R Event Study

David Zimmermann

2016-11-23

Contents

1 Introduction

The following script provides an application of an event study. The data and and the script itself can be found here: https://github.com/DavZim/Classes/tree/master/Advanced%20Corporate%20Finance/Event%20Study

To run this notebook, make sure that you have at least RStudio version 1.0.44 installed and run R version 3.3.2.

The general outline for this document is to first load the data, merge the necessary data, inspect the data both visually and via tables, estimate a CMRM (constant-mean-return model), calculate the ARs (abnormal returns), and CARs (cumulative abnormal returns) and then test for significance using a t-test.

2 Preparation

2.1 Load libraries

```
library(dplyr)     # for data manipulation
library(ggplot2)   # for plotting
library(lubridate) # for dates
library(readr)     # for data loading
library(scales)    # for plotting
library(tidyr)     # for tidy data
```

2.2 Load data

```
mar_wide <- read_csv("data/market.csv")
ret_wide <- read_csv("data/returns.csv")
events <- read_csv("data/events.csv")

# reshape returns and market to long format
returns <- gather(ret_wide, key = company,
              value = ret, -date)
market <- gather(mar_wide, key = country,
                  value = mret, -date)

# date formatting
returns <- returns %>% mutate(date = dmy(date))
market <- market %>% mutate(date = dmy(date))
events <- events %>% mutate(event = dmy(event))
```

2.3 Inspect data

```
returns
```

```
## # A tibble: 26,604 × 3
##          date  company         ret
##        <dttm>    <chr>       <dbl>
## 1  1997-01-02 Chrysler  0.034094044
## 2  1997-01-03 Chrysler  0.014647928
## 3  1997-01-06 Chrysler  0.028884879
```

```
## 4  1997-01-07 Chrysler  0.000000000
## 5  1997-01-08 Chrysler -0.007015588
## 6  1997-01-09 Chrysler  0.000000000
## 7  1997-01-10 Chrysler  0.010597731
## 8  1997-01-13 Chrysler -0.020973193
## 9  1997-01-14 Chrysler  0.007140830
## 10 1997-01-15 Chrysler  0.000000000
## # ... with 26,594 more rows
```

```
market
```

```
## # A tibble: 17,736 × 3
##          date country         mret
##        <dttm>   <chr>        <dbl>
## 1  1997-01-02      us -0.007505188
## 2  1997-01-03      us  0.014872343
## 3  1997-01-06      us  0.000371782
## 4  1997-01-07      us  0.007912912
## 5  1997-01-08      us -0.005192890
## 6  1997-01-09      us  0.008200646
## 7  1997-01-10      us  0.006127263
## 8  1997-01-13      us -0.000456746
## 9  1997-01-14      us  0.012215926
## 10 1997-01-15      us -0.002693593
## # ... with 17,726 more rows
```

```
events
```

```
## # A tibble: 6 × 2
##       company      event
##         <chr>     <dttm>
## 1    Chrysler 1998-05-06
## 2   BellSouth 2006-03-06
## 3   Engelhard 2006-01-03
## 4 Norsk Hydro 2006-12-18
## 5  Pilkington 2005-10-31
## 6         INA 1999-09-14
```

2.4 Merge Data

```
comps <- c("Chrysler", "BellSouth", "Engelhard", "Norsk Hydro", "Pilkington", "INA")
counts <- c("us", "us", "us", "norway", "uk", "italy")
countries <- data_frame(company = comps, country = counts)

# merge into one dataset
merged <- left_join(returns, countries, by = "company")
merged <- left_join(merged, market, by = c("date", "country"))
merged <- left_join(merged, events, by = "company")
merged
```

```
## # A tibble: 26,604 × 6
##          date  company          ret country         mret      event
##        <dttm>    <chr>        <dbl>   <chr>        <dbl>     <dttm>
## 1  1997-01-02 Chrysler  0.034094044      us -0.007505188 1998-05-06
## 2  1997-01-03 Chrysler  0.014647928      us  0.014872343 1998-05-06
```

```
## 3  1997-01-06 Chrysler  0.028884879      us  0.000371782 1998-05-06
## 4  1997-01-07 Chrysler  0.000000000      us  0.007912912 1998-05-06
## 5  1997-01-08 Chrysler -0.007015588      us -0.005192890 1998-05-06
## 6  1997-01-09 Chrysler  0.000000000      us  0.008200646 1998-05-06
## 7  1997-01-10 Chrysler  0.010597731      us  0.006127263 1998-05-06
## 8  1997-01-13 Chrysler -0.020973193      us -0.000456746 1998-05-06
## 9  1997-01-14 Chrysler  0.007140830      us  0.012215926 1998-05-06
## 10 1997-01-15 Chrysler  0.000000000      us -0.002693593 1998-05-06
## # ... with 26,594 more rows
```

2.5 Estimation and Events

```
# calculate the event-time as the difference in days to the event
merged <- merged %>% group_by(company) %>%
  mutate(date_index = 1:n(),
         event_index = max(ifelse(event == date, date_index, 0)),
         event_time = date_index - event_index)

merged
```

```
## Source: local data frame [26,604 x 9]
## Groups: company [6]
##
##          date  company          ret country         mret      event
##        <dttm>    <chr>        <dbl>   <chr>        <dbl>     <dttm>
## 1  1997-01-02 Chrysler  0.034094044      us -0.007505188 1998-05-06
## 2  1997-01-03 Chrysler  0.014647928      us  0.014872343 1998-05-06
## 3  1997-01-06 Chrysler  0.028884879      us  0.000371782 1998-05-06
## 4  1997-01-07 Chrysler  0.000000000      us  0.007912912 1998-05-06
## 5  1997-01-08 Chrysler -0.007015588      us -0.005192890 1998-05-06
## 6  1997-01-09 Chrysler  0.000000000      us  0.008200646 1998-05-06
## 7  1997-01-10 Chrysler  0.010597731      us  0.006127263 1998-05-06
## 8  1997-01-13 Chrysler -0.020973193      us -0.000456746 1998-05-06
## 9  1997-01-14 Chrysler  0.007140830      us  0.012215926 1998-05-06
## 10 1997-01-15 Chrysler  0.000000000      us -0.002693593 1998-05-06
## # ... with 26,594 more rows, and 3 more variables: date_index <int>,
## #   event_index <dbl>, event_time <dbl>
```

Now we want to split our sample into estimation-sample ($[-230, +31]$) and event-sample ($[-30, +30]$). We also want to have a quick visualization of the return correlations to the market.

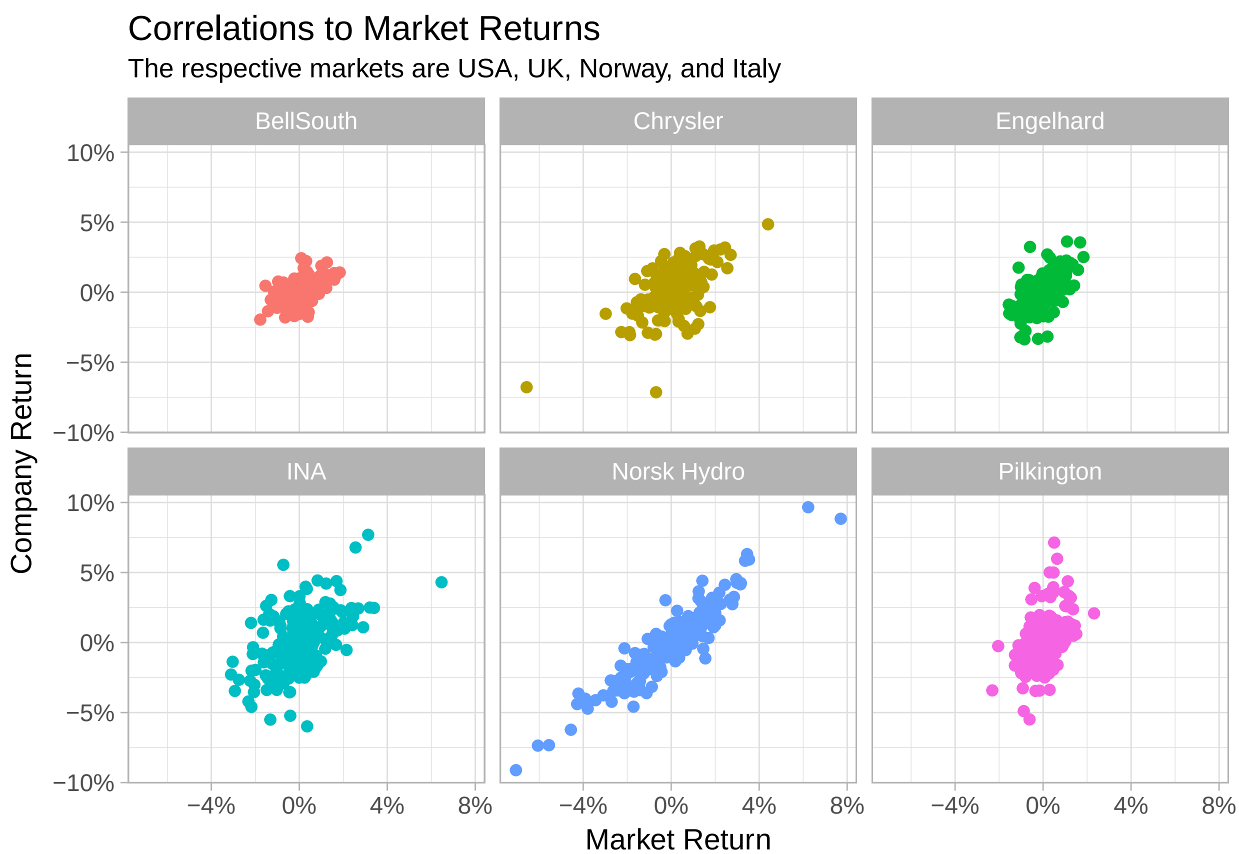
```
# windows
estimation_window <- c(-230, -31)
event_window <- c(-30, 30)

# filter returns
estimation <- merged %>% filter(event_time >= estimation_window[1] &
                                  event_time <= estimation_window[2])

event <- merged %>% filter(event_time >= event_window[1] &
                             event_time <= event_window[2])

# have a look at the data
estimation
```

```
## Source: local data frame [1,200 x 9]
## Groups: company [6]
##
##          date  company          ret country         mret      event
##        <dttm>    <chr>        <dbl>   <chr>        <dbl>     <dttm>
## 1  1997-06-18 Chrysler  0.003837413      us -0.004615305 1998-05-06
## 2  1997-06-19 Chrysler -0.003822743      us  0.009952588 1998-05-06
## 3  1997-06-20 Chrysler  0.000000000      us  0.000297446 1998-05-06
## 4  1997-06-23 Chrysler -0.011549495      us -0.020220303 1998-05-06
## 5  1997-06-24 Chrysler  0.023356325      us  0.018024913 1998-05-06
## 6  1997-06-25 Chrysler -0.003805923      us -0.007569669 1998-05-06
## 7  1997-06-26 Chrysler  0.011449065      us -0.005733615 1998-05-06
## 8  1997-06-27 Chrysler -0.011319467      us  0.003875103 1998-05-06
## 9  1997-06-30 Chrysler  0.003820463      us -0.002630133 1998-05-06
## 10 1997-07-01 Chrysler -0.009514806      us  0.006586111 1998-05-06
## # ... with 1,190 more rows, and 3 more variables: date_index <int>,
## #   event_index <dbl>, event_time <dbl>
```

```
event
```

```
## Source: local data frame [366 x 9]
## Groups: company [6]
##
##          date  company          ret country         mret      event
##        <dttm>    <chr>        <dbl>   <chr>        <dbl>     <dttm>
## 1  1998-03-25 Chrysler -0.017288936      us -0.002219378 1998-05-06
## 2  1998-03-26 Chrysler -0.013199413      us -0.000496591 1998-05-06
## 3  1998-03-27 Chrysler  0.001487251      us -0.004668205 1998-05-06
## 4  1998-03-30 Chrysler -0.005930890      us -0.001507903 1998-05-06
## 5  1998-03-31 Chrysler -0.005975612      us  0.007873770 1998-05-06
## 6  1998-04-01 Chrysler  0.006011535      us  0.006251873 1998-05-06
## 7  1998-04-02 Chrysler -0.020895968      us  0.009735458 1998-05-06
## 8  1998-04-03 Chrysler -0.003051572      us  0.002176478 1998-05-06
## 9  1998-04-06 Chrysler  0.021407254      us -0.003866529 1998-05-06
## 10 1998-04-07 Chrysler  0.002996760      us -0.009708936 1998-05-06
## # ... with 356 more rows, and 3 more variables: date_index <int>,
## #   event_index <dbl>, event_time <dbl>
```

```
# Graph data
theme_set(theme_light())
ggplot(estimation %>% filter(ret != 0), aes(x = mret, y = ret, color = company)) +
  geom_point() +
  facet_wrap(~company) +
  scale_x_continuous(labels = percent) +
  scale_y_continuous(labels = percent) +
   theme(legend.position = "none") +
  labs(title = "Correlations to Market Returns",
       subtitle = "The respective markets are USA, UK, Norway, and Italy",
       x = "Market Return", y = "Company Return")
```



3 Estimation

3.1 Calculate the CMRM

Although we have many options, this script uses the constant-mean-return model to calculate expected returns (for simplicity reasons mainy).

The expected return is given by

$$E\left[R_{i,t}|X_t\right]$$

using the CMRM (constant mean return model), we get

$$E[R_{i,t}|X_t] = \overline{R_{i,t}}$$

```r
cmrm <- estimation %>% group_by(company) %>% summarise(cmrm = mean(ret))
cmrm
```

```
## # A tibble: 6 × 2
##        company          cmrm
##          <chr>         <dbl>
## 1   BellSouth  3.948146e-04
## 2    Chrysler  1.732147e-03
## 3  Engelhard -2.613456e-05
## 4         INA  1.122800e-04
```

```
## 5 Norsk Hydro  4.191843e-04
## 6  Pilkington  1.466970e-03
```

Next, we want to merge the expected returns into the event-dataset to be able to calculate the next steps.

```
# select only necessary variables
event <- event %>% select(company, ret, event_time)
event <- left_join(event, cmrm, by = "company")
event
```

```
## Source: local data frame [366 x 4]
## Groups: company [?]
##
##     company          ret event_time        cmrm
##       <chr>        <dbl>      <dbl>       <dbl>
## 1  Chrysler -0.017288936        -30 0.001732147
## 2  Chrysler -0.013199413        -29 0.001732147
## 3  Chrysler  0.001487251        -28 0.001732147
## 4  Chrysler -0.005930890        -27 0.001732147
## 5  Chrysler -0.005975612        -26 0.001732147
## 6  Chrysler  0.006011535        -25 0.001732147
## 7  Chrysler -0.020895968        -24 0.001732147
## 8  Chrysler -0.003051572        -23 0.001732147
## 9  Chrysler  0.021407254        -22 0.001732147
## 10 Chrysler  0.002996760        -21 0.001732147
## # ... with 356 more rows
```

3.2 Calculate the Abnormal Returns

The abnormal return in period t for company i is given by

$$AR_{i,t} = R_{i,t} - E\left[R_{i,t}\right]$$

which we can calculate in R like this

```
event <- event %>% mutate(ar = ret - cmrm)
event
```

```
## Source: local data frame [366 x 5]
## Groups: company [6]
##
##     company          ret event_time        cmrm            ar
##       <chr>        <dbl>      <dbl>       <dbl>         <dbl>
## 1  Chrysler -0.017288936        -30 0.001732147 -0.0190210827
## 2  Chrysler -0.013199413        -29 0.001732147 -0.0149315597
## 3  Chrysler  0.001487251        -28 0.001732147 -0.0002448957
## 4  Chrysler -0.005930890        -27 0.001732147 -0.0076630367
## 5  Chrysler -0.005975612        -26 0.001732147 -0.0077077587
## 6  Chrysler  0.006011535        -25 0.001732147  0.0042793883
## 7  Chrysler -0.020895968        -24 0.001732147 -0.0226281147
## 8  Chrysler -0.003051572        -23 0.001732147 -0.0047837187
## 9  Chrysler  0.021407254        -22 0.001732147  0.0196751073
## 10 Chrysler  0.002996760        -21 0.001732147  0.0012646133
## # ... with 356 more rows
```

3.3 Calculate the Cumulative Abnormal Returns

The CARs are given by

$$CAR_{i,t} = \sum_{k=1}^{t} AR_{i,t-k}$$

with a known distribution of

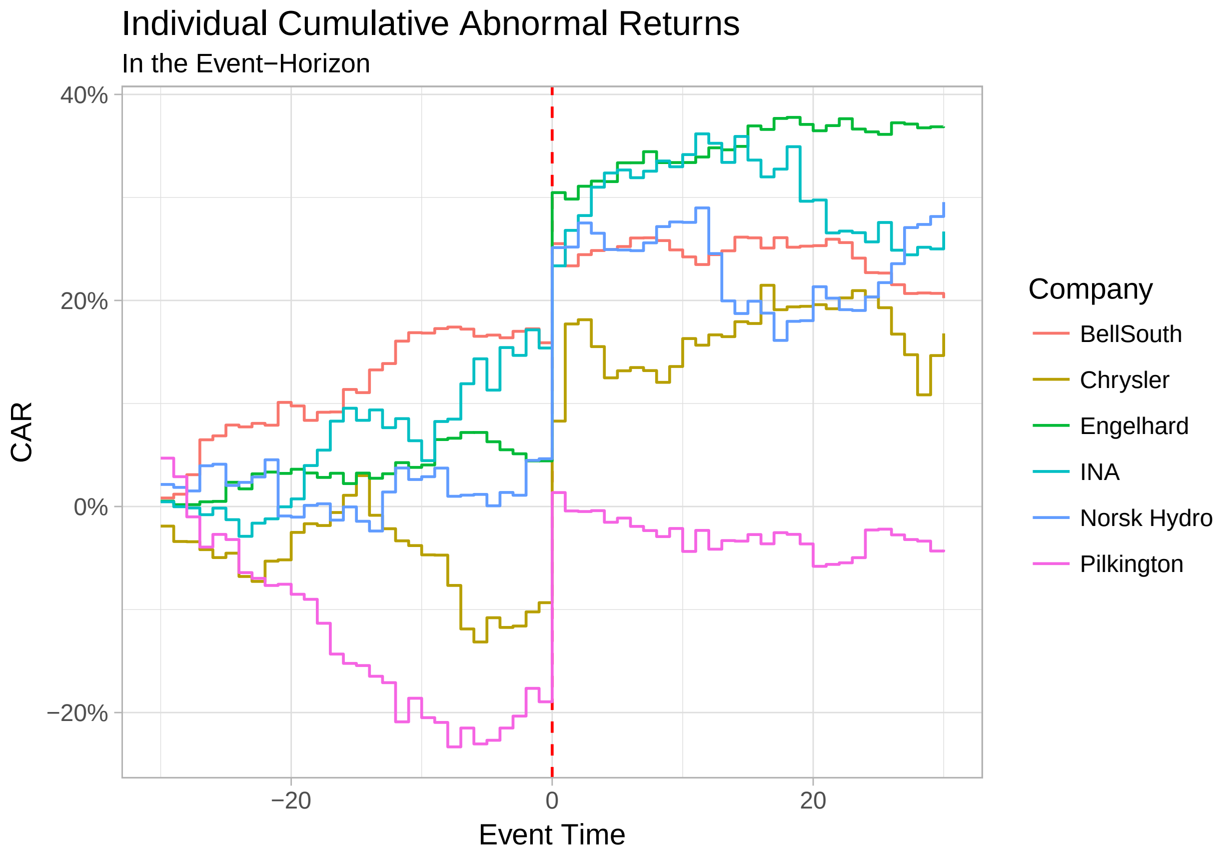
$$CAR_{i,t} \sim N(0, \sigma^2_{i,t})$$

In R we can calculate the CARs like this

```
indiv_event <-  event %>% group_by(company) %>% mutate(car = cumsum(ar))
indiv_event
```

```
## Source: local data frame [366 x 6]
## Groups: company [6]
##
##     company          ret event_time        cmrm            ar         car
##       <chr>        <dbl>      <dbl>       <dbl>         <dbl>       <dbl>
## 1  Chrysler -0.017288936        -30 0.001732147 -0.0190210827 -0.01902108
## 2  Chrysler -0.013199413        -29 0.001732147 -0.0149315597 -0.03395264
## 3  Chrysler  0.001487251        -28 0.001732147 -0.0002448957 -0.03419754
## 4  Chrysler -0.005930890        -27 0.001732147 -0.0076630367 -0.04186057
## 5  Chrysler -0.005975612        -26 0.001732147 -0.0077077587 -0.04956833
## 6  Chrysler  0.006011535        -25 0.001732147  0.0042793883 -0.04528895
## 7  Chrysler -0.020895968        -24 0.001732147 -0.0226281147 -0.06791706
## 8  Chrysler -0.003051572        -23 0.001732147 -0.0047837187 -0.07270078
## 9  Chrysler  0.021407254        -22 0.001732147  0.0196751073 -0.05302567
## 10 Chrysler  0.002996760        -21 0.001732147  0.0012646133 -0.05176106
## # ... with 356 more rows
```

```
ggplot(indiv_event, aes(x = event_time, y = car, color = company)) +
  geom_vline(xintercept = 0, color = "red", linetype = "dashed") +
  geom_step() +
  scale_y_continuous(labels = percent) +
  labs(title = "Individual Cumulative Abnormal Returns", subtitle = "In the Event-Horizon",
       x = "Event Time", y = "CAR", color = "Company")
```



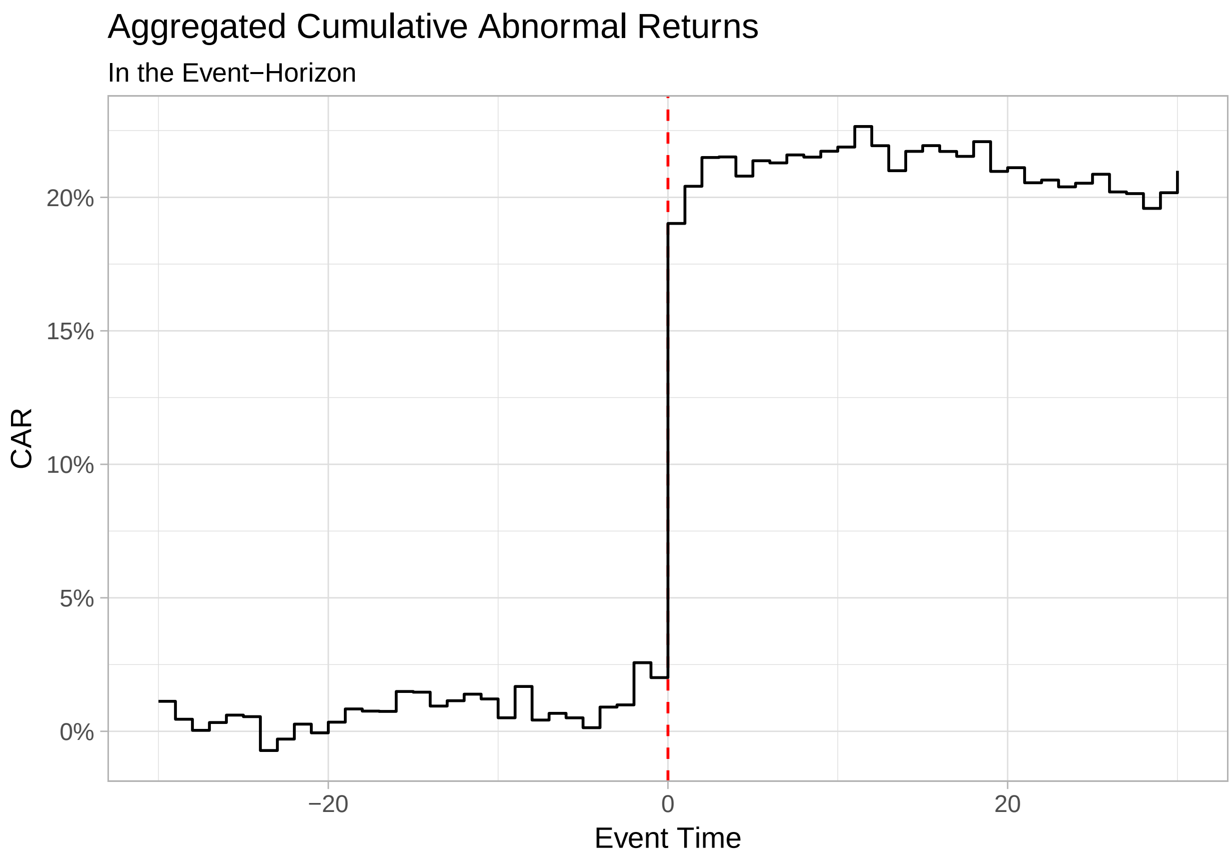
We can also calculate aggregated values (AAR as the average abnormal return) per day, which is handy, for example for ploting

```
# aggregated
agg_event <- event %>% group_by(event_time) %>% summarise(aar = mean(ar))
agg_event <- agg_event %>% mutate(car = cumsum(aar))
agg_event
```

```
## # A tibble: 61 × 3
##    event_time          aar           car
##         <dbl>        <dbl>         <dbl>
## 1         -30  0.011231117  0.0112311168
## 2         -29 -0.006714701  0.0045164158
## 3         -28 -0.004152784  0.0003636322
## 4         -27  0.002924491  0.0032881228
## 5         -26  0.002781145  0.0060692680
## 6         -25 -0.000583495  0.0054857730
## 7         -24 -0.012677222 -0.0071914490
## 8         -23  0.004285656 -0.0029057925
## 9         -22  0.005594058  0.0026882660
## 10        -21 -0.003264866 -0.0005766002
## # ... with 51 more rows
```

```
ggplot(agg_event, aes(x = event_time, y = car)) +
  geom_vline(xintercept = 0, color = "red", linetype = "dashed") +
  geom_step() +
  scale_y_continuous(labels = percent) +
```

```
  labs(title = "Aggregated Cumulative Abnormal Returns", subtitle = "In the Event-Horizon",
       x = "Event Time", y = "CAR")
```



4 Testing

To test for signifance, we mainly use t-test in this script, other tests include Boehmer et al. (1991) and Corrado (1989), among others.

The variance of the $CARs$, are known to be distributed with a variance of

$$\sigma_{i,t}^2 = \frac{1}{N(N-1)} \sum_{j=1}^{N} \left(CAR_{j,t} - \overline{CAR_{j,t}}\right)^2$$

4.1 T-test

The first chunk uses a t-test to test the individual ARs (the question we are trying to answer: Is the abnormal return in time-period t different from zero?).

```
test1 <- indiv_event %>%
  group_by(event_time) %>%
  summarise(mean_ar = mean(ar),
         var_ar = 1/(n()*(n() - 1)) * sum((ar - mean_ar)^2),
         t_value = mean_ar / sqrt(var_ar),
         p_value = pt(abs(t_value), df = n(), lower.tail = F)*2)
```

```
test1
```

```
## # A tibble: 61 × 5
##    event_time      mean_ar       var_ar    t_value    p_value
##         <dbl>        <dbl>        <dbl>      <dbl>      <dbl>
## 1         -30  0.011231117 7.949270e-05  1.2596773 0.25456700
## 2         -29 -0.006714701 1.112093e-05 -2.0135209 0.09071096
## 3         -28 -0.004152784 5.959287e-05 -0.5379503 0.60996980
## 4         -27  0.002924491 8.869356e-05  0.3105304 0.76666185
## 5         -26  0.002781145 7.400027e-06  1.0223674 0.34604683
## 6         -25 -0.000583495 3.485041e-05 -0.0988401 0.92448471
## 7         -24 -0.012677222 2.968817e-05 -2.3266571 0.05890964
## 8         -23  0.004285656 1.197490e-05  1.2384586 0.26180002
## 9         -22  0.005594058 1.828100e-05  1.3083591 0.23863386
## 10        -21 -0.003264866 1.180128e-04 -0.3005390 0.77390761
## # ... with 51 more rows
```

The following chunk uses CARs to see if the price-development (which is represented by the CARs) is different from zero, instead of a snapshot of a single day as we did in the example above.

```
# test2 with CARs
stars <- function(p) {
  ifelse(p < 0.001, "***",
         ifelse(p < 0.01, "**",
                ifelse(p < 0.05, "*", "")))
}

test2 <- indiv_event %>%
  group_by(event_time) %>%
  summarise(mean_car = mean(car),
         var_car = 1/(n()*(n() - 1)) * sum((car - mean_car)^2),
         t_value = mean_car / sqrt(var_car),
         p_value = pt(abs(t_value), df = n(), lower.tail = F)*2)

test2 %>% mutate(sign = stars(p_value)) %>%
  select(event_time, car = cumsum(mean_car), t_value, sign) %>%
  filter(event_time %in% -3:6) # look only at the frame [-3, 6], to have less output
```

```
## # A tibble: 10 × 4
##    event_time         car   t_value  sign
##         <dbl>       <dbl>     <dbl> <chr>
## 1          -3 0.009899399 0.1654311
## 2          -2 0.025715019 0.4437065
## 3          -1 0.020103760 0.3567753
## 4           0 0.190211690 4.0655770    **
## 5           1 0.204140265 4.5541234    **
## 6           2 0.214914837 4.5239831    **
## 7           3 0.215126659 4.3204857    **
## 8           4 0.207946197 3.9035773    **
## 9           5 0.213698238 3.9638917    **
## 10          6 0.212891828 3.8996172    **
```

4.2 Testing over Aggregated Times

In the next step we want to look not at a single time-point, but at aggregated times, in this example, we want to see if the price in the time-horizon $[-3, +3]$ is different from zero.

```
time_window <- c(-3, 3)
test3 <- indiv_event %>% filter(event_time >= time_window[1] &
                                  event_time <= time_window[2]) %>%
  select(company, ar) %>%
  group_by(company) %>% summarise(car = sum(ar))

# using the same logic as before
test3 %>% summarise(mean_car = mean(car),
                    var_car = 1/(n()*(n() - 1)) * sum((car - mean_car)^2),
                    t_value = mean_car / sqrt(var_car),
                    p_value = pt(abs(t_value), df = n(), lower.tail = F)*2,
                    sign = stars(p_value))
```

```
## # A tibble: 1 × 5
##    mean_car     var_car  t_value      p_value  sign
##       <dbl>       <dbl>    <dbl>        <dbl> <chr>
## 1 0.2060564 0.000893218 6.894572 0.0004597725   ***
```

So we can see, that we have detected highly significant returns in the time-period $[-3, +3]$. If we want to test multiple time-periods we can do it like this.

4.3 Multiple Time Windows

It may seem a bit more complicated, but we are essentially doing the same thing as before, but use a lapply-function to loop over the row-numbers and repeat the process.

```
time_windows <- data_frame(min = c(-1, 0, -1, -3),
                           max = c(0, 1, 1, 3))

list_events <- lapply(1:nrow(time_windows), function(i) {
  tmp <- indiv_event %>% filter(event_time >= time_windows$min[i] &
                           event_time <= time_windows$max[i]) %>%
    select(company, ar) %>%
    group_by(company) %>%
    summarise(car = sum(ar)) %>%
    summarise(mean_car = mean(car),
              var_car = 1/(n()*(n() - 1)) * sum((car - mean_car)^2),
              t_value = mean_car / sqrt(var_car),
              p_value = pt(abs(t_value), df = n(), lower.tail = F)*2,
              sign = stars(p_value)) %>%
    mutate(range = paste0("[", time_windows$min[i], ", ",
                          time_windows$max[i], "]"))
  return(tmp %>% select(range, car = mean_car, t_value, p_value, sign))
})
# lapply returns a list of data_frames, to bind them into a single df, we use
# do.call in combination with rbind.
mult_events <- do.call(rbind, list_events)
mult_events
```

```
## # A tibble: 4 × 5
```

```
##     range       car  t_value      p_value  sign
## *   <chr>     <dbl>    <dbl>        <dbl> <chr>
## 1 [-1, 0] 0.1644967 5.271110 0.0018812081    **
## 2  [0, 1] 0.1840365 5.849613 0.0011014852    **
## 3 [-1, 1] 0.1784252 5.062289 0.0023054626    **
## 4 [-3, 3] 0.2060564 6.894572 0.0004597725   ***
```