Chapter 6

Norman Lim

2025-10-30

Hands-On: Working with Line Data

Step 1: Load necessary packages:

```
library(pacman)
p_load(sf, tidyverse, osmdata, ggspatial, patchwork)
```

Step 2: Fetch Line Data.

Getting bounding box for Shenzhen, China:

```
city_bbox <- osmdata::getbb("Shenzhen, China")
city_bbox
```

```
##         min       max
## x 113.67936 115.38906
## y  21.82136  23.01653
```

Building running OSM Query:

```
osm_motorways_query <- osmdata::opq(
  bbox = city_bbox,
  timeout = 60
) |>
  osmdata::add_osm_feature(key = "highway", value = "motorway") |>
  osmdata::osmdata_sf()

osm_motorways_query
```

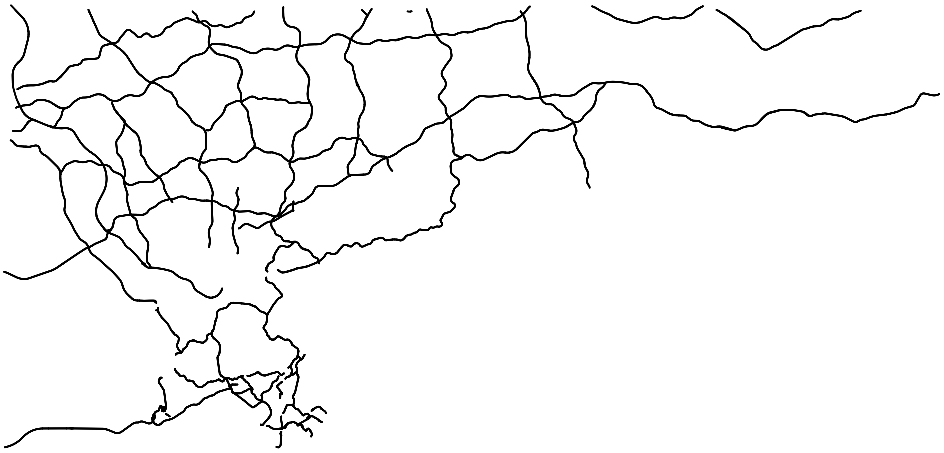
```
## Object of class 'osmdata' with:
##                  $bbox : 21.8213642,113.6793565,23.016534,115.3890648
##         $overpass_call : The call submitted to the overpass API
##                  $meta : metadata including timestamp and version numbers
##            $osm_points : 'sf' Simple Features Collection with 32627 points
##             $osm_lines : 'sf' Simple Features Collection with 6212 linestrings
##          $osm_polygons : 'sf' Simple Features Collection with 0 polygons
##        $osm_multilines : NULL
##     $osm_multipolygons : NULL
```

Extracing the lines object:

```r
city_motorways_sf <- osm_motorways_query$osm_lines
```

Plotting:

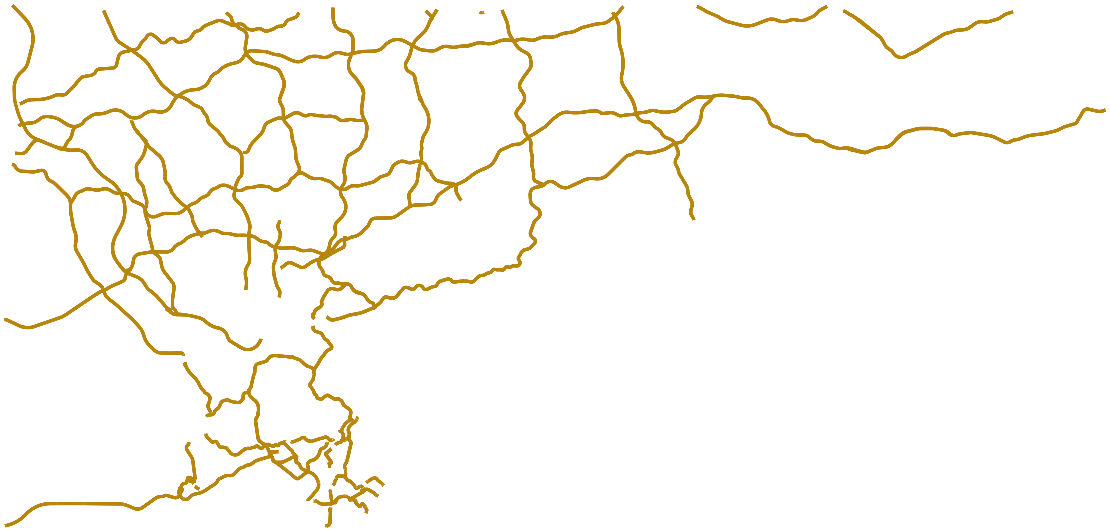
```r
plot(sf::st_geometry(city_motorways_sf))
```



Step 3: Styling Lines in ggplot2

```r
styled_motorway_map <- ggplot() +
  # add the motorway lines using geom_sf
  # ste style attribues OUTSIDE aesZ() for a fixed style for all motorways
  geom_sf(
    data = city_motorways_sf,
    color = "darkgoldenrod",
    linewidth = 0.6
  ) +
  labs(
    title = "Motorways in Shenzhen",
    caption = "Data: OpenStreetMap contributors"
  ) +
  theme_minimal() +
  theme(
    axis.text = element_blank(),
    axis.ticks = element_blank(),
    panel.grid = element_blank()
  )

styled_motorway_map
```

Motorways in Shenzhen



Data: OpenStreetMap contributors

Step 4: Simplify Lines using `sf::st_simplify`

Transforming motorways to local projected CRS (ESPG:4526):

```
# checking current CRS first
original_crs <- sf::st_crs(city_motorways_sf)
print(paste("Original CRS:", original_crs$input))
```

```
## [1] "Original CRS: EPSG:4326"
```

```
# Transforming to local CRS
city_motorways_sf_proj <- city_motorways_sf |> sf::st_transform(crs = 4526)
```

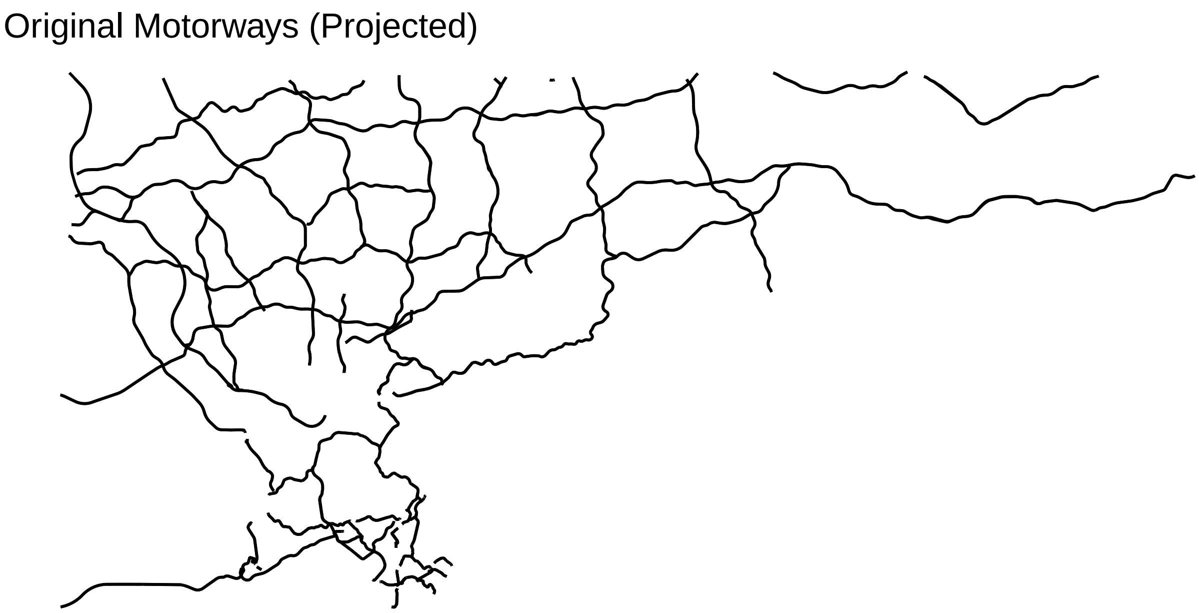
Proceed with line simplification:

```
tolerance_meters <- 5

city_motorways_sf_simplified <- sf::st_simplify(
  city_motorways_sf_proj, dTolerance = tolerance_meters
)
```

Plotting Original vs Simplified:

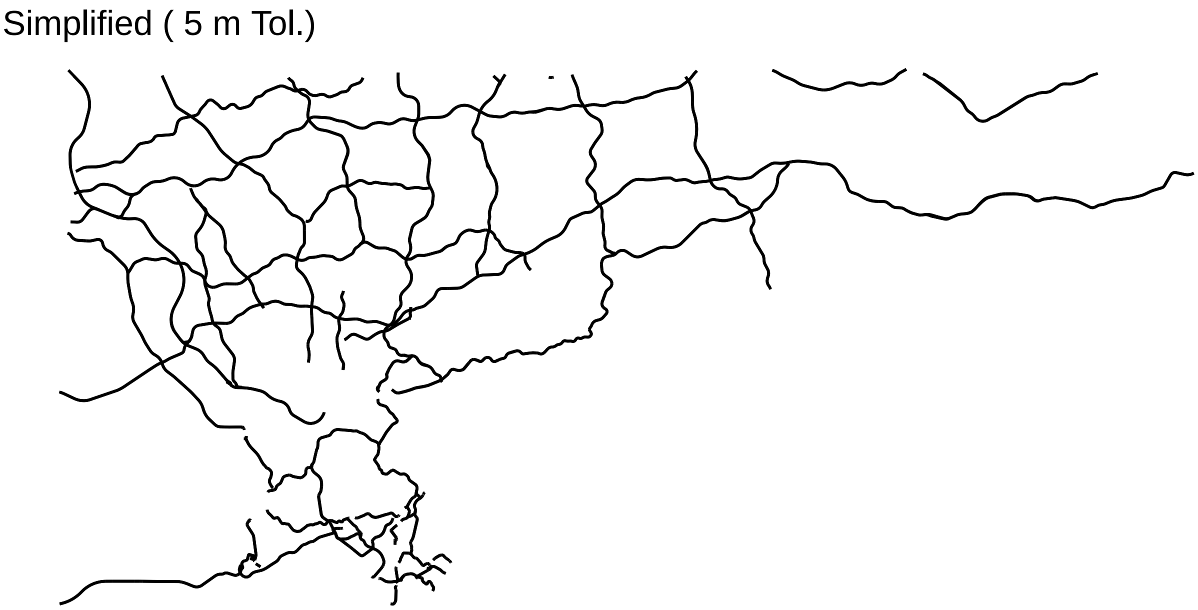
Creating plot object for the original projected motorway:

```r
# Original Projected CRS
plot_original_motorways <- ggplot() +
  geom_sf(
    data = city_motorways_sf_proj,
    linewidth = 0.5,
    color = "black"
  ) +
  labs(
    title = "Original Motorways (Projected)"
  ) +
  theme_void()

plot_original_motorways
```



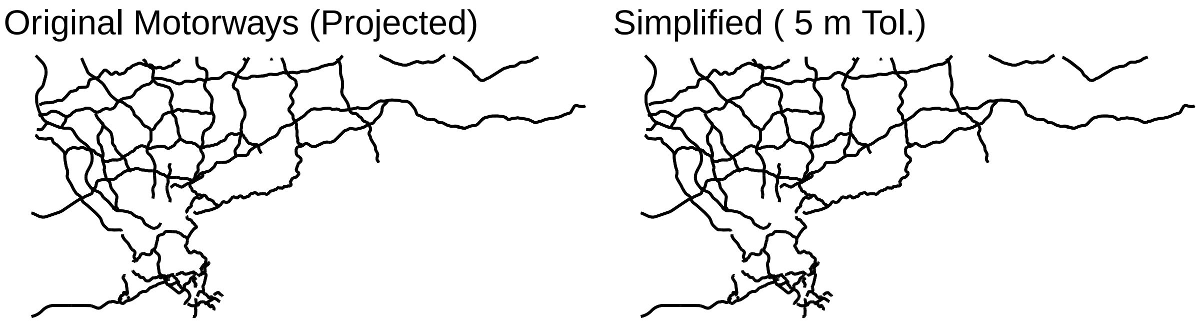
Creating plot object for the simplified motorway:

```r
plot_simplified_motorways <- ggplot() +
  geom_sf(
    data = city_motorways_sf_simplified,
    linewidth = 0.5,
    color = "black"
  ) +
  labs(
    title = paste("Simplified (", tolerance_meters, "m Tol.)", sep = " ")
  ) +
  theme_void()

plot_simplified_motorways
```



Combine plots side-by-side using `patchwork`:

```
comparison_plot_motorways <- plot_original_motorways + plot_simplified_motorways
comparison_plot_motorways
```



Original Motorways (Projected)

Simplified (5 m Tol.)

Creating Simple Flow Maps

Flow maps visualize movement or connection between origins and destinations.

Basic example using `geom_segment` in `ggplot2`:

Defining start and end points:

```
origin_x <- 4.86   # Approx. Longitude near Vondelpark West
origin_y <- 52.358 # Approx. Latitude
dest_x <- 4.88
dest_y <- 52.36
```

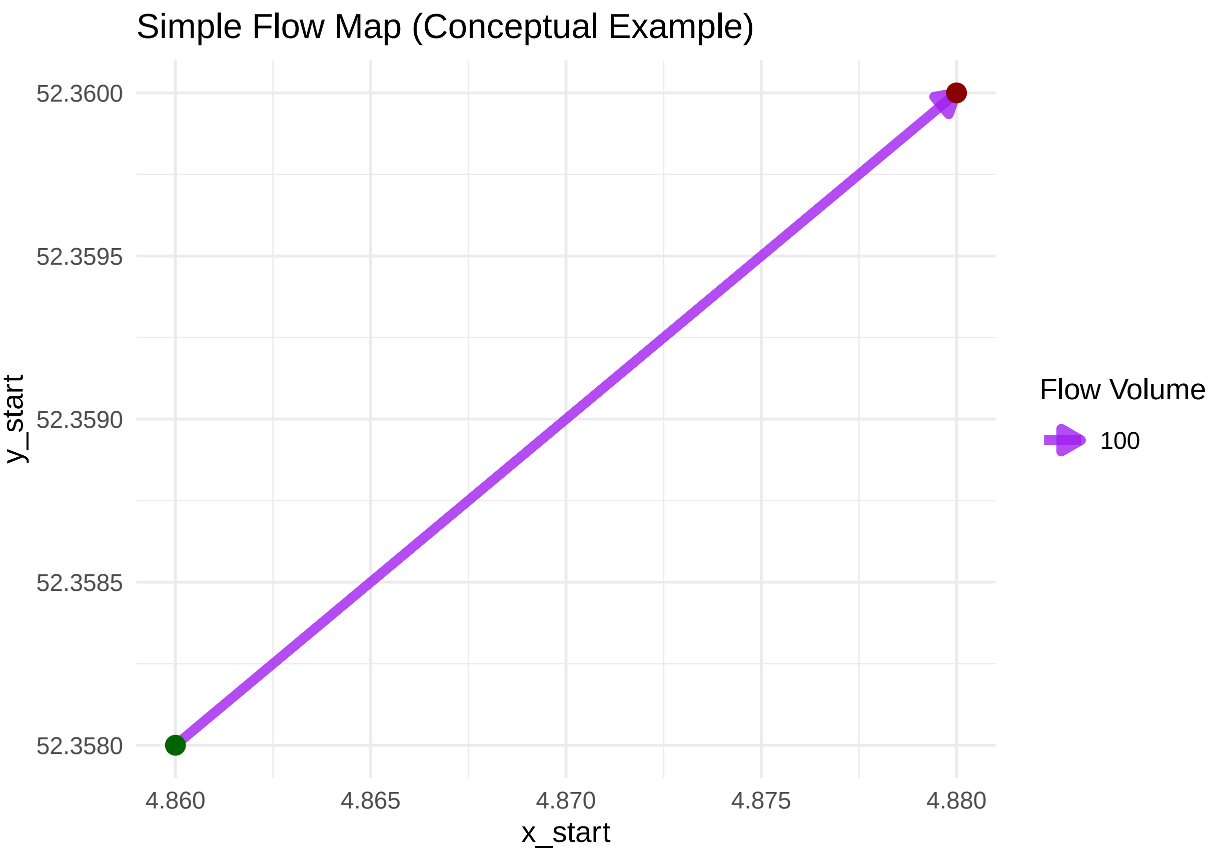
Creating a data frame for the flow segment:

```
flow_data <- data.frame(
  x_start = origin_x,
  y_start = origin_y,
  x_end = dest_x,
  y_end = dest_y,
  volume = 100   # A fictional flow volume
)

flow_data
```

```
##   x_start y_start x_end y_end volume
## 1    4.86  52.358  4.88 52.36    100
```

Plotting the flow map using `geom_segment`:

```
flow_data |> ggplot() +
  # Add the segment layer
  geom_segment(
    aes(
      x = x_start, y = y_start, xend = x_end, yend = y_end,
      linewidth = volume
    ),
    color = "purple",
    alpha = 0.8,
    arrow = arrow(length = unit(0.3, "cm"), type = "closed")
  ) +
  # Control how volume maps to thickness
  scale_linewidth(
    range = c(0.5, 3), # min/max thickness
    name = "Flow Volume"
  ) +
  # Add points for origin/destination
  geom_point(
    aes(x = x_start, y = y_start),
    color = "darkgreen", size = 3
  ) +
  geom_point(
    aes(x = x_end, y = y_end),
    color = "darkred", size = 3
  ) +
  # Add labels
  labs(
    title = "Simple Flow Map (Conceptual Example)"
  ) +
  theme_minimal()
```



Project Exercise: Create a Styled River or Transportation Network Map

Goal: Create a map of Kolkata showing major rivers styled nicely using `ggplot2`.

Step 1: Load necessary packages:

```
# Already done.
```

Step 2: Get Line Data:

Get bounding box for Kolkata, India.

```
kolkata_bbox <- osmdata::getbb("Kolkata, India")
kolkata_bbox
```

```
##        min      max
## x 88.23363 88.46108
## y 22.45203 22.61883
```

Build an `osmdata` query using `opq()` for the bounding box.

```
kolkata_rivers_query <- osmdata::opq(bbox = kolkata_bbox, timeout = 120) |>
  osmdata::add_osm_feature(key = "waterway", value = "river") |>
  osmdata::osmdata_sf()

kolkata_rivers_query
```

```
## Object of class 'osmdata' with:
##                  $bbox : 22.4520292,88.233628,22.6188255,88.4610776
##         $overpass_call : The call submitted to the overpass API
##                  $meta : metadata including timestamp and version numbers
##            $osm_points : 'sf' Simple Features Collection with 1492 points
##             $osm_lines : 'sf' Simple Features Collection with 41 linestrings
##          $osm_polygons : 'sf' Simple Features Collection with 0 polygons
##        $osm_multilines : 'sf' Simple Features Collection with 3 multilinestrings
##     $osm_multipolygons : NULL
```

Extract lines from `$osm_lines`.

```
kolkata_rivers_sf <- kolkata_rivers_query$osm_lines
dim(kolkata_rivers_sf)
```

```
## [1] 41 21
```

Step 3: Clean the data:

Retain only the necessary columns and filter the empty geometry column.

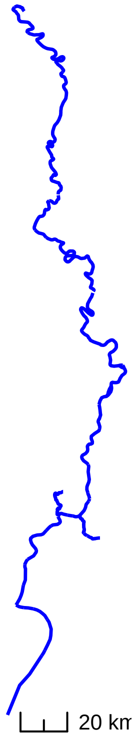
```
kolkata_rivers_sf <- kolkata_rivers_sf |>
  select(name, osm_id, waterway, geometry) |>
  filter(!st_is_empty(geometry))

dim(kolkata_rivers_sf)
```

```
## [1] 41  4
```

Step 4: Plot!

```
kolkata_rivers_sf |> ggplot() +
  geom_sf(
    color = "blue",
    linewidth = 0.6
  ) +
  labs(
    title = "Major Rivers in Kolkata, India",
    caption = "Data: OpenStreetMap"
  ) +
  theme_void() +
  ggspatial::annotation_scale(
    location = "bl",
    width_hint = 0.4,
    style = "ticks"
  ) +
  theme(
    plot.title = element_text(hjust = 0.5, face = "bold")
  )
```

Major Rivers in Kolkata, India



Data: OpenStreetMap