```r
knitr::opts_chunk$set(fig.show = 'hide',
                      fig.width = 8.4,
                      fig.height = 7,
                      out.width = "8.4in")
```

```r
library(data.table)
library(dplyr)

##
## Attaching package:  'dplyr'
##
## The following objects are masked from 'package:data.table':
##
##     between, last
##
## The following objects are masked from 'package:stats':
##
##     filter, lag
##
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(magrittr)
library(ggplot2)
library(nycflights13)  # for airports
library(fasttime)
library(grattan)

## Loading required package:  devEMF
##
## Attaching package:  'grattan'
##
## The following object is masked from 'package:datasets':
##
##     Orange
```

```r
pre2008_flights <-
  rbindlist(lapply(list.files(path = "../flights/1987-2008/",
                    pattern = "csv$",
                    full.names = TRUE), fread))
```

```r
pre2008.names <-
  names(pre2008_flights)

read_and_report <-
  function(filename){
    year <- gsub("^.*(2[0-9]{3}).{3,4}csv$", "\\1", filename)
    if(grepl("1.csv", filename, fixed = TRUE))
      cat(year)
    fread(filename, select = pre2008.names, showProgress = FALSE)
  }

gc(1,1)
post2008_flights <-
  rbindlist(lapply(list.files(path = "../flights", recursive = TRUE, pattern = "2[0-9]{3
                              full.names = TRUE),
                   read_and_report))

flights <- rbindlist(list(pre2008_flights, post2008_flights), use.names = TRUE)
readr::write_csv(flights, path = "../1987-2015-On-Time-Performance.csv")
```

```r
Sys.time()

## [1] "2016-01-03 22:33:43 AEDT"

flights <- fread("../1987-2015-On-Time-Performance.csv")

##
Read 0.0% of 165931626 rows
Read 0.5% of 165931626 rows
Read 1.0% of 165931626 rows
Read 1.5% of 165931626 rows
Read 2.0% of 165931626 rows
Read 2.5% of 165931626 rows
Read 3.1% of 165931626 rows
Read 3.6% of 165931626 rows
Read 4.1% of 165931626 rows
Read 4.6% of 165931626 rows
Read 5.1% of 165931626 rows
Read 5.7% of 165931626 rows
Read 6.2% of 165931626 rows
Read 6.7% of 165931626 rows
Read 7.2% of 165931626 rows
Read 7.7% of 165931626 rows
```

```
Read 8.3% of 165931626 rows
Read 8.8% of 165931626 rows
Read 9.3% of 165931626 rows
Read 9.8% of 165931626 rows
Read 10.3% of 165931626 rows
Read 10.9% of 165931626 rows
Read 11.4% of 165931626 rows
Read 11.9% of 165931626 rows
Read 12.4% of 165931626 rows
Read 12.9% of 165931626 rows
Read 13.4% of 165931626 rows
Read 14.0% of 165931626 rows
Read 14.5% of 165931626 rows
Read 15.0% of 165931626 rows
Read 15.5% of 165931626 rows
Read 16.0% of 165931626 rows
Read 16.6% of 165931626 rows
Read 17.1% of 165931626 rows
Read 17.6% of 165931626 rows
Read 18.1% of 165931626 rows
Read 18.6% of 165931626 rows
Read 19.2% of 165931626 rows
Read 19.7% of 165931626 rows
Read 20.2% of 165931626 rows
Read 20.7% of 165931626 rows
Read 21.2% of 165931626 rows
Read 21.8% of 165931626 rows
Read 22.3% of 165931626 rows
Read 22.8% of 165931626 rows
Read 23.3% of 165931626 rows
Read 23.8% of 165931626 rows
Read 24.3% of 165931626 rows
Read 24.8% of 165931626 rows
Read 25.4% of 165931626 rows
Read 25.9% of 165931626 rows
Read 26.4% of 165931626 rows
Read 26.9% of 165931626 rows
Read 27.4% of 165931626 rows
Read 27.9% of 165931626 rows
Read 28.4% of 165931626 rows
Read 28.9% of 165931626 rows
Read 29.5% of 165931626 rows
Read 30.0% of 165931626 rows
```

```
Read 30.5% of 165931626 rows
Read 31.0% of 165931626 rows
Read 31.5% of 165931626 rows
Read 32.0% of 165931626 rows
Read 32.5% of 165931626 rows
Read 33.0% of 165931626 rows
Read 33.6% of 165931626 rows
Read 34.1% of 165931626 rows
Read 34.6% of 165931626 rows
Read 35.1% of 165931626 rows
Read 35.6% of 165931626 rows
Read 36.1% of 165931626 rows
Read 36.6% of 165931626 rows
Read 37.1% of 165931626 rows
Read 37.6% of 165931626 rows
Read 38.2% of 165931626 rows
Read 38.7% of 165931626 rows
Read 39.2% of 165931626 rows
Read 39.7% of 165931626 rows
Read 40.2% of 165931626 rows
Read 40.7% of 165931626 rows
Read 41.2% of 165931626 rows
Read 41.7% of 165931626 rows
Read 42.2% of 165931626 rows
Read 42.7% of 165931626 rows
Read 43.3% of 165931626 rows
Read 43.7% of 165931626 rows
Read 44.3% of 165931626 rows
Read 44.8% of 165931626 rows
Read 45.3% of 165931626 rows
Read 45.8% of 165931626 rows
Read 46.3% of 165931626 rows
Read 46.8% of 165931626 rows
Read 47.3% of 165931626 rows
Read 47.8% of 165931626 rows
Read 48.4% of 165931626 rows
Read 48.9% of 165931626 rows
Read 49.4% of 165931626 rows
Read 49.9% of 165931626 rows
Read 50.4% of 165931626 rows
Read 50.9% of 165931626 rows
Read 51.4% of 165931626 rows
Read 52.0% of 165931626 rows
```

```
Read 52.5% of 165931626 rows
Read 53.0% of 165931626 rows
Read 53.5% of 165931626 rows
Read 54.0% of 165931626 rows
Read 54.6% of 165931626 rows
Read 55.1% of 165931626 rows
Read 55.6% of 165931626 rows
Read 56.1% of 165931626 rows
Read 56.6% of 165931626 rows
Read 57.1% of 165931626 rows
Read 57.7% of 165931626 rows
Read 58.2% of 165931626 rows
Read 58.7% of 165931626 rows
Read 59.2% of 165931626 rows
Read 59.7% of 165931626 rows
Read 60.3% of 165931626 rows
Read 60.8% of 165931626 rows
Read 61.3% of 165931626 rows
Read 61.8% of 165931626 rows
Read 62.3% of 165931626 rows
Read 62.8% of 165931626 rows
Read 63.4% of 165931626 rows
Read 63.9% of 165931626 rows
Read 64.4% of 165931626 rows
Read 64.9% of 165931626 rows
Read 65.4% of 165931626 rows
Read 65.9% of 165931626 rows
Read 66.5% of 165931626 rows
Read 67.0% of 165931626 rows
Read 67.5% of 165931626 rows
Read 68.0% of 165931626 rows
Read 68.5% of 165931626 rows
Read 69.1% of 165931626 rows
Read 69.6% of 165931626 rows
Read 70.1% of 165931626 rows
Read 70.6% of 165931626 rows
Read 71.1% of 165931626 rows
Read 71.6% of 165931626 rows
Read 72.1% of 165931626 rows
Read 72.6% of 165931626 rows
Read 73.1% of 165931626 rows
Read 73.6% of 165931626 rows
Read 74.2% of 165931626 rows
```

```
Read 74.7% of 165931626 rows
Read 75.2% of 165931626 rows
Read 75.7% of 165931626 rows
Read 76.2% of 165931626 rows
Read 76.7% of 165931626 rows
Read 77.2% of 165931626 rows
Read 77.7% of 165931626 rows
Read 78.2% of 165931626 rows
Read 78.8% of 165931626 rows
Read 79.3% of 165931626 rows
Read 79.8% of 165931626 rows
Read 80.3% of 165931626 rows
Read 80.8% of 165931626 rows
Read 81.3% of 165931626 rows
Read 81.8% of 165931626 rows
Read 82.3% of 165931626 rows
Read 82.8% of 165931626 rows
Read 83.3% of 165931626 rows
Read 83.8% of 165931626 rows
Read 84.3% of 165931626 rows
Read 84.8% of 165931626 rows
Read 85.4% of 165931626 rows
Read 85.9% of 165931626 rows
Read 86.4% of 165931626 rows
Read 86.9% of 165931626 rows
Read 87.4% of 165931626 rows
Read 87.9% of 165931626 rows
Read 88.4% of 165931626 rows
Read 88.9% of 165931626 rows
Read 89.4% of 165931626 rows
Read 89.9% of 165931626 rows
Read 90.4% of 165931626 rows
Read 90.9% of 165931626 rows
Read 91.4% of 165931626 rows
Read 91.9% of 165931626 rows
Read 92.4% of 165931626 rows
Read 92.9% of 165931626 rows
Read 93.4% of 165931626 rows
Read 93.9% of 165931626 rows
Read 94.5% of 165931626 rows
Read 95.0% of 165931626 rows
Read 95.5% of 165931626 rows
Read 96.0% of 165931626 rows
```

```
Read 96.5% of 165931626 rows
Read 97.0% of 165931626 rows
Read 97.5% of 165931626 rows
Read 98.0% of 165931626 rows
Read 98.5% of 165931626 rows
Read 99.0% of 165931626 rows
Read 99.5% of 165931626 rows
Read 165931626 rows and 29 (of 29) columns from 15.111 GB file in 00:04:16

# flights <- readRDS("../1987-2015-On-Time-Performance.rds")
```

```
sample.frac = 0.1
sample.weight.int = as.integer(round(1/sample.frac))
flights <- flights[sample(.N, .N * sample.frac)]
```

```
# First we want a time for each flight. This is more difficult that it might seem.
# We need to concatenate the Year, Month, and DayofMonth fields, but we also need
# to take into account the various time zones of the airports in the database.
integer.cols <- grep("Time$", names(flights))

Sys.time()

## [1] "2016-01-03 22:38:20 AEDT"

for (j in integer.cols){
  set(flights, j = j, value = as.integer(flights[[j]]))
}
Sys.time()

## [1] "2016-01-03 22:38:20 AEDT"

# See stackoverflow: links and comments under my question
create_DepDateTime <- function(DT){
  setkey(DT, Year, Month, DayofMonth, DepTime)
  unique_dates <- unique(DT[,list(Year, Month, DayofMonth, DepTime)])
  unique_dates[,DepDateTime := fastPOSIXct(sprintf("%d-%02d-%02d %s", Year, Month, DayofM
                                                    sub("([0-9]{2})([0-9]{2})", "\\1:\\2:0
                                                        perl = TRUE)),
                                            tz = "GMT")]
  DT[unique_dates]
}
```

```
create_ArrDateTime <- function(DT){
  setkey(DT, Year, Month, DayofMonth, ArrTime)
  unique_dates <- unique(DT[,list(Year, Month, DayofMonth, ArrTime)])
  unique_dates[,ArrDateTime := fastPOSIXct(sprintf("%d-%02d-%02d %s", Year, Month, DayofM
                                                    sub("([0-9]{2})([0-9]{2})", "\\1:\\2:0
                                                        perl = TRUE)),
                                            tz = "GMT")]
  DT[unique_dates]
}
flights <- create_DepDateTime(flights)
flights <- create_ArrDateTime(flights)
#flights[,`:=`(Year = NULL, Month = NULL, DayofMonth = NULL, DepTime = NULL, ArrTime = N
Sys.time()

## [1] "2016-01-03 22:39:37 AEDT"
```

```
# Now we join it to the airports dataset from nycflights13 to obtain time zone informati
Sys.time()

## [1] "2016-01-03 22:39:37 AEDT"

airports <- as.data.table(airports)
airports <- airports[,list(faa, tz)]
gc(1,1)

##              used   (Mb) gc trigger   (Mb)  max used   (Mb)
## Ncells     533293   28.5    7974897  426.0    533293   28.5
## Vcells 324371914 2474.8  973788657 7429.5 324371914 2474.8

setnames(airports, old = c("faa", "tz"), new = c("Origin", "tzOrigin"))
setkey(airports, Origin)
setkey(flights, Origin)
flights <- flights[airports]
setnames(airports, old = c("Origin", "tzOrigin"), new = c("Dest", "tzDest"))
setkey(flights, Dest)
flights <- flights[airports]
rm(airports)
gc(1,1)

##              used   (Mb) gc trigger   (Mb)  max used   (Mb)
## Ncells     533306   28.5    5103933  272.6    533306   28.5
## Vcells 354325754 2703.3  973788657 7429.5 354325754 2703.3
```

```r
# The joins produce NAs when the airports table isn't present in the flights table.
flights <- flights[!is.na(Origin)]
gc(1,1)

##              used   (Mb) gc trigger   (Mb)  max used   (Mb)
## Ncells    533321   28.5    4083146  218.1    533321   28.5
## Vcells 354303746 2703.2  973788657 7429.5 354303746 2703.2

Sys.time()

## [1] "2016-01-03 22:39:56 AEDT"
```

```r
Sys.time()

## [1] "2016-01-03 22:39:56 AEDT"

setkey(flights, DepDateTime)
flights[,`:=`(DepDateTimeZulu = DepDateTime - lubridate::hours(tzOrigin),
              ArrDateTimeZulu = ArrDateTime - lubridate::hours(tzDest) )]
Sys.time()

## [1] "2016-01-03 22:41:52 AEDT"
```

```r
# Flights typically follow a weekly cycle, so we should obtain the week in the dataset.
# Pretty quick!
Sys.time()

## [1] "2016-01-03 22:41:52 AEDT"

setkey(flights, Year, Month, DayofMonth)
unique_dates <- unique(flights)
unique_dates <- unique_dates[,list(Year, Month, DayofMonth)]
unique_dates[,Week := (Year - 1987L) * 52 + data.table::yday(sprintf("%d-%02d-%02d", Year
unique_dates[,Week := Week - min(Week)]
flights <- flights[unique_dates]
Sys.time()

## [1] "2016-01-03 22:41:57 AEDT"
```

Flights 1987-2015

Hugh P

January 3, 2016

1

There were 164 million flights from 1987-10-01 05:00:00 to 2015-11-01 09:43:00.

2 San Francisco

```
SanFran_flights <-
  flights %>%
  filter(Origin %in% c("SFO", "OAK") | Dest %in% c("SFO", "OAK"))
```

```
SanFran_flights %>%
  filter(!(Origin %in% c("SFO", "OAK") & Dest %in% c("SFO", "OAK"))) %>%
  mutate(SF_airport = ifelse(Origin %in% c("SFO", "OAK"),
                             Origin,
                             Dest)) %>%
  count(Week, SF_airport) %>%
  mutate(Date = Week,
         n = n) %>%  # sample
  ggplot(aes(x = Date, y = n, color = SF_airport, group = SF_airport)) +
  geom_line()
  stat_smooth(n = 10000, span = 0.01, se = TRUE)
```

```
carriers <- as.data.table(airlines)
if("carrier" %in% names(carriers))
  setnames(carriers, old = "carrier", new = "UniqueCarrier")

setkey(carriers, UniqueCarrier)
set(carriers, j = 1L, value = as.character(carriers[[1L]]))
set(carriers, j = 2L, value = gsub("^([A-Za-z]+)\\s.*$", "\\1", carriers[[2L]]))

SanFran_flights %>%
  filter(Origin %in% c("SFO", "OAK")) %>%
  count(Year, Month, Origin, UniqueCarrier) %>%
  group_by(UniqueCarrier) %>%
  filter(sum(n) > (2015 - 1987) * 12 * 30)  %>%
  mutate(Date = Year + (Month - 1)/12) %>%
  setkey(UniqueCarrier) %>%
  merge(carriers) %>%
  ggplot(aes(x = Date, y = n * sample.weight.int, color = name, group = interaction(name
  geom_smooth(span = 0.25, se = FALSE) +
```

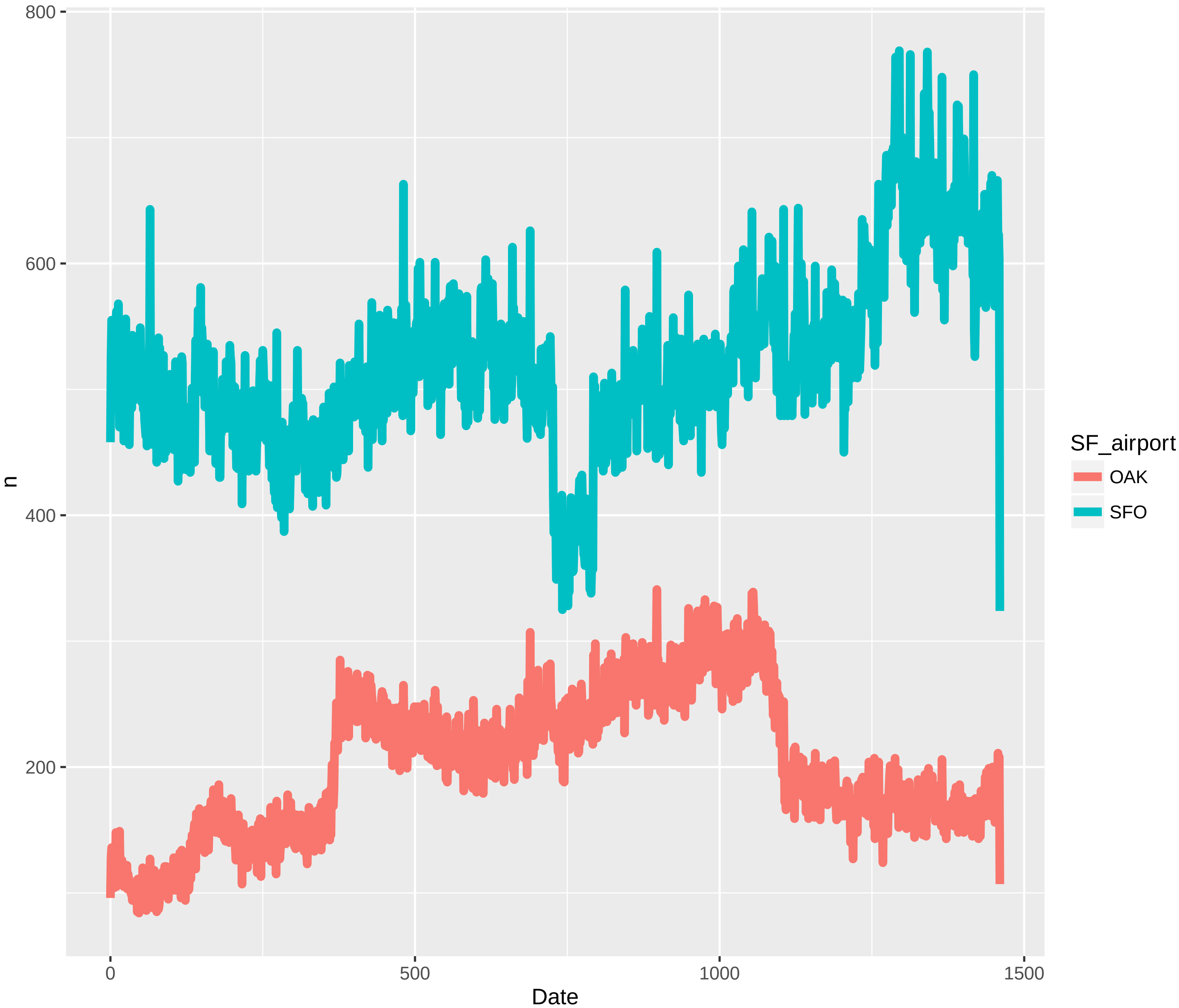

Figure 2.1: Number of depatures over time from Oakland and San Francisco Intl.

```r
  geom_text(aes(label = ifelse(Date == max(Date),
                               name,
                               NA_character_),
                vjust = ifelse(name == "Southwest" & Origin == "SFO",
                                 -0.5,
                                 0.5)),
            nudge_x = 0.75,
            size = 5) + theme(legend.position = "none") +
  annotate("blank", x = 2019, y = 0) +
  facet_grid(Origin ~ .) +
  theme(text = element_text(size = 16))

## Warning:  Removed 3926 rows containing missing values (geom_text).
```

After September 11, flights from SFO fell, whereas OAK's volume did notFlights fell more in SFO than they did in OAK because most of OAK's flights are from Southwest, which did not change its flight patterns. Furthermore, United was affected more than most airlines from the aftermath of the attacks.

```r
flights %>%
  count(Year)

## Source: local data table [29 x 2]
##
##     Year      n
##    (int)  (int)
## 1   1987 129860
## 2   1988 516321
## 3   1989 498605
## 4   1990 521873
## 5   1991 502872
## 6   1992 503344
## 7   1993 501444
## 8   1994 513989
## 9   1995 528486
## 10  1996 530212
## ..   ...    ...
```

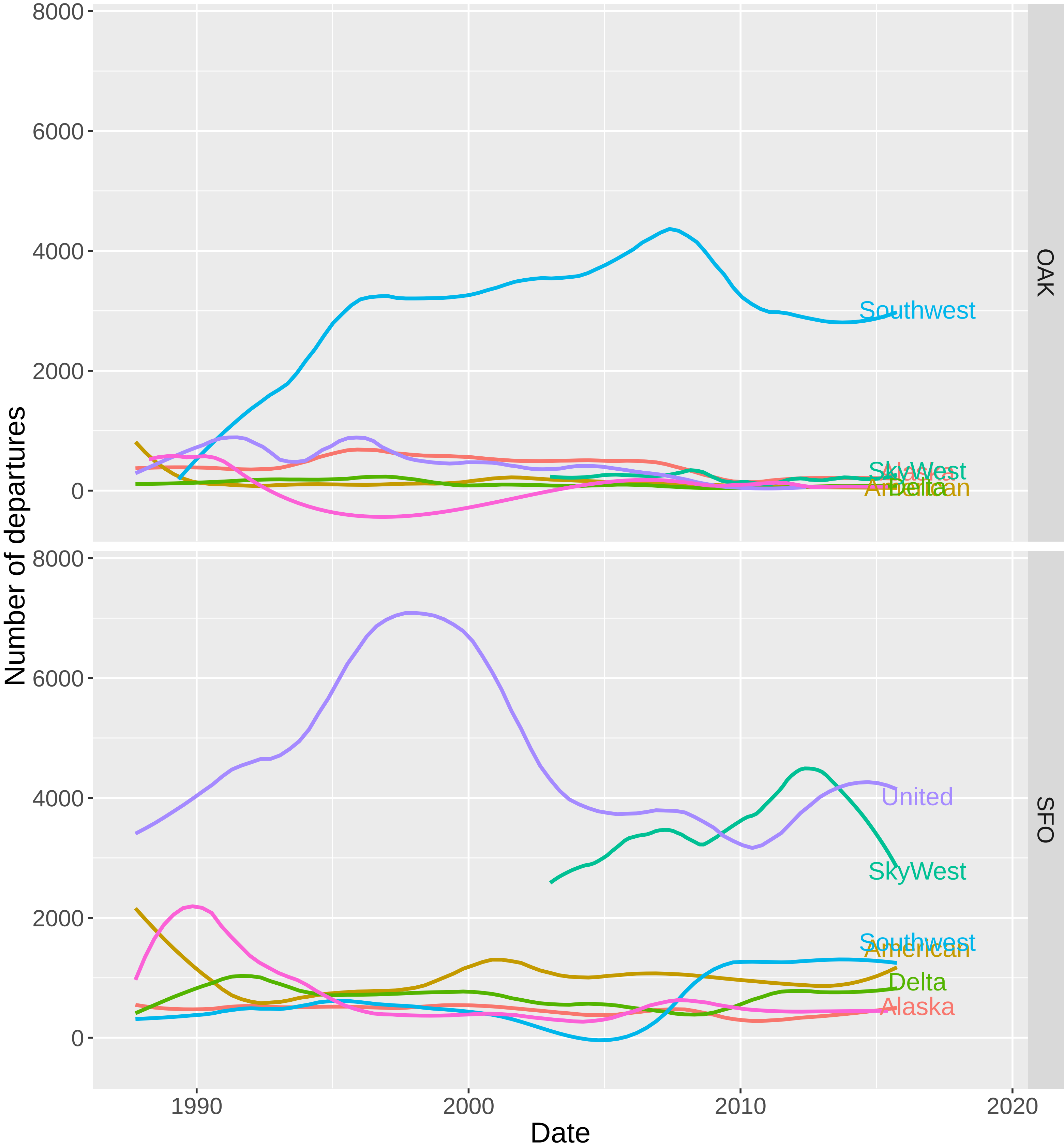

Figure 2.2: Number of depatures over time from Oakland and San Francisco Intl.