Homework 4: Diffusion of Tetracycline

We continue examining the diffusion of tetracycline among doctors in Illinois in the early 1950s, building on our work in lab 6. You will need the data sets `ckm_nodes.csv` and `ckm_network.dat` from the labs.

1. Clean the data to eliminate doctors for whom we have no adoption-date information, as in the labs. Only use this cleaned data in the rest of the assignment.

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.2.1     v purrr   0.3.4
## v tibble  3.0.1     v dplyr   0.8.3
## v tidyr   1.1.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
```

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

```
setwd("D:/github/Rlab/")
ckm_nodes<-read.csv("data/ckm_nodes.csv",header=T)
ckm_network<-read.table("data/ckm_network.dat")

ckm_network = ckm_network[which(!is.na(ckm_nodes$adoption_date))
                          ,which(!is.na(ckm_nodes$adoption_date))]
ckm_nodes <- ckm_nodes %>%
    filter(!is.na(adoption_date))
```

2. Create a new data frame which records, for every doctor, for every month, whether that doctor began prescribing tetracycline that month, whether they had adopted tetracycline before that month, the number of their contacts who began prescribing strictly *before* that month, and the number of their contacts who began prescribing in that month or earlier. Explain why the dataframe should have 6 columns, and 2125 rows. Try not to use any loops.

```
new = data.frame(matrix(rep(c(1:125),17),nrow=125))
new = stack(new)
names(new)<-c("doc","adoption_date")
new$adoption_date = substr(new$adoption_date,2,3)
new$adoption_date[which(new$adoption_date==17)] = Inf

adoption_date_net = data.frame(t(matrix(rep(ckm_nodes$adoption_date,125),nrow=125)))
compare_small = adoption_date_net<t(adoption_date_net)
compare_notbig = adoption_date_net<=t(adoption_date_net)

compare_small_net = compare_small&ckm_network
compare_notbig_net = compare_notbig&ckm_network

small = rowSums(as.matrix(compare_small_net))
notbig = rowSums(as.matrix(compare_notbig_net))
```

```
new <- new %>% mutate(if_start= ifelse(ckm_nodes$adoption_date[doc]==adoption_date,TRUE,FALSE))%>%
    mutate(if_before = ifelse(ckm_nodes$adoption_date[doc]<adoption_date,TRUE,FALSE))%>%
    mutate(connect_before = rep(small,17))%>%
    mutate(connect_not_after = rep(notbig,17))
```

3. Let

   p_k = Pr(A doctor starts prescribing tetracycline this month | Number of doctor's contacts prescribing before this mont

   and

   q_k = Pr(A doctor starts prescribing tetracycline this month | Number of doctor's contacts prescribing this month = k)

   We suppose that p_k and q_k are the same for all months.
   a. Explain why there should be no more than 21 values of k for which we can estimate p_k and q_k directly from the data.

```
max(rowSums(ckm_network))
```

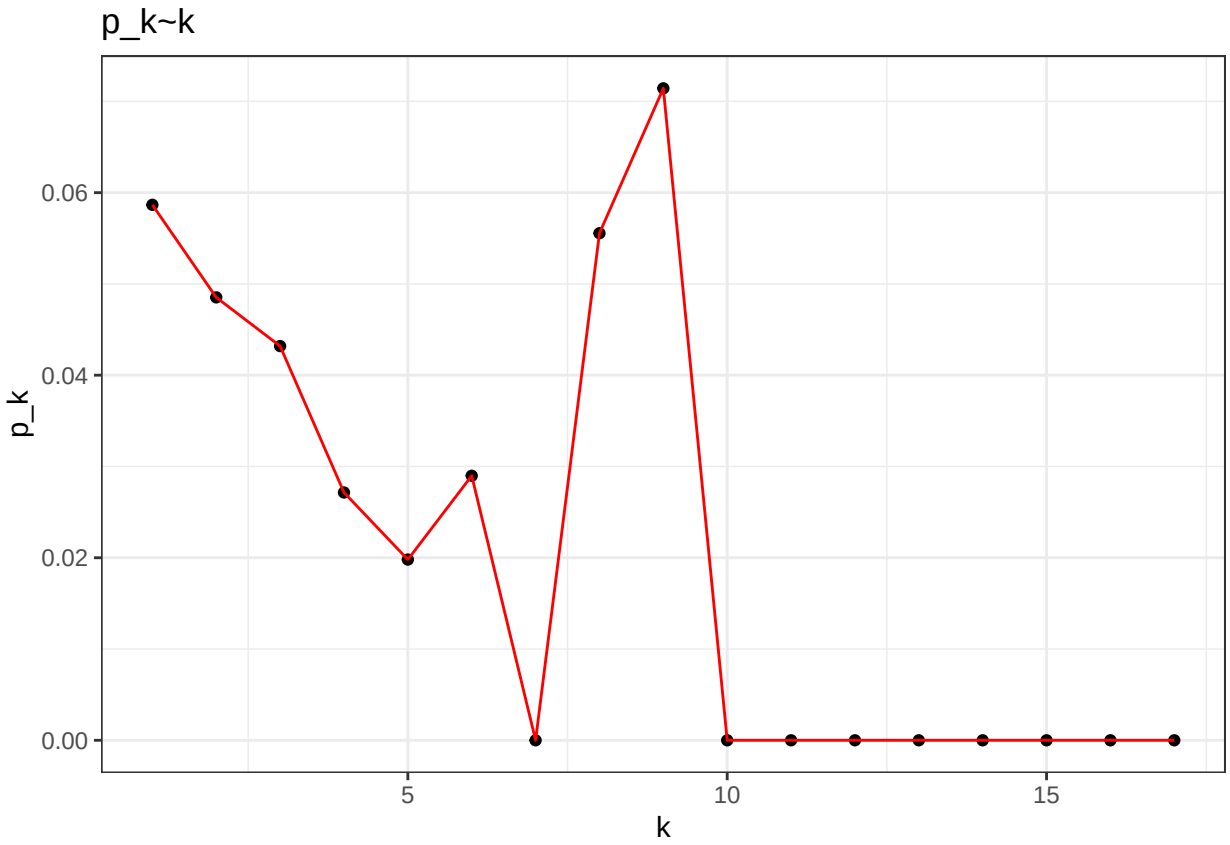
```
## [1] 20
```

explain: because a doctor has at most 20 connections, thus k must no more than 20, there are 21 choices.

   b. Create a vector of estimated p_k probabilities, using the data frame from (2). Plot the probabilities against the number of prior-adoptee contacts k.

```
doctor_month_before = matrix(nrow = 125,ncol = 17)
for(i in 1:125){
    for(j in 1:17){
        doctor_month_before[i,j] =
          length(which(ckm_nodes$adoption_date[which(ckm_network[i,]==1)]<j))
    }
}
p_pro = rep(0,max(doctor_month_before)+1)
doc_per_situation = rep(0,max(doctor_month_before)+1)
for(i in 0:max(doctor_month_before)){
    all = length(which(doctor_month_before==i))#all the cases
    #when a doctor have i conections have take pills
    num_start = 0# all the cases of a doctor starts take pills
    #when i connections started to take pills before.
    for(j in 1:125){
        if(ckm_nodes$adoption_date[j]!=Inf &
           doctor_month_before[j,ckm_nodes$adoption_date[j]]==i){
            num_start = num_start + 1
            doc_per_situation[i] = doc_per_situation[i] + 1
        }
    }
    p_pro[i] = num_start/all
}
p_pro = na.omit(p_pro)

ggplot(data = data.frame(p_pro)) +
  geom_point(aes(x = 1:length(p_pro), y = p_pro)) +
  geom_line(aes(x = 1:length(p_pro), y = p_pro),colour='red') +
  labs(title = "p_k~k",   y = "p_k",   x = "k") +
  theme_bw()
```

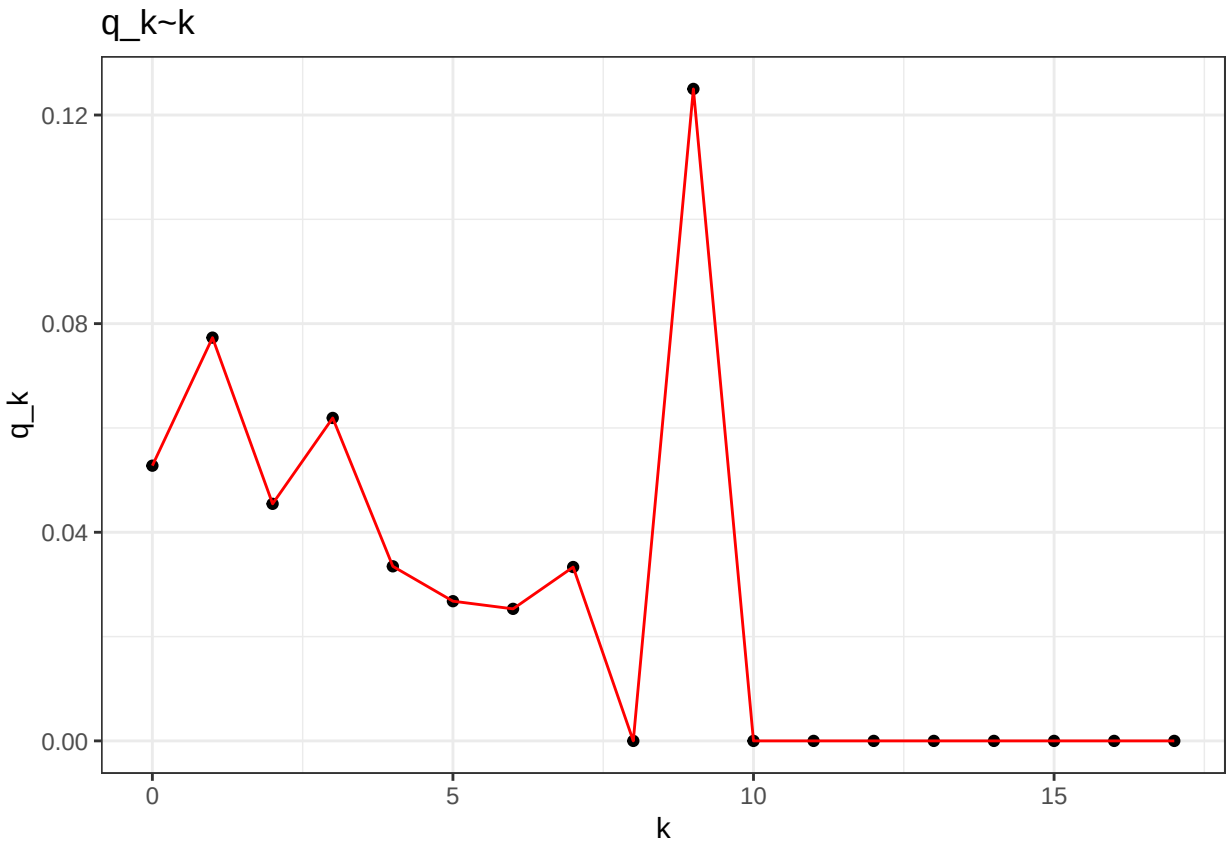


c. Create a vector of estimated q_k probabilities, using the data frame from (2). Plot the probabili

```
doctor_month_not_after = matrix(nrow = 125,ncol = 17)
for(i in 1:125){
    for(j in 1:17){
        doctor_month_not_after[i,j] =
          length(which(ckm_nodes$adoption_date[which(ckm_network[i,]==1)]<=j))
    }
}

q_pro = rep(0,max(doctor_month_not_after)+1)
for(i in 0:max(doctor_month_not_after)){
    all = length(which(doctor_month_not_after==i))#all the cases
    #when a doctor have i conections have take pills
    num_start = 0# all the cases of a doctor start to take pills
    #when i connections started to take pills before.
    for(j in 1:125){
        if(ckm_nodes$adoption_date[j]!=Inf &
           doctor_month_not_after[j,ckm_nodes$adoption_date[j]]==i){
            num_start = num_start + 1
        }
    }
    q_pro[i+1] = num_start/all
}
q_pro = na.omit(q_pro)
ggplot(data = data.frame(q_pro)) +
  geom_point(aes(x = 0:(length(q_pro)-1), y = q_pro)) +
```

```
  geom_line(aes(x = 0:(length(q_pro)-1), y = q_pro),colour='red') +
  labs(title = "q_k~k",    y = "q_k",    x = "k") +
  theme_bw()
```



4. Because it only conditions on information from the previous month, p_k is a little easier to interpret than q_k. It is the probability per month that a doctor adopts tetracycline, if they have exactly k contacts who had already adopted tetracycline.
   a. Suppose $p_k = a + bk$. This would mean that each friend who adopts the new drug increases the probability of adoption by an equal amount. Estimate this model by least squares, using the values you constructed in (3b). Report the parameter estimates.

```
#since when k>=10 and k=7, the pro is all zero or NA because of the lack of data,
#we delete these probabilities.
x = c(1:6,8,9)
lm.fit = lm(p_pro[x]~x)
summary(lm.fit)
```

```
##
## Call:
## lm(formula = p_pro[x] ~ x)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.024703 -0.016212  0.004184  0.011001  0.021452
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
```

```
## (Intercept) 0.037664   0.013681   2.753   0.0332 *
## x           0.001368   0.002519   0.543   0.6066
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.01877 on 6 degrees of freedom
## Multiple R-squared:  0.04686,    Adjusted R-squared:  -0.112
## F-statistic: 0.295 on 1 and 6 DF,  p-value: 0.6066
```

b. Suppose $p_k = e^{a+bk}/(1+e^{a+bk})$. Explain, in words, what this model would imply about the impa

```
#adding one more adoptee friend will make a given doctor's probability of
#adoption bigger and closer to 1, but not bigger than 1(when b>0)
mse1<-function(param,x_=x,y=p_pro[x]){
    a = param[1]
    b = param[2]
    estimate = exp(a+b*x_)/(1+exp(a+b*x_))
    MSE = mean((y-estimate)^2)
    return(MSE)
}
res = nlm(mse1, c(a=0.1,b=0.1))
res
```

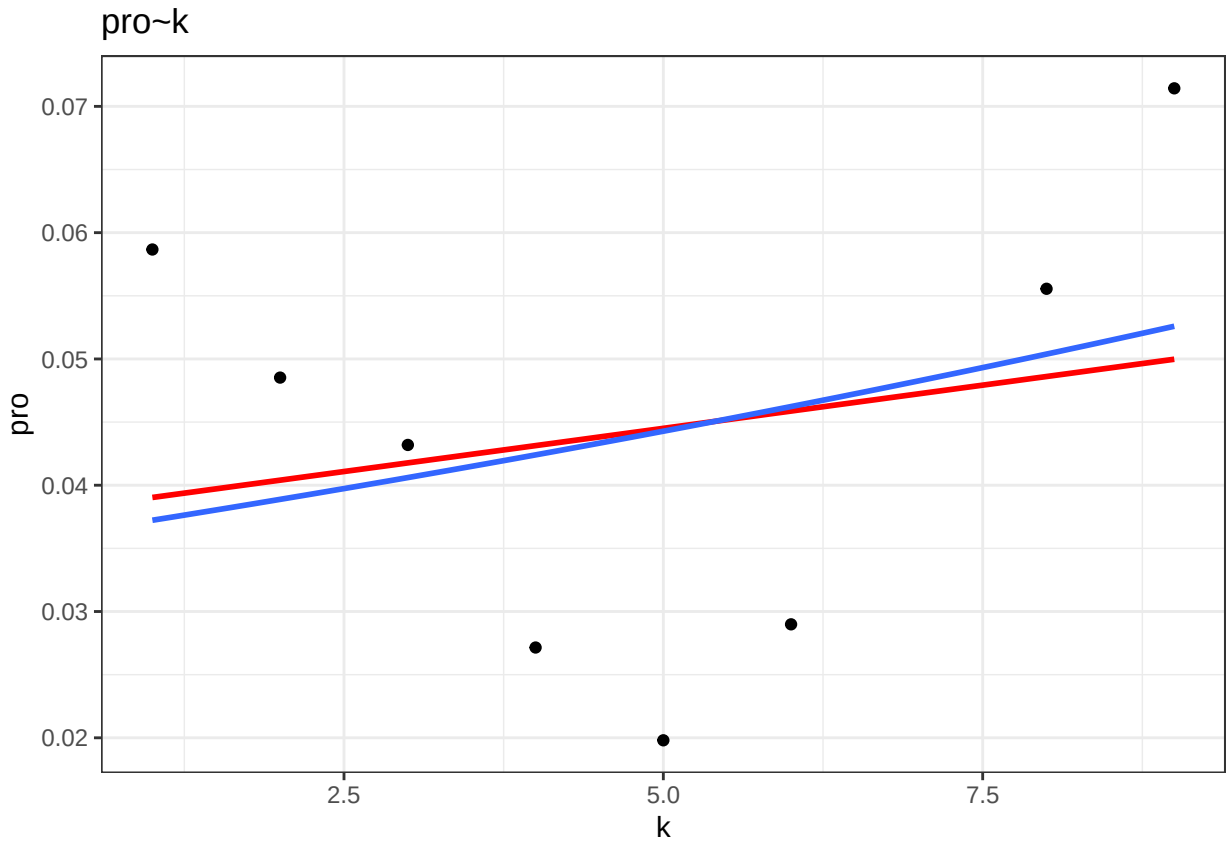
```
## $minimum
## [1] 0.0002587565
##
## $estimate
## [1] -3.29797406  0.04519583
##
## $gradient
## [1] 4.036762e-09 1.630597e-08
##
## $code
## [1] 1
##
## $iterations
## [1] 31
```

```
estimate = exp(a+b*x)/(1+exp(a+b*x))
MSE = mean((y-estimate)^2)
return(MSE)
```

c. Plot the values from (3b) along with the estimated curves from (4a) and (4b). (You should have one plot, with k on the horizontal axis, and probabilities on the vertical axis .) Which model do you prefer, and why?

```
lm.predict1 = predict(lm.fit)
p.estimate = exp(res$estimate[1]+res$estimate[2]*x)/(1+exp(res$estimate[1]+res$estimate[2]*x))
ggplot(data = data.frame(p_pro[x])) +
  geom_point(aes(x = x, y = p_pro[x])) +
  geom_smooth(aes(x = x, y = lm.predict1),colour='red') +
  geom_smooth(aes(x = x, y = p.estimate)) +
  labs(title = "pro~k",   y = "pro",   x = "k") +
  theme_bw()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



I prefer the second model, because the probability can not exist 1.

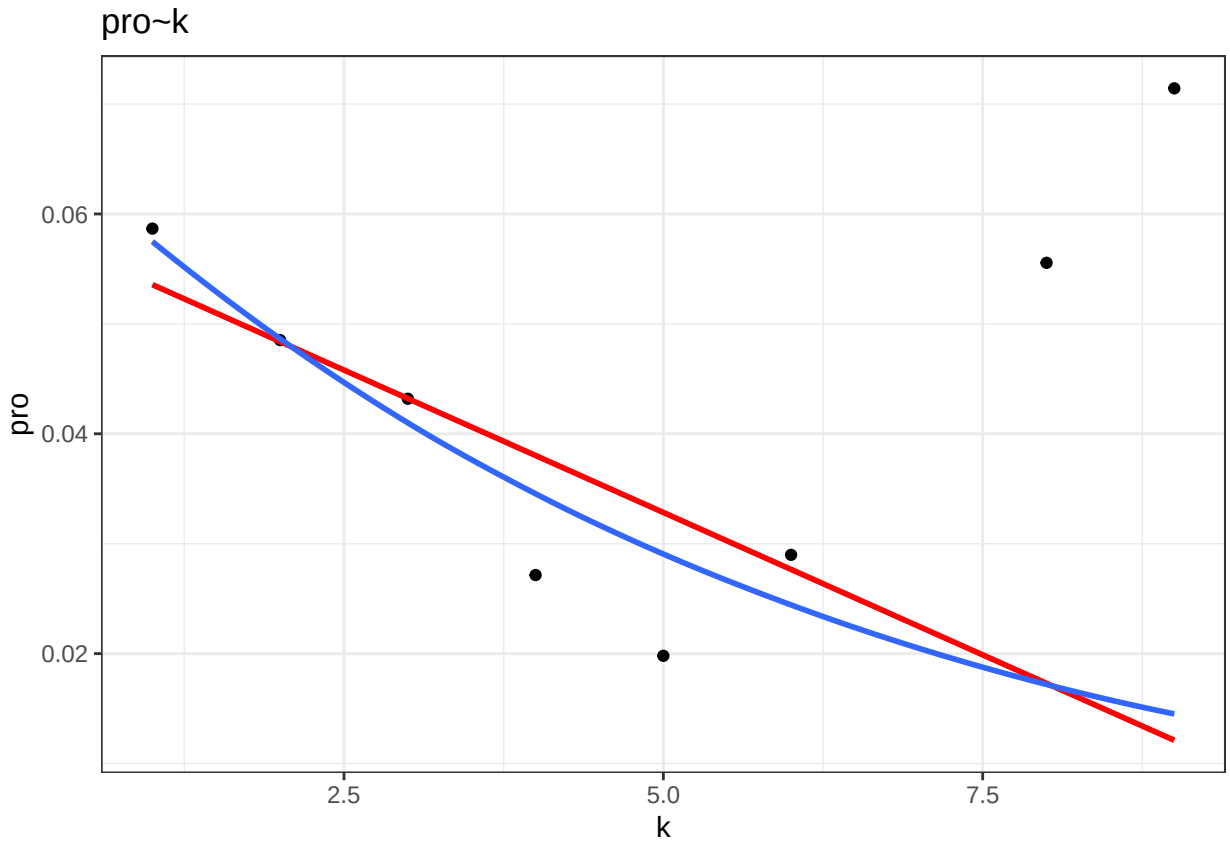
For quibblers, pedants, and idle hands itching for work to do: The p_k values from problem 3 aren't all equally precise, because they come from different numbers of observations. Also, if each doctor with k adoptee contacts is independently deciding whether or not to adopt with probability p_k, then the variance in the number of adoptees will depend on p_k. Say that the actual proportion who decide to adopt is $\hat{p}_k$. A little probability (exercise!) shows that in this situation, $\mathbb{E}[\hat{p}_k] = p_k$, but that $\text{Var}[\hat{p}_k] = p_k(1-p_k)/n_k$, where n_k is the number of doctors in that situation. (We estimate probabilities more precisely when they're really extreme [close to 0 or 1], and/or we have lots of observations.) We can estimate that variance as $\hat{V}_k = \hat{p}_k(1-\hat{p}_k)/n_k$. Find the $\hat{V}_k$, and then re-do the estimation in (4a) and (4b) where the squared error for p_k is divided by $\hat{V}_k$. How much do the parameter estimates change? How much do the plotted curves in (4c) change?

```
Vk = p_pro[x]*(1-p_pro[x])/doc_per_situation[x]
#4(a)
mse1<-function(param,x_=x,y=p_pro[x],V =Vk){
    a = param[1]
    b = param[2]
    estimate = a+x_*b
    MSE = mean((y-estimate)^2/V)
    return(MSE)
}
res1 = nlm(mse1, c(a=0.1,b=0.1))
#4(b)
mse2<-function(param,x_=x,y=p_pro[x],V =Vk){
    a = param[1]
    b = param[2]
    estimate = exp(a+b*x_)/(1+exp(a+b*x_))
    MSE = mean((y-estimate)^2/V)
```

```
    return(MSE)
}
res2 = nlm(mse2, c(a=-3,b=0.04))
#4(c)
p.estimate1 = res1$estimate[1]+res1$estimate[2]*x
p.estimate2 = exp(res2$estimate[1]+res2$estimate[2]*x)/(1+exp(res2$estimate[1]+res2$estimate[2]*x))

ggplot(data = data.frame(p_pro[x])) +
  geom_point(aes(x = x, y = p_pro[x])) +
  geom_smooth(aes(x = x, y = p.estimate1),colour='red') +
  geom_smooth(aes(x = x, y = p.estimate2)) +
  labs(title = "pro~k",    y = "pro",    x = "k") +
  theme_bw()
```

```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



The parameters of linier model change from 0.0377,0.00137 to 0.0587,-0.00518; the parameters of exponential model change from -3,30,0.0452 to -2.62,-0.178.

The fitting line in the plot are more aline with the first 4 dots.