (e.g. 400 m) to the Grvaity index: a baseline distance from Origin *i* in which no distance decay is applied and Destinations contribute their full weights towards the index. This allows treating all nearby Destinations as equally accessible. The plateau radius can be set to zero to ensure that distance decay is always included. If the plateau distance is greater than or equal to the search radius, it has no effect.

- **The K-Nearest-Neighbor (KNN)** accessibility metric offers a customized WalkScore index that measures combined access to a basket of destinations (WalkScore, 2021). It builds on the Gravity index, with three important differences. First, only K-nearest Destinations for each Destination category are used to compute the index (e.g., if k = 3 for bus stops, then only three nearest bus stops), ignoring other Destinations of that category. Unlike the Gravity index, where maximum results are found at locations with most Destinations, the k ceiling effect in the KNN accessibility metric ensures that peak results can occur in many different locations within the same city, where the predefined minimum level of Destination accessibility is met. Second, the relative impact of each of the K-nearest Destinations is controlled by predefined weights, explained below and in Table 2. And third, the KNN accessibility index allows the contributions of multiple Destination types to be combined into a single aggregate access score that can be normalized into a zero to one range. The index is defined as follows:

$$\text{KNNA } [i]^r = \sum_{j \in G-\{i\}, d[i,j] \le r} \frac{C_k}{e^{\beta * d[i,j] - \text{plateau}}} \quad (3)$$

where Ck is a predefined coefficient for the K^{th} nearest Destination, given in an array of the form [0.5, …, 0.1]. For instance, the array [0.5, 0.3, 0.2] means that the first reachable Destination increases KNN-Access by 0.5 (if it lies within the plateau radius), the second by 0.3 and the third by 0.2. Any additional accessible Destinations beyond the third nearest will not contribute to the accessibility score. Beyond the plateau radius, Destination contributions are further penalized by the distance decay rate, similar to the Gravity index. Different destination types (e.g. transit stops, shops, parks) can be assigned a different coefficient array.

For example, a white paper published by WalkScore[6] uses the following arrays for each type of commercial amenity:

```
amenity_weight = {
    "grocery": [3],
    "restaurants": [.75, .45, .25, .25, .225,
     .225, .225, .225, .2, .2],
    "shopping": [.5, .45, .4, .35, .3],
    "coffee": [1.25, .75],
    "banks": [1],
    "parks": [1],
    "schools": [1],
    "books": [1],
    "entertainment": [1],
}
```

Within an 800-meter (1/2 mile) search radius, an Origin gets a KNN access score of 3 for reaching one grocery store within the given plateau radius, 3 for reaching ten different restaurants, 2 scores for reaching five different retail shops, 2 scores for reaching 2 coffee shops, and 1 for reaching a bank, a park, a school, a bookstore and an entertainment venue. A complete WalkScore estimate also contains other destination types and neighborhood qualities, including public transit access, land use diversity, and intersection density, which we ignore here for the sake of brevity. The overall KNN-Access score sums the contributions of individual Destination types, which Madina finally normalizes into a 0-1 range. A key benefit of producing accessibility estimates in Madina is the ability to automatically analyze different Destinations with different specifications within a single routine. Calling the UNA

[6] See WalkScore method here: https://www.dropbox.com/scl/fi/ kz76k8jt0guyfaajbab2m/WalkScore-methodology-2011.pdf?rlkey= 8j2q9nst8lhxvkly6xhcf88jc&dl=0

Table 2
Pairing table used to measure KNN accessibility scores from homes in NYC.

Network Cost	Origin Name	Destination Name	Radius	Beta	KNN Weight	Plateau	Turns	Turn Threshold	Turn Penalty
Geometric	Homes	Grocery	800	0.001	[3]	400	FALSE	0	0
Geometric	Homes	Restaurants	800	0.001	[0.75, 0.45, 0.25, 0.25, 0.225, 0.225, 0.225, 0.225, 0.2, 0.2]	400	FALSE	0	0
Geometric	Homes	Shopping	800	0.001	[0.5, 0.45, 0.4, 0.35, 0.3]	400	FALSE	0	0
Geometric	Homes	Coffee	800	0.001	[1.25, 0.75]	400	FALSE	0	0
Geometric	Homes	Banks	800	0.001	[1]	400	FALSE	0	0
Geometric	Homes	Parks	800	0.001	[1]	400	FALSE	0	0
Geometric	Homes	Schools	800	0.001	[1]	400	FALSE	0	0
Geometric	Homes	Books	800	0.001	[1]	400	FALSE	0	0
Geometric	Homes	Entertainment	800	0.001	[1]	400	FALSE	0	0

`KNN_accessibility()` workflow calculates these accessibility measures for all Origin points using the settings specified in the"pairing.csv". An example pairing table is shown in Table 2, when all layers containing Origins and Destinations are stored in a Madina project folder Cities/NYC/Data:

```
import madina.una.workflows as una_wf
una_wf.KNN_accessibility(
    city_name="NYC",
    pairings_file="pairings.csv",
)
```

Conducting an analogous workflow in Rhinoceros 3D or GIS would necessitate specifying different Origin-Destination accessibility estimates sequentially, requiring extra time and effort. The KNN accessibility index is also not available in these other platforms. Though analogous specifications have been proposed elsewhere, they are typically pre-computed for fixed spatial units, such as Census polygons in select cities (Environmental Protection Agency, Ogy and User Guide, 2021). Madina allows users to specify their own KNN accessibility indices that can be customized to specific user groups (e.g. elderly, children), destination bundles, and locations.Computing results with Reach, Gravity or KNN accessibility indices can produce policy relevant evidence of how access to critical urban Destinations (e.g. schools, libraries, grocery stores, transit stations, green areas etc.) varies in space. Combined with demographic data, this can reveal socio-economic disparities in access to opportunities and inform policy makers where future investments are needed. The KNN accessibility results can also be used as elasticity coefficients for trip generation rates in pedestrian flow models, which we further discuss in section 3.4 below.

3.3. Workflow for pedestrian flow modeling

A unique functionality of Madina involves estimating foot traffic volumes on network segments. Understanding how built environments produce variations in foot-traffic on different streets is essential to identifying sidewalks or crosswalks with highest levels of pedestrian activity. Though pedestrian counts may be available from some streets in a city, they usually constitute a small fraction of all streets in a district or city. A pedestrian flow model can help understand which sidewalks, crosswalks and footpaths are most trafficked for particular trip types and estimate their corresponding use volumes.

Pedestrian flow estimation in Madina is achieved with the `una.betweenness` function. Building on the graph theory concept of betweenness (Freeman, 1977), the `una.betweenness` function estimates the spatial distribution of pedestrian trips between Origin-Destination pairs using a highly customized version of the betweenness algorithm (Sevtsuk, 2021a; Sevtsuk and Kalvo, 2024), and a set of user-input parameters. Similar to the accessibility set-up described above, the `una.betweenness` function requires two key inputs: a) a folder of data geometry layers (including a network-, Origin-, and Destination-geoJSON files, and b) a pairing CSV table that contains specifications about which Origins and Destinations are paired to generate trips, along with user-defined model settings and parameters. Illustrated in Table 3, this pairing table offers a convenient and user-friendly solution to specifying which types of trips to model without requiring users to edit code.

Calling the UNA betweenness flow simulation runs the workflow using the input data specified in the pairing.csv (from a folder called "Cities/NYC/Data"). The output data are automatically stored in a folder called "Cities/NYC/Simulations":

```
import madina.una.workflows as una_wf
una_wf.betweenness_flow_simulation(
    city_name="NYC",
    pairings_file="pairings.csv"
)
```

For each O-D pair specified in the pairing table (e.g. home to metro station trips), the betweenness tool estimates the distribution of these types of pedestrian trips over the network and produces a new column in the input network's attribute table, showing the number of estimated trips passing through each network segment. The results are generated and stored in the Cities/Cityname/Simulations subfolder, under the same directory location where the Python notebook is stored. The output includes a GeoJSON copy of the network file with segment IDs and estimated trip volumes for each O-D pair shown in a newly added column, a CSV file with the same information without geometry objects, and an HTML file, which can be opened in a browser for rapid visualization of results. Fig. 3 illustrates the resulting graphics from this HTML file. If the pairing table contains several rows for different O-D pairs, a combined "betweenness record CSV" and a "betweenness record geoJSON" file are additionally generated to summarize all O-D flow types in a single table with different columns, where rows represent input network segment IDs and columns represent the estimated flows for different O-D types (e.g. flows from homes to metro stations, from homes to parks etc.). This makes it convenient to estimate a number of different pedestrian flow types over a network in Madina, and to have the results showing how many trips for each flow type pass through each network segment in a single summary table with multiple result columns.The following sub-sections describe how Madina can be used to estimate pedestrian and cycling trip generation, distribution and route assignment. We do not discuss the mode choice estimation here, which can be done externally, before defining pedestrian trip generation and distribution rates, or in the calibration phase, based on observed count data.

3.4. Trip generation

The number of trips generated from each Origin corresponds to a specified weight column in the Origin points input layer. If a set of residential address points or Census block *cent*roids are used as Origins, for instance, each point object on the layer can each include an attribute indicating the number of residents, which determines the number of pedestrian journeys generated from each point. If a number of different

Table 3
Example pairing table used in the Betweenness function to estimate pedestrian flows.

Network Cost	Origin	OriginWeight	Destination	Destination Weight	Radius	Beta	Decay	Decay Mode	Closest destination	Detour	Turn Penalty
Geometric	Homes	pop total	Schools	CAPACITY	800	0.001	TRUE	exponent	FALSE	1.15	FALSE
Geometric	Homes	pop total	Metro	line ent st	800	0.001	TRUE	exponent	FALSE	1.15	FALSE
Geometric	Homes	pop total	Parks	area sqm	800	0.001	TRUE	exponent	FALSE	1.15	FALSE
Geometric	Homes	pop total	Jobs	EMPNUM	800	0.001	TRUE	exponent	FALSE	1.15	FALSE
Geometric	Homes	pop total	Amenities	cnt	800	0.001	TRUE	exponent	FALSE	1.15	FALSE
Geometric	Jobs	EMPNUM	Metro	line ent st	800	0.001	TRUE	exponent	FALSE	1.15	FALSE
Geometric	Jobs	EMPNUM	Amenities	cnt	800	0.001	TRUE	exponent	FALSE	1.15	FALSE
Geometric	Amenities	cnt	Amenities	cnt	800	0.001	TRUE	exponent	FALSE	1.15	FALSE

Origin types are used for trip generation, then a pairing table CSV can be used to specify the Origin types and weights, as shown in (Table 3). If no Origin weighs are specified, a default count option is used, simply routing one trip from each Origin.

While the weight field generally determines the number of trips generated from each Origin, this trip generation rate can also be set to be elastic with respect to destination availability. Pedestrian travel behavior literature has shown that when people have better access to Destinations within reasonable walking radii, they generate more per capita trips to such destinations on foot (Clifton et al., 2016; Sevtsuk and Kalvo, 2018; Bas et al., 2023). The Madina pedestrian flow simulation tool allows users to specify whether to enable this induced demand elasticity effect using the optional "Elastic Weight" column in the pairing table. If enabled, Madina first computes a KNN access score from each Origin to the given Destination type, ranging between 0 and 1, and then multiplies the given Origin weight with this KNN accessibility score (see (Sevtsuk et al., 2024) for more details). At locations with high Destination access (e.g. KNN access = 1), Origin weights are multiplied by one and trip generation stays unchanged. At locations with lower levels of Destination accessibility, Origin weights are multiplied by KNN access values between zero and one, proportionately reducing pedestrian trip generation rates to the given Destination type from such Origins.

3.5. Trip distribution

From each Origin, trips are distributed to all Destinations that are found within a specified network search radius (e.g. 800 m or 1/2 mile) from the Origin using the Huff Destination choice model (Huff, 1963b; Sevtsuk et al., 2024). The probability that a person at origin point [j] will patronize a particular Destination point [k] is given as a ratio between [j]'s gravity accessibility to that particular Destination, divided by [j]'s sum of gravity accessibilities to all available Destinations, including [k] (Sevtsuk and Kalvo, 2022):

$$\text{Probability}[j,k] = \frac{\text{Gravity}[j,k]}{\sum_{\text{Gravity}[j]}^{r}} \quad (4)$$

Since the gravity calculations are determined by both the proximity of the Destinations and the attractiveness of the Destinations (e.g. Destination weight), the probability of visiting any Destination is determined by both factors simultaneously as well as the presence of competing Destinations. This probability is constrained between 0 and 1. The number of trips that are sent from Origin [j] to a particular Destination [k] is found by multiplying the weight of [j] (optionally the elastic weight described above) with the probability [j,k]:

$$\text{Trips}[jk] = W[j] \bullet \text{Probability}[jk] \quad (5)$$

Given that the Gravity accessibility accounts for both spatial proximity and destination attractiveness, Destinations with greater destination weights (defined in the pairing.csv) and those that are reachable with lower travel costs (e.g. nearer Destinations) obtain larger probabilities and proportional trip allocations, while the total number of trips generated to all Destinations combined remains constrained by the origin weight. How Destination proximity affects destination probabilities is controlled by an exponential decay function in the Gravity accessibility function (Eq. 2) with a user-specified beta coefficient that represents an assumption about pedestrians' sensitivity to distance in a given context, which can also be defined in the pairing table (Table 3). Note that unlike many microscopic pedestrian travel behavior models (Horni et al., 2016; Clifton et al., 2016; Zhang et al., 2023), which use a destination choice model to determine the single most likely Destination for each trip, the UNA framework here allocates a proportion of trips simultaneously to all available Destinations within the given search radius. Even if only one trip is generated from an Origin, this trip is divided into fractional parts and dispatched to all available Destinations, with the parts corresponding to Destination probabilities (Sevtsuk, 2021b). This allows the model to capture idiosyncratic Destination preferences of different user types.

3.6. Route assignment

Once trip generation and distribution are set, Madina computes route assignments for each O-D pair. Again, unique to the UNA framework, we do not seek a single highest likelihood route for each trip (e.g. the least-cost route) but rather find a number of plausible routes and assign each of them a proportional share of trips, corresponding to the route costs. Plausible routes are determined as routes that are up to a given"detour ratio" longer than the shortest available path, also set in the pairing table (Table 3). Detour ratio represents the ratio between the allowable path length and the shortest available path length: a detour ratio of "1.15″ finds all routes that are up to 15 % longer than the shortest route. All paths that are identified within the detour ratio constraint between an O—D pair are weighted equally.[7] A more detailed and illustrated explanation of the UNA route assignment algorithm is provided in Supplementary Materials. Technical details and an in-depth description of the procedures and outputs is provided in Madina documentation.[8]

Though we lack room in this paper to demonstrate the empirical validity of the UNA pedestrian flow estimation methods implemented in Madina, we invite readers to consult other papers, where the same UNA methods (albeit outside of the Madina Python environment) have been validated with observed pedestrian route choice behavior in San Francisco, CA (Sevtsuk and Kalvo, 2021) and with pedestrian count data in Melbourne, Australia (Sevtsuk et al., 2021).

4. New York City analysis example

We illustrate the implementation of Madina's pedestrian accessibility metrics and pedestrian flow estimates in New York City—one of

[7] As part of future implementation, we are working on a sampling framework that can penalize longer paths over shorter paths. See Sevtsuk and Kalvo (2021) for reference.

[8] Technical documentation is available here: https://madinadocs. readthedocs.io/en/latest/

Table 4
Data sources for NYC pedestrian volume model.

Layer	Data Source	Resolution
Sidewalk Network	NYC Planimetric	Sidewalk, crosswalk and footpath segments, corrected and validated with both automated and manual cleaning procedures.
Home locations	Census Data	Original data at the Census block level was converted to n center-points along each perimeter edge of the block, assigning each point 1/n of the original block population.
Job locations	InfoGroup USA 2019 Business Historical Data	Address level points, describing the number of employees at each business establishment.
Commercial Amenity locations	InfoGroup USA 2019 Business Historical Data	Address points, including NAICS codes: 44–45 (retail), 722 (food and beverage services), 491, 811–812 (personal services), and 7111, 712, 713 (entertainment services).
School locations	NYC.gov	Schools, with school student capacity information as weights.
MTA metro train entrances	NYC.gov	Each subway station is represented by one or multiple entry points. Each entry is assigned a weight w, where w = number of lines in station / number of entries to station.
Park points	NYC.gov	Polygons of NYC parks were converted into a 100 × 100-meter grid of points, such that each point represents a unit of area.T his point-grid representation allows for granular, partial accessibility estimation to large urban parks.

the largest and most complex pedestrian networks in the U.S., housing approximately 8.5 million people. We conduct our analysis using a detailed pedestrian infrastructure network comprising all sidewalks, crosswalks and footpaths in NYC, which was assembled, corrected and checked during an extensive data preparation process in our research group. This network was topologically interconnected for trip routing, and visually validated and corrected over 2019 aerial imagery in NYC. [9] Analyses of this scale would be prohibitive in Rhinoceros 3D or ArcGIS versions of Urban Network Analysis due to performance and memory issues. The results below were computed on a regular desktop computer with an 18-core 3.00 GHz Intel® Xeon® W-2295 processor and 128GB of RAM.

We use the Census Block residential population estimates as Origins for the analysis below–these are the Origin points from which accessibility is evaluated, and from which pedestrian trips are generated. Because each Census Block centroid would create a single point object at its centroid, we have further split each Block into multiple frontage points, one per each perimeter street around the Block, allocating each frontage an equal fraction of the Block's original population count. Table 4 describes the data sources we use for the analyses below.

4.1. Accessibility analyses

Fig. 2 (a) illustrates the Reach accessibility index from New York City Census Blocks to MTA metro stations (shown in red dots) within an 800 m walking radius. The color-coded values show that pedestrians in the zoomed-in area of the city can reach up to four subway stations from some Block fronts (highest values colored in dark red), while the gray origins reach no subway stations within the given walking radius. This city-wide calculation, using 120,254 origins (frontage points), 1868 destinations (subway entrance points), and a pedestrian network of 315,577 edges took 2 h, 4 min and 49 s (7,480 s) to complete on the above hardware.

The run-times for both Madina accessibility and pedestrian flow calculations can be significantly reduced (e.g. by about 100×) if feather file type inputs are used. This functionality is available for advanced users.

Fig. 2 (b) shows the implementation of the Gravity index for the same Origins, Destinations, an 800 m search radius, and a beta value of"0.001″ in meters. While Reach accessibility in Fig. 2 (a) follows discrete levels that change when new Destinations become available within the search radius, Gravity accessibility Fig. 2 (b) yields a continuous spatial gradient, where even immediately neighboring Origins have different accessibility results. This is because each origin has a unique distance to the Destination set around it, which is not accounted for in the Reach index. Note how the Gravity values are also lower than Reach results because of the distance function in the denominator of the index. Using the same city-wide inputs as above, the Gravity calculation also took the same 7480 s to complete as Reach, since the two indices can be computed simultaneously in the `una.accessibility` function.

Fig. 2 (c) illustrates the implementation of the KNN accessibility function. Using address level business establishment data from InfoGroup, we specified a basket of daily amenities as Destinations, similar to those used in the popular WalkScore metric, discussed above (Infogroup, 2019; WalkScore, 2021). The Destinations and settings used for the KNN access results are shown in Table 2. The figure shows how considering this particular bundle of Destinations yields a perfect accessibility score of"1″ in many dense parts of Manhattan and Brooklyn, while parts of Staten Island, the Bronx, Queens and eastern Brooklyn exhibit lower-than-one levels of pedestrian access. The city-wide KNN accessibility calculation took 2 h, 37 min, and 19 s (9,439 s) on the above dataset and hardware using GeoJSON input files.

4.2. Pedestrian flow analysis

Fig. 3(a) illustrates how the UNA betweenness algorithm in Madina distributes trips from a single Census Block in NYC, whose population count we have evenly redistributed to four street fronts for higher granularity, to the nearest MTA metro rail station along all routes that are up to 15 % longer than the shortest route (detour ratio = 1.15). Line weights symbolize the resulting foot-traffic estimates on network edges. Fig. 3(b) expands the destination set to all metro station entrances that are available within an 800 m walkshed from the origin points. Note, how due to the Huff Destination choice model, not all trips accrue to the nearest MTA station, but to all available stations withing the search radius, with the nearer station entrances attracting slightly more trips. We use the count of subway lines at each station as the Destination weight (Destination Weight column of Table 3), so stations with more lines also attract more trips. Note that here we only illustrate flows from one trip type (homes to metro stations), which correspond only to the third row of pairings shown in Table 3.

Fig. 3(c), and its zoom detail 3(d), illustrate the same analysis expanded to all Census Block frontages and all metro entrances throughout the pedestrian network of NYC. The city-wide computation of home-to-metro-station flows took 3 h, 31 min and 34 s (12,694 s) on the abovementioned hardware.

These simulated trips do not represent actual pedestrian flows from homes to subway stations, as they depend on trip generation weights used at Origins, and remain uncalibrated at this stage. Nevertheless, these raw estimates can be useful for examining critical routes from homes to transit stations. A similar pairing between homes and school locations, for instance, can be used to highlight critical network

[9] The underlying source for the network was the NYC planimetric pedestriannetwork dataset (https://github.com/CityOfNewYork/nyc-planimetrics), which shows most pedestrian network segments, but which had not been prepared for topological connectivity and routability. Our research team used both automated and manual corrections across all parts of this network to make it routable, checking connectivity manually, intersection by intersection across the whole city by overlaying the network with 2019 aerial imagery in GIS

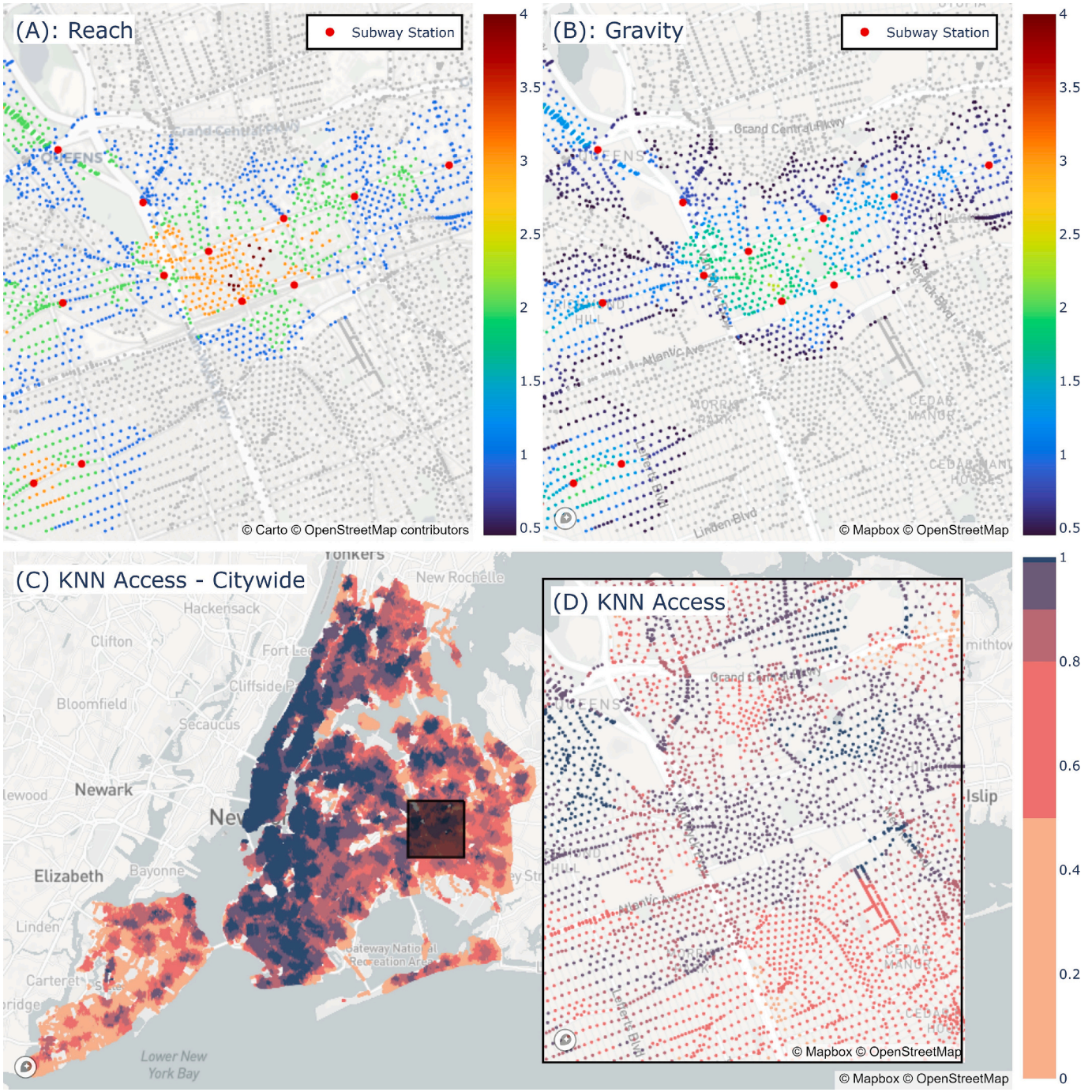

Fig. 2. Three Accessibility measures between homes and metro stations. The square in [C] shows where Jamaica, Queens is relative to New York City, and shown in [a, b, d] [a] Reach from each Census block front, to metro stations within an 800 m walk along the sidewalk network [b] Gravity accessibility from Census Block fronts to metro station entrances, within an 800 m walkshed, and a beta of 0.001 [c] City-wide KNN access scores for Block frontages in New York City based on the specifications in Table 2, [d] showing the KNN access scores around the Jamaica neighborhood in Queens. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

segments for pedestrian journeys between homes and schools, even if their estimated values are uncalibrated.

The pedestrian flow estimates from the Madina betweenness tool can be calibrated to the actual pedestrian flow volumes on each network segment. Observed pedestrian count data from multiple locations across the network for typical time periods (e.g. AM or PM peak hour on weekdays) can be used to fit a regression model for a given time interval, using regression parameters (or functions in the case of machine learning models) to scale individual flow estimates to their relative contributions to total pedestrian counts during the specified time intervals. Since these calibration procedures can be performed using different approaches (e.g. ordinary least squares, different machine learning models, ridge regressions etc.), we do not describe them in detail here, but refer the reader to (Sevtsuk et al., 2021; Sevtsuk et al., 2024) for more details.

Comparing pedestrian flow estimates in existing and proposed settings can help analysts predict how proposed interventions contribute to both the generation and distribution of pedestrian trips in study areas. This demonstrates a core value of having a pedestrian model–as the impacts of planned changes cannot be observed by definition (i.e. they

Fig. 3. (a) Estimated trips from a single Census block, whose population counts we have evenly redistributed to four street-fronts for higher granularity, to the nearest metro station (with two competing entrances) along all routes that are up to 15 % longer than the shortest route. (b) Same analysis as (a) is expanded to all six metro stations that are available within an 800 m walkshed from the Origin block. (c) The same analysis as (b) expanded to all metro stations and Census Blocks in NYC, with [d] zooming to a part of the Bronx to illustrate the spatial resolution of the pedestrian flow estimates. The city-wide computation of home-to-metro-station flows took 3 h, 31 min and 34 s (12,694 s) on the above-mentioned hardware using GeoJSON file inputs.

don't exist yet), a model can guide decision makers to prioritize infrastructure improvements at locations that affect most constituents, or fast track permitting approvals for projects that increase, rather than decrease pedestrian activity.

5. Summary and discussion

5.1. Workflows for scalable, reproducible and expandable pedestrian analysis

Though several of the analysis workflows presented in Madina have been available as separate functions or tools in the Rhinoceros3D UNA toolbox (Sevtsuk and Kalvo, 2024), Madina introduces several unique contributions. First, Madina offers the first end-to-end Python package for importing, analyzing and visualizing pedestrian or bicycle trips over

spatial networks, without requiring any proprietary tools or separate desktop software. This makes spatial network analysis workflows for non-motorized mobility modeling available to a wider audience. Second, the implementation of the tools in an open-source Python environment enables users to use pedestrian accessibility and flow modeling tools in innovative and custom ways, creating their own combinations of analysis procedures and combining complex analysis routines into joint workflow code. Such customization has so far been unavailable in the Rhino UNA tools, where users are instead required to manually specify different analysis procedures step-by-step, requiring more time and producing more opportunity for error.

Regarding the UNA analysis functions themselves, Madina also introduces several new tools for pedestrian network analysis. In particular, the KNN accessibility analysis has not been implemented in the Rhinoceros3D UNA toolbox so far. Implementation in Madina, with pairing CSV tables as user friendly inputs, allows access scores to different Destination types to be conveniently aggregated into a single WalkScore-type accessibility score. Also unique to Madina is the option to use the KNN accessibility results as trip-generation elasticity coefficients in the `una.betweenness` function, so as to temper the trip generation levels according to Destination availability around each unique location. Computing several different types of O-D flows in Madina is substantially easier than specifying analogous flows manually in Rhinoceros3D for each flow type separately, along with analysis parameters that must be entered by hand. Furthermore, Madina implements an efficient algorithm for finding pedestrian detours and redundant routes that are up to a given percentage longer than the lowest-cost path. Routing trips along numerous plausible routes, instead of a single lowest cost route, is computationally notoriously expensive (see Madina documentation for details). Madina's implementation of combinatorial route assignment, along with its multi-threaded application, enables redundant routes to be analyzed for large datasets that have been so far prohibitive in other platforms.

Beyond the specific contributions to non-motorized trip modeling, Madina also contributes to urban science software tools in general, by providing a platform to help promote scalable, reproducible and expandable urban research. This is supported by layers, network management, and interoperability capabilities. Though it may sound trivial, converting a shapefile (SHP) into a network, running network analysis tasks on it, and then passing the results back to a shapefile, and creating visual results, typically requires using several open-source libraries that are challenging to integrate. Madina streamlines the process in an end-to-end workflow.

Most published urban science research suffers from difficulties in reproducibility, since multiple steps of the analysis are carried out manually across multiple software environments. A key advantage for maintaining a single script for a research project, is that every step is explicitly documented. All utilized functions, along with their inputs, outputs, and parameter settings are recorded in the script. This makes it possible for the research community at large to inspect the process and help identify issues. Attempts to reproduce published research that lacks this level of documentation often fail to produce identical results even with identical inputs.

When using fragmented software to carry out spatial analysis procedures, a step that depends on a CAD software must be carried out completely before starting a following step that depends on GIS, which then needs to be carried out completely before starting a statistical analysis in Statahis sequential process makes it hard to collaborate, manage files, diagnose and detect mistakes, and might result in work repetition if an error is made in an earlier stage of the process. Using a unified Python script that depends on a raw data folder, eliminates the need to store, track, pass, and exchange intermediate results, reducing chances for error. It also allows analytic steps to be carried out concurrently by multiple people. Researchers working on intermediate steps of the analysis could use synthetic or sample input. When all the steps are completed, it becomes simple to integrate all steps in a single script. *Re*-running the script after fixing an error is significantly less time consuming than having to repeat multiple tasks, multiple times.

5.2. Use cases

Pedestrian accessibility and flow modeling workflows available in Madina have value for both planning and policy practice and research. Detailed, address-level accessibility measurements using the Reach, Gravity or KNN access score can help document variations in present destination availability, for instance among grocery stores, green spaces, daycare centers, transit stops, etc. These estimates can help guide planning decisions about where new facilities are needed, where new housing could be zoned to facilitate access to existing Destinations, or where changes in the networks (e.g. new sidewalks, through-block walkways) can improve pedestrian access. Accessibility outcomes can also be measured and compared among users with different physical ability– peripatetic users who need no help walking, users with walking-aids, users in wheelchairs.

Pedestrian flow estimates from Madina can have a range of practical applications. The most trafficked routes to transit stations, schools or other public destination can inform where to conduct pedestrian safety and comfort audits, and where necessary, intervene with infrastructure improvements. As mentioned above, the raw estimates of pedestrian flow can be calibrated against observed pedestrian counts to achieve model of total dominant foot-traffic during given time periods, as demonstrated in (Sevtsuk, 2021a; Sevtsuk et al., 2021; Sevtsuk et al., 2023). This can offer a pedestrian equivalent of average daily traffic (ADT) estimates, widely available for vehicular traffic and often used for infrastructure investment decision. This can help allocate city resources for pedestrian realm improvements where they impact most constituents (Sevtsuk et al., 2023). Pedestrian volume estimates can also provide an important denominator for various hazard data–-traffic crashes, urban heat, noise, air pollution, etc. Normalizing raw incidence data with exposure levels can produce incident probabilities, whose distributions can substantially differ from raw incident locations. Finally, a pedestrian volume model can be used to forecast how changes in infrastructure, land uses, or urban design affect, and desirably increase, non-motorized mobility, thus helping city governments coordinate planning decisions in ways that increase the mode share of non-motorized trips (Sevtsuk et al., 2021, 2023).

5.3. Potential expansion and future work

As all open-source projects, there is ample room for expansion and improvement in Madina. We note four particular areas of need. First, for better interoperability, it would be desirable to adopt a wider range of data formats, such as tab separated values and raster data formats to enhance the generation of data inputs. Adding functionality to connect Madina to APIs to access external data sources would potentially eliminate the present need to store input layers as geoJSON or SHP files. There is also ample room to improve visualization capabilities.

Second, Madina enables integration with other Python libraries. The GeoPandas, NetworkX and Deck.GL libraries that Madina presently uses, provide a robust base for its current functionality. Integrating connections to other spatial analysis libraries, or statistical analysis and machine learning packages, could potentially streamline pedestrian flow model calibration, and other output analysis procedures within Madina, eliminating the need for additional steps.

Third, there is ample room to expand UNA functionality in Madina. As the research on pedestrian and cycling mobility evolves, so do the tools that model acive activity. The next version of Madina will already incorporate functionality to handle elevation data as impedance costs. Implementing directional flow outputs on network segments are also high on our development list.

And fourth, using existing tools that Madina already offers, users can produce and share more workflows for common urban analysis tasks.

The UNA accessibility and pedestrian flow workflows demonstrated in this paper emphasize the efficiencies of workflow-oriented spatial analysis tasks, specified in scripts. Other frequently needed spatial research analysis procedures can be explored, implemented and openly shared with analogous script-based workflows to expand pedestrian mobility research from one city or neighborhood to another.

Madina is freely available on GitHub at https://github.com/City-Form-Lab/madina with documentation at https://madinadocs.readthedocs.io.

Funding

The work was supported by a DAR Group grant at MIT, administered by the Leventhal Center for Advance Urbanism entitled “Beirut Community Streets”.

CRediT authorship contribution statement

Andres Sevtsuk: Writing – original draft, Validation, Supervision, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Abdulaziz Alhassan:** Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization.

Declaration of competing interest

None.

Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.jtrangeo.2025.104130.

Data availability

Data will be made available on request.

References

Bas, J., Al-Khasawneh, M.B., Erdogan, S., Cirillo, C., 2023. How the Designof Complete Streets Affects Mode Choice: Understanding the Behavioral Responses to the Level of Traffic Stress, Transportation Research Part a: Policy and Practice, 173, p. 103698. https://doi.org/10.1016/j.tra.2023.103698. https://www.sciencedirect.com/science/article/pii/S0965856423001180.

Batty, M., 2003. Agent-Based Pedestrian Modelling. https://discovery.ucl.ac.uk/id/eprint/237/.

Bhat, C., Handy, S., Kockelman, K., Mahmassani, H., Chen, Q., Weston, L., 2000. Development of an Urban Accessibility Index: Literature Review.

Boeing, G., 2017. OSMnx: new methods for acquiring, constructing, analyzing, and visualizing complex street networks. Comput. Environ. Urban. Syst. 65, 126–139. https://doi.org/10.1016/J.COMPENVURBSYS. 2017.05.004.

Cervero, R., K., 1997. Kockelman, sity, diversity, and design. Transp. Res. Part D: Transp. Environ. 2, 199–219. http://www.sciencedirect.com/science/article/pii/S1361920997000096.

City of Boston, 2022. Go Boston 2030. https://www.boston.gov/departments/transportation/go-boston-2030#report-chapters.

City of Los Angels, 2019. Los Angeles Green New Deal Sustainability Plan. https://plan.lamayor.org.

Clifton, K.J., Singleton, P.A., Muhs, C.D., 2016. Development of destination choice models for pedestrian travel. Transp. Res. A Policy Pract. 94, 255–265. https://doi.org/10.1080/01944363.2020.1864758 (https://doi.org/10.1016/j.tra.2016.09.017).

Cooper, C.H., Chiaradia, A.J., 2020. sDNA: 3-d Spatial Network Analysis for GIS, CAD, Command Line & Python, SoftwareX, 12, 100525. https://doi.org/10.1016/J.SOFTX.2020.100525.

Crucitti, P., Latora, V., Porta, S., 2006. Centrality in networks of urban streets. Chaos 16 (1). https://doi.org/10.1063/1.2150162/321923 (URL /aip/cha/article/16/1/015113/321923/Centrality-in-networks-of-urban-streets).

J. V. den Bossche, K. Jordahl, M. Fleischmann, J. McBride, M. Richards, J. Wasserman, A. G. Badaracco, A. D. Snow, J. Gerard, J. Tratner, B. Ward, M. Perry, C. Farmer, G. A. Hjelle, M. Taves, E. ter Hoeven, M. Cochran, Raymondgh, M. Bartos, R. Bell, L. Culbertson, G. Caria, N. Eubank, Sangarshanan, J. Flavin, S. Rey, J. Gardiner, Kaushik, Maxalbert, Geopandas (2024). doi:https://doi.org/10.5281/zenodo.10460075.

Dogan, T., Samaranayake, S., Saraf, N., 2018. Urbano: A new tool to promote cess analysis for amenities and public transport. In: SIMAUD ‘18: Proceedings of the Symposium on Simulation for Architecture and Urban Design, pp. 1–9. https://doi.org/10.5555/3289750.3289778.

Environmental Protection Agency, Ogy and User Guide, 2021. https://www.epa.gov/sites/default/files/2021-06/documents/national_walkability_index_methodology_and_ user_guide_june2021.pdf.

Freeman, L.C., 1977. A set of measures of centrality based on betweenness. Sociometry 40, 35–41.

Friis, C., Svensson, L., 2013. Pedestrian Microsimulation a Comparative Study Between the Software Programs Vissim and Viswalk MSc Thesis. https://hdl.handle.net/20.500.12380/185439.

Geurs, K.T., van Wee, B., 2004. Accessibility evaluation of land-use and transport strategies: review and research directions. J. Transp. Geogr. 12, 127–140.

S. Gillies, C. der Wel, J. den Bossche, M. W. Taves, J. Arnott, B. C. Ward, others, Shapely (2023). doi:https://doi.org/10.5281/zenodo.8436711.

Glaeser, E.L., 2011. Triumph of the City: How our Greatest Invention Makes us Richer, smarter, Greener, Healthier, and Happier. Penguin Press. https://www.penguinrandomhouse.com/books/303439/triumph-of-the-city-by-edward-glaeser/.

Grasser, G., Van Dyck, D., Titze, S., Stronegger, W., 2013. Objectively measured walkability and active transport and weight-related outcomes in adults: a systematic review. Int. J. Public Health 58 (4), 615–625. https://doi.org/10.1007/S00038-012-0435-0/FIGURES/1. https://link.springer.com/article/10.1007/s00038-012-0435-0.

Hagberg, A.A., Schult, D.A., Swart, P.J., 2008. Exploring network structure, dynamics, and function using networkX. In: Proceedings of the 7th Python in Science Conference, Pasadena, CA USA, pp. 11–15. In: https://conference.scipy.org/proceedings/SciPy2008/paper_2/.

Handy, S.L., Boarnet, M.G., Ewing, R., Killingsworth, R.E., 2002. How the built environment affects physical activity: views from urban planning. Am. J. Prev. Med. 23 (2), 64–73. https://doi.org/10.1016/S0749-3797(02)00475-0.

Hansen, W.G., 1959. How accessibility shapes land use. J. Am. Plan. Assoc. 25, 73–76.

Horni, A., Nagel, K., Axhausen, K.W., 2016. Lation MATSim. Ubiquity Press. https://doi.org/10.5334/BAW.

Huff, D., 1963a. A probabilistic analysis of shopping center trade areas. Land Econ. 39, 81–90.

Huff, D., 1963b. A probabilistic analysis of shopping center trade areas. Land Econ. 39, 81–90.

Infogroup, 2019. ReferenceUSA Business Historical Data Files, Harvard Dataverse. https://doi.org/10.7910/DVN/GW2P3G/VPA2UC. https://doi.org/10.7910/DVN/GW2P3G/VPA2UC.

Jaber, A., Wagner, N., Papaioannou, D., 2017. Benchmarking Accessibility to Services Across Cities. http://gaates.org/wp-content/uploads/2019/12/accessibility-proximity-transport-performance_2.pdf.

Kazerani, A., Winter, S., 2009. Can Betweenness centrality explain traffic flow?. In: In: 12th AGILE International Conference on Geographic Information Science, Hannover, Germany https://agile-gi.eu/images/conferences/2009/documents/111.pdf.

Krish, S., 2011. A practical generative design method. Comput. Aided Des. 43, 88–100. https://doi.org/10.1016/J.CAD.2010.09.009. https://www.sciencedirect.com/science/article/abs/pii/S0010448510001764.

Li, Y., Tsukaguchi, H., 2005. Relationship between network topology and pedestrian route choice behavior. J. East. Asia Soc. Transp. Stud. 6, 241–248.

Lue, G., Miller, E.J., 2019. Estimating a Toronto pedestrian route choice model using smartphone gps data. Travel Behav. Soc. 14, 34–42. https://doi.org/10.1016/j.tbs.2018.09.008. http://www.sciencedirect.com/science/article/pii/S2214367X18300425.

New York City, 2014. New York City’s Roadmap to 80x50. https://www.nyc.gov/assets/sustainability/downloads/pdf/publications/NewYorkCity%27sRoadmapto80x50_Final.pdf.

Olszewski, P., Wibowo, S.S., 2005. Using equivalent walking distance to assess pedestrian accessibility to transit stations in Singapore. Transport. Res. Record 1927, 38–45. https://doi.org/10.1177/0361198105192700105.

PTV, 2012. Pedestrians’ big Debut in Traffic Simulation PTV America, PTV Compass, pp. 4–8. https://www.yumpu.com/en/document/view/4277652/pedestrians-big-debut-in-traffic-simulation-ptv-america.

Puusepp, R., Looke, T., Cerrone, D., Männigo, K., 2018. Simulating pedestrian movement. In: Humanizing Digital Reality. Springer, Singapore, pp. 547–557. https://doi.org/10.1007/978-981-10-6611-5{_}46.

Sevtsuk, A., 2010. Path and Place: Path and Place : A Study of Urban Geometry and Retail Activity in Cambridge and Somerville, Ma.

Sevtsuk, A., 2018. ing pedestrian and bicycle trips in cities. https://cityform.mit.edu/projects/una-rhino-toolbox.

Sevtsuk, A., 2021a. Estimating pedestrian flows on street networks: revisiting the betweenness index. J. Am. Plan. Assoc. 87, 512–526. https://doi.org/10.1080/01944363.2020.1864758. https://doi.org/10.1080/01944363.2020.1864758.

Sevtsuk, A., 2021b. Estimating pedestrian flows on street networks. J. Am. Plan. Assoc. 87 (4), 512–526. https://doi.org/10.1080/01944363.2020.1864758. https://www.tandfonline.com/doi/abs/10.1080/01944363.2020.1864758.

Sevtsuk, A., Kalvo, R., 2018. Patronage of urban commercial clusters: a network-based extension of the Huff model for balancing location and size. Environ. Plann. B 45 (3), 508–528. https://doi.org/10.1177/2399808317721930.

Sevtsuk, A., Kalvo, R., 2021. Predicting pedestrian flow along city streets: a cisco. Int. J. Sustain. Transp. https://doi.org/10.1080/15568318.2020.1858377.

Sevtsuk, A., Kalvo, R., 2022. Predicting pedestrian flow along city streets: a comparison of route choice estimation approaches in downtown San Francisco. Int. J. Sustain. Transp. 16 (3), 222–236. https://doi.org/10.1080/15568318.2020.1858377. https://www.tandfonline.com/doi/abs/10.1080/15568318.2020.1858377.

Sevtsuk, A., Kalvo, R., 2024. Modeling pedestrian activity in cities with urban network analysis. Environ. Plann. B. https://doi.org/10.1177/23998083241261766.

Sevtsuk, A., Mekonnen, M., 2012. Urban Network analysis: a new toolbox for measuring city form in ArcGIS. In: SimAUD '12: Proceedings of the 2012 Symposium on Simulation for Architecture and Urban Design, pp. 1–10. https://doi.org/10.5555/2339453.2339471. https://dl.acm.org/doi/10.5555/2339453.2339471.

Sevtsuk, A., Basu, R., Chancey, B., 2021. We shape our buildings, but do they then shape us? A longitudinal analysis of pedestrian flows and development activity in Melbourne. PLoS One 16 (9), e0257534. https://doi.org/10.1371/JOURNAL.PONE.0257534. https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0257534.

Sevtsuk, A., Kollar, J., Pratama, D., Basu, R., Alhassan, A., Chancey, B., Haddad, J., Halabi, M., Makhlouf, R., Zeid, M. Abou, 2023. Bility in Beirut: an Approach Using Pedestrian Modeling, Participatory Design, and Scenario Analysis, SSRN, p. 4618238. https://doi.org/10.2139/SSRN. https://papers.ssrn.com/abstract=4618238.

Sevtsuk, A., Kollar, J., Pratama, D., Haddad, J., Basu, R., Alhassan, A., Chancey, B., Makhlouf, R., Halabi, J., Abou-Zeid, M., 2024. Pedestrian-oriented development in Beirut: a framework for estimating urban design impacts on pedestrian flows through modeling, participatory design, and scenario analysis. Cities 149. https://doi.org/10.1016/j.cities.2024.104927.

Urban Computing Foundation, 2016. Deck.gl: WebGL2 Powered Geospatial Visualization Layers. https://deck.gl.

US EPA, 2024. Sources of Greenhouse Gas Emissions. https://www.epa.gov/ghgemissions/sources-greenhouse-gas-emissions.

WalkScore, 2021. Walk Score Api. https://www.walkscore.com/professional/api.php.

Ye, P., Wu, B., Fan, W., 2016. Modified Betweenness-Based Measure for Prediction of Traffic Flow on Urban Roads, 2563, pp. 144–150. https://doi.org/10.3141/2563-19.

Yen, J.Y., 1971. Finding the K shortest Loopless paths in a network. Manag. Sci. 17 (11), 712–716. http://www.jstor.org/stable/2629312.

Zacharias, J., 2001. Pedestrian behavior and perception in urban walking environments. J. Plan. Lit. 16, 3–18.

Zhang, Q., Moeckel, R., Clifton, K.J., 2022. Assessing pedestrian impacts of future land use and transportation scenarios. J. Transp. Land Use 15 (1), 547–566. https://doi.org/10.5198/JTLU.2022.2117. https://www.jtlu.org/index.php/jtlu/article/view/2117.

Zhang, Q., Moeckel, R., Clifton, K.J., 2023. MoPeD meets MITO: a hybrid modeling framework for pedestrian travel demand. Transportation 1–21. https://doi.org/10.1007/S11116-022-10365-X/FIGURES/11.