Creating plot of municipalities around Amstelveen

Han Oostdijk (www.hanoostdijk.nl)

March 11, 2016

Introduction

We will use R to create a plot of Amstelveen and its surrounding municipalities. Input is the file **Gemeentegrenzen.gml** that was downloaded from www.pdok.nl on 7 March 2016.

Used libraries

```
library(rgdal)
library(ggplot2)
library(rgeos)
library(maptools)
library(dplyr)
library(magrittr)
```

Read the data set into a data.frame

Read the data set and check the layers

We unpacked from the zip-file the file with boundaries for the municipalities. This file is in **gml** format and this can be read with **readOGR** function of the **rgdal** package.

```
dsn        = "D:/data/maps/Gemeentegrenzen.gml"
mylayers   = ogrListLayers(dsn)
mun       = readOGR(dsn, mylayers[1])
```

```
## OGR data source with driver: GML
## Source: "D:/data/maps/Gemeentegrenzen.gml", layer: "Gemeenten"
## with 393 features
## It has 3 fields
```

```
class_mun = class(mun)
print(class_mun)
```

```
## [1] "SpatialPolygonsDataFrame"
## attr(,"package")
## [1] "sp"
```

From the output we see that we have read layer **Gemeenten** (Dutch for **Municipalities**) and that the resulting object **mun** is of class **SpatialPolygonsDataFrame**.

Convert the object to coordinates with standard longitude and latitude

I just selected the CRS in the code and apparently this works: in the final plot (Figure 1 on page 5) the correct coordinates are printed.

```
mun <- spTransform(mun, CRS("+init=epsg:4238"))
```

Convert the object to a data.frame with coordinates

We use the **fortify** function to obtain a data.frame with the coordinates of the municipalities. Because we lose the describing information in this way, we have to merge this information with the coordinates. I have not found an automated way specify the merge key, so the merging is still an ad-hoc procedure. Here I use the **rownames** of the data.frame as the merge-key.

```
mun <- spTransform(mun, CRS("+init=epsg:4238"))

mun.f <- fortify(mun) # mun.f <- fortify(mun,region='id')
```

```
## Regions defined for each Polygons
```

```
head(mun.f,n=3)
```

```
##       long      lat order  hole piece id group
## 1 6.231139 52.42330     1 FALSE     1  0   0.1
## 2 6.227967 52.42363     2 FALSE     1  0   0.1
## 3 6.227646 52.42366     3 FALSE     1  0   0.1
```

```
head(mun@data,n=3)
```

```
##                                    gml_id Code      Gemeentenaam
## 0 ide3995447-fcdd-4a78-b94b-60298a0a71c1 0177            Raalte
## 1 id4378b5f0-e537-4556-a9dc-f18fd284db63 0798      Hilvarenbeek
## 2 idbcf03185-b586-44a4-851a-32c28a431dff 1903 Eijsden-Margraten
```

```
mun@data$id = rownames(mun@data)
mun.f <- merge(mun.f, mun@data, by.x = "id", by.y = "id")
head(mun.f,n=3)
```

```
##   id     long      lat order  hole piece group                                    gml_id
## 1  0 6.231139 52.42330     1 FALSE     1   0.1 ide3995447-fcdd-4a78-b94b-60298a0a71c1
## 2  0 6.227967 52.42363     2 FALSE     1   0.1 ide3995447-fcdd-4a78-b94b-60298a0a71c1
## 3  0 6.227646 52.42366     3 FALSE     1   0.1 ide3995447-fcdd-4a78-b94b-60298a0a71c1
##   Code Gemeentenaam
## 1 0177       Raalte
## 2 0177       Raalte
## 3 0177       Raalte
```

Create the data.frame that is needed for the plot

First we have to determine which municipalities surround Amstelveen. We do this by considering the smallest rectangle (parallel to longitude and latitude circles) that contains Amstelveen. Then we determine which municipaties have coordinates in or touching this rectangle. For these municipalities we select all the coordinates.

Smallest rectangle containing Amstelveen

```
mun.b = mun.f %>%
  filter(Gemeentenaam == c('Amstelveen'))  %>%
  summarize(
    minlat  = min(lat),
    maxlat  = max(lat),
    minlong = min(long),
    maxlong = max(long))
```

Municipalities with coordinates in or at the rectangle

```
mun.b = mun.f %>%
  group_by(Gemeentenaam) %>%
  filter(long>=mun.b$minlong & long<=mun.b$maxlong &
      lat >= mun.b$minlat & lat <= mun.b$maxlat)  %>%
  summarize(n= n()) %>%
  ungroup() %>%
  select(Gemeentenaam)
```

All coordinates of municipalities with coordinates in or at the rectangle

By doing an *inner-join* we keep the coordinates from **mun.f** for only the municipalities that were selected.

```
mun.a = mun.f %>%
  inner_join(mun.b,by=c('Gemeentenaam'='Gemeentenaam')) %>%
  ungroup() %>%
  mutate( Gemeentenaam = Gemeentenaam,
          fill         = as.character(Gemeentenaam),
          fill         = factor(ifelse(hole==T,NA,fill)))
  #   %>% arrange(Gemeentenaam,group,piece,order) # apparently not necessary
```

Create the data.frame with label information and fill information

We want to plot in the centre of the municipality a label with its name. That is why we again calculate the smallest rectangle that encloses the municipality and take as centre the midpoint of this rectangle. For the labels we will use the same fill attributes.

Calculate centre of municipalities

We calculate the centre and use 'Gemeentenaam' to color the municipalities.

```
mun.n = mun.a %>%
  group_by(Gemeentenaam) %>%
  summarize(minlat  = min(lat),
            maxlat  = max(lat),
            minlong = min(long),
            maxlong = max(long),
            fill    = last(Gemeentenaam)) %>%
  mutate(   cenlat  = (minlat+maxlat)/2,
            cenlong = (minlong+maxlong)/2) %>%
  select(  Gemeentenaam,lat=cenlat,long=cenlong,fill)
```

Plot the municipalities

The final plot can be found in Figure 1 on page 5. The formatting functions were found on StackOverflow. They make use of plotmath.

```
format_WE <- function(x) {
  xf = sprintf('%.1f',x) ;   d  = "*degree"
  ifelse(x < 0, parse(text=paste0(xf,d, "*~W")),
    ifelse(x > 0, parse(text=paste0(xf,d, "*~E")),xf))
}
format_NS <- function(x) {
  xf = sprintf('%.1f',x) ;  d  = "*degree"
  ifelse(x < 0, parse(text=paste0(xf,d, "*~S")),
    ifelse(x > 0, parse(text=paste0(xf,d, "*~N")),xf))
}
ggplot( mun.a,
  aes(long, lat, color=fill, fill=fill, label=fill)) +
  scale_fill_hue() +
  geom_polygon(aes(group = group),color='black') +
  geom_label(data=mun.n,color='black',show.legend=FALSE) +
  labs(x = "longitude", y = "latitude") +
  scale_x_continuous(labels=format_WE) +
  scale_y_continuous(labels=format_NS) +
  theme(legend.title=element_blank())
  # ggtitle("Municipalities around Amstelveen")
```

Session Info

```
sessionInfo()
```

```
## R version 3.2.4 (2016-03-10)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 10586)
##
## locale:
## [1] LC_COLLATE=English_United States.1252  LC_CTYPE=English_United States.1252
## [3] LC_MONETARY=English_United States.1252 LC_NUMERIC=C
## [5] LC_TIME=English_United States.1252
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] knitr_1.12.3    magrittr_1.5    dplyr_0.4.3     maptools_0.8-39 rgeos_0.3-17
## [6] ggplot2_2.1.0   rgdal_1.1-3     sp_1.2-2
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_0.12.3      munsell_0.4.3    colorspace_1.2-6 lattice_0.20-33  R6_2.1.2
##  [6] stringr_1.0.0    plyr_1.8.3       tools_3.2.4      parallel_3.2.4   grid_3.2.4
## [11] gtable_0.2.0     DBI_0.3.1        htmltools_0.3    lazyeval_0.1.10  assertthat_0.1
## [16] yaml_2.1.13      digest_0.6.9     formatR_1.3      evaluate_0.8.3   rmarkdown_0.9.5
## [21] labeling_0.3     stringi_1.0-1    scales_0.4.0     foreign_0.8-66
```

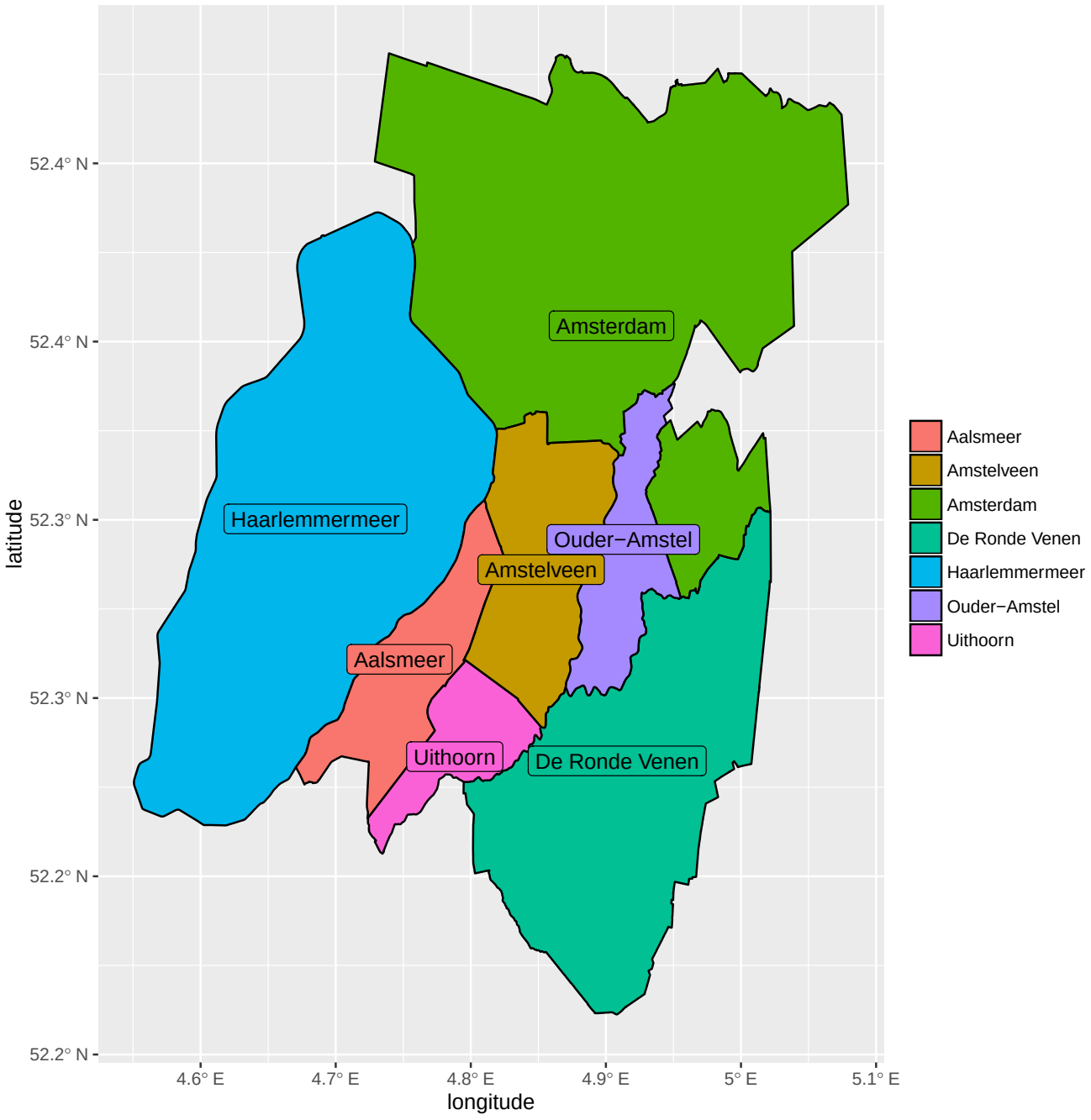

Figure 1: Municipalities around Amstelveen