Osmenrich: An R package to enrich geocoded data

Leonardo Jaya Vida[1, 2], **Jonathan de Bruin**[1, 2], **and Erik-Jan van Kesteren**[1, 2]

1 Utrecht University **2** ODISSEI

DOI: DOIunavailable

Software

- Review
- Repository
- Archive

Editor: Pending Editor

Reviewers:

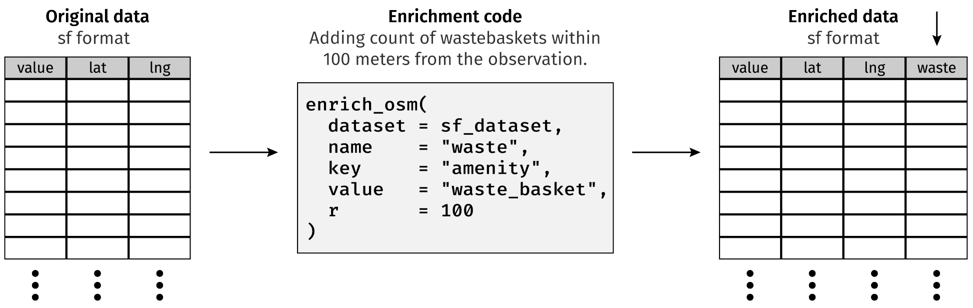
- @Pending Reviewers

Submitted: N/A
Published: N/A

Summary

The `osmenrich` package provides a user-friendly way to enrich geographic datasets in `R`, for example observations on a map, with geographic features around those observations. Additionally, it can weigh these features by their (walking / driving / cycling) distance or duration from the observations. This package builds on existing infrastructure to interface with OpenStreetMap (`osmdata`, Padgham et al., 2017), and works well with the existing ecosystem of packages for (geospatial) data analysis, namely `sf` (E. Pebesma, 2018) and the `tidyverse` (Hadley Wickham et al., 2019). This package streamlines and standardizes the process going from a raw dataset with observations and locations to a `tidy` (Wickham, 2014), rich dataset with multiple relevant geographical features about those locations. Figure 1 shows graphically the basic workflow of `osmenrich`.

The R package `osmenrich` is available on GitHub.



```
enrich_osm(
  dataset = sf_dataset,
  name    = "waste",
  key     = "amenity",
  value   = "waste_basket",
  r       = 100
)
```

Figure 1: Basic workflow of `osmenrich`.

Statement of need

Geographic data is valuable in research where the environment influences the process under investigation. For example, the `osmenrich` package is useful in the analysis of data retrieved from citizen science projects such as plastic spotter or the great backyard bird count. At the same time, within the R ecosystem multiple software solutions exist for extracting data from geographic information systems (Cooley, 2020; Padgham et al., 2017; Walker, 2020). However, to include these geographic data in further analysis (e.g. carrying out kriging in `gstat`) (E. J. Pebesma, 2004) the data often need further processing and, crucially, *aggregation*. Within this problem space, `osmenrich` contributes by:

- Creating a user-friendly interface to OpenStreetMap, abstracting away the necessary API calls (see section *Main function*).
- Defining standardized ways to aggregate geographic information based on kernels (see section *Aggregation*).

- Allowing distance measures based on routing, such as duration by foot or distance by car (see section *Routing*).

Using our package, researchers can focus on investigating research questions, rather than spending time figuring out how to aggregate geographic data. The `osmenrich` package is especially suited for questions surrounding interactions between a process and its close physical environment, such as gathering real-world feature objects within a determined distance from observations to improve a prediction process. Throughout this paper, objects with geocoded data are called "*reference objects*," while objects of interest are called "*feature objects*."

Main function

To enrich data, the `osmenrich` package uses the main function `enrich_osm()`. This function takes a dataset containing geocoded *reference objects* in `sf` format, retrieves specified *feature objects* from a local or remote OpenStreetMap server (see *Routing* section), computes the enrichment using specified parameters and outputs an enriched tidy `sf` dataset.

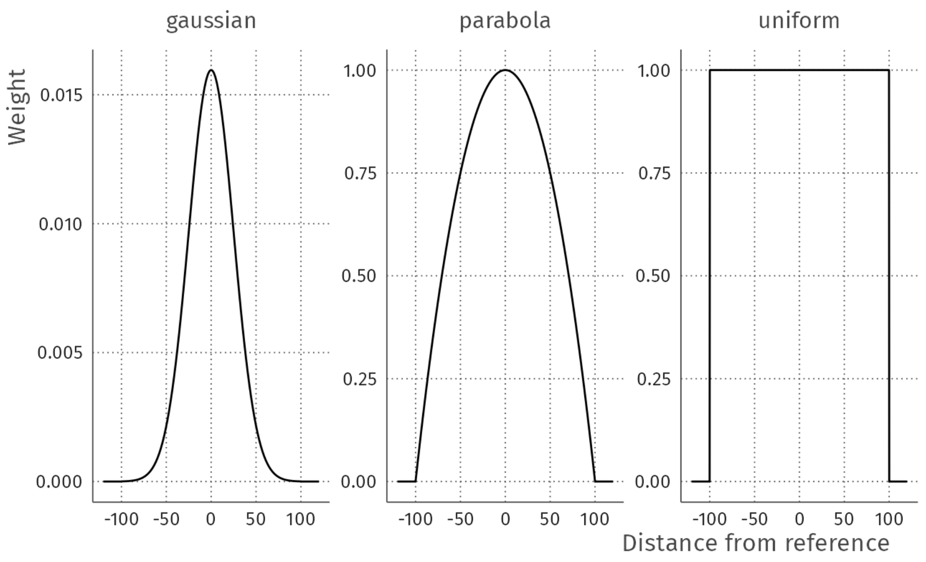
```r
enrich_osm(
  dataset = sf_dataset,
  name = "waste",
  key = "amenity", # Syntax borrowed from OpenStreetMap
  value = "waste_basket", # Syntax borrowed from OpenStreetMap
  r = 100
)
```

The code listing above shows an example of a basic enrichment of *reference points* with the number of waste baskets in the surrounding 100 meters. Specifically, the function uses the bounding box created by the *reference objects* from the input dataset and searches for the specified *feature objects* in OpenStreetMap with parameters `key` and `value` within the radius `r` (in meters) around each of the *reference objects*. The `key` and `value` parameters are also used as tags in OpenStreetMap to describe physical features of map elements. The user is able to search for them using official OpenStreetMap documentation. Finally the `enrich_osm` function creates a new column named after the parameter `name` containing the enriched data. See Section *Full usage example* for an example usage of this function.

Aggregation

To convert the retrieved features to a single number per reference object, an aggregation step is performed by `osmenrich`. In the `enrich_osm` function, there are three parameters that control aggregation: `kernel`, `reduce_fun`, and `measure`.

1. The `kernel` determines the weighing function applied on the distances (or durations) between the objects retrieved and the reference points. The `osm enrich` package provides three different kernels to be used out-of-the-box (`uniform`, `gaussian` and `parabola`) and defaults to `kernel = unifor m`. However, the package allows users to also specify custom-made kernels.



2. The aggregation function parameter `reduce_fun`, is used to reduce the weighted vectors of distances (or durations) into single numbers. This parameter defaults to `reduce_fun = sum`, however it accepts any standard R function, such as `mean` or `median`.

Specifying these variables in the `enrich_osm()` function allows the user to choose specific types of weights and aggregation to be applied on the feature objects retrieved from OpenStreetMap.

```r
enrich_osm(
  [...],
  r = 100, # Radius for feature objects retrieval
  kernel = "gaussian", # Weighing function
  reduce_fun = "mean" # Aggregation function
)
```

Routing

To retrieve *feature objects* around the *reference objects* and the distances (or durations) between them, the `osmenrich` package uses an OpenStreetMap server and, optionally, one or more instances of the Open Source Routing Machine (OSRM).

The package leverages publicly available servers to enable basic data enrichment without the need of setting up any local instance of these servers. However, for large data enrichment tasks and for tasks involving the computation of distances (or durations) between objects using specific profiles, the setup of one or more of these servers is required.

We created a GitHub repository hosting the instructions and the `docker_compose.yml` files needed to set up these servers. To facilitate the routing of users to the right setup for their needs, we provide three use cases and their respective recommended setup.

Once the desired server is set up, the user can set the parameter `measure` to specify which profile to use to compute distances (or durations) between the objects. Depending on the routing servers available, the `osmenrich` package can retrieve metrics computed on three different types of profile (`car`, `bike` or `foot`).

```r
# If available, specify the address of local OSRM instance
# options(osrm.server = "http://localhost:<port>/")
# Specify the address of the public or local Overpass (OSM) instance
# osmdata::set_overpass_url("http://localhost:<port>/api/interpreter")
enrich_osm(
  [...],
  measure = "distance_by_foot" # Choose a specific metric that leverages the OSRM
)
```

Full usage example

osmenrich is available on GitHub and can be installed and loaded into the R session with the remotes package.

```r
library(osmenrich)
```

We have included an example dataset of common swift's nests provided openly by the city of Utrecht, the Netherlands. A researcher may want to investigate the effect of natural material availability on nesting behaviour. In this example, we use "trees" as a proxy for natural material availability.

```r
head(common_swift)
# Simple feature collection with 6 features and 1 field
# Geometry type: POINT
# Dimension:     XY
# Bounding box:  xmin: 5.086195 ymin: 52.0879 xmax: 5.08662 ymax: 52.08866
# CRS:           EPSG:4326
# # A tibble: 6 x 2
#  nest_count             geometry
#        <dbl>          <POINT [°]>
# 1          1  (5.086195 52.0884)
# 2          2 (5.086602 52.08866)
# 3          1  (5.086619 52.0884)
# 4          1 (5.086604 52.08837)
# 5          1    (5.08633 52.0879)
# 6          1  (5.08662 52.08826)

# Enrichment step
enriched_common_swift <-
  common_swift %>%
  enrich_osm(
    name = "tree_1km",
    key = "natural",
    value = "tree",
    kernel = "gaussian",
    r = 1000
  )

head(enriched_common_swift)
# Simple feature collection with 6 features and 2 fields
# Geometry type: POINT
# Dimension:     XY
# Bounding box:  xmin: 5.086195 ymin: 52.0879 xmax: 5.08662 ymax: 52.08866
# CRS:           EPSG:4326
# # A tibble: 6 x 3
#   nest_count             geometry tree_1km
#        <dbl>          <POINT [°]>    <dbl>
# 1           1  (5.086195 52.0884)  0.00799
# 2           2 (5.086602 52.08866)  0.00692
# 3           1  (5.086619 52.0884)  0.00689
# 4           1 (5.086604 52.08837)  0.00693
# 5           1   (5.08633 52.0879)  0.00753
# 6           1  (5.08662 52.08826)  0.00688
```

As shown in the output of `enriched_common_swift`, following the enrichment step the dataset gains an additional column named `tree_1km`. This column contains the sum of the numbers of trees within 1km around each nest location, weighed by the `gaussian` kernel.

Acknowledgements

This work was supported by the Open Data Infrastructure for Social Science and Economic Innovations (ODISSEI) in the Netherlands (https://ror.org/03m8v6t10).

References

Cooley, D. (2020). *Googleway: Accesses google maps APIs to retrieve data and plot maps*. https://CRAN.R-project.org/package=googleway

Hadley Wickham et al. (2019). Welcome to the tidyverse. *Journal of Open Source Software*, *4*(43), 1686. https://doi.org/10.21105/joss.01686

Padgham, M., Rudis, B., Lovelace, R., & Salmon, M. (2017). Osmdata. *The Journal of Open Source Software*, *2*(14). https://doi.org/10.21105/joss.00305

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

Pebesma, E. J. (2004). Multivariable geostatistics in S: The gstat package. *Computers & Geosciences*, *30*, 683–691.

Walker, K. (2020). *Mapboxapi: R interface to 'mapbox' web services*. https://CRAN.R-project.org/package=mapboxapi

Wickham, H. (2014). Tidy data. *Journal of Statistical Software, Articles*, *59*(10), 1–23. https://doi.org/10.18637/jss.v059.i10