Project 2

Anton Stråhle

15 februari 2020

Exercise 1

```
library(knitr)
```

```
## Warning: package 'knitr' was built under R version 3.5.3
```

```
library(readr)
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 3.5.3
```

```
## -- Attaching packages ---------------------------------------------------------------------------
```

```
## v ggplot2 3.1.0     v purrr   0.2.5
## v tibble  2.1.3     v dplyr   0.7.8
## v tidyr   0.8.2     v stringr 1.3.1
## v ggplot2 3.1.0     v forcats 0.3.0
```

```
## Warning: package 'tibble' was built under R version 3.5.3
```

```
## -- Conflicts ------------------------------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
data <- read.delim("Projekt2_Grupp9.txt", sep = ";") %>%
  mutate(ClaimYear = floor((ClaimDay-1)/365) + 1,
         PaymentYear = floor((PaymentDay-1)/365) + 1)
```

```
## Warning: The `printer` argument is deprecated as of rlang 0.3.0.
## This warning is displayed once per session.
```

```
ct <- data %>% filter(ClaimYear >= 11) %>%
  group_by(ClaimYear, PaymentYear) %>%
  summarize(Total = sum(ClaimCost)) %>%
  mutate(DevelopmentYear = PaymentYear-ClaimYear + 1) %>%
  select(-PaymentYear) %>%
  mutate(Total = cumsum(Total)) %>%
  spread(value = Total, key = DevelopmentYear) %>%
  ungroup() %>%
  select(-ClaimYear)

#Estimates the incremental factors f based on the observed triangle (not trapezoid!)

estimateF <- function(ct){

  n <- nrow(ct)

  fVec <- c()

  for(i in 1:(n-1)){
```

```
    f <- sum(ct[1:(n-i),i+1])/sum(ct[1:(n-i),i])

    fVec[i] <- f

  }

  fVec

}


fillct <- function(ct){

  n <- nrow(ct)

  fHat <- estimateF(ct)

  for(i in 2:n){

    for(j in (n-i+2):n){

    ct[i,j] <- ct[i,j-1]*fHat[j-1]

    }

  }

    ct

}


fullct <- fillct(ct)

fHat <- estimateF(ct)


kable(fullct, caption = "Full claims triangle predicted with CL", row.names = c(1:10))
```

```
## Warning in if (is.na(row.names)) row.names = has_rownames(x): the condition
## has length > 1 and only the first element will be used
```

```
## Warning in if (row.names) {: the condition has length > 1 and only the
## first element will be used
```

Table 1: Full claims triangle predicted with CL

	1	2	3	4	5	6	7	8	9	10
1	17382857	32064142	40583164	45163613	47828744	48961319	49629613	49764835	49967431	50125945
2	11940035	25909330	33875305	38571174	41264712	42394449	43047834	43527148	43618976	43757351
3	9570218	23414758	32580043	38193737	40325432	41673704	42146307	42605671	42740132	42875718
4	8312376	18695679	24490118	27902308	29520382	30686881	31162979	31411199	31510331	31610292
5	14023889	30863800	41644000	47146057	50657766	52457015	53184477	53608102	53777286	53947886
6	10893434	23539789	31331880	35431428	37881787	39070369	39612187	39927707	40053716	40180781

	1	2	3	4	5	6	7	8	9	10
7	7998715	21327320	29327038	33897863	36095968	37228518	37744794	38045440	38165509	38286583
8	10466529	23251932	30887979	35177716	37458813	38634123	39169892	39481889	39606491	39732137
9	12161281	27441687	36492033	41560063	44255023	45643572	46276547	46645149	46792359	46940800
10	10766067	23733536	31560923	35944119	38274913	39475830	40023271	40342065	40469382	40597765

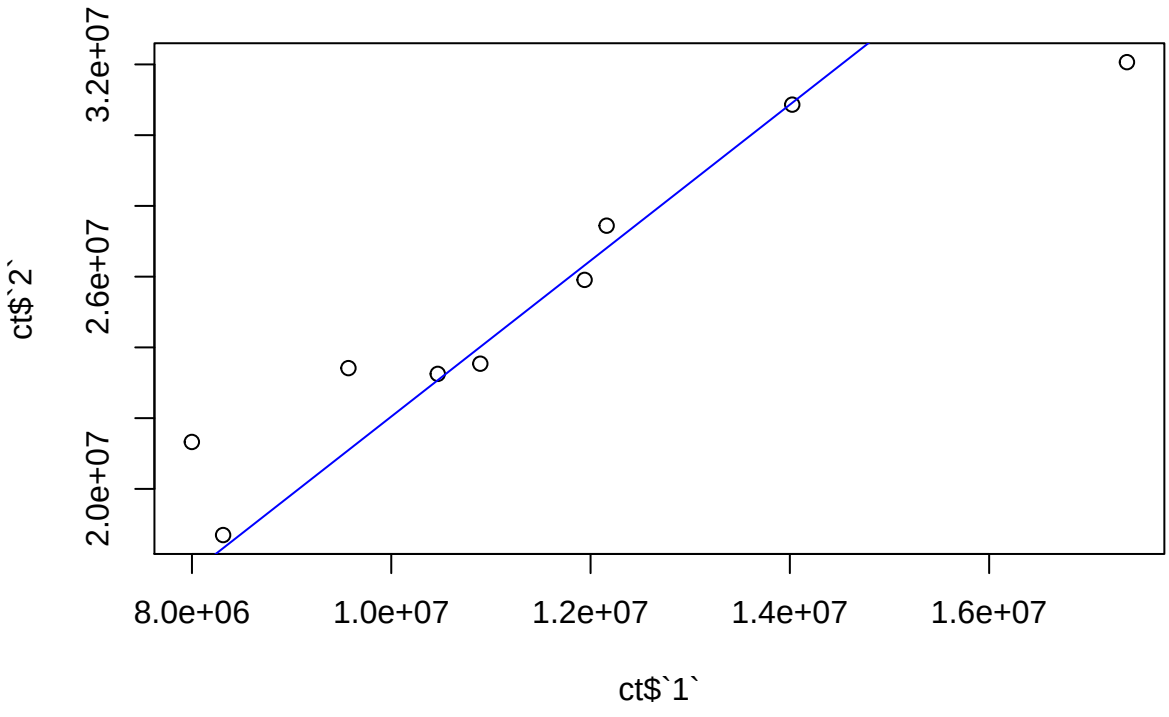
Exercise 2

We now want to check whether or not Mack's undrlying assumptions are met in our case. The assumptions are as follows.

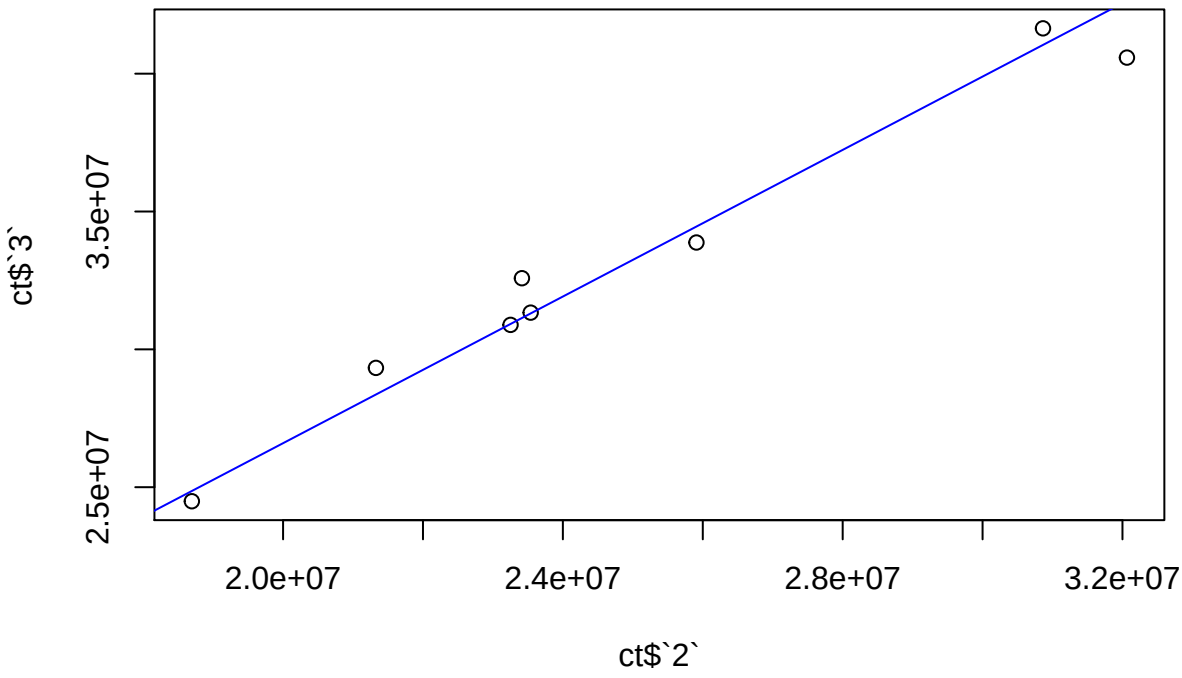
1. $E[C_{i,k+1}|C_{i,1},...,Ci,k] = f_k C_{i,k}$
2. Independent accident years
3. $Var(C_{i,k+1}|C_{i,1},...,Ci,k) = \sigma_k^2 C_{i,k}$

We begin by examing whether or not the we have an approximate linear relationship between $C_{i,k}$ and $C_{i,k+1}$ for $i = 1,...,10$

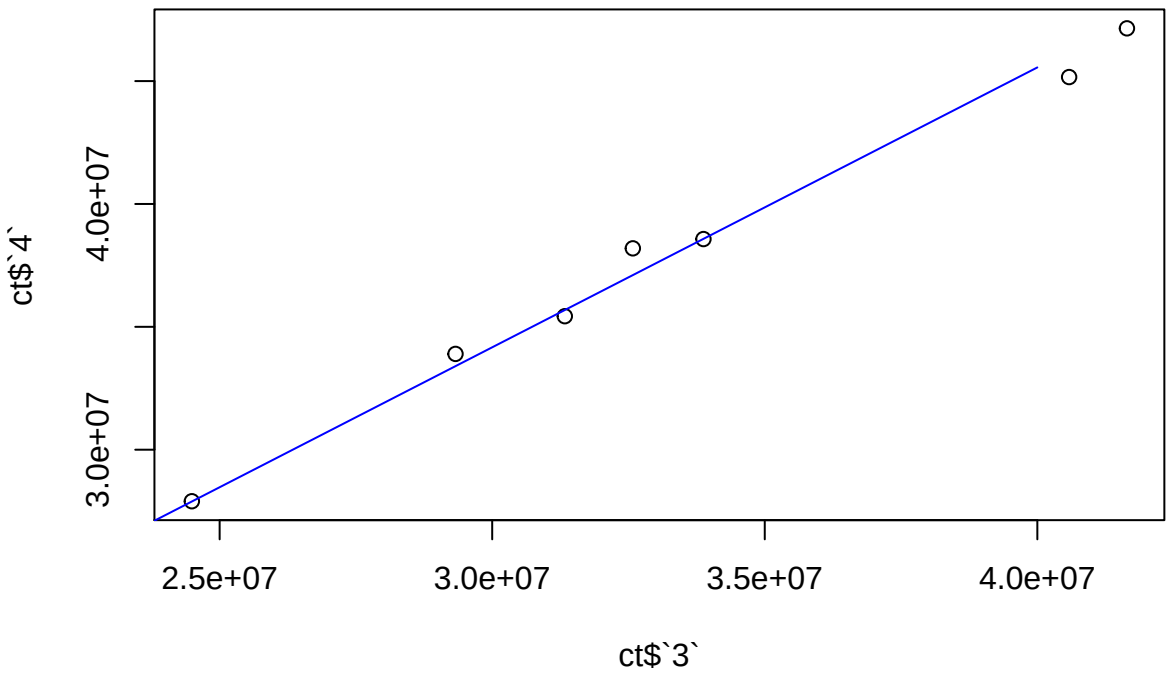
```
#Gör om gör fint!

plot(ct$`1`, ct$`2`)
lines(seq(5e6,2e7, by = 10000), sapply(X = seq(5e6,2e7, by = 10000), FUN = function(x){ x*fHat[1]}), co
```



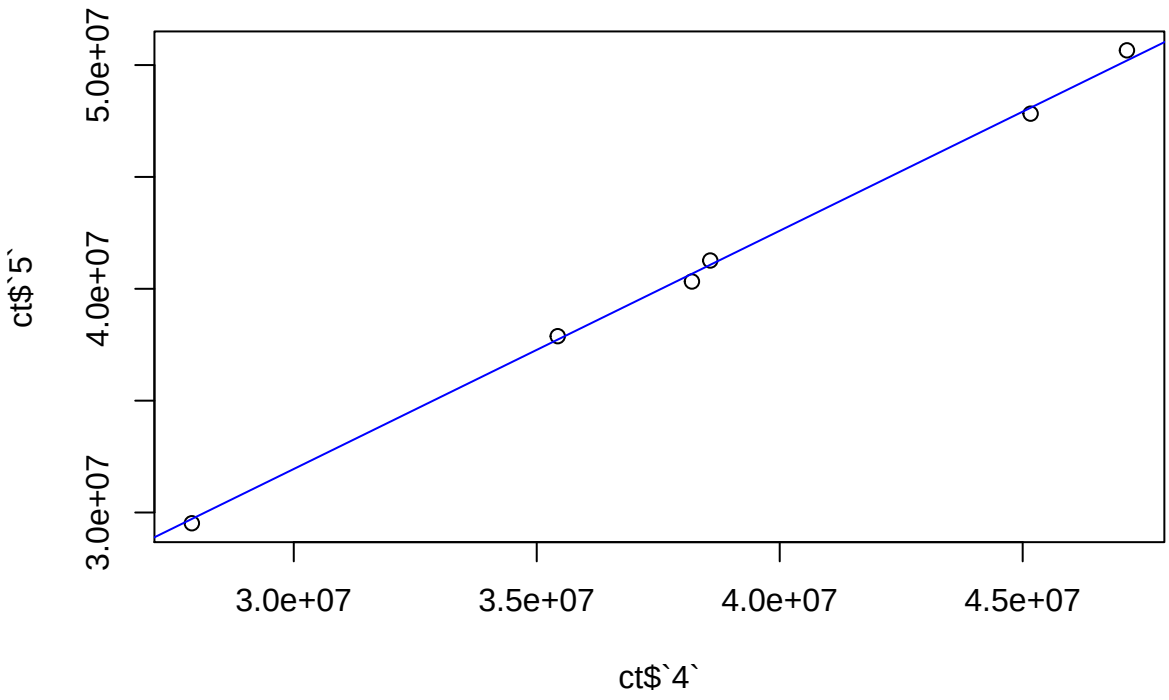

```
plot(ct$`2`, ct$`3`)
lines(seq(1e7,4e7, by = 10000), sapply(X = seq(1e7,4e7, by = 10000), FUN = function(x){ x*fHat[2]}), co
```



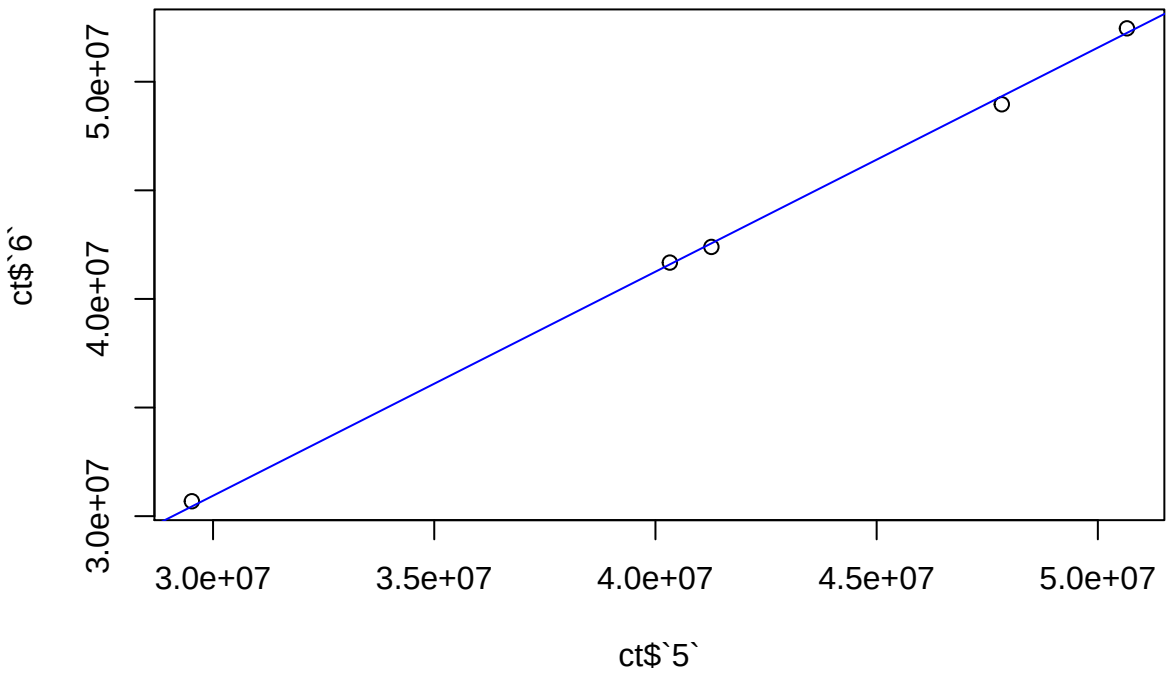

```
plot(ct$`3`, ct$`4`)
lines(seq(1e7,4e7, by = 10000), sapply(X = seq(1e7,4e7, by = 10000), FUN = function(x){ x*fHat[3]}), col
```



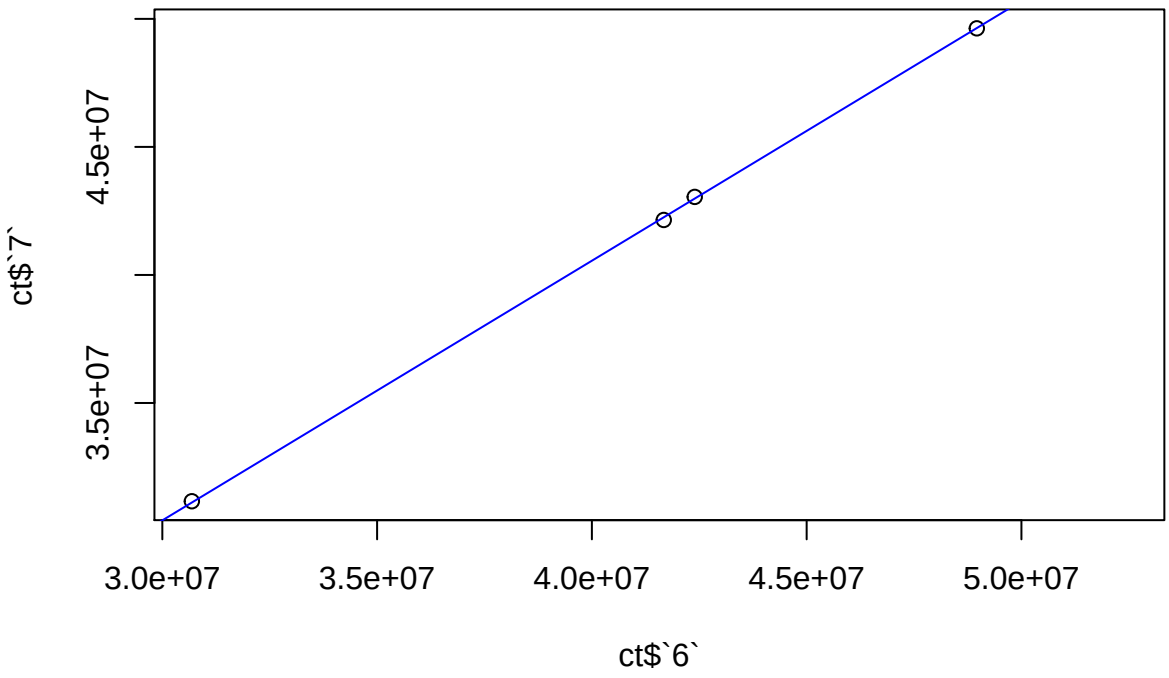

```
plot(ct$`4`, ct$`5`)
lines(seq(1e7,5e7, by = 10000), sapply(X = seq(1e7,5e7, by = 10000), FUN = function(x){ x*fHat[4]}), col
```



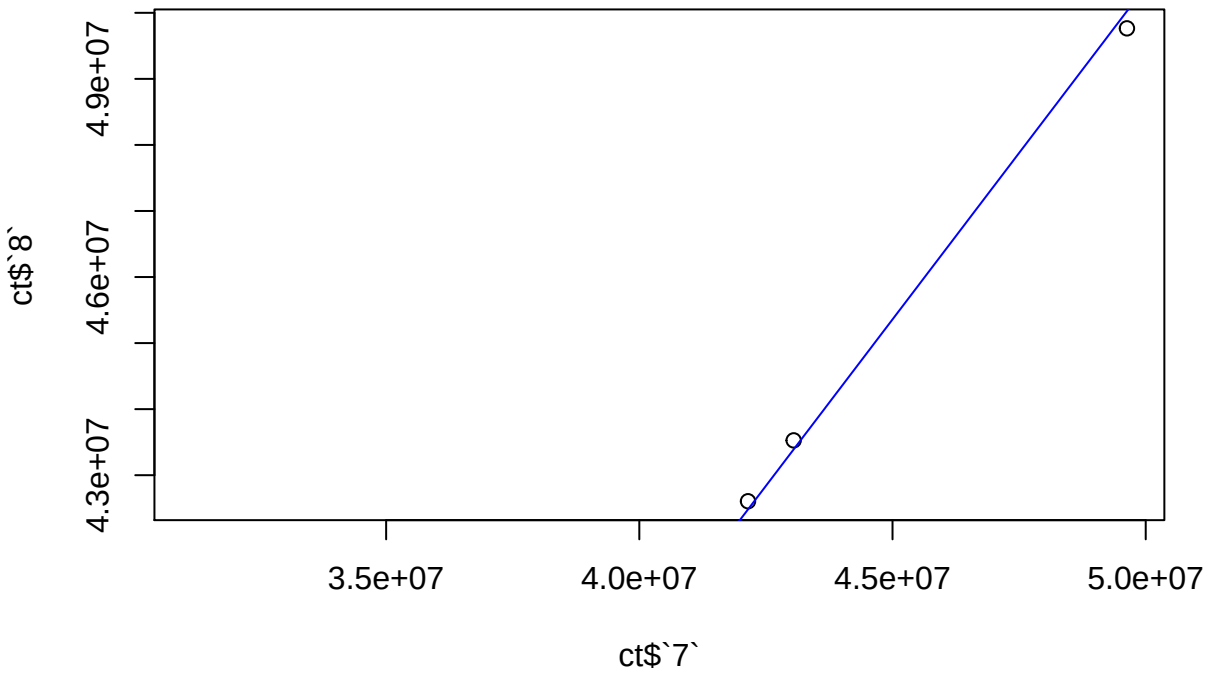

```
plot(ct$`5`, ct$`6`)
lines(seq(1e7,6e7, by = 10000), sapply(X = seq(1e7,6e7, by = 10000), FUN = function(x){ x*fHat[5]}), col
```



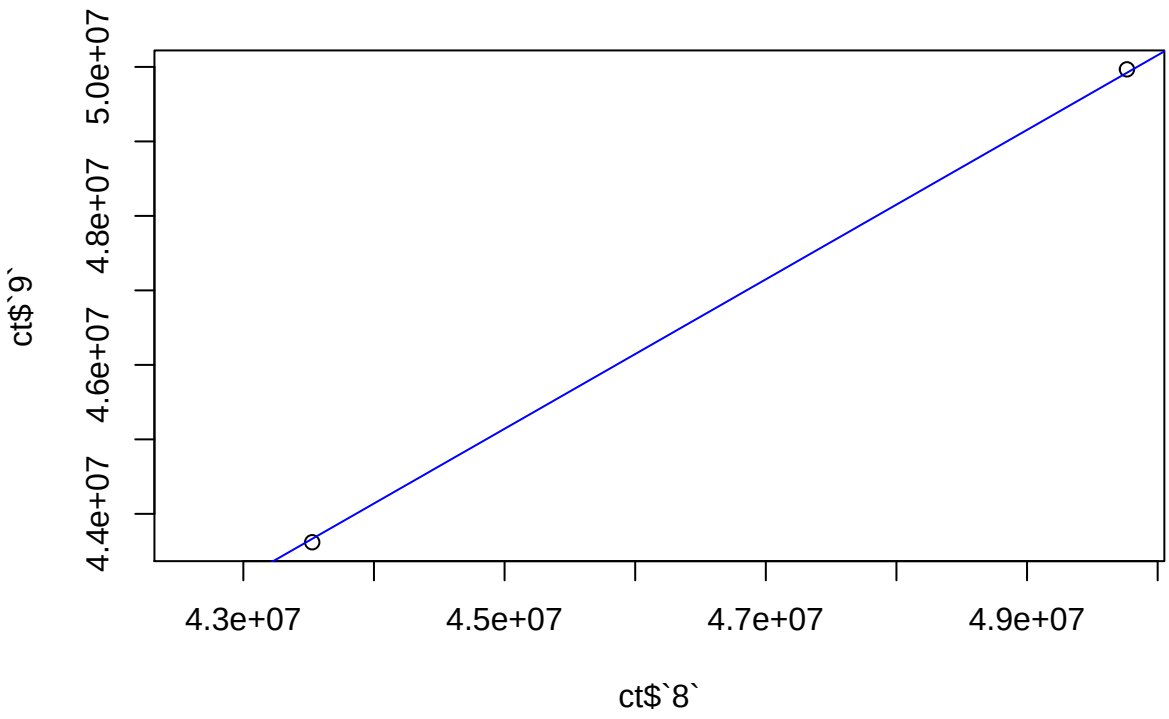

```
plot(ct$`6`, ct$`7`)
lines(seq(1e7,6e7, by = 10000), sapply(X = seq(1e7,6e7, by = 10000), FUN = function(x){ x*fHat[6]}), col
```




```
plot(ct$`7`, ct$`8`)
lines(seq(1e7,6e7, by = 10000), sapply(X = seq(1e7,6e7, by = 10000), FUN = function(x){ x*fHat[7]}), col
```




```
plot(ct$`8`, ct$`9`)
lines(seq(1e7,6e7, by = 10000), sapply(X = seq(1e7,6e7, by = 10000), FUN = function(x){ x*fHat[8]}), col
```

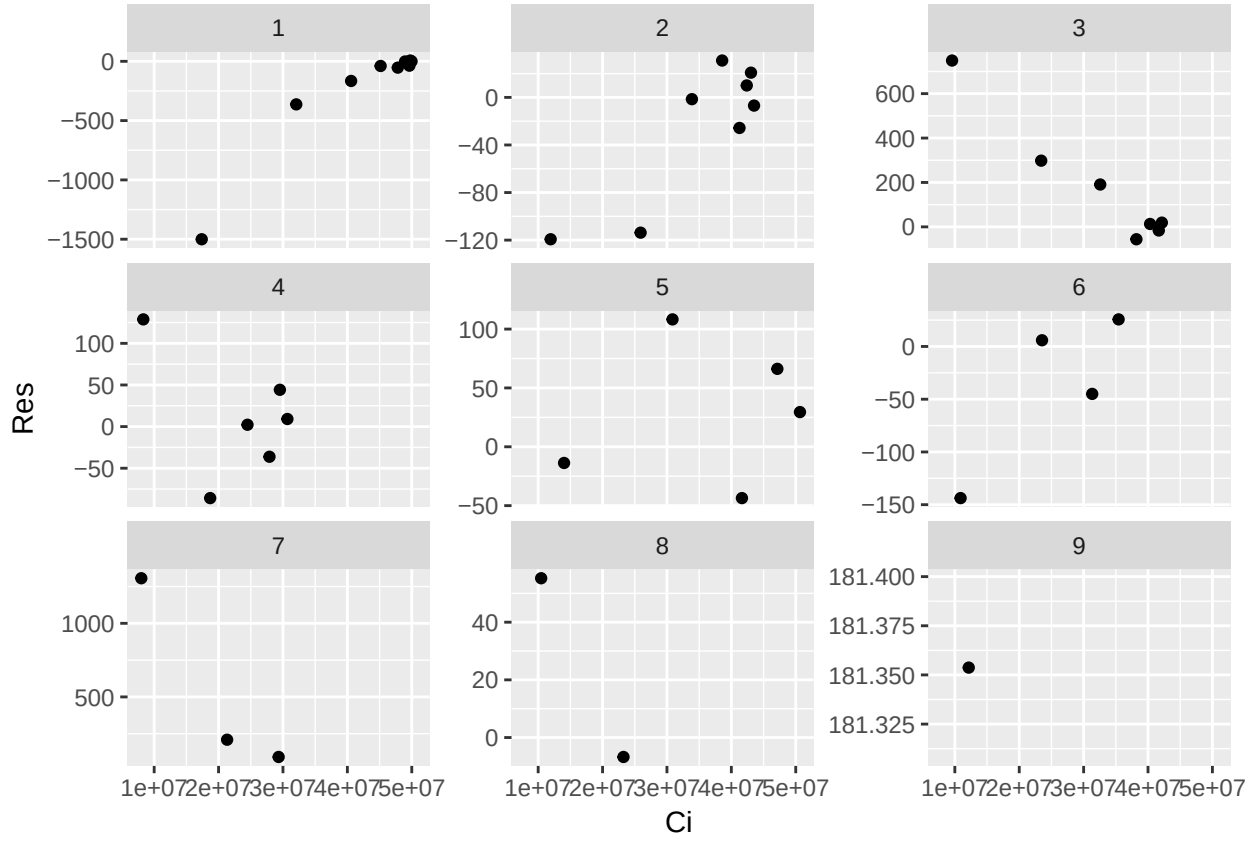



```
#Skoja dem blev faktiskt inte så fula
```

We note that the linear approximation seems to hold. We now want to plot the weighted residuals.

```
residualHelper <- function(ct){

  fHat <- estimateF(ct)

  n <- nrow(ct)

  df <- data.frame(matrix(ncol = 3, nrow = 0))

  for(i in 1:(n-1)){

    for(j in 2:(n-i+1)){

      res <- (ct[i,j] - ct[i,j-1]*fHat[j-1])/sqrt(ct[i,j-1])

      df <- rbind(df, unlist(c(i,unname(res), ct[i,j-1])))

    }

  }

  df

}
```

```
resFrame <- residualHelper(ct) %>%
  setNames(c("Year", "Res", "Ci"))

ggplot(resFrame, aes(x = Ci, y = Res)) +
  geom_point() +
  facet_wrap(~Year, scales = "free_y")
```




```
#Notera att vi bara beaktar år med fler än 6 punkter enl Mack!
#OBS enbart använt vanliga f. Kan vara bra att kolla resterande
```

All years with more than 6 points, as suggested by Mack, seem to showcase random behaviour and no signs of any systematic deviations.

Lastly we want to examine whether or not we have any calender year effects. An example of a scenario where the assumption of independent years might be violated is if there is an overhaul of the way claims are handled.

```
n <- nrow(ct)

fList <- list()

for(k in 1:(n-1)){

  fk <- c()

  for(i in 1:(n-k)){

    temp <- unlist(ct[i, k+1])/unlist(ct[i,k])
```

```
    names(temp) <- k+i-1 #år

    fk <- c(fk, temp)

  }

  fList[[k]] <- sort(fk)

}

fList
```

```
## [[1]]
##        1        6        2        5        8        4        9        3
## 1.844584 2.160915 2.169954 2.200802 2.221551 2.249138 2.256480 2.446627
##        7
## 2.666343
##
## [[2]]
##        2        3        5        9        7        6        8        4
## 1.265687 1.307456 1.309935 1.328405 1.331018 1.349283 1.375093 1.391432
##
## [[3]]
##        3        8        7        4        6        9        5
## 1.112866 1.130843 1.132121 1.138622 1.139329 1.155857 1.172305
##
## [[4]]
##        6        7        4        9        5        8
## 1.055813 1.057991 1.059011 1.069158 1.069833 1.074486
##
## [[5]]
##        5        6        7        9        8
## 1.023680 1.027378 1.033435 1.035518 1.039515
##
## [[6]]
##        8        6        7        9
## 1.011341 1.013649 1.015412 1.015515
##
## [[7]]
##        7        9        8
## 1.002725 1.010899 1.011134
##
## [[8]]
##        9        8
## 1.002110 1.004071
##
## [[9]]
##        9
## 1.003172
```

```
#Detta blev ju väldigt fult men det får fram budskapet
#Gör en tabell där man räknar kardinaleteten av
#sFk och lFk för alla k
```

We might have some seasonal dependence idk??

Exercise 3

From the paper by Mack we have that

$$Var(C_{i,I}) = C_{i,I}^2 \sum_{k=I+1-i}^{I-1} \frac{\sigma_k^2}{f_k^2} \left(\frac{1}{C_{i,k}} + \frac{1}{\sum_{j=1}^{I-k} C_{j,k}} \right)$$

where we estimate σ_k^2 by

$$\hat{\sigma}_k^2 = \frac{1}{I-k-1} \sum_{j=1}^{I-k} C_{j,k} \left(\frac{C_{j,k+1}}{C_{j,k} - \hat{f}_k} \right)^2$$

which is an unbiased estimator.

```
#WE now need the last 11 years

data2 <- rbind(data, c(1, 3735, 7021, 0, 10, 19)) #det är dumt men det gör det MYCKET lättare då den få

ct11 <- data2 %>% filter(ClaimYear >= 10) %>%
  group_by(ClaimYear, PaymentYear) %>%
  summarize(Total = sum(ClaimCost)) %>%
  mutate(DevelopmentYear = PaymentYear-ClaimYear + 1) %>%
  select(-PaymentYear) %>%
  mutate(Total = cumsum(Total)) %>%
  spread(value = Total, key = DevelopmentYear) %>%
  ungroup() %>%
  select(-ClaimYear)


estimateSigma <- function(ct){

  n <- nrow(ct)

  C <- fillct(ct)

  fHat <- estimateF(ct)

  sigmaHat <- c()

  for(k in 1:(n-2)){

    sum <- 0

    for(j in 1:(n-k)){

      sum = sum + C[j,k]*(C[j,k+1]/(C[j,k]) - fHat[k])^2

    }

    sigmaHat = c(sigmaHat, unlist(sum/(n-k-1)))
```

```
  }

  sigmaHat

}

sigmaHat <- estimateSigma(ct11) #estimates sigma^2


riskCalculation <- function(ct, ct11){

  n <- nrow(ct)

  C <- fillct(ct)

  fHat <- estimateF(ct)

  sigmaHat <- estimateSigma(ct11) #estimates sigma^2

  risks <- c()

  for(i in 2:n){

    sum <- 0

    for(k in (n+1-i):(n-1)){

      sum = sum + sigmaHat[k]/fHat[k]^2*(1/C[i,k] + 1/sum(C[1:(n-k),k]))

    }

    risks <- c(risks, unlist(sum*C[i,n]^2))

  }

  risks

}

risks <- cbind(2:10,unname(riskCalculation(ct, ct11)), fullct[2:10,10]) %>%
  data.frame() %>%
  setNames(c("Year", "Reserve Risk", "Reserve"))

kable(risks, caption = "Reserve risk for coming years")
```

Table 2: Reserve risk for coming years

Year	Reserve Risk	Reserve
2	1.931057e+10	43757351
3	2.347830e+10	42875718
4	5.400392e+10	31610292
5	1.210298e+11	53947886
6	1.874059e+11	40180781

Year	Reserve Risk	Reserve
7	2.740393e+11	38286583
8	7.776764e+11	39732137
9	3.381822e+12	46940800
10	2.069653e+13	40597765

```
#Blankrader för att lösa aggregeringsproblems

data3 <- data %>%
  filter(ClaimYear <= 10) %>%
  rbind(c(1,1,1,0,1,10)) %>%
  rbind(c(1,1,1,0,10,19))

#Not vi har skador med utvecklingsår 1+ vilket vi ej ska ha?? Behöver fråga.

ct <- data3 %>%
  group_by(ClaimYear, PaymentYear) %>%
  summarize(Total = sum(ClaimCost)) %>%
  mutate(DevelopmentYear = PaymentYear-ClaimYear + 1) %>%
  select(-PaymentYear) %>%
  mutate(Total = cumsum(Total)) %>%
  spread(value = Total, key = DevelopmentYear) %>%
  ungroup() %>%
  select(-ClaimYear)

estimateFAlt <- function(ct, known){

  n <- nrow(ct)

  fVec <- c()

  for(i in 1:(n-1)){

    f <- sum(ct[1:(n-i),i+1])/sum(ct[1:(n-i),i])

    fVec[i] <- f

  }

  fVec

}


fillAlt <- function(){




}
```