The German fuel prices data set and examples in R

Presentation at useR!2017 BRUSSELS

Boris Vaillant, Quantitative Consulting, Bonn

Now available on GitHub: https://github.com/borva/fuel

What is this talk about?

- a tour of some interesting (German) open data sets
- material for an open workshop on the end-to-end use of R in an analytical toolchain
- some examples of results on fuel prices — *not* a scientific assessment of competitiveness in the German retail fuel market
- maybe a good basis to build on, for whoever likes to join in the analysis

Who am I?

2001
PhD in Mathematics from Bonn University

2001 – 2005
Consultant and project manager at Simon-Kucher & Partners

2006 – 2008
Freelance Consultant, Quantitative Consulting

2009 – 2016
Head of Analytics, Technology and Data Science / Quality assurance at IRI Germany

2017

A few key facts about the German retail fuel market

Fuel and the economy

~ 14K fuel stations

~ 55m vehicles

of which ~ 45m private cars

~ 82m inhabitants

~ 44K€ GDP / inhabitant

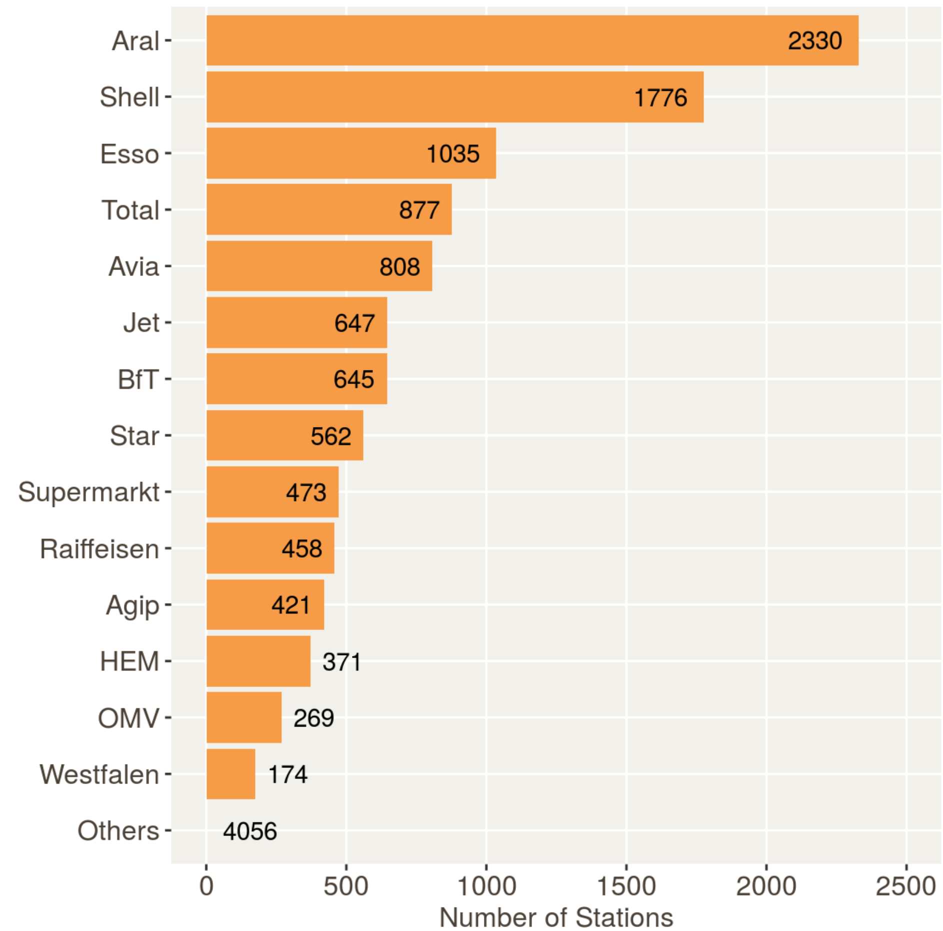
~ 22K€ disposable income / inhabitant

~ 43bn L diesel / yr

~ 25bn L petrol / yr

~ 1K€ spend on fuel / inhabitant

Fuel stations per brand



from Station Master Data MTS-K & www.tankerkoenig.de

Background of the MTS-K dataset

- Historically, strong daily and weekly price fluctuations for fuel have raised suspicions of consumers, media and the German Bundeskartellamt (fair trade office)

> "Confusion at the Pumps: Big Oil's Strategy for Jacking Up Gas Prices", April 05, 2012
> http://www.spiegel.de/international/business/how-big-oil-attempts-to-confuse-german-petrol-consumers-a-825750.html.

- 2011 study on gas prices from the Kartellamt (http://www.bundeskartellamt.de/SharedDocs/Publikation/DE/Sektoruntersuchungen/Sektoruntersuchung%20Kraftstoffe%20-%20Zusammenfassung.pdf).
- Conclusion: Need to increase price transparency for consumers, creation of "MTS-K" (market transparency unit for fuels)
- Since 2013: fuel stations have to report all price changes. Data is provided to "market information services". Most common use are online price-finder apps
- MTS-K also publishes yearly reports (https://www.bundeskartellamt.de/SharedDocs/Publikation/DE/Berichte/Dritter_Jahresbericht_MTS-K.pdf)
- The service "www.tankerkoenig.de" is making the dataset available as a Postgres dump (from June 2014 onwards) under CC4.0

The data consists of a table of reported price changes …

```r
summary(origprices)
```

```
##      stid                date                             e10
##  Length:844652      Min.   :2014-06-08 09:50:01   Min.   :  -1
##  Class :character   1st Qu.:2015-01-05 06:46:01   1st Qu.:1259
##  Mode  :character   Median :2015-07-14 12:46:01   Median :1349
##                     Mean   :2015-06-22 11:15:54   Mean   :1322
##                     3rd Qu.:2015-12-12 23:06:01   3rd Qu.:1459
##                     Max.   :2016-05-01 23:53:01   Max.   :9999
##        e5            diesel         changed
##  Min.   :  -1   Min.   :  -1   Min.   : 1
##  1st Qu.:1279   1st Qu.:1069   1st Qu.:20
##  Median :1369   Median :1169   Median :21
##  Mean   :1350   Mean   :1172   Mean   :18
##  3rd Qu.:1479   3rd Qu.:1269   3rd Qu.:21
##  Max.   :9999   Max.   :9999   Max.   :63
```

… and a master table describing the fuel stations

```r
str(origstations)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    14902 obs. of  8 variables:
##  $ stid   : chr  "00006210-0037-4444-8888-acdc00006210" "00016899-3247-4444-8888-acdc00000007" "00041414-208c-4444-8888-a
##  $ name   : chr  "Beducker - Qualität günstig tanken" "Röttenbach" "SELGROS" "NeebTank" ...
##  $ brand  : chr  "Beducker" "BFT Pickelmann" NA NA ...
##  $ street : chr  "Donauwörther Str." "Lohmühlweg" "Oststrasse" "Lochhamer Schlag" ...
##  $ place  : chr  "Meitingen" "Röttenbach" "Hilden" "Gräfelfing" ...
##  $ lat    : num  48.6 49.7 51.2 48.1 48.2 ...
##  $ lng    : num  10.85 10.92 6.95 11.45 12.52 ...
##  $ ot_json: chr  "{\"openingTimes\":[{\"applicable_days\":127,\"periods\":[{\"startp\":\"00:00\",\"endp\":\"00:00\"}]}]}'
```

The data set (and similar data) is already being investigated in some detail

Boehnke, Jörn. “Pricing Strategies, Competition, and Consumer Welfare Evidence from the German and Austrian Retail Gasoline Market.” *Unpublished Manuscript*, 2014. http://scholar.harvard.edu/files/boehnke/files/boehnke_pricing_strategies_competition_and_consumer_welfare.pdf.

Dewenter, Ralf, Ulrich Heimeshoff, and Hendrik Lüth. “The Impact of the Market Transparency Unit for Fuels on Gasoline Prices in Germany, May 2016.” *Forthcoming in: Applied Economics Letters*, n.d.

Dewenter, Ralf, and Ulrich Schwalbe. “Preisgarantien Im Kraftstoffmarkt.” *Perspektiven Der Wirtschaftspolitik* 17, no. 3 (2016): 276–288.

Eibelshäuser, Steffen, and Sascha Wilhelm. “Price Competition at Work: Intraday Edgeworth Cycles in the German Retail Gasoline Market,” 2016. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2879392.

Frondel, Manuel, Colin Vance, and Alex Kihm. “Time Lags in the Pass-through of Crude Oil Prices: Big Data Evidence from the German Gasoline Market.” *Applied Economics Letters* 23, no. 10 (2016): 713–717.

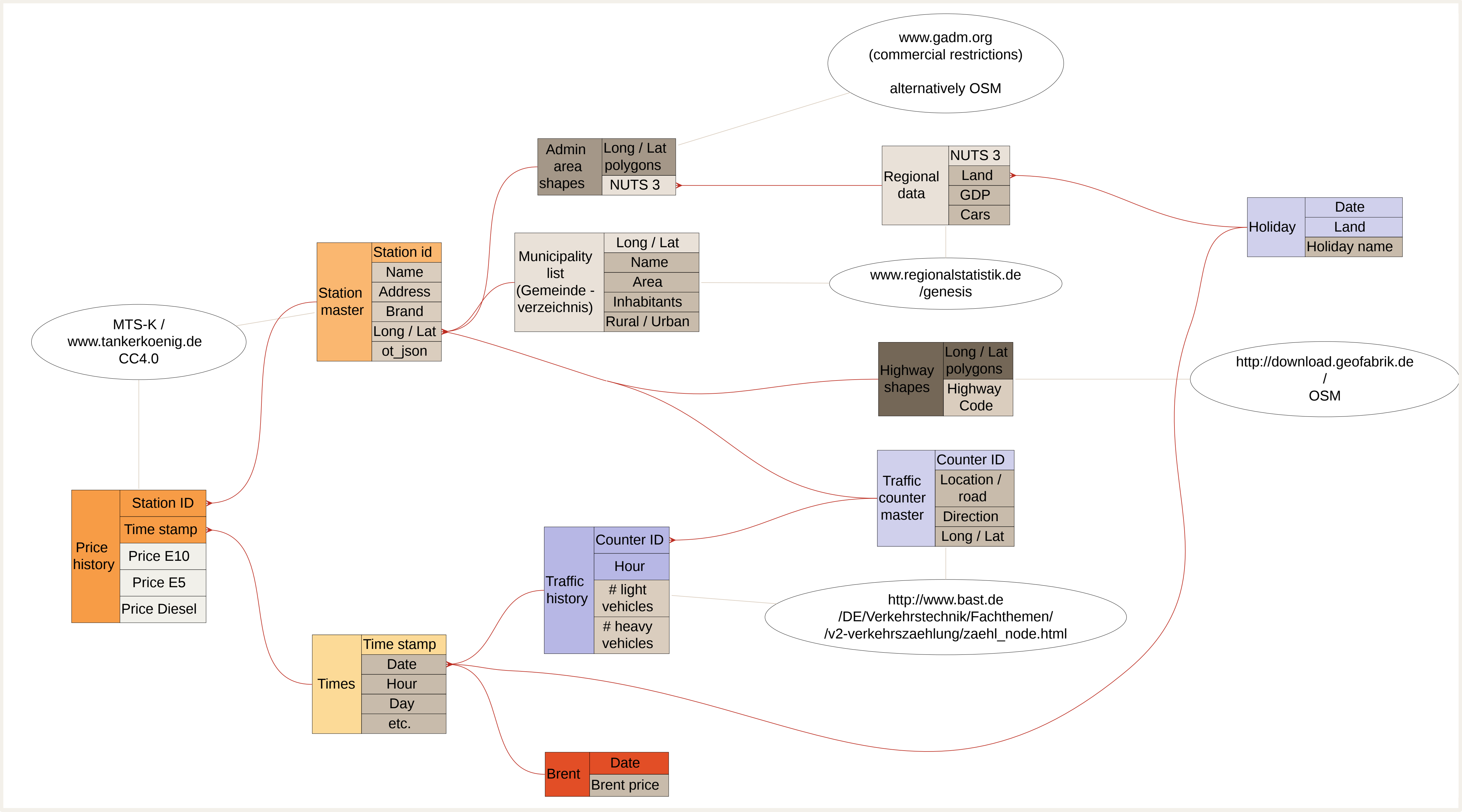
Haucap, Justus, Ulrich Heimeshoff, and Manuel Siekmann. “Price Dispersion and Station Heterogeneity on German Retail Gasoline Markets, January 2015.” *Forthcoming in: The Energy Journal*, n.d.

Kihm, Alexander, Nolan Ritter, and Colin Vance. “Is the German Retail Gas Market Competitive?,” 2014. https://papers.ssrn.com/sol3/papers.cfm?abstract_id=2566251.

Ritter, Nolan, Alex Kihm, and Colin Vance. “Do Major Brands Have Market Power in the German Retail Gas Market?” In *Energy & the Economy, 37th IAEE International Conference, June 15-18, 2014*. International Association for Energy Economics, 2014. http://www.iaee.org/proceedings/article/7876.

Siekmann, Manuel. “Characteristics, Causes, and Price Effects: Empirical Evidence of Intraday Edgeworth Cycles.” DICE Discussion Paper, 2017. http://www.dice.hhu.de/fileadmin/redaktion/Fakultaeten/Wirtschaftswissenschaftliche_Fakultaet/DICE/Discussion_Paper/252_Siekmann.pdf.

A closer look at the data structure and auxiliary data



The workshop is structured into 7 sections A – G

	Workshop section	Topics covered	Packages + Tools
A	Collection of regional data tables from destatis	Reading, cleaning and consolidating multiple socio-demographic data files	`tidyverse`, esp. `purrr`, `stringr`
B	Preparation and cleaning of the station master	Tidying, creation of brand and highway markers using regular expressions, parsing of json-information on opening hours	`tidyverse`, `stringr`, `jsonlite`
C	Geo-operations on the station master	Identification of NUTS 3-region per station, station distance matrix to competitors, highways, traffic counters etc.	`tidyverse`, `sp`, `rgdal`, `rgeos`, `geosphere`
D	Preparation of price data for models	Reading from Postgres, cleaning strange prices, imputing and aggregating the price data. Calculation of competitor prices	`tidyverse`, `RPostgres`, `lubridate`
E	Creation of different models	Moving to AWS, Test of different models (Linear, Panel, Spatial), collection of results	`lm`, `lfe`, `plm`, `spdep`, `splm`, http://www.louisaslett.com
F	Analysis	Analyses, preparation and visualisation of results	`ggplot2`, `ggmap`, `stringr`
G	Presentation	This document	`rmarkdown`, `pandoc`, `ggplot2`, `DiagrammeR`

A `purrr`-based workflow is used to read multiple files into one single database

The data on https://www.regionalstatistik.de/genesis/ is divided into many different smaller tables

Code	Inhalt
171-01-4	Gebietsstand: Gebietsfläche in qkm - Stichtag 31.12. - regionale Tiefe: Kreise und krfr. Städte
171-01-5	Gebietsstand: Gebietsfläche in qkm - Stichtag 31.12. - regionale Tiefe: Gemeinden, Samt-/Verbandsgemeinden
171-01-5-B	Gebietsstand: Gebietsfläche in qkm - Stichtag 31.12. - regionale Ebenen
171-31-4	Feststellung des Gebietsstandes: Zahl der Gemeinden - Stichtag 31.12. - regionale Tiefe: Kreise und krfr. Städte
171-31-4-B	Feststellung des Gebietsstandes: Zahl der Gemeinden - Stichtag 31.12. - regionale Ebenen
173-01-4	Bevölkerungsstand: Bevölkerung nach Geschlecht - Stichtag 31.12. - regionale Tiefe: Kreise und krfr. Städte
173-01-5	Bevölkerungsstand: Bevölkerung nach Geschlecht - Stichtag 31.12. - regionale Tiefe: Gemeinden, Samt-/Verbandsgemeinden

and is consolidated into a single table like this

```
## 'data.frame':    466 obs. of  6 variables:
##  $ KreisCode                   : chr  "01" "01000" "01001" "01002" ...
##  $ KreisName                   : chr  "Schleswig-Holstein" "Schleswig-Holstein" "Flensburg, Kreisfreie Stadt" "Kiel, Lar
##  $ BIP je Einwohner_EUR        : num  29331 29331 39092 44274 37492 ...
##  $ Flaeche                     : num  15802.5 15802.5 56.7 118.6 214.2 ...
##  $ KFZ_Pkw                     : num  1583822 1583822 41440 105759 94706 ...
##  $ Angebotene Gästebetten_Anzahl: num  173986 173986 1619 4189 9332 ...
```

A

The purrr-code for compiling these tables starts somewhat like this

```r
# start with csv files in directory
tibble(fileFullPath =
  grep(".csv", sort(dir(PATH, full.names = TRUE)), value = TRUE)) %>%

# make a shortened text load of each file
mutate(textLines =
         map(fileFullPath, readLines, encoding = "latin1") %>%
         map(substr,  start = 1, stop = 100) %>%

# identify data start and end points
mutate( indexStart = map(textLines, ~grep("^;", .)),
        posStart = map_int(indexStart, max),
        posEnd = map(textLines, ~grep("^_", .)) %>%
          map_int(min) - posStart - 1L) %>%

# make a first guess at column names
mutate(
  header =
    pmap(list(file = fileFullPath,   n_max = posStart-1),
         read_csv2, locale = locale(encoding = "latin1"),
         na = "NA", skip = 1, col_names = FALSE),
  LongNames = map(header, . %>% map(paste0, collapse = "_")) %>%
          map(unlist) %>% map(as.vector)) # etc etc
```

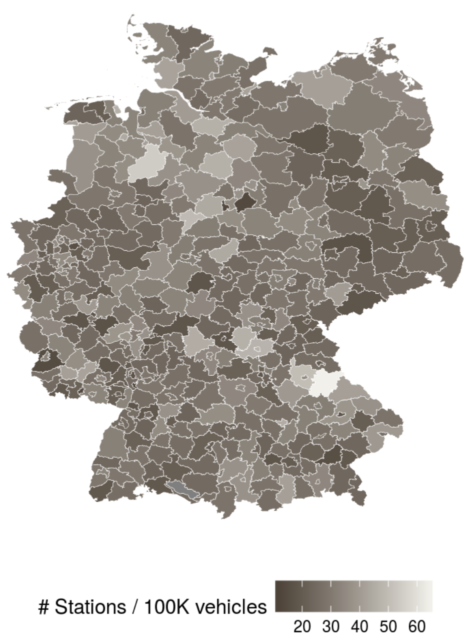
Examples of original data

```
## [[1]]
##  [1] "GENESIS-Tabelle: 400-51-4-B"
##  [2] "Baulandverkäufe - Jahressumme
##  [3] "regionale Ebenen;;;;;;"
##  [4] "Statistik der Kaufwerte für Bau
##  [5] ";;;;;Baulandverkäufe;Baulandve
##  [6] ";;;;;Insgesamt;baureifes Land"
##  [7] "2015;01001;Flensburg, Kreisfre
##  [8] "2015;01001;Flensburg, Kreisfre
##  [9] "2015;01001;Flensburg, Kreisfre
## [10] "2015;01001;Flensburg, Kreisfre
```

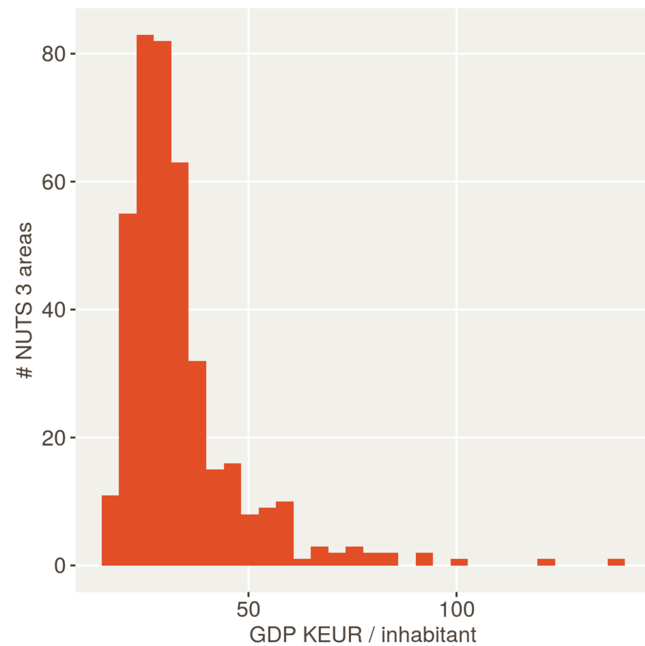
C

Linking the stations master to the Destatis data provides information on the economic background and competitve situation for each station

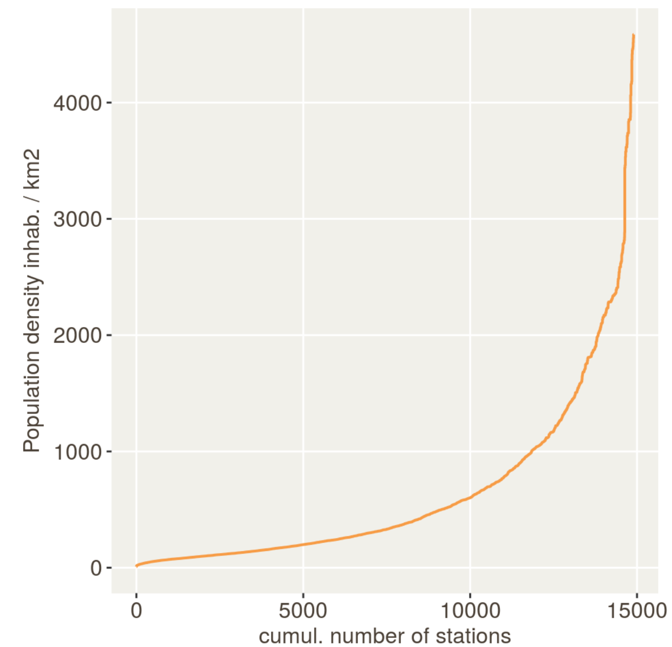
Station density in the NUTS 3 areas



Count of NUTS 3 by average GDP



Stations in rural vs. urban areas



C

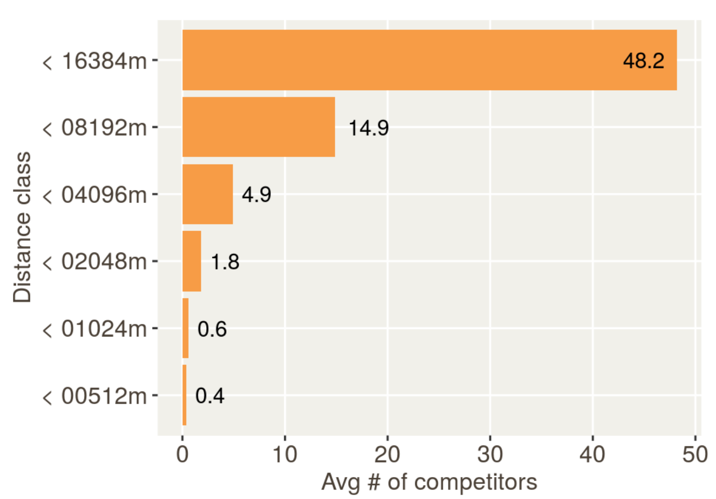
The match of stations to (NUTS 3-) administrative areas works best by matching long/lat to polygons. ZIP/PLZ-code-based methods proved to be much less reliable

```r
library(sp)

kreisShapes <- readRDS(gzcon(url(
    "http://biogeo.ucdavis.edu/data/gadm2.8/rds/DEU_adm2.rds")))

pointsDF <- stationsAll %>% select(long = lng, lat = lat)
pointsSP <- SpatialPoints(pointsDF, proj4string = CRS(proj4string(kreisShapes)))
kreisMatch <- over(pointsSP, kreisShapes)

# only 5 problems. Examples:
# tankpool24 Straße von Malakka 26388 Wilhelmshaven N 53.58208 E 8.13587 ## on seashore
# Clemens Tenhagen BFT Weselerstr.17 47665 Sonsbeck N 51.36410 E 6.22430 ## already in NL?
```

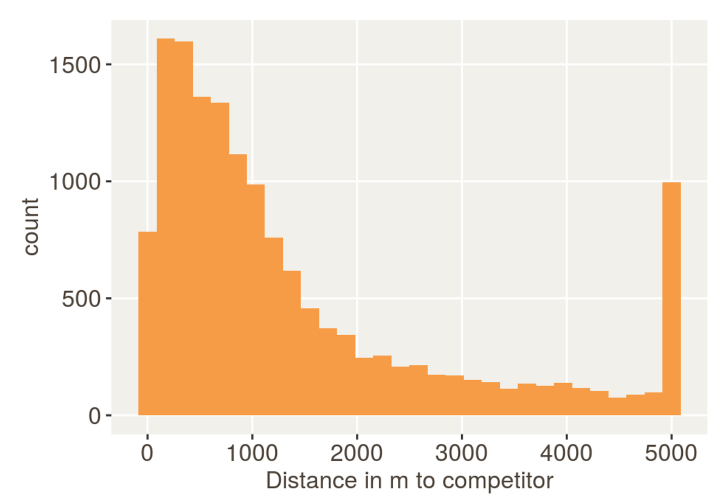
C To understand the local competitive situation between fuel stations we create the distance matrix

```r
library(geosphere)
distvec <- as.integer(distm(pointsSP, fun = distHaversine)) #

NStations <- length(stationsAll$StationID)

distDF <- tibble(distComp = distvec,    # competitor dataframe
                 StationID = rep(stationsAll$StationID, NStations),
                 CompID = rep(stationsAll$StationID, rep(NStations, NStations) ))  %>%
  filter(distComp <= 16383, StationID != CompID)

gc() # Voodoo?
```

Competitors per distance class



Distribution of distances



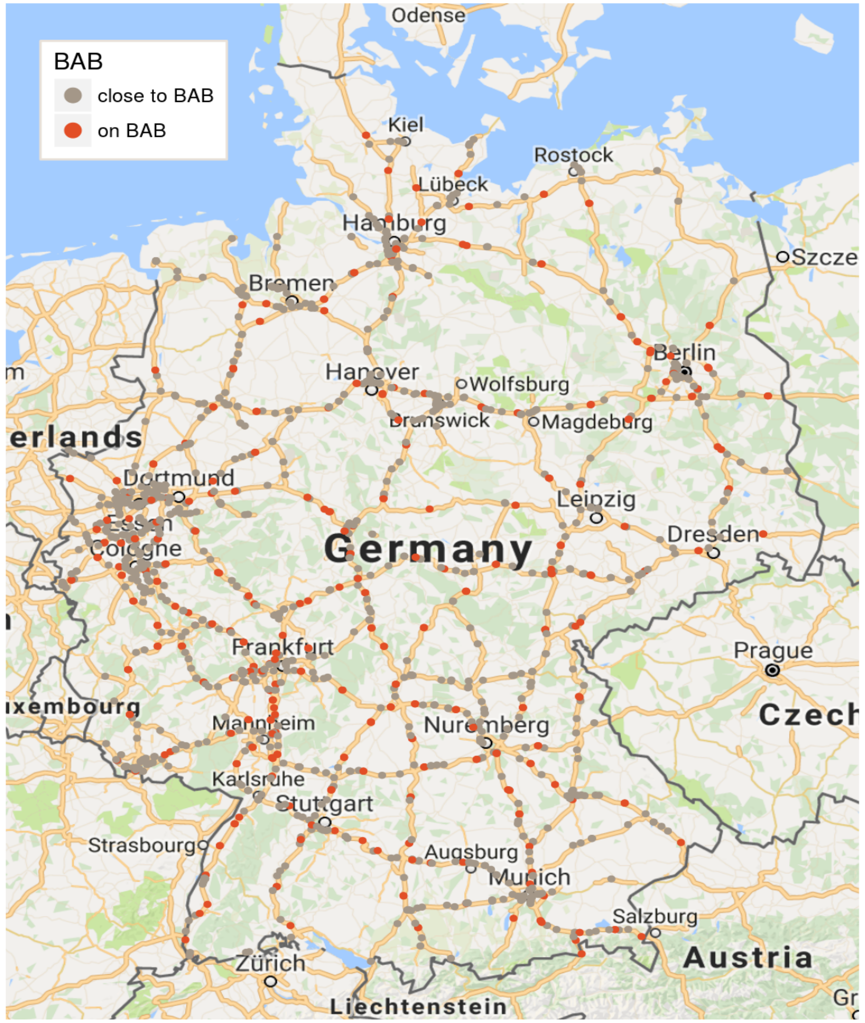
C

Similarly, identification of fuel stations on – and close to – highways is key to understanding the competitive situation

Identification of stations close to highways / highway links works using rgeos and OSM data

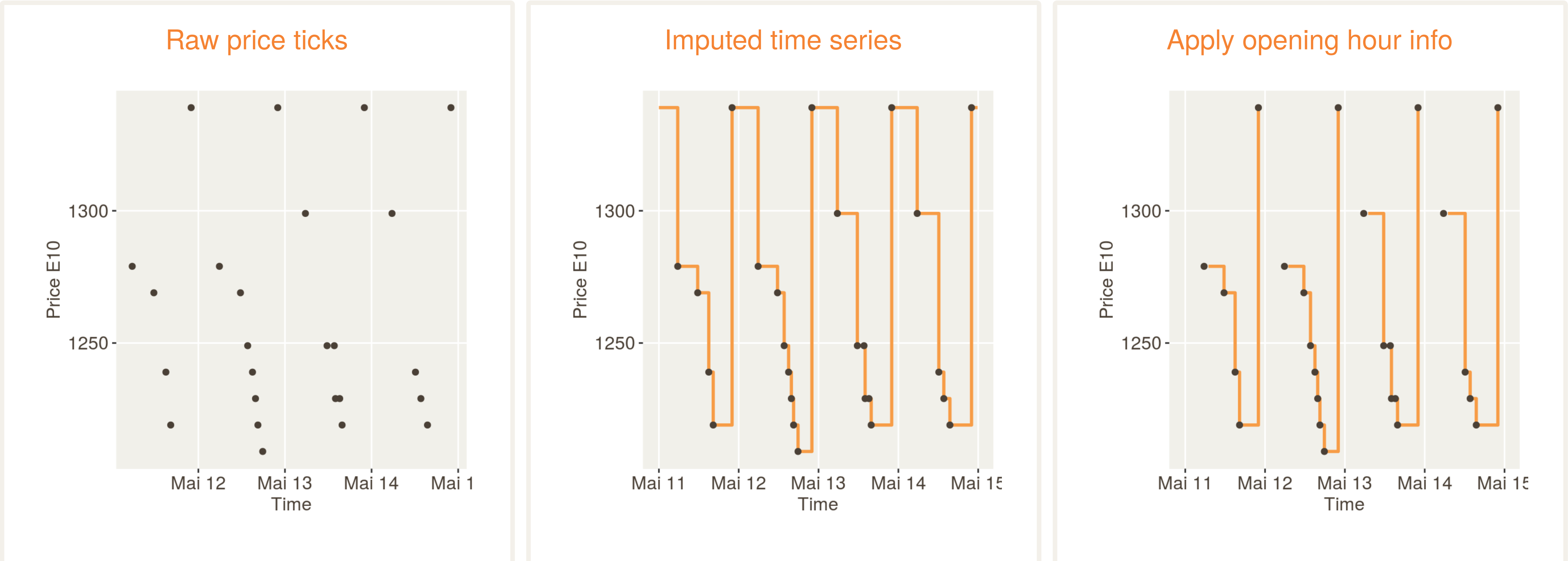
```r
library(rgeos)

hwLinkMatch <-
  gWithinDistance(
    pointsSP, ## Station long / lat
    highwayLinkShapes, ## OSM data
    dist = 0.005, ## "cheat" ~500m around HWs
    byid=TRUE)
```

The exact identification of highway stations, requires additional text search and manual verification



D

An important step in cleaning the price data is to convert price ticks into a (more) regular time series



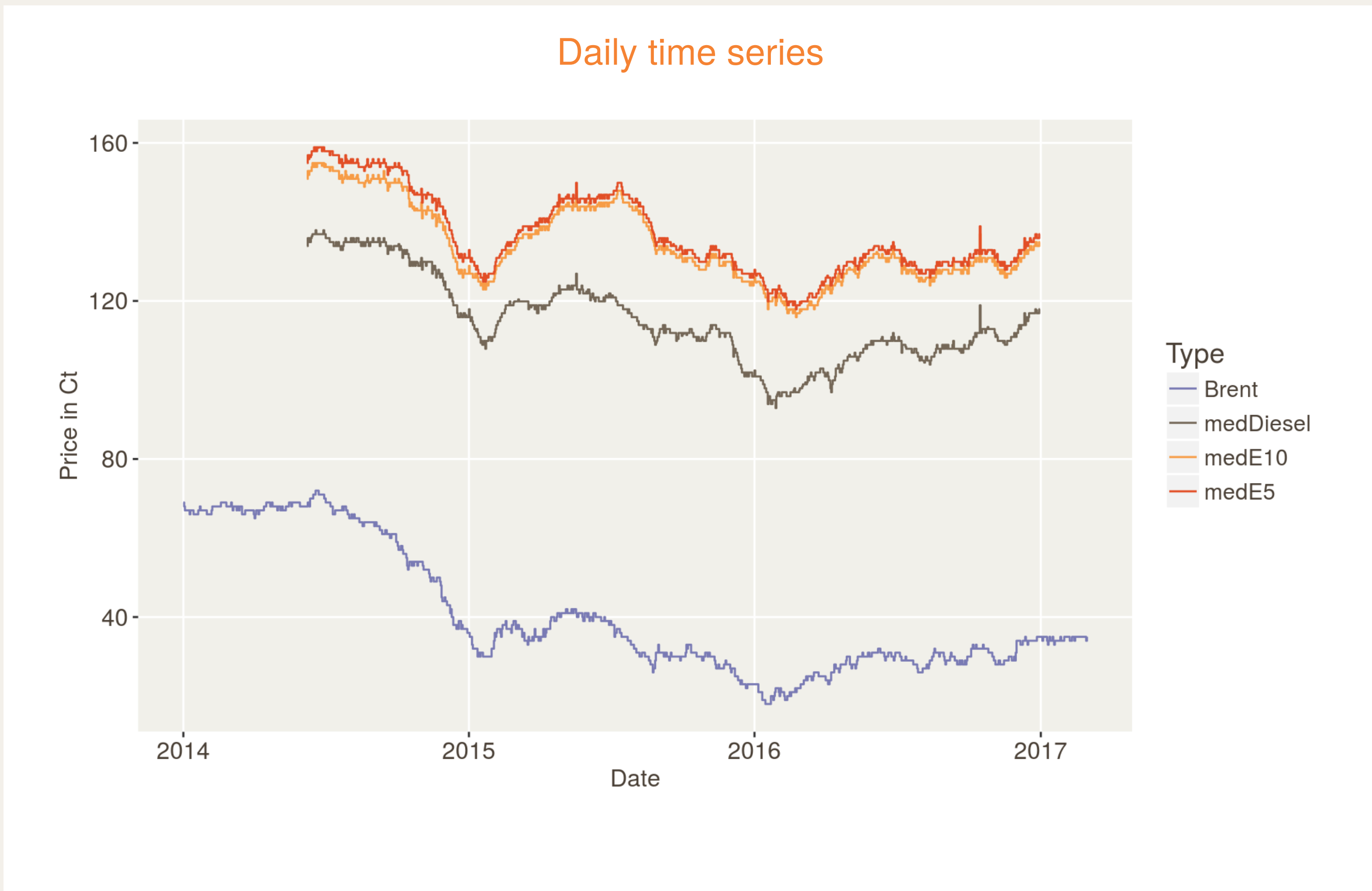
D The imputation process ("locf") works very well using just the `rep`-function

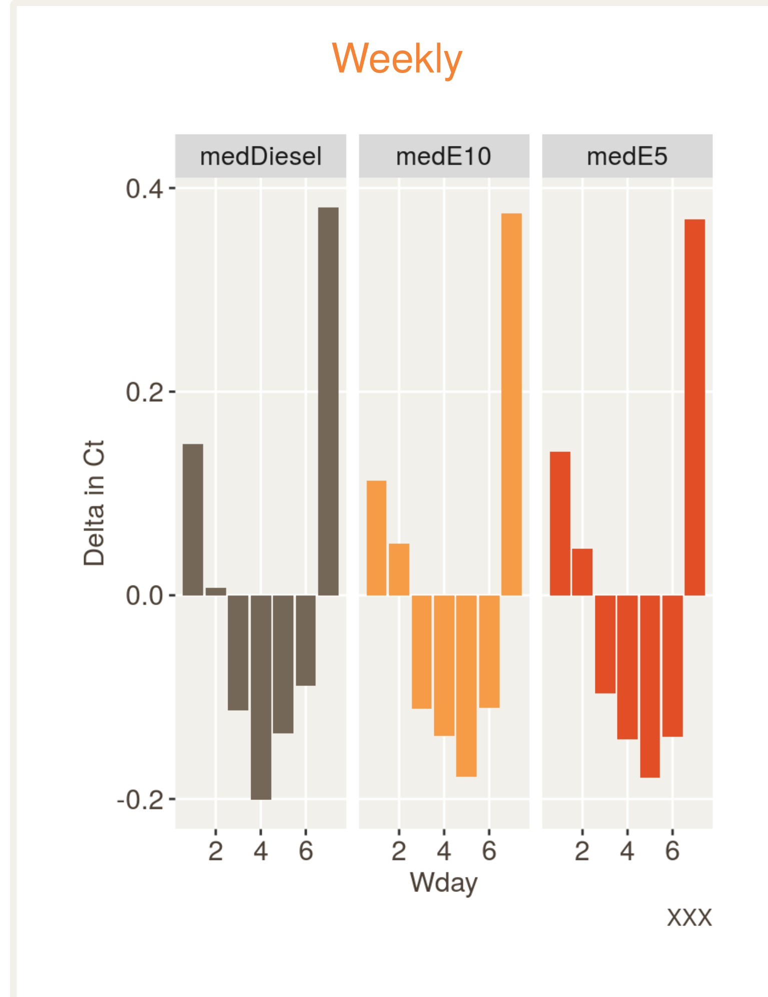
- (not shown) nudge all price ticks onto a regular 10 min grid
- determine length of gap to next price tick
- repeat current price tick accordingly

```
pricesRed <- prices %>%
  arrange(StationID, TimeID) %>%
  group_by(StationID) %>%
  mutate(
    TimeAfter = lead(TimeID, 1L),    #important: consecutive TimeIDs
    TimeAfter = ifelse(is.na(TimeAfter), TimeID + 1L, TimeAfter),
    ForwardGap = TimeAfter - TimeID)

pricesGrid <-
  tibble(
    lastE10 = rep(pricesRed$e10, pricesRed$ForwardGap),
    lastTimeID = rep(pricesRed$TimeID, pricesRed$ForwardGap))

## etc
```

D

Daily development and weekly price pattern

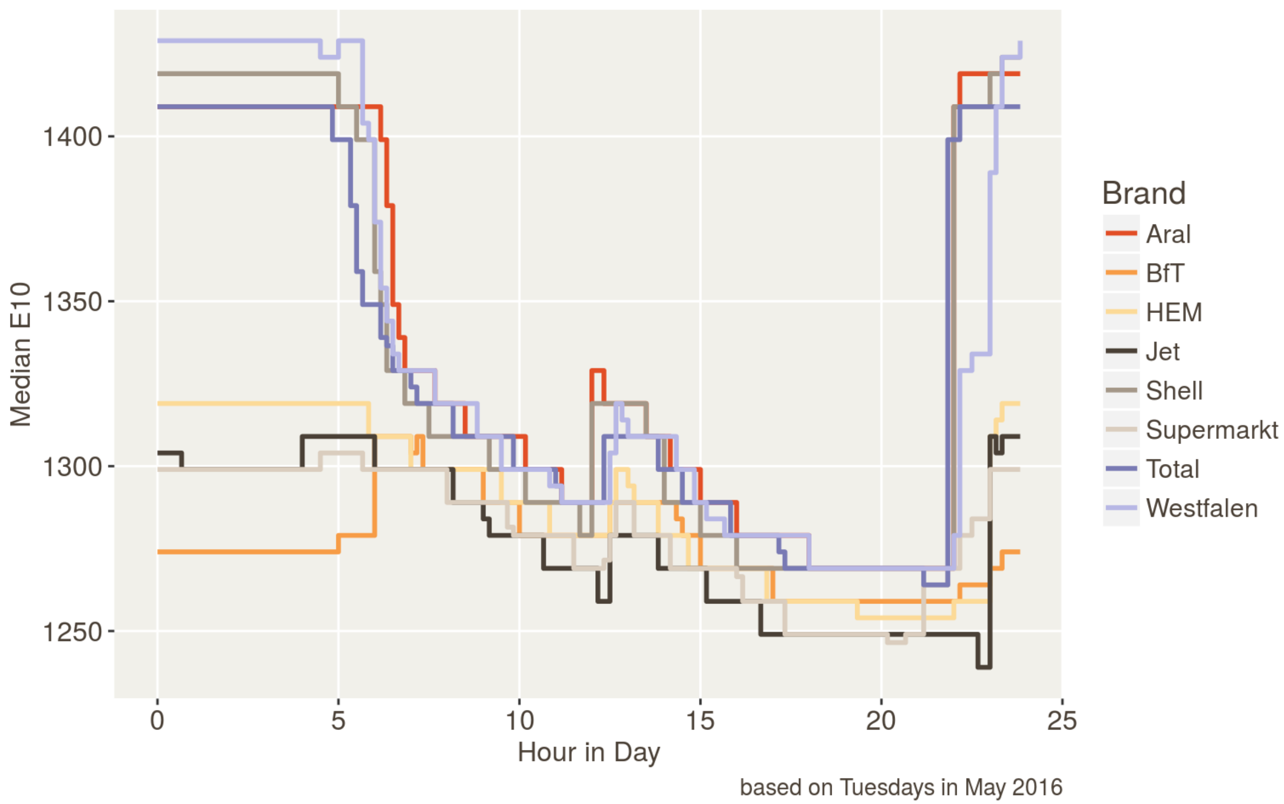




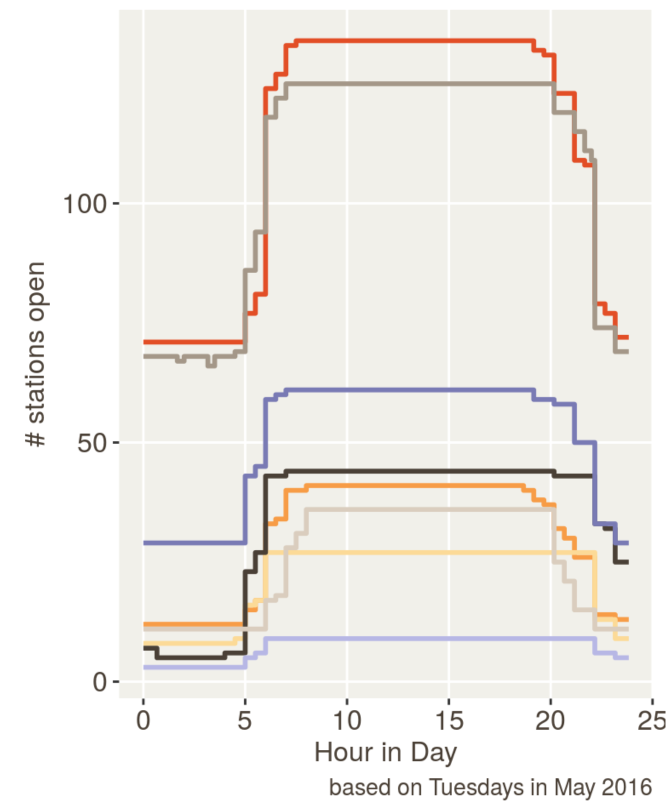
D

Including opening hours is essential to correctly understand the intra-day price pattern

Typical daily price movements



Typical opening hours



E

Two simple models of prices in 2016

- Daily median prices for 2016
- “Market” explanatory variables scaled to [0, 1]
- Natural splines on long / lat for spatial distribution
- Brand and day of the week as further external inputs
- Model 1: pure lm – no competitor price
- Model 2: splm::spgm – model to include “spatial lag” - average of prices of 10 nearest competitors

```
mod1 <- lm(medE10 ~ 1 +
            Brent +
            factor(Wday)+
            ns(lat, df = 6):ns(lng, df = 6) +
            isBAB + closeBAB +
            brandCl +
            hotels_Sc + StatsKKm2_Sc + popdens_Sc + compdens_Sc,
          data = pricesAgg)
```

F Automated cleanup of coefficient names is mostly a (mild) exercise in regular expressions (1)

```
keysdf <-  tribble(
  ~kword, ~kgroup, ~kname, ~kremove, ~kshow, ~kfactor, ~kunit,
  "Wday",  "Time", "Day of Week", "factor\\(\\w+\\)[\\.]*",  TRUE, 0.1, "Ct",
  "CompMean", "Price In", "Competitor Price", ".",  TRUE, 1, "Factor",
  "popdens_Sc", "Market", "Population Density", ".", TRUE, 100, "Ct",
  ... etc ...)
regstring <- paste(paste0("\\b", keysdf$kword), collapse="|")

resultDF <- resultDF %>%
  mutate(CoefKey = term %>% str_extract(regstring))%>%
  left_join(select(keysdf, CoefKey = kword, CoefCl = kgroup,
                   CoefShort = kname, kremove, kshow, kfactor)) %>%
  mutate(CoefDetail = str_replace_all(term, kremove,""),
         Term = paste(CoefShort, CoefDetail),
         Effect = estimate*kfactor)
```

F Automated cleanup of coefficient names is mostly a (mild) exercise in regular expressions (2)

```
## # A tibble: 6 x 16
##    Name   One        term CoefKey CoefCl CoefShort kremove kshow kfactor
##   <chr> <int>       <chr>   <chr>  <chr>     <chr>   <chr> <lgl>   <dbl>
## 1    LM     1 brandClAral brandCl  Brand           brandCl  TRUE     0.1
## 2    LM     1 brandClAvia brandCl  Brand           brandCl  TRUE     0.1
## 3    LM     1  brandClBfT brandCl  Brand           brandCl  TRUE     0.1
## 4    LM     1 brandClEsso brandCl  Brand           brandCl  TRUE     0.1
## 5    LM     1  brandClHEM brandCl  Brand           brandCl  TRUE     0.1
## 6    LM     1  brandClJet brandCl  Brand           brandCl  TRUE     0.1
## # ... with 7 more variables: CoefDetail <chr>, Term <chr>, estimate <dbl>,
## #   std.error <dbl>, statistic <dbl>, p.value <dbl>, Effect <dbl>
```

F

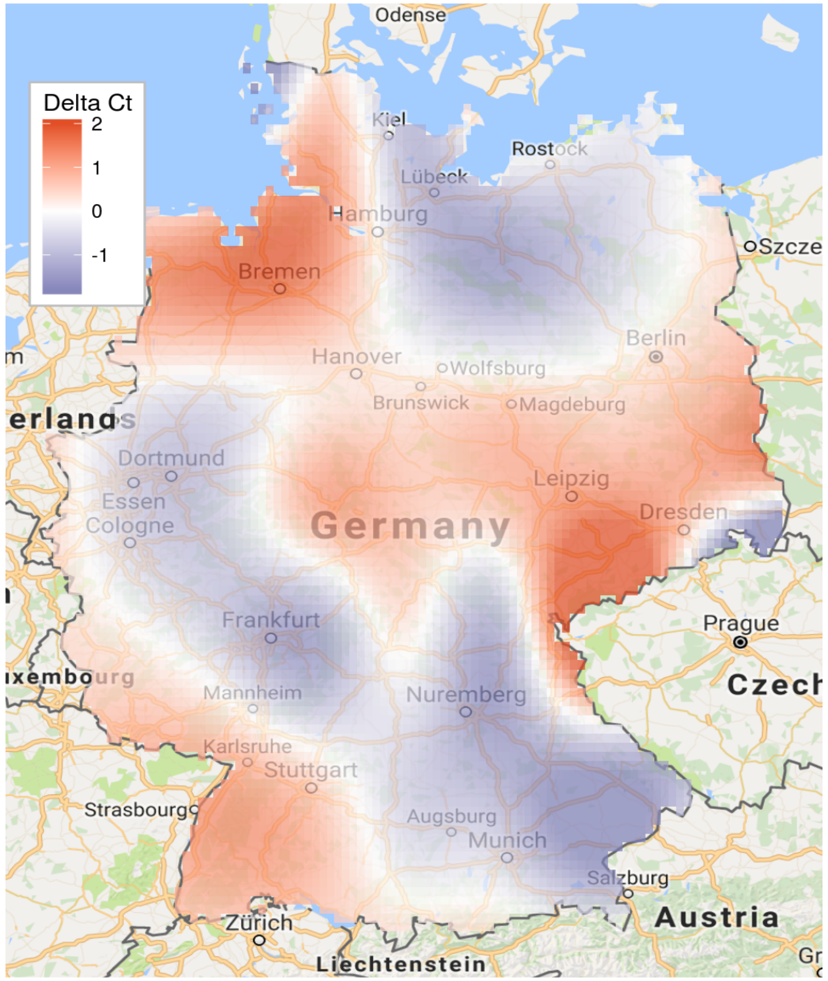
Again, `purrr` can be put to use for the production of the multiple ggplot outputs

```
resultGG <- filter(resultDF, !is.na(kshow) & kshow & p.value < 0.05) %>%
  mutate(CoefCl2 = CoefCl) %>%
  nest(-Name, -CoefCl) %>%
  mutate(gg = map(data,
      ~ggplot(.) + ADJP +
        geom_col(aes(x = factor(Term, levels= rev(levels(factor(Term)))),
                     y =  Effect,
                     fill = CoefCl2)) +
        coord_flip() + ... etc ...
```

```
## # A tibble: 6 x 4
##          Name   CoefCl                 data        gg
##         <chr>    <chr>               <list>    <list>
## 1          LM    Brand <tibble [14 x 15]> <S3: gg>
## 2          LM Location  <tibble [2 x 15]> <S3: gg>
## 3          LM   Market  <tibble [4 x 15]> <S3: gg>
## 4          LM Price In  <tibble [2 x 15]> <S3: gg>
## 5          LM     Time  <tibble [6 x 15]> <S3: gg>
## 6 LM + Comp    Brand <tibble [14 x 15]> <S3: gg>
```

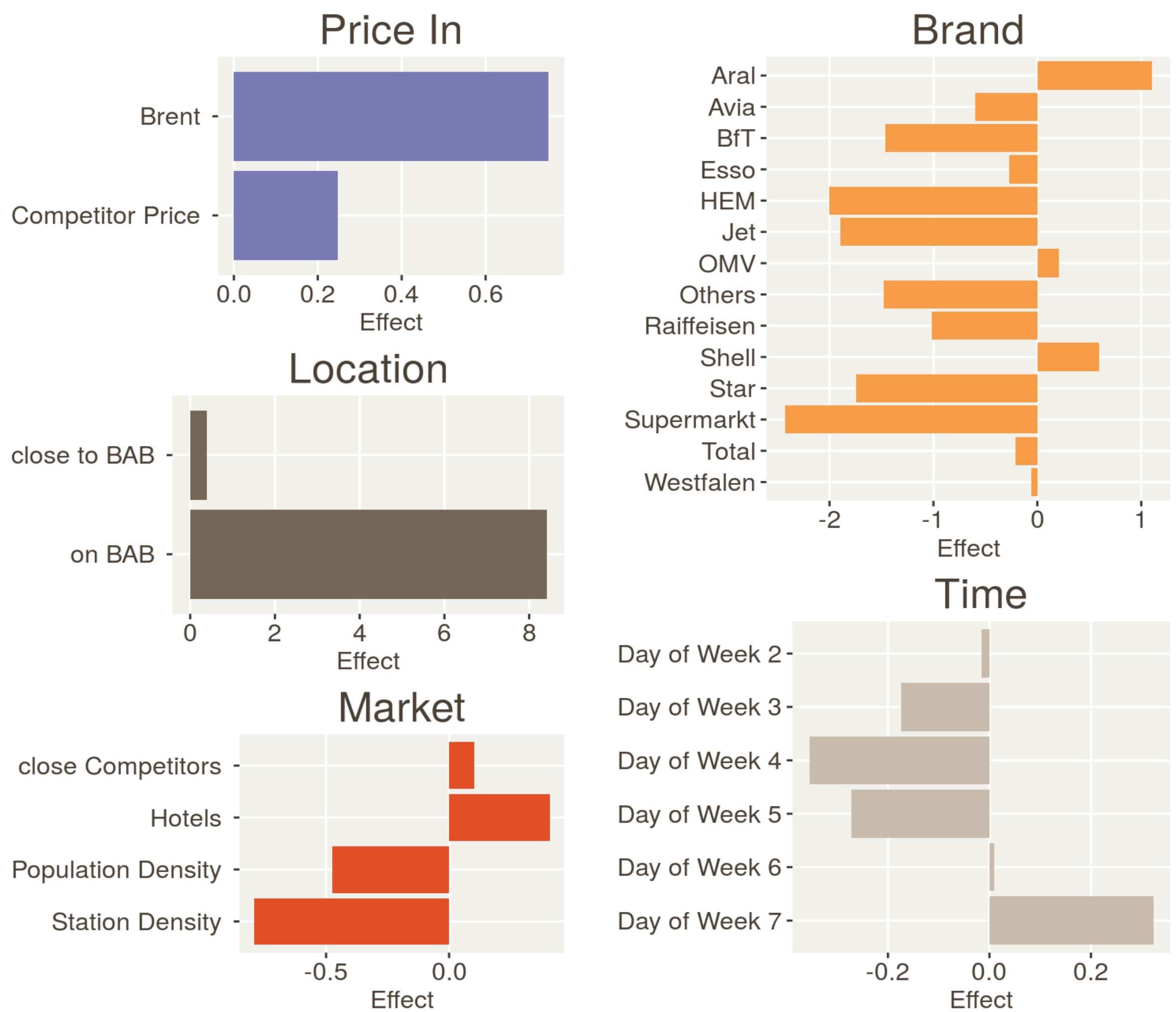
F

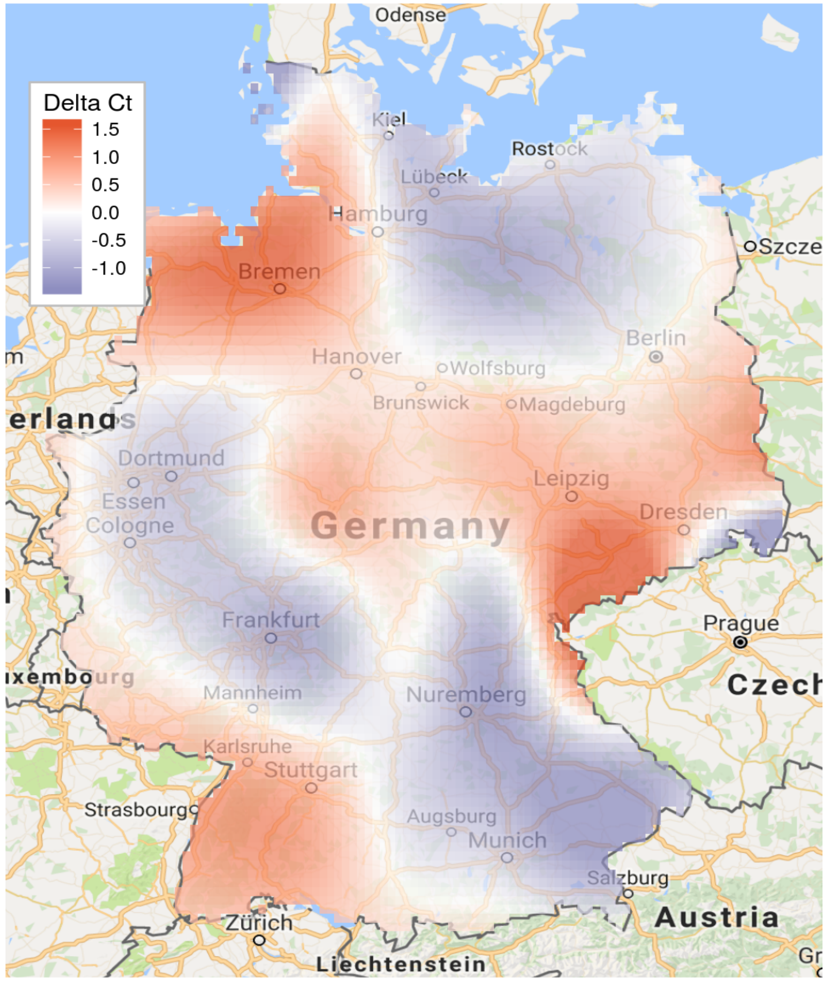
Results for the simple linear model



F

Results for spatial lag (i.e. competitor prices) panel model





Some (traditionalist ?) learnings

Nested data frames and list-columns have proved to be useful data structures in many applications in this project

—

Workflow with `purrr` allows to get problems out of the way sooner (Worked best on initial data consolidation and creation of multiple outputs)

—

Scale up diligently: Make sure you know what to do on smaller pieces of the data. Also, thriftiness (e.g. using integers where possible) still counts.

—

Geo-matching beats text- / ZIP-code- based methods

—

`dplyr` – don't forget `ungroup()`.

—

In Germany, get your fuel at 5pm

—

Check out https://github.com/borva/fuel