The Journal of Open Source Software

# Georouting: An AI-friendly Python Library for Routing and Origin–destination (OD) Matrix Calculation

**Xiaokang Fu** [1] **and Devika Jain** [2]

**1** State Key Laboratory of Information Engineering in Surveying, Mapping and Remote Sensing, Wuhan University, 129 Luoyu Road, Wuhan, China **2** Center for Geographic Analysis, Harvard University, 1737 Cambridge Street, MA, USA

**DOI:** 10.xxxxxx/draft

**Software**

- Review
- Repository
- Archive

**Editor:**

**Submitted:** 19 November 2025
**Published:** unpublished



## Summary

georouting is a Python package that provides a unified interface to multiple web routing services and open-source local routing solutions, currently including Google Maps, Bing Maps, Esri services, Baidu Maps, TomTom Maps, Here Maps, Mapbox, OpenRouteService, an OSRM local/demo server (Luxen & Vetter, 2011) with helper functions to auto-build and start Docker or Singularity backends (suitable for HPC environments), and routing through OSMnx (Boeing, 2025) and iGraph (Csardi & Nepusz, 2006). It enables researchers and practitioners to compute travel routes, travel times, and travel distances between large numbers of origin–destination (OD) pairs using a consistent and high-level API. The package is designed to integrate naturally with the scientific Python ecosystem by returning results as pandas DataFrames (McKinney & others, 2011) and geopandas GeoDataFrames (Jordahl et al., 2020), and by exposing convenient methods for map-based visualization of routes and accessibility patterns. In many applied domains—such as transport planning, health-care accessibility, logistics, and urban analytics—accurate travel-time and distance estimates are needed to replace simple Euclidean distances or crude network approximations. georouting lowers the barrier to using production-grade web routing services by handling provider-specific request formats, batching strategies, and response parsing behind a single API. For OSRM, it includes Geofabrik region utilities (with optional size lookup) and a one-shot helper that downloads extracts, builds, and launches a local service with the chosen profile (car/foot/bicycle) via Docker or Singularity/Apptainer. The package is available on both PyPI and conda-forge, comes with automated tests and continuous integration, and includes AI-oriented documentation (via `llms.txt` (Jeremy Howard, 2024)) to make it easier for users to get support from large language models. The package is open source (MIT license) and available at https://github.com/wybert/georouting.

## Statement of Need

There is a growing need in the health-care, geospatial, and data-science communities for tools that can compute realistic travel distances and travel times at scale. There are many tools available including web-based routing services like Google Maps, Bing/Azure Maps, Esri, Baidu, TomTom, Here, Mapbox, OpenRouteService, and open source routing engines like OSRM and pgRouting (pgRouting contributors, 2025). Researchers often need to compare results from different providers or switch providers due to coverage, accuracy, pricing, or licensing constraints. However, each tool has its own workflow and often limited built-in visualization support, making it difficult to cross verify results or switch providers without significant refactoring. georouting addresses this gap by providing a single, extensible, Pythonic, and AI-friendly routing interface that abstracts over tool-specific details while still exposing tool-specific strengths. Users can change routing methods by switching a class (e.g., `GoogleRouter`, `OSRMRouter`, `EsriRouter`, `MapboxRouter`, `TomTomRouter`, `HereRouter`,

The Journal of Open Source Software

`OpenRouteServiceRouter`, or an `OSMnx`-based router) without refactoring their analysis pipeline. Common operations—such as computing a route between two coordinates, building a full OD distance matrix, or computing distances only for specific OD pairs—are implemented in a backend-agnostic way. It also provides a `plot_route()` method to visualize the routing results on an interactive map. We provide llms.txt files to make it easier for users to get AI-powered help when working with the package using vibe coding tools like Cursor or Claude Code. For users who need a local engine, `georouting` ships helper functions to fetch Geofabrik extracts (with optional size metadata), generate an OSRM Dockerfile, and auto-build/run an OSRM backend with the desired profile (car/foot/bicycle) and region via Docker or Singularity/Apptainer (suited to HPC environments). The package is intended for geospatial researchers, transportation and health geographers, operations researchers, and data scientists who need reproducible and scalable routing and accessibility analyses from Python. `georouting` has been used in the rapidroute project (Xiaokang Fu and Devika Kakkar, 2024), enabling researchers to compute drive times for millions of OD pairs at the national scale within minutes on high-performance computing systems, as well as in a study evaluating drive-time estimation methods on geospatial big data (Fu et al., 2023).

## Georouting Functionality and Design

The core of `georouting` is a set of router classes that inherit from a common `WebRouter` base class. Each router encapsulates the logic for interacting with a specific routing service while sharing a common public interface:

- `get_route(origin, destination)` returns a route object with methods to access total distance, travel time, detailed step-by-step information, and `plot_route` function to visualizing the route on a map, as well as a `GeoDataFrame` representation of the route.
- `get_distance_matrix(origins, destinations, append_od=True)` builds a full distance or duration matrix between sets of origins and destinations, returning a `DataFrame` where rows correspond to OD pairs.
- `get_distances_batch(origins, destinations, append_od=True)` computes distances and durations only for specific OD pairs, using internal batching strategies to respect provider API limits.

The underlying route objects (e.g., `GoogleRoute`, `BingRoute`, `OSRMRoute`) are wrappers around the raw JSON responses returned by each provider. They expose a small, consistent set of methods such as `get_distance()`, `get_duration()`, `get_route()`, and `get_route_geopandas()`. For map-based visualization, route objects can be plotted directly, producing interactive maps (e.g., via `folium`) suitable for exploratory analysis in a Jupyter environment (see Figure 1).

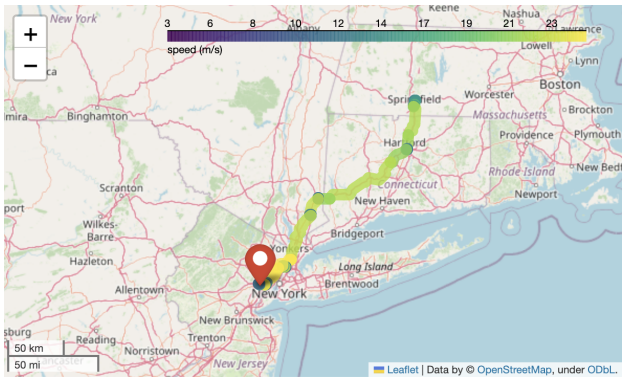


**Figure 1:** Figure.

To support large OD problems, `georouting` implements helper utilities that partition OD pairs

into batches that respect provider-specific API limits while avoiding redundant calls. These utilities work with vectorized data structures (e.g., Numpy arrays or pandas DataFrames) and are designed to minimize the number of remote API requests. For users needing on-prem/HPC execution, `georouting` ships OSRM helpers that can download Geofabrik extracts, build images, and launch local OSRM backends via Docker or Singularity/Apptainer—allowing high-throughput OD computation without relying on external APIs. The package is written in pure Python and integrates closely with widely used geospatial libraries such as geopandas and `osmnx`, enabling workflows that combine web-based routing with open street network analysis.

The project follows standard Python packaging practices, ships with an automated test suite that covers core routing and batching logic (including OSRM-based tests that do not require API keys), and is continuously tested on multiple Python versions using GitHub Actions. Documentation is built with MkDocs, including narrative tutorials, usage examples, and API reference pages generated from docstrings. In addition, the project maintains `llms.txt` and `llms-full.txt` files to provide machine-readable documentation for large language models, making it easier for users to obtain high-quality automated assistance.

## Acknowledgements

The development of `georouting` was inspired by the design of the geopy package (Contributors, 2025) for geocoding in Python, as well as by the growing ecosystem of open-source geospatial tools such as geopandas and `osmnx`. The author thanks the maintainers of these projects. The author also thanks early users and contributors who provided feedback on API design, documentation, and test coverage.

## References

Boeing, G. (2025). Modeling and analyzing urban networks and amenities with OSMnx. *Geographical Analysis*, *57*(4), 567–577. https://doi.org/10.1111/gean.70009

Contributors, G. (2025). *Geopy/geopy*. geopy.

Csardi, G., & Nepusz, T. (2006). The igraph software package for complex network research. *InterJournal, Complex Systems*, 1695.

Fu, X., Kakkar, D., Chen, J., Moynihan, K., Hegland, T., & Blossom, J. (2023). A comparative study of methods for drive time estimation on geospatial big data: A case study in USA. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *48*.

Jeremy Howard. (2024). *The /llms.txt file*. https://github.com/answerdotai/llms-txt

Jordahl, K., Bossche, J. V. den, Fleischmann, M., Wasserman, J., McBride, J., Gerard, J., Tratner, J., Perry, M., Badaracco, A. G., Farmer, C., Hjelle, G. A., Snow, A. D., Cochran, M., Gillies, S., Culbertson, L., Bartos, M., Eubank, N., maxalbert, Bilogur, A., … Leblanc, F. (2020). *Geopandas/geopandas: v0.8.1* (Version v0.8.1). Zenodo. https://doi.org/10.5281/zenodo.3946761

Luxen, D., & Vetter, C. (2011). Real-time routing with OpenStreetMap data. *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 513–516. https://doi.org/10.1145/2093973.2094062

McKinney, W., & others. (2011). Pandas: A foundational python library for data analysis and statistics. *Python for High Performance and Scientific Computing*, *14*(9), 1–9.

pgRouting contributors. (2025). *pgRouting: Routing on PostgreSQL*. https://doi.org/10.5281/zenodo.15004469

Xiaokang Fu and Devika Kakkar. (2024). *Rapidroute*. https://rapidrouteapp.onrender.com/routing_app