# R Event Study

*David Zimmermann*

## 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 or higher.

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(tidyverse) # for most of the data and visualisation functions
library(scales)    # scales for plotting
library(lubridate) # easy date handling
#############
# Alternatively, load the libraries individually
# 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

# used for visualisation
theme_set(theme_light())
```

## 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 <- ret_wide %>%
  pivot_longer(-date, names_to = "company", values_to = "ret")

market <- mar_wide %>%
  pivot_longer(-date, names_to = "country", values_to = "mret")

# 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 x 3
```

```
##    date       company          ret
##    <date>     <chr>          <dbl>
##  1 1997-01-02 Chrysler     0.0341
##  2 1997-01-02 BellSouth    0.0185
##  3 1997-01-02 Engelhard    0.00654
##  4 1997-01-02 Norsk Hydro -0.00481
##  5 1997-01-02 Pilkington  -0.0120
##  6 1997-01-02 INA          0.00787
##  7 1997-01-03 Chrysler     0.0146
##  8 1997-01-03 BellSouth   -0.00607
##  9 1997-01-03 Engelhard    0.0195
## 10 1997-01-03 Norsk Hydro  0.00489
## # ... with 26,594 more rows
```

```
market
```

```
## # A tibble: 17,736 x 3
##    date       country      mret
##    <date>     <chr>       <dbl>
##  1 1997-01-02 us      -0.00751
##  2 1997-01-02 norway  -0.00260
##  3 1997-01-02 uk      -0.0241
##  4 1997-01-02 italy   -0.00600
##  5 1997-01-03 us       0.0149
##  6 1997-01-03 norway   0.00511
##  7 1997-01-03 uk       0.00795
##  8 1997-01-03 italy   -0.00366
##  9 1997-01-06 us       0.000372
## 10 1997-01-06 norway   0.0106
## # ... with 17,726 more rows
```

```
events
```

```
## # A tibble: 6 x 2
##   company     event
##   <chr>       <date>
## 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

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

# 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 x 6
##    date       company          ret country     mret event
##    <date>     <chr>          <dbl> <chr>      <dbl> <date>
##  1 1997-01-02 Chrysler     0.0341  us       -0.00751 1998-05-06
##  2 1997-01-02 BellSouth    0.0185  us       -0.00751 2006-03-06
##  3 1997-01-02 Engelhard    0.00654 us       -0.00751 2006-01-03
##  4 1997-01-02 Norsk Hydro -0.00481 norway   -0.00260 2006-12-18
##  5 1997-01-02 Pilkington  -0.0120  uk       -0.0241  2005-10-31
##  6 1997-01-02 INA          0.00787 italy    -0.00600 1999-09-14
##  7 1997-01-03 Chrysler     0.0146  us        0.0149  1998-05-06
##  8 1997-01-03 BellSouth   -0.00607 us        0.0149  2006-03-06
##  9 1997-01-03 Engelhard    0.0195  us        0.0149  2006-01-03
## 10 1997-01-03 Norsk Hydro  0.00489 norway    0.00511 2006-12-18
## # ... 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
```

```
## # A tibble: 26,604 x 9
## # Groups:   company [6]
##    date       company      ret country     mret event      date_index
##    <date>     <chr>      <dbl> <chr>      <dbl> <date>          <int>
##  1 1997-01-02 Chrysl~  0.0341  us       -0.00751 1998-05-06          1
##  2 1997-01-02 BellSo~  0.0185  us       -0.00751 2006-03-06          1
##  3 1997-01-02 Engelh~  0.00654 us       -0.00751 2006-01-03          1
##  4 1997-01-02 Norsk ~ -0.00481 norway   -0.00260 2006-12-18          1
##  5 1997-01-02 Pilkin~ -0.0120  uk       -0.0241  2005-10-31          1
##  6 1997-01-02 INA      0.00787 italy    -0.00600 1999-09-14          1
##  7 1997-01-03 Chrysl~  0.0146  us        0.0149  1998-05-06          2
##  8 1997-01-03 BellSo~ -0.00607 us        0.0149  2006-03-06          2
##  9 1997-01-03 Engelh~  0.0195  us        0.0149  2006-01-03          2
## 10 1997-01-03 Norsk ~  0.00489 norway    0.00511 2006-12-18          2
## # ... with 26,594 more rows, and 2 more variables: 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],
```

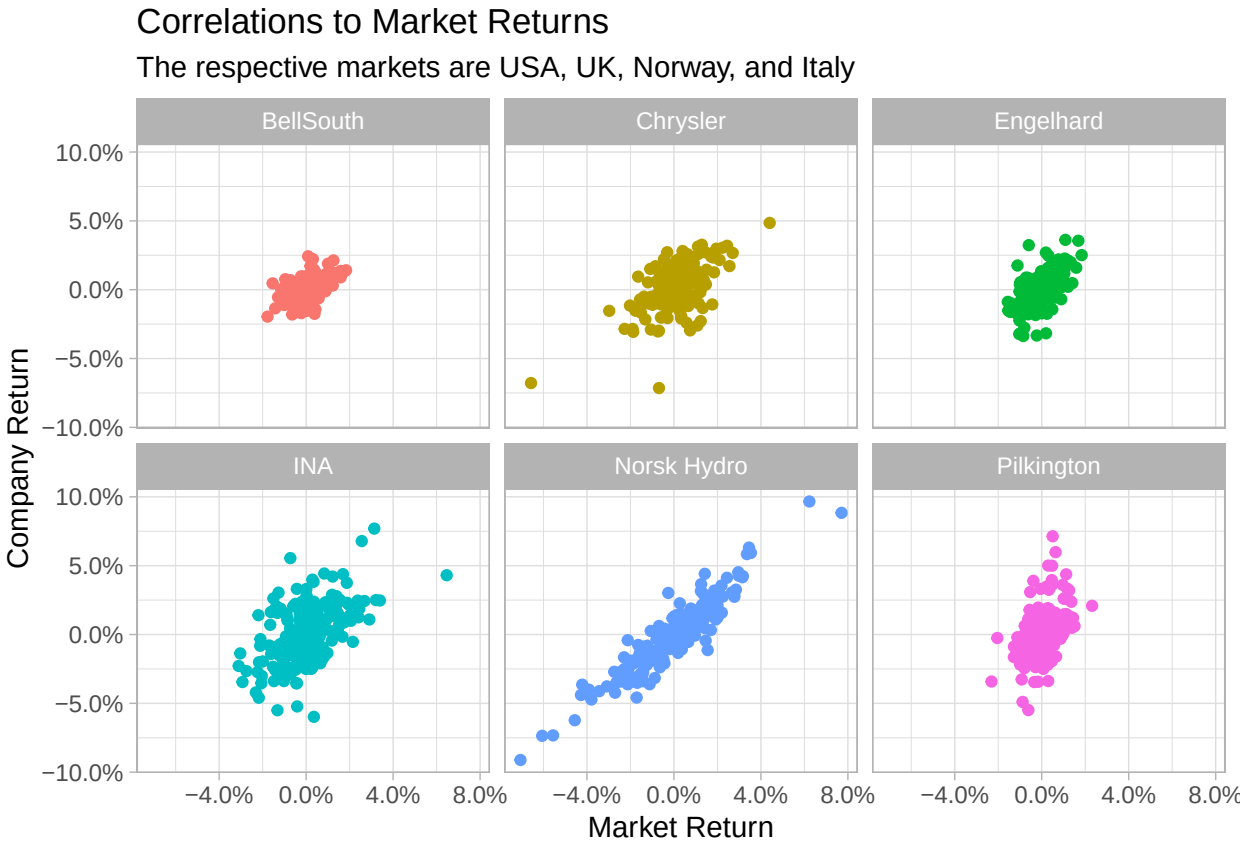
```
        event_time <= estimation_window[2])

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

# have a look at the data
estimation
```

```
## # A tibble: 1,200 x 9
## # Groups:   company [6]
##    date       company      ret country     mret event      date_index
##    <date>     <chr>      <dbl> <chr>      <dbl> <date>          <int>
##  1 1997-06-18 Chrysl~  0.00384 us      -4.62e-3 1998-05-06        120
##  2 1997-06-19 Chrysl~ -0.00382 us       9.95e-3 1998-05-06        121
##  3 1997-06-20 Chrysl~  0       us       2.97e-4 1998-05-06        122
##  4 1997-06-23 Chrysl~ -0.0115  us      -2.02e-2 1998-05-06        123
##  5 1997-06-24 Chrysl~  0.0234  us       1.80e-2 1998-05-06        124
##  6 1997-06-25 Chrysl~ -0.00381 us      -7.57e-3 1998-05-06        125
##  7 1997-06-26 Chrysl~  0.0114  us      -5.73e-3 1998-05-06        126
##  8 1997-06-27 Chrysl~ -0.0113  us       3.88e-3 1998-05-06        127
##  9 1997-06-30 Chrysl~  0.00382 us      -2.63e-3 1998-05-06        128
## 10 1997-07-01 Chrysl~ -0.00951 us       6.59e-3 1998-05-06        129
## # ... with 1,190 more rows, and 2 more variables: event_index <dbl>,
## #   event_time <dbl>
```

```
event
```

```
## # A tibble: 366 x 9
## # Groups:   company [6]
##    date       company      ret country     mret event      date_index
##    <date>     <chr>      <dbl> <chr>      <dbl> <date>          <int>
##  1 1998-03-25 Chrysl~ -0.0173  us      -2.22e-3 1998-05-06        320
##  2 1998-03-26 Chrysl~ -0.0132  us      -4.97e-4 1998-05-06        321
##  3 1998-03-27 Chrysl~  0.00149 us      -4.67e-3 1998-05-06        322
##  4 1998-03-30 Chrysl~ -0.00593 us      -1.51e-3 1998-05-06        323
##  5 1998-03-31 Chrysl~ -0.00598 us       7.87e-3 1998-05-06        324
##  6 1998-04-01 Chrysl~  0.00601 us       6.25e-3 1998-05-06        325
##  7 1998-04-02 Chrysl~ -0.0209  us       9.74e-3 1998-05-06        326
##  8 1998-04-03 Chrysl~ -0.00305 us       2.18e-3 1998-05-06        327
##  9 1998-04-06 Chrysl~  0.0214  us      -3.87e-3 1998-05-06        328
## 10 1998-04-07 Chrysl~  0.00300 us      -9.71e-3 1998-05-06        329
## # ... with 356 more rows, and 2 more variables: event_index <dbl>,
## #   event_time <dbl>
```

```
# Graph data
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

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

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

```
## # A tibble: 6 x 2
##   company           cmrm
##   <chr>            <dbl>
## 1 BellSouth    0.000395
## 2 Chrysler     0.00173
## 3 Engelhard   -0.0000261
## 4 INA          0.000112
## 5 Norsk Hydro  0.000419
## 6 Pilkington   0.00147
```

## 3.2 CAPM

To calculate the capm we can use the simplified market-model (estimates the intercept (risk-free rate) instead of imposing it) which uses a linear regression of the form $return \sim marketreturn$, using the `broom`-library we can do the following:

```
capm <- estimation %>%
  group_by(company) %>%
  # "do" a regression using do() from the broom-package (tidyverse)
  # see https://github.com/tidyverse/broom
  do(fit = lm(ret ~ mret, data = .)) %>%
  # get the coefficients: intercept and slope (alpha and beta)
  # and discard the model itself (fit)
  mutate(alpha = coefficients(fit)[1],
         beta = coefficients(fit)[2],
         fit = NULL)
capm
```

```
## Source: local data frame [6 x 3]
## Groups: <by row>
##
## # A tibble: 6 x 3
##   company           alpha  beta
##   <chr>             <dbl> <dbl>
## 1 BellSouth   -0.00000498 0.681
## 2 Chrysler     0.000699   0.851
## 3 Engelhard   -0.000335   1.06
## 4 INA         -0.000117   0.870
## 5 Norsk Hydro -0.000616   1.27
## 6 Pilkington   0.000906   1.22
```

```
event_capm <- left_join(event, capm, by = "company") %>%
  # compute the expected return
  mutate(capm = alpha + mret * beta,
         alpha = NULL,
         beta = NULL)
event_capm
```

```
## # A tibble: 366 x 10
## # Groups:   company [6]
##    date       company      ret country     mret event      date_index
##    <date>     <chr>      <dbl> <chr>      <dbl> <date>          <int>
##  1 1998-03-25 Chrysl~ -0.0173  us      -2.22e-3 1998-05-06        320
##  2 1998-03-26 Chrysl~ -0.0132  us      -4.97e-4 1998-05-06        321
##  3 1998-03-27 Chrysl~  0.00149 us      -4.67e-3 1998-05-06        322
##  4 1998-03-30 Chrysl~ -0.00593 us      -1.51e-3 1998-05-06        323
##  5 1998-03-31 Chrysl~ -0.00598 us       7.87e-3 1998-05-06        324
##  6 1998-04-01 Chrysl~  0.00601 us       6.25e-3 1998-05-06        325
##  7 1998-04-02 Chrysl~ -0.0209  us       9.74e-3 1998-05-06        326
##  8 1998-04-03 Chrysl~ -0.00305 us       2.18e-3 1998-05-06        327
##  9 1998-04-06 Chrysl~  0.0214  us      -3.87e-3 1998-05-06        328
## 10 1998-04-07 Chrysl~  0.00300 us      -9.71e-3 1998-05-06        329
## # ... with 356 more rows, and 3 more variables: event_index <dbl>,
## #   event_time <dbl>, capm <dbl>
```

Nonetheless, we will continue the tests using the CMRM and leave the testing of the CAPM to the interested

reader.

### 3.3 Merge Returns

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
```

```
## # A tibble: 366 x 4
## # Groups:   company [6]
##    company       ret event_time    cmrm
##    <chr>       <dbl>      <dbl>   <dbl>
##  1 Chrysler -0.0173         -30 0.00173
##  2 Chrysler -0.0132         -29 0.00173
##  3 Chrysler  0.00149        -28 0.00173
##  4 Chrysler -0.00593        -27 0.00173
##  5 Chrysler -0.00598        -26 0.00173
##  6 Chrysler  0.00601        -25 0.00173
##  7 Chrysler -0.0209         -24 0.00173
##  8 Chrysler -0.00305        -23 0.00173
##  9 Chrysler  0.0214         -22 0.00173
## 10 Chrysler  0.00300        -21 0.00173
## # ... with 356 more rows
```

### 3.4 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
```

```
## # A tibble: 366 x 5
## # Groups:   company [6]
##    company       ret event_time    cmrm        ar
##    <chr>       <dbl>      <dbl>   <dbl>     <dbl>
##  1 Chrysler -0.0173         -30 0.00173 -0.0190
##  2 Chrysler -0.0132         -29 0.00173 -0.0149
##  3 Chrysler  0.00149        -28 0.00173 -0.000245
##  4 Chrysler -0.00593        -27 0.00173 -0.00766
##  5 Chrysler -0.00598        -26 0.00173 -0.00771
##  6 Chrysler  0.00601        -25 0.00173  0.00428
##  7 Chrysler -0.0209         -24 0.00173 -0.0226
##  8 Chrysler -0.00305        -23 0.00173 -0.00478
##  9 Chrysler  0.0214         -22 0.00173  0.0197
## 10 Chrysler  0.00300        -21 0.00173  0.00126
## # ... with 356 more rows
```

### 3.5 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

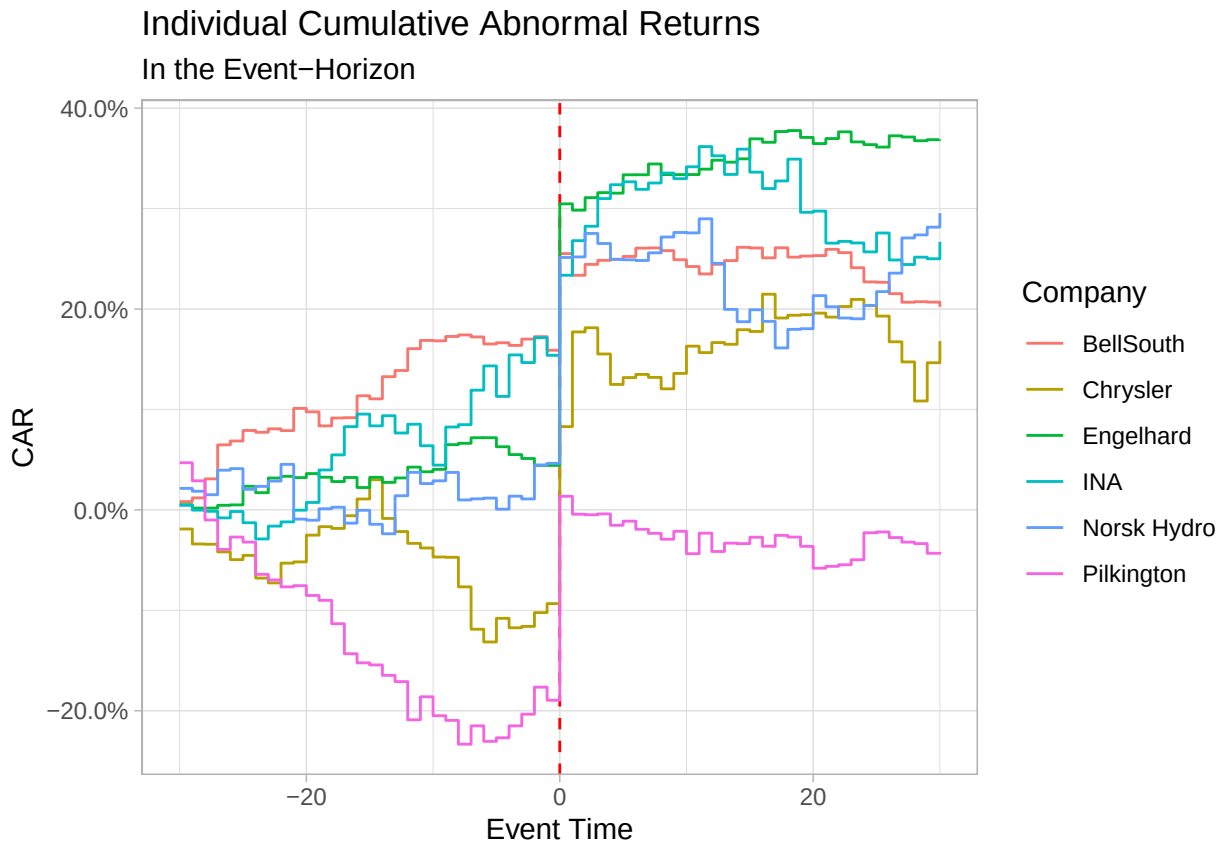
$$CAR_{i,t} \sim N(0, \sigma^2_{i,t})$$

In R we can calculate the CARs like this

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

```
## # A tibble: 366 x 6
## # Groups:   company [6]
##    company       ret event_time    cmrm        ar     car
##    <chr>       <dbl>      <dbl>   <dbl>     <dbl>   <dbl>
##  1 Chrysler -0.0173         -30 0.00173 -0.0190   -0.0190
##  2 Chrysler -0.0132         -29 0.00173 -0.0149   -0.0340
##  3 Chrysler  0.00149        -28 0.00173 -0.000245 -0.0342
##  4 Chrysler -0.00593        -27 0.00173 -0.00766  -0.0419
##  5 Chrysler -0.00598        -26 0.00173 -0.00771  -0.0496
##  6 Chrysler  0.00601        -25 0.00173  0.00428  -0.0453
##  7 Chrysler -0.0209         -24 0.00173 -0.0226   -0.0679
##  8 Chrysler -0.00305        -23 0.00173 -0.00478  -0.0727
##  9 Chrysler  0.0214         -22 0.00173  0.0197   -0.0530
## 10 Chrysler  0.00300        -21 0.00173  0.00126  -0.0518
## # ... 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")
```



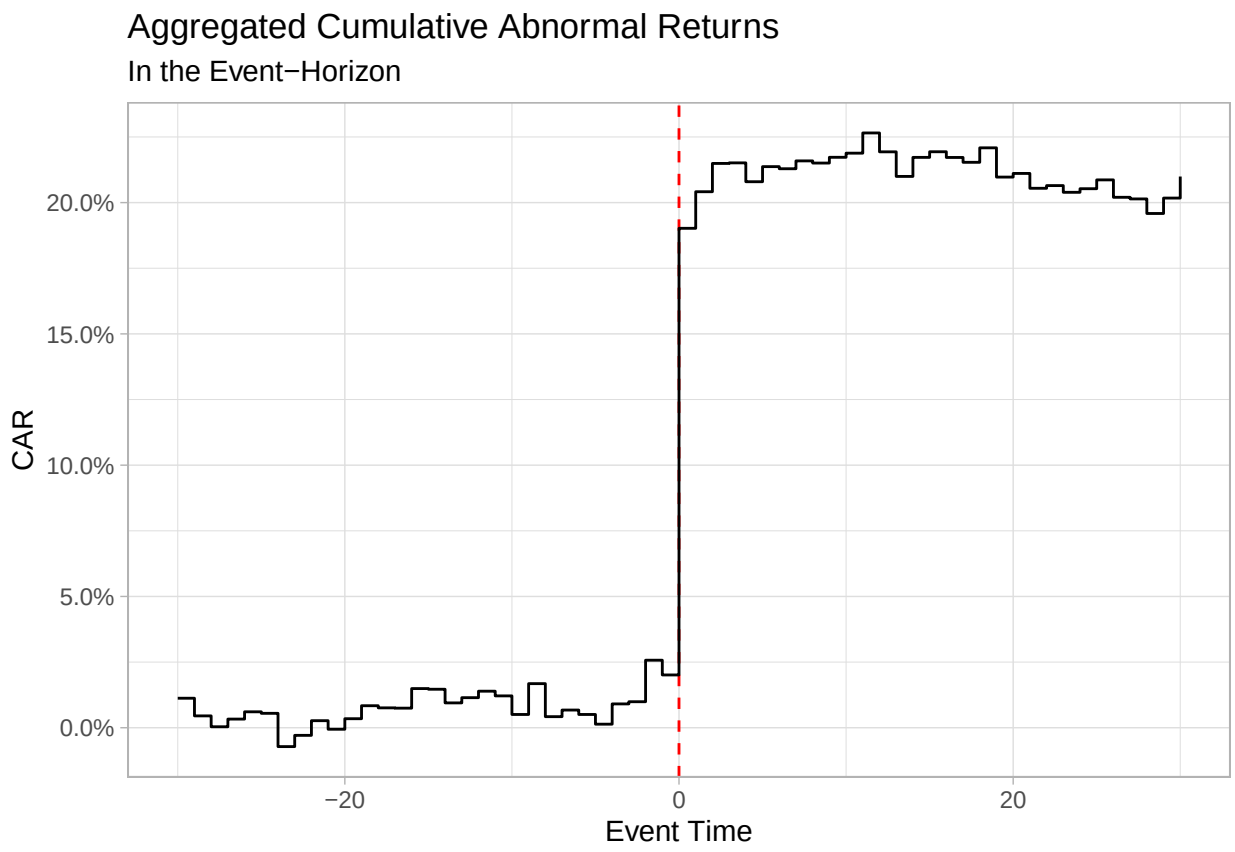

We can also calculate aggregated values ($AAR$ as the average abnormal return) per day, which is handy, for example for ploting

```
# 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 x 3
##    event_time       aar       car
##         <dbl>     <dbl>     <dbl>
##  1        -30  0.0112    0.0112
##  2        -29 -0.00671   0.00452
##  3        -28 -0.00415   0.000364
##  4        -27  0.00292   0.00329
##  5        -26  0.00278   0.00607
##  6        -25 -0.000583  0.00549
##  7        -24 -0.0127   -0.00719
##  8        -23  0.00429  -0.00291
##  9        -22  0.00559   0.00269
## 10        -21 -0.00326  -0.000577
## # ... 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

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

## 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 x 5
##    event_time   mean_ar     var_ar t_value p_value
##         <dbl>     <dbl>      <dbl>   <dbl>   <dbl>
##  1        -30  0.0112   0.0000795   1.26    0.255
##  2        -29 -0.00671  0.0000111  -2.01    0.0907
##  3        -28 -0.00415  0.0000596  -0.538   0.610
##  4        -27  0.00292  0.0000887   0.311   0.767
##  5        -26  0.00278  0.00000740  1.02    0.346
##  6        -25 -0.000583 0.0000349  -0.0988  0.924
##  7        -24 -0.0127   0.0000297  -2.33    0.0589
##  8        -23  0.00429  0.0000120   1.24    0.262
##  9        -22  0.00559  0.0000183   1.31    0.239
## 10        -21 -0.00326  0.000118   -0.301   0.774
## # ... 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),
                 car = cumsum(mean_car)) %>%
  select(event_time, car, t_value, sign) %>%
  filter(event_time %in% -3:6) # look only at the frame [-3, 6], to have less output
```

```
## # A tibble: 10 x 4
##    event_time   car t_value sign
##         <dbl> <dbl>   <dbl> <chr>
##  1         -3 0.184   0.165 " "
##  2         -2 0.210   0.444 " "
##  3         -1 0.230   0.357 " "
##  4          0 0.421   4.07  **
##  5          1 0.625   4.55  **
##  6          2 0.840   4.52  **
##  7          3 1.05    4.32  **
##  8          4 1.26    3.90  **
##  9          5 1.48    3.96  **
## 10          6 1.69    3.90  **
```

## 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 x 5
##   mean_car  var_car t_value  p_value sign
##      <dbl>    <dbl>   <dbl>    <dbl> <chr>
## 1    0.206 0.000893    6.89 0.000460 ***
```

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 <- tibble(min = c(-1, 0, -1, -3),
                       max = c(0, 1, 1, 3))

# map_dfr maps inputs to a function and returns a tibble with bind_rows
mult_events <- map_dfr(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))
})

mult_events
```

```
## # A tibble: 4 x 5
##   range     car t_value  p_value sign
##   <chr>   <dbl>   <dbl>    <dbl> <chr>
## 1 [-1, 0] 0.164    5.27 0.00188  **
## 2 [0, 1]  0.184    5.85 0.00110  **
## 3 [-1, 1] 0.178    5.06 0.00231  **
## 4 [-3, 3] 0.206    6.89 0.000460 ***
```