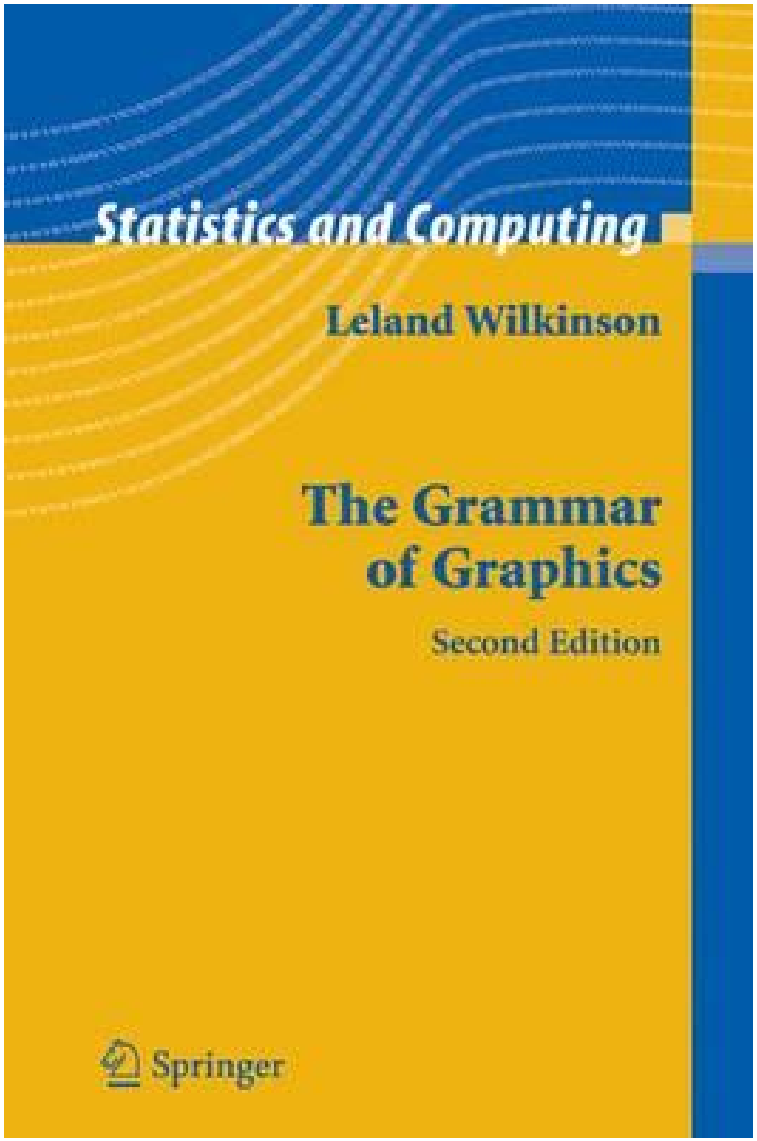
ggplot2 and maps

Marcin Kierczak

11/10/2016

The grammar of graphics

Hadley Wickham's **ggplot2** package implements the grammar of graphics described in Leland Wilkinson's book by the same title. It offers a very flexible and efficient way of generating plots based on data and is gaining more and more popularity.



The ggplot() function

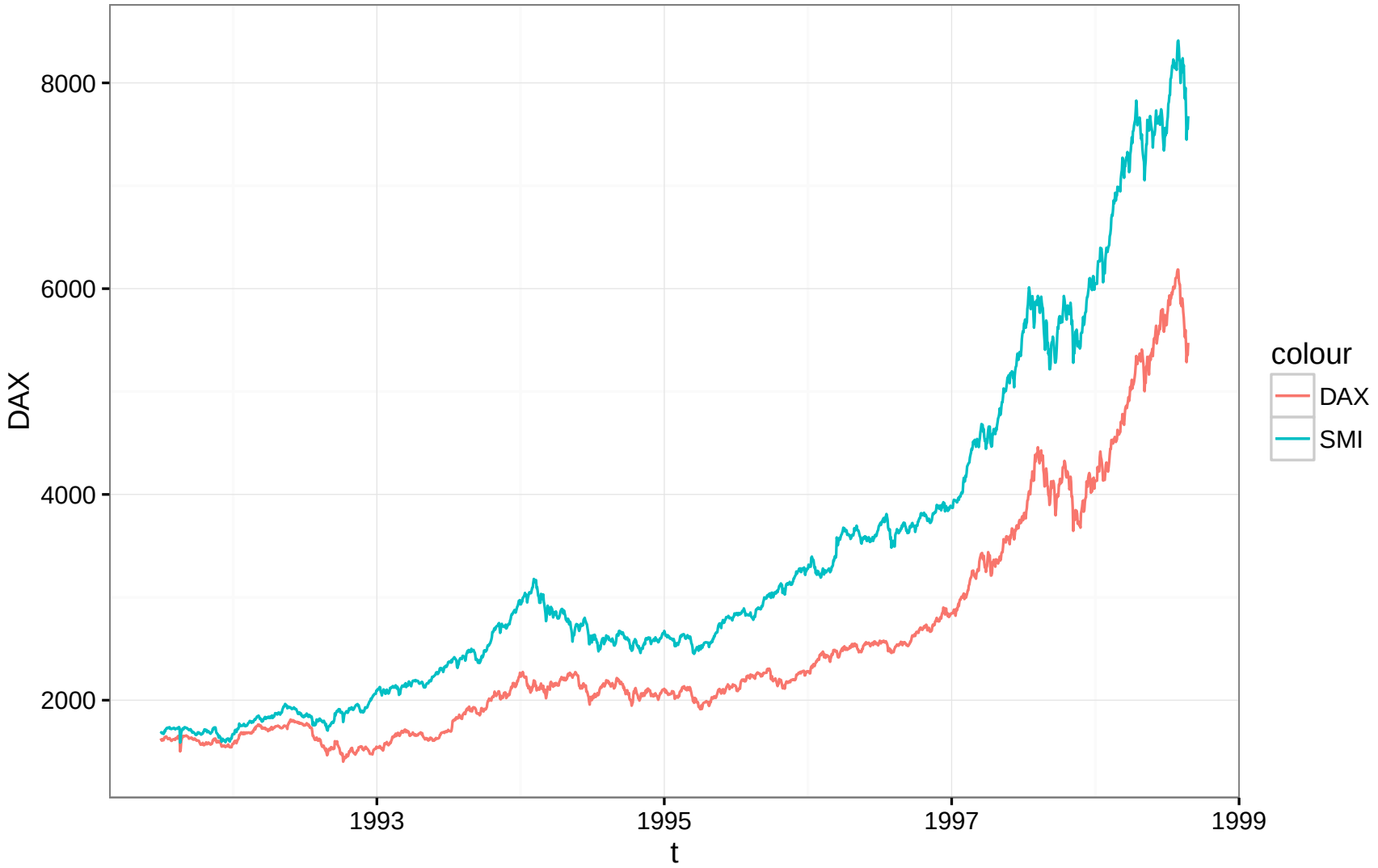
In the ggplot2 package, the default plotting function is called ggplot(). It is relatively easy to use. Let us see some examples:

```
library(ggplot2)
data(EuStockMarkets)
data.eu <- as.data.frame(EuStockMarkets)
t <- time(EuStockMarkets)
data.eu <- data.frame(t, data.eu)
stock.plot <- ggplot(data=data.eu, aes(x=t)) +
  geom_line(aes(y=DAX, col='DAX')) +
  geom_line(aes(y=SMI, col='SMI')) +
  theme_bw()
class(stock.plot)
```

```
## [1] "gg"     "ggplot"
```

ggplot() – example plot 1

```
## Don't know how to automatically pick scale for object of
```



The ggplot() function – example 1

```
summary(stock.plot)
```

```
## data: t, DAX, SMI, CAC, FTSE
##   [1860x5]
## mapping:  x = t
## faceting: facet_null()
## -----------------------------------
## mapping: y = DAX, colour = DAX
## geom_line: na.rm = FALSE
## stat_identity: na.rm = FALSE
## position_identity
##
## mapping: y = SMI, colour = SMI
## geom_line: na.rm = FALSE
## stat_identity: na.rm = FALSE
## position_identity
```

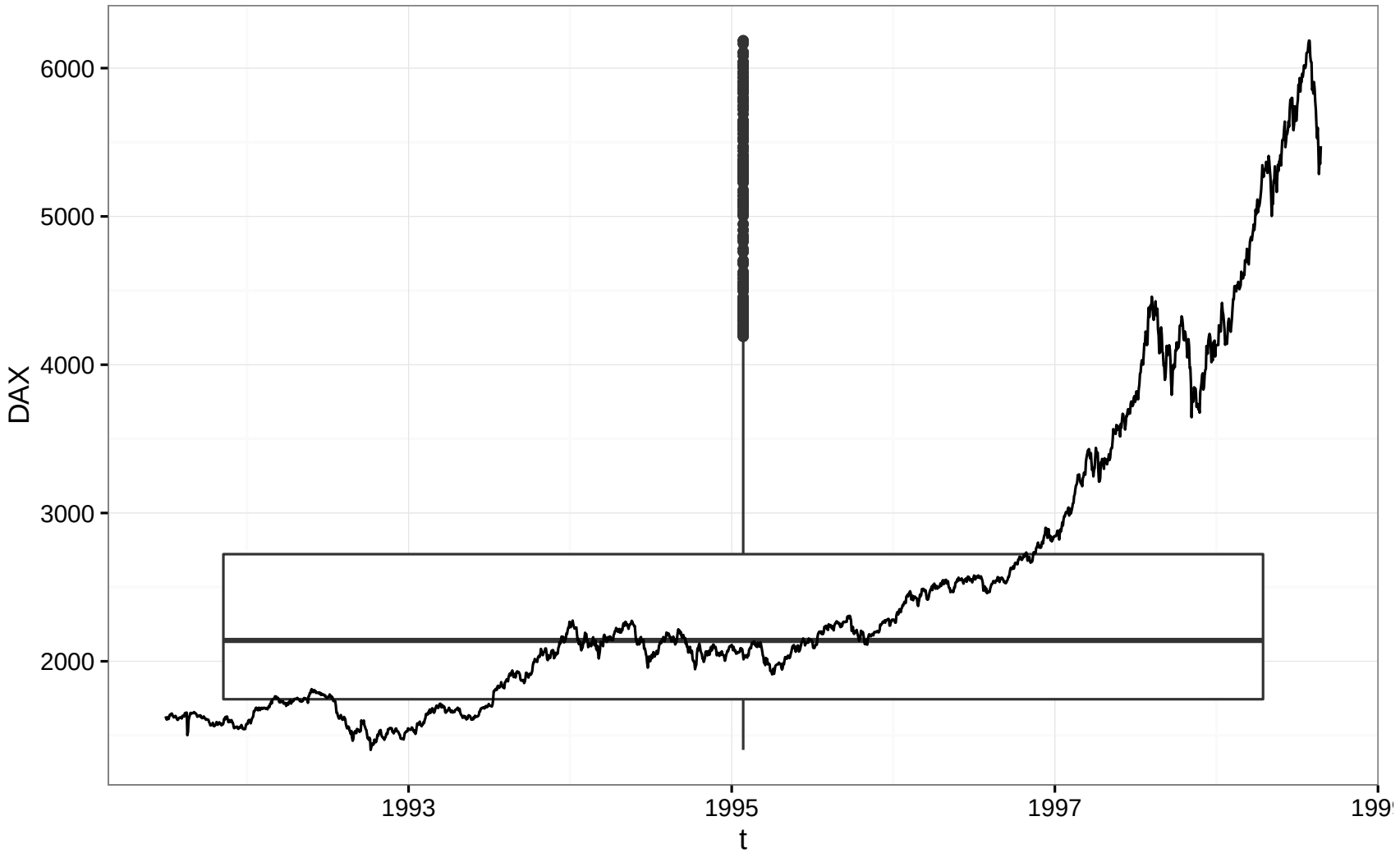
ggplot() – another example

```
stock.plot <- ggplot(data=data.eu, aes(x=t, y=DAX)) +
  geom_boxplot() +
  geom_line() +
  theme_bw()
summary(stock.plot)
```

```
## data: t, DAX, SMI, CAC, FTSE
##   [1860x5]
## mapping:  x = t, y = DAX
## faceting: facet_null()
## -----------------------------------
## geom_boxplot: outlier.colour = NULL, outlier.shape = 19
## stat_boxplot: na.rm = FALSE
## position_dodge
##
## geom_line: na.rm = FALSE
## stat_identity: na.rm = FALSE
```

ggplot() – another example plot

```
## Don't know how to automatically pick scale for object of
```



Visualising John Snow cholera data

On the 31 August 1854 a major outbreak of cholera occured in London's SOHO. A physician, John Snow, put all reported deaths on a map of London and identified the focal point of the epidemics. It turned out, that the area has been supplied in water by a particular pump. Snow ordered the pump being closed and stopped the epidemics. He also provided an indirect proof that cholera is a waterborne disease. Let us try to recreate his work using modern tools.

The data

First, we need to get Snow's original data in the digital form. Luckily, it can be obtained from, e.g. [http://blog.rtwilson.com/updated-snow-gis-data/].

We will need a couple of packages to work with maps:

```
# an extension of ggplot2 for spatial data
# vizualisations
library(ggmap)
 # various tools, e.g. to convert between datums
library(maptools)
```

```
## Loading required package: sp
```

```
## Checking rgeos availability: TRUE
```

The data cted.

```
library(sp)
# a Geospatial Data Abstraction Library,
# also useful for datum conversions etc.
library(rgdal)
```

```
## rgdal: version: 1.1-10, (SVN revision 622)
##  Geospatial Data Abstraction Library extensions to R suc
##  Loaded GDAL runtime: GDAL 1.11.4, released 2016/01/25
##  Path to GDAL shared files: /Library/Frameworks/R.framew
##  Loaded PROJ.4 runtime: Rel. 4.9.1, 04 March 2015, [PJ_V
##  Path to PROJ.4 shared files: /Library/Frameworks/R.fram
##  Linking to sp version: 1.2-3
```

```
# for Voronoi tesseleation
library(deldir)
```

```
## deldir 0.1-12
```

Reading the data

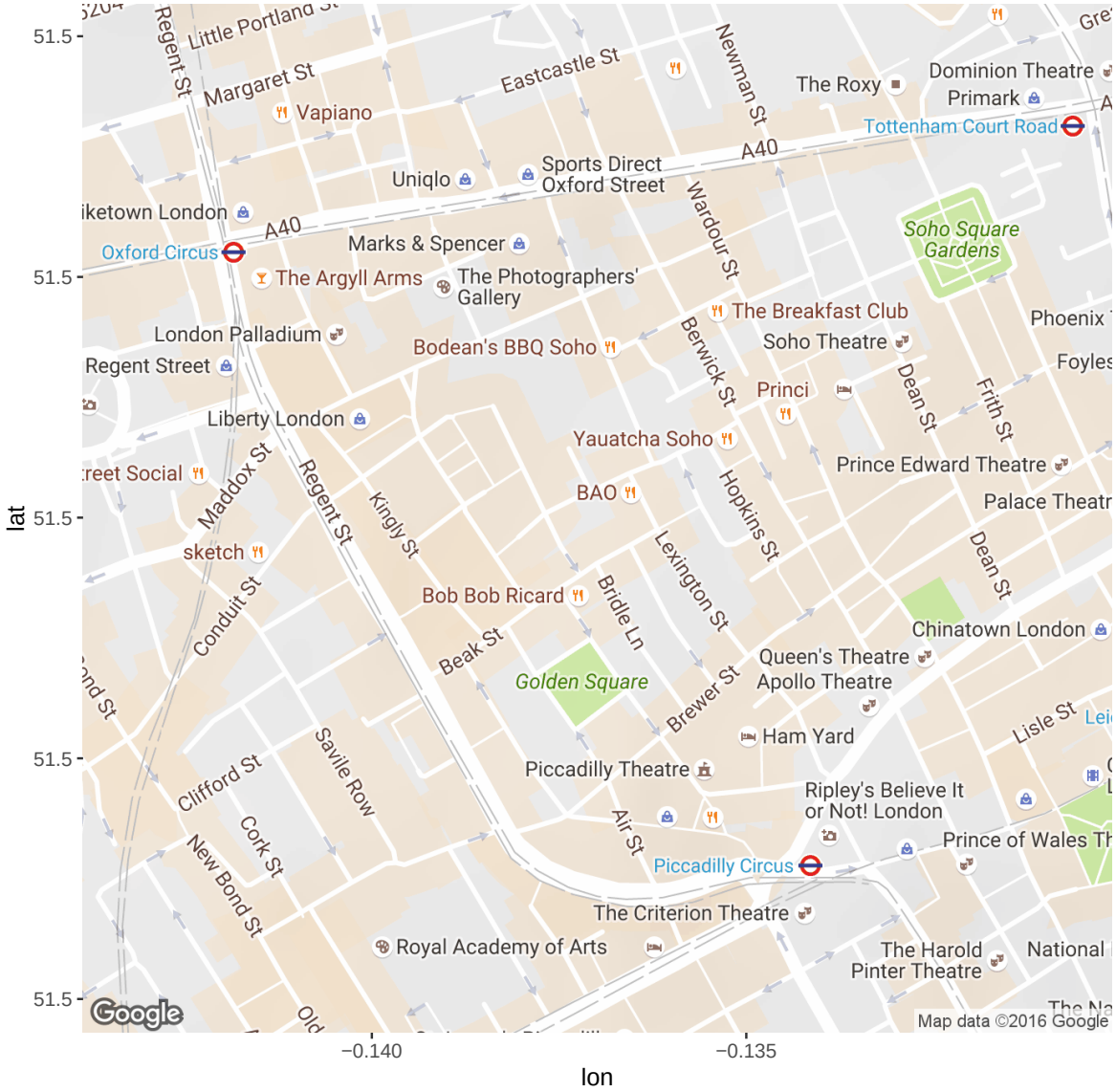
```
# download SOHO map from Google Maps
google.london <- get_map(c(-.137,51.513), zoom=16)
```

```
## Map from URL : http://maps.googleapis.com/maps/api/stati
```

```
# and make it to a ggmap object
london <- ggmap(google.london)
# now, read the downloaded Snow data
deaths <- readShapePoints("~/Dropbox/Rcourse/Labs/Lab - map
pumps <- readShapePoints("~/Dropbox/Rcourse/Labs/Lab - maps
```

Plot London

```
london
```



Make a data frame for ggplot2

```
tmp.deaths <- data.frame(deaths@coords)
tmp.pumps <- data.frame(pumps@coords)
tmp <- rbind(tmp.deaths, tmp.pumps)
# we need a column telling ggplot if it
# is a death case or a pump
tmp$type <- c(rep('death', times=dim(tmp.deaths)[1]),
              rep('pump', times=dim(tmp.pumps)[1]))
```

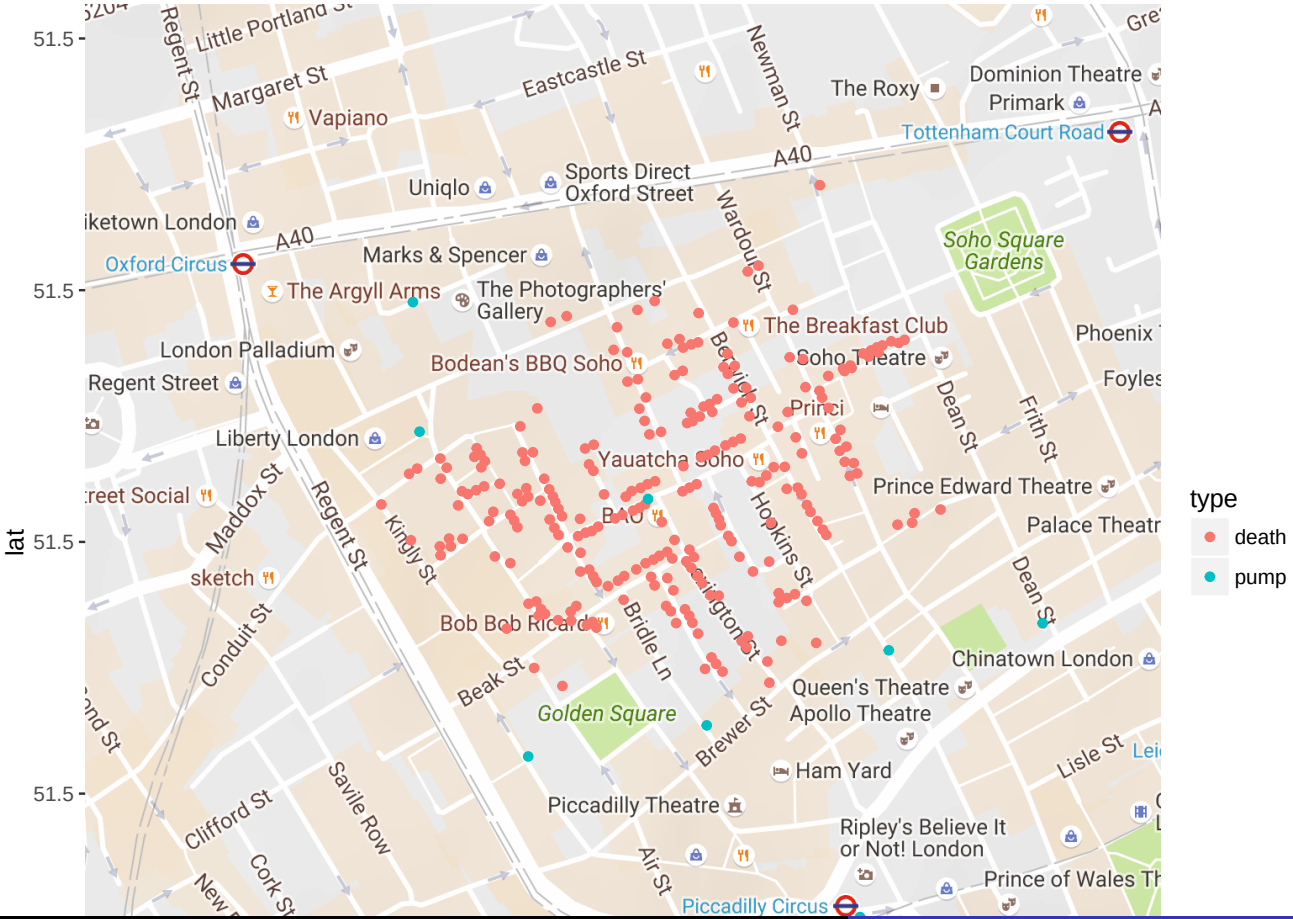
Transform the datum

```
# Transform coordinates to WGS84 datum used by Google
# Check EPSG codes online

# create object of coordinates class
coordinates(tmp)=~coords.x1+coords.x2
# set the projection in the object
proj4string(tmp)=CRS("+init=epsg:27700")
# transform the projection to WGS84
tmp = spTransform(tmp, CRS("+proj=longlat +datum=WGS84"))
# adjust in the data frame
tmp <- data.frame(tmp@coords, type=tmp@data$type)
```

Plot Snow's data

```
london +
  geom_point(mapping=
               aes(x=coords.x1, y=coords.x2, col=type),
               data=tmp)
```



Further analyses

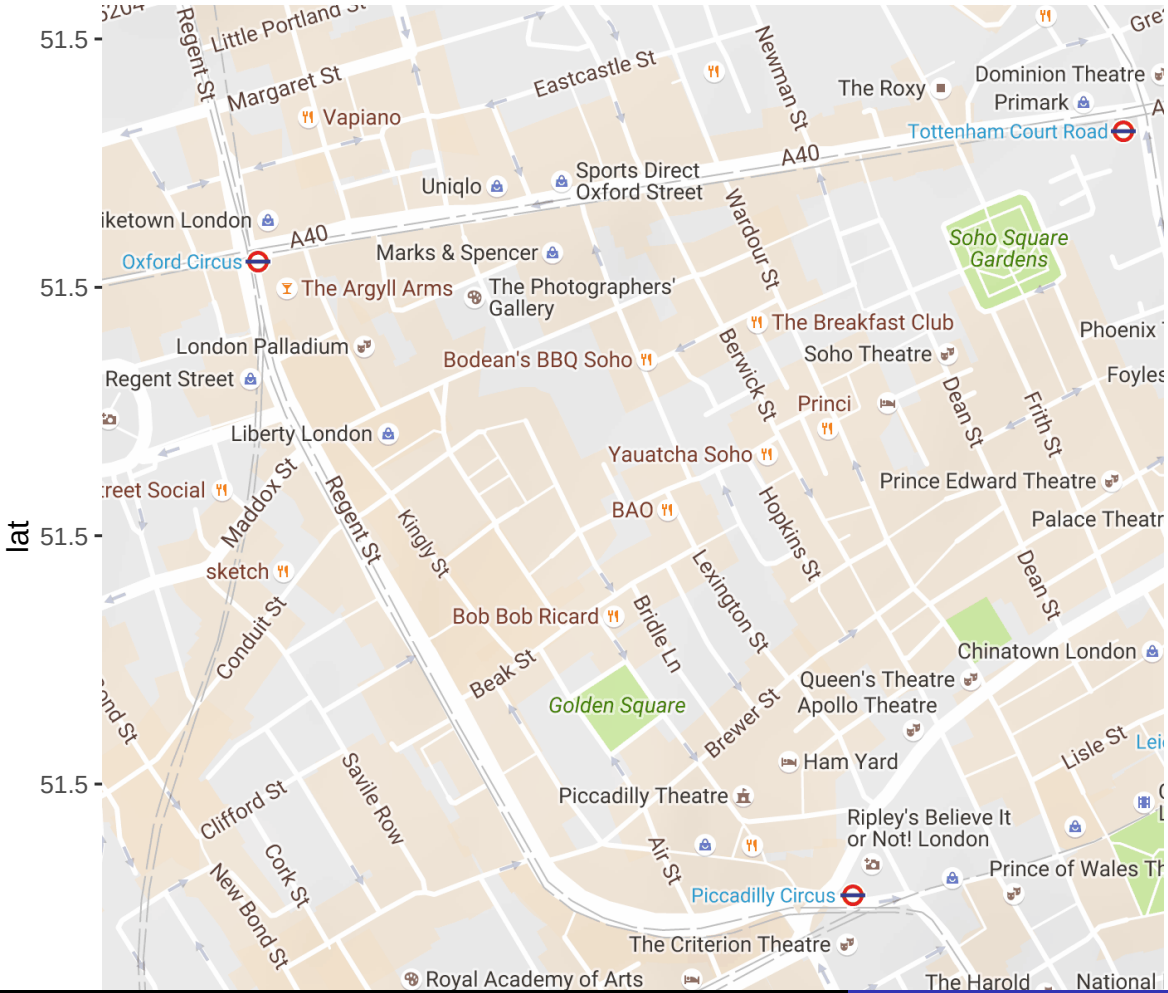
Well, so far, so good, but it still does not give the answer to our question on where the cholera source is...

```
# do Voronoi tesselation
voronoi <- deldir(tmp[tmp$type=='pump',])
```

```
##
##      PLEASE NOTE:  The components "delsgs" and "summary"
##  object returned by deldir() are now DATA FRAMES rather
##  matrices (as they were prior to release 0.0-18).
##  See help("deldir").
##
##      PLEASE NOTE: The process that deldir() uses for det
##  duplicated points has changed from that used in versior
##  0.0-9 of this package (and previously). See help("deldi
```

Step 1

```
# plot SOHO
snow.plot <- london
snow.plot
```

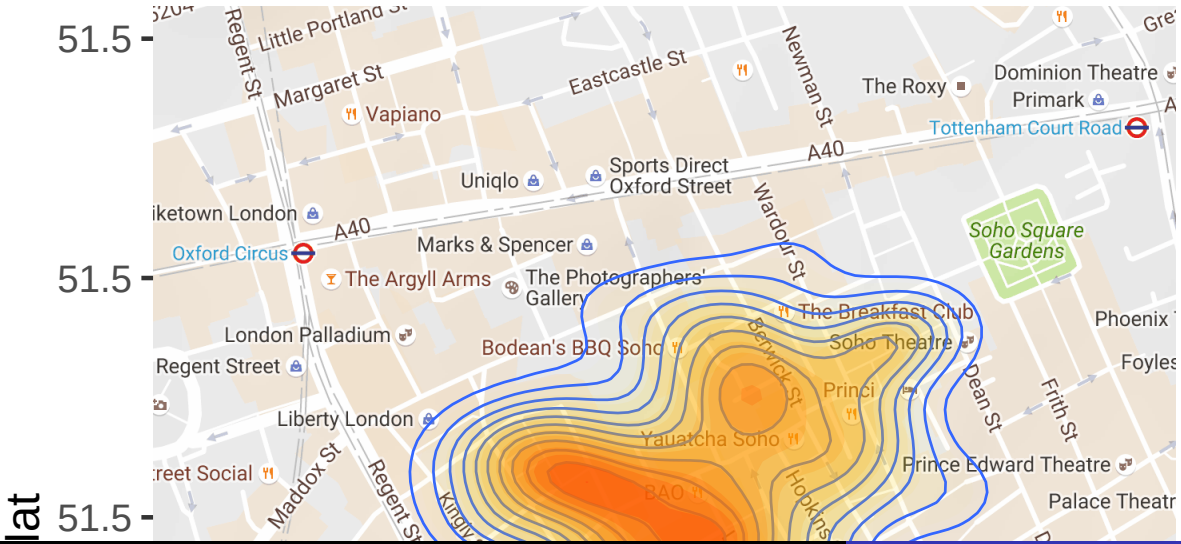


Step 2

```
# plot death density lines
snow.plot <- snow.plot + geom_density2d(data = tmp[tmp$type
    "death", ], aes(x = coords.x1, y = coords.x2),
    size = 0.3)
snow.plot
```

Step 3

```
# plot death gradient
snow.plot <- snow.plot + stat_density2d(data = tmp[tmp$type
    "death", ], aes(x = coords.x1, y = coords.x2,
    fill = ..level.., alpha = ..level..),
    size = 0.01, bins = 16, geom = "polygon") +
    scale_fill_gradient(low = "yellow",
        high = "red", guide = FALSE) +
    scale_alpha(range = c(0, 0.3), guide = FALSE)
snow.plot
```



Step 4

```
# plot pumps and deaths
snow.plot <- snow.plot + geom_point(mapping = aes(x = coord
    y = coords.x2, col = type), data = tmp)
snow.plot
```

Step 5

```
# plot Voronoi tesselation
snow.plot <- snow.plot + geom_segment(aes(x = x1,
    y = y1, xend = x2, yend = y2), size = 0.7,
    data = voronoi$dirsgs, color = "grey")
snow.plot
```