

B5 Das osmdata Paket

Jan-Philipp Kolb

23 Oktober 2018

Themen dieses Abschnitts

- Das R-Paket `osmdata` mit dem man OSM-Daten herunterladen kann und das auf der **Overpass API** beruht.
- Das Paket `osmplotr`



Mark Padgham - Import 'OpenStreetMap' Data as Simple Features or Spatial Objects

Das osmdata Paket

- Mit dem Paket kann man Daten von OpenStreetMap importieren
- Die OSM Daten sind unter **ODbL licence** zu haben

```
install.packages("osmdata")
```

```
library(osmdata)
```

```
citation("osmdata")
```

Das Paket osmplotr

```
library("osmplotr")  
library("osmdata")
```

Beispiel Kindergärten in Mannheim

```
bbox <- getbb("Mannheim")  
dat_osm <- extract_osm_objects(key='building',  
                                value="kindergarten",  
                                bbox=bbox)
```

Einen Rahmen definieren um Daten zu bekommen

- Der Rahmen kann entweder erstellt werden, indem die Koordinaten angegeben werden:

```
q <- opq(bbox = c(52.3, 4.7, 52.4, 5.1))
```

- oder indem man den Befehl getbb verwendet:

```
bb <- getbb('Ladenburg')
```

- In bb sind nun vier Werte gespeichert, die den Rahmen definieren
- Befehl opq - eine Overpass Anfrage erstellen

```
q <- opq(bbox = bb)
```

Die Grenze von Mannheim

- Erst mit dem Argument `format_out=polygon` Befehl `getbb` erhält man das Polygon:

```
bb_poly <- getbb(place_name = "Ladenburg",  
                 format_out = "polygon")
```

- Das Ergebnis ist sind zwei Vektoren mit den Longitude und Latitude Koordinaten.

```
##           [,1]      [,2]  
## [1,] 8.569720 49.49107  
## [2,] 8.569858 49.49101  
## [3,] 8.569999 49.49096  
## [4,] 8.570342 49.49085
```

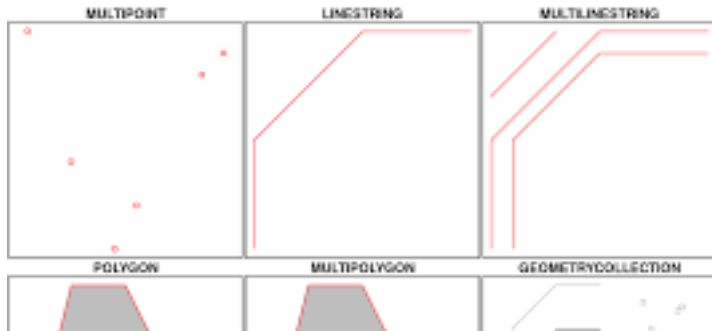

Das Paket für simple feature (sf)

Simple Features for R

- Das Paket `sf` ist ein Paket um geometrische Operationen durchzuführen.

```
library(sf)
```

- **Vignette für das Paket `sf`**



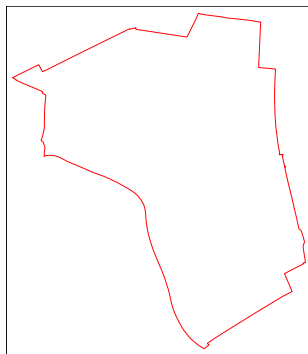
Die Funktion `st_linestring`

Create simple feature from a numeric vector, matrix or list

```
library(sf)
ls <- st_linestring(bb_poly)
sfc <- st_sfc(ls)
```

Den linestring plotten

```
library(tmap)  
qtm(sfc)
```



Einrichtungen (amenity)

OSM map features

- Alle benannten Objekte findet man, wenn man OSM map features in eine Suchmaschine eingibt.
- Achtung, wenn man bspw. alle Objekte mit dem Schlüssel amenity für eine Stadt heraussucht, bekommt man einen recht großen Datensatz

```
q <- add_osm_feature (q, key = 'amenity')  
osmdata_xml(q, '../data/Ladenburg_amenity.osm')
```

Was dahinter steckt

```
dat <- sf::st_read ('../data/Ladenburg_amenity.osm',  
                    layer = 'points',  
                    quiet = TRUE)
```

```
nrow(dat)  
?osmdata_sf
```

Die Funktion `osmdata_sf`

- Die Funktion `osmdata_sf` gibt ein `osmdata` Objekt im `sf` Format.

```
library(magrittr)
dat1 <- opq(bbox = 'Ladenburg') %>%
  add_osm_feature(key = 'shop', value = 'bakery') %>%
  osmdata_sf ()
```

```
unlist(lapply(dat1,nrow))
```

```
##          osm_points          osm_lines          osm_polygons          os
##                16                0                0
## osm_multipolygons
##                0
```

Alles in eine Karte plotten

****Der Start mit dem Paket tmap**

```
library(tmap)
tm_shape(sfc)
tm_bubbles(dat, size=2)
```

Beispiel Fahrradparkplätze

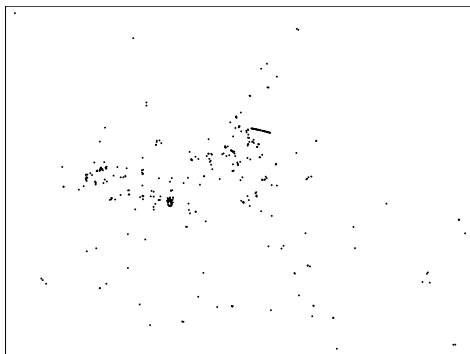
- OSM map features

```
q <- add_osm_feature (q, key = 'amenity', value = "bicycle_parking")
osmdata_xml(q, '../data/Amsterdam_amenity_bicycle_parking.osm')
```

```
dat <- sf::st_read ('../data/Amsterdam_amenity_bicycle_parking',
                    layer = 'points',
                    quiet = TRUE)
```


Die Daten plotten

```
library(tmap)  
qtm(dat)
```



Sehen was dahinter steckt

```
dat <- sf::st_read ('../data/Amsterdam_amenity.osm',  
                    layer = 'points',  
                    quiet = TRUE)
```

```
nrow(dat)  
names(dat)
```

Bar's in Mannheim

```
?add_osm_feature
```

```
q <- opq (bbox = 'Mannheim')  
q <- add_osm_feature (q, key = "amenity", value = 'bar')  
osmdata_xml (q, 'data/Mannheim_bar.osm')  
  
dat_bar <- sf::st_read ('../data/Mannheim_bar.osm',  
                        layer = 'points', quiet = TRUE)
```

Bus stations in Amsterdam

```
q <- opq (bbox = 'Amsterdam')
q <- add_osm_feature (q, key = "amenity",
                      value = 'bus_station')
osmdata_xml (q, 'data/Amsterdam_bus_station.osm')

dat_bus <- sf::st_read ('../data/Amsterdam_bus_station.osm',
                       layer = 'points', quiet = TRUE)
nrow(dat_bus)

?sf::st_read
```

An alternative

- Main vignette osmdata
- OpenStreetMap Data Structure

```
q <- opq (bbox = 'Amsterdam')
q <- add_osm_feature (q, key = "public_transport",
                      value = 'station')
osmdata_xml (q, '../data/Amsterdam_bus_pubtrans.osm')

dat_bus <- sf::st_read ('../data/Amsterdam_bus_pubtrans.osm',
                       layer = 'points', quiet = TRUE)
nrow(dat_bus)
```

Further information about public transport

Stop area

```
dat3 <- opq(bbox = 'Amsterdam') %>%  
  add_osm_feature(key = 'railway',  
                  value = 'tram_stop') %>%  
  osmdata_sf ()
```

```
dat3$osm_points$geometry
```

Das Ergebnis graphisch darstellen

```
# install.packages("osmplotr")
library("osmplotr")
bbox <- getbb("Amsterdam")
dat_pa <- extract_osm_objects(key='highway',
                              value="primary",
                              bbox=bbox)
dat_sa <- extract_osm_objects(key='highway',
                              value="secondary",
                              bbox=bbox)
dat_ta <- extract_osm_objects(key='highway',
                              value="tertiary",
                              bbox=bbox)
```

Die Graphik erzeugen

```
map <- osm_basemap(bbox = bbox, bg = c("#F5F5DC"))
map <- add_osm_objects(map, dat_pa, col = c("#00008B"))
map <- add_osm_objects(map, dat_sa, col = "green")
map <- add_osm_objects(map, dat_ta, col = "lightblue")
map <- add_osm_objects(map, dat3$osm_points, col = c("red"))
print_osm_map(map)
```


Get an overview of the available features

```
features <- available_features()  
head(features,n=20)
```

```
## [1] "4wd only" "abandoned"  
## [3] "abutters" "access"  
## [5] "addr" "addr:city"  
## [7] "addr:conscriptionnumber" "addr:country"  
## [9] "addr:district" "addr:flats"  
## [11] "addr:full" "addr:hamlet"  
## [13] "addr:housename" "addr:housenumber"  
## [15] "addr:inclusion" "addr:interpolation"  
## [17] "addr:place" "addr:postcode"  
## [19] "addr:province" "addr:state"
```

Changing the API

```
api_list <- c('http://overpass-api.de/api/interpreter',  
             'https://lz4.overpass-api.de/api/interpreter',  
             'https://z.overpass-api.de/api/interpreter',  
             'https://overpass.kumi.systems/api/interpreter')  
  
api_to_use <- sample(1:length(api_list), 1)  
  
set_overpass_url(api_list[api_to_use])
```

Die wichtigsten Funktionen im Paket osmdata

```
# https://rdrr.io/cran/osmdata/man/osmdata\_sp.html  
?osmdata_sp
```

- Github repo of the osmdata package
- Vignette for the package osmdata on github
- osmdata Homepage
- Overpass API - query form
- Overpass API/Language Guide
- Overpass Turbo
- `**osmplotr` tutorial
- **Geocomputation with R**
- **osmar - JOS**