homework3

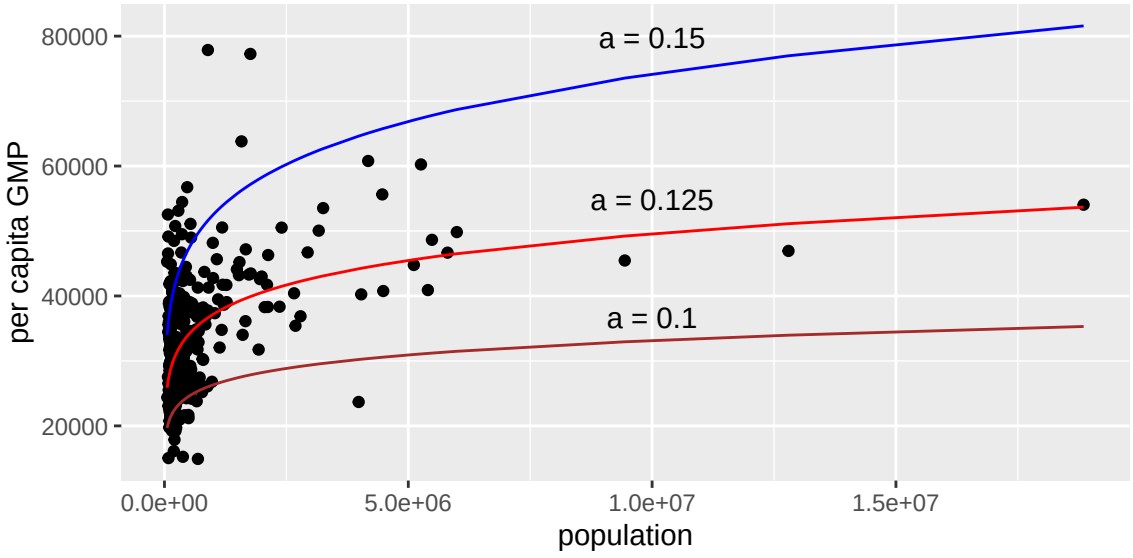
Huang Xubin 3180102999

2021/7/14

1

```
my_model <- function(x, param){
  return(param$y0 * x^param$a)
  invisible(TRUE)
}
```

```
# str(gmp)
gmp %>% ggplot() + geom_point(aes(x = pop, y = pcgmp)) +
  labs(x = "population", y = "per capita GMP") +
  geom_line(aes(x = pop, y = my_model(pop, list(y0 = 6611, a = 0.125))),
            col = 'red') +
  geom_line(aes(x = pop, y = my_model(pop, list(y0 = 6611, a = 0.1))),
            col = 'brown') +
  geom_line(aes(x = pop, y = my_model(pop, list(y0 = 6611, a = 0.15))),
            col = 'blue') +
  geom_text(data = data.frame(x = 1e7, y = 75000),
            aes(x, y, label = "a = 0.15"), vjust = -1) +
  geom_text(data = data.frame(x = 1e7, y = 50000),
            aes(x, y, label = "a = 0.125"), vjust = -1) +
  geom_text(data = data.frame(x = 1e7, y = 35000),
            aes(x, y, label = "a = 0.1"), vjust = 0)
```



2

```
mse <- function(param, N = gmp$pop, Y = gmp$pcgmp){
  stopifnot(length(param) == 2)
  Y_wave <- param[1] * N^param[2]
  return(mean((Y - Y_wave)^2))
}
# mse(c(6611, 0.15))
# mse(c(5000,0.10))
```

4

```
my_another_function <- function(y0_begin, a_begin){
  stopifnot(is.null(dim(a_begin)) && is.null(dim(y0_begin)) &&
              length(a_begin) == length(y0_begin))
  minimum <- c(); y0 <- c(); a <- c()
  n <- length(a_begin)
  for(i in 1 : n){
    result <- nlm(mse, c(y0_begin[i], a_begin[i]))
    minimum <- c(minimum, result$minimum)
    y0 <- c(y0, result$estimate[1])
    a <- c(a, result$estimate[2])
  }
  invisible(return(data.frame(y0_begin = y0_begin, a_begin = a_begin,
                              y0_estimate = y0, a_estimate = a,
                              minimum = minimum)))
  invisible(TRUE)
}
```

```
a_begin <- c(-1, 0.1, 0.125, 0.15, 0.16, 0.17, 0.18, 0.19, 0.2, 0.6)
y0_begin <- rep(c(6611), times = length(a_begin))
defaultW <- getOption("warn"); options(warn = -1)
my_another_function(y0_begin, a_begin)
```

```
##    y0_begin a_begin y0_estimate   a_estimate    minimum
## 1      6611  -1.000    6494.440    0.1276781   61853982
## 2      6611   0.100    6611.000    0.1263177   61857060
## 3      6611   0.125    6611.000    0.1263177   61857060
## 4      6611   0.150    6611.000    0.1263182   61857060
## 5      6611   0.160    6611.000    0.1263177   61857060
## 6      6611   0.170    6611.000    0.1263182   61857060
## 7      6611   0.180    6611.000   -4.9867529 1168662933
## 8      6611   0.190    6610.999  -62.9018009 1168662933
## 9      6611   0.200    6610.998 -145.0805304 1168662933
## 10     6611   0.600    6610.997 -333.9485720 1168662933
```

```
options(warn = defaultW)
```

The two variables minimum and estimate represents the minimum of the object function where the method nlm stops on and the corresponding estimates of the parameters of the origin problem, respectively. In our example above I tried several (more than 3) initial points and it turns out that there is a kind of steep peak lies between the initial values 0.17 and 0.18 of a.

5

```
# Input: initial point, X and Y
# Return: estimated values of parameters, and the final MSE value the method "nlm" got.
plm <- function(param, X = gmp$pop, Y = gmp$pcgmp){
  stopifnot(is.null(dim(param)) && length(param) == 2)
  result <- nlm(mse, param, X, Y)
  return(list(y0 = result$estimate[1], a = result$estimate[2], MSE = result$minimum))
  invisible(TRUE)
}
```

```
# # using function plm:
# plm(c(6611, 0.15))
# plm(c(5000, 0.1))

# using the function defined my own:
y0_begin <- c(6611, 5000)
a_begin <- c(0.15, 0.1)

defaultW <- getOption("warn")
options(warn = -1)

my_another_function(y0_begin, a_begin)
```

```
##   y0_begin a_begin y0_estimate a_estimate  minimum
## 1     6611    0.15        6611  0.1263182 61857060
## 2     5000    0.10        5000  0.1475913 62521484
```

```
options(warn = defaultW)
```

The estimates are listed in the table above, which clearly shows difference between the two initial points and the reason same as that in section 4 (there is a peak of the object value lies between the two initial points), together with the lesser MSE of 61857060 w.r.t. the finer initial point $(y_0,\ a) = (6611,\ 0.15)$.

6

a

The standard error of the mean, SEM, can be estimated by

$$\widehat{\sigma}_{\bar{x}} = \frac{\sigma_x}{\sqrt{n}}$$

where σ the standard deviation, x the sample with n observations and $\bar{x}$ the mean of sample. In our case x is just the per-capita GMP, and by the formula above the mean and the corresponding SEM can easily be calculated as

```
# mean:
pcgmp.mean <- mean(gmp$pcgmp); pcgmp.mean
```

```
## [1] 32922.53
```

and

```
# SEM:
pcgmp.mean.sd <- sd(gmp$pcgmp) / sqrt(length(gmp$pcgmp)); pcgmp.mean.sd
```

```
## [1] 481.9195
```

b

The sub function that carries out the mission of calculating the estimate under given the omitted index of the original initial point.

```
sub_func.mean.omit <- function(pcgmp = gmp$pcgmp, omit_idx){
  return(mean(pcgmp[-omit_idx]))
  invisible(TRUE)
}
```

c

Calculating the jackknifed.means by a "for" loop and the sub function above.

```
jackknifed.means <- c()
n = length(gmp$pcgmp)
for(i in 1 : n){
  jackknifed.means <- c(jackknifed.means, sub_func.mean.omit(gmp$pcgmp, i))
}
# jackknifed.means
```

d

Calculating by hand the variance and standard error of the mean.

```
jackknife.means.var <- var(jackknifed.means) * (n - 1)^2 / n
jackknife.means.sd <- sqrt(jackknife.means.var); jackknife.means.sd
```

```
## [1] 481.9195
```

This answer compares to the one in part (a) is:

```
jackknife.means.sd - pcgmp.mean.sd
```

```
## [1] 1.875833e-12
```

That's delicious.

7

```
# Input: the data, and the given index to be omitted
# Return: the omitted data
sub_func.omit <- function(data, omit_idx){
  data_dims <- dim(data)
  if (is.null(data_dims) || (length(data_dims) == 1)) {
    return(data[-omit_idx])
  } else {
    return(the_data[-omit_idx, ])
  }
  invisible(TRUE)
}

# Input: initial guess of parameters, X and Y
# Return: the jackknife.sd of each parameter
plm.jackknife <- function(param, X = gmp$pop, Y = gmp$pcgmp){
  if (is.null(dim(X))) {
    n <- length(X)
  }
  else {
    n <- nrow(X)
  }

  sub_func.omit_for_sapply <- function(omit_idx){
    return(plm(param, sub_func.omit(X, omit_idx), sub_func.omit(Y, omit_idx))[1 : 2])
  }

  jackknife.ests <- matrix(unlist(sapply(1 : n, sub_func.omit_for_sapply)), ncol = n)
  jackknife.ests.var <- apply(jackknife.ests, 1, var)
  jackknife.var <- jackknife.ests.var * (n - 1)^2 / n
  jackknife.sd <- sqrt(jackknife.var)
  names(jackknife.sd) <- c("y0.jackknife.sd", "a.jackknife.sd")
  return(jackknife.sd)
  invisible(TRUE)
}
```

```
defaultW <- getOption("warn")
options(warn = -1)
plm(c(6611, 0.125))
```

```
## $y0
## [1] 6611
##
## $a
## [1] 0.1263177
##
## $MSE
## [1] 61857060
```

```
plm.jackknife(c(6611, 0.125))
```

```
## y0.jackknife.sd  a.jackknife.sd
##    1.136653e-08    9.901003e-04
```

```
options(warn = defaultW)
```

The standard errors of the two parameters are quite small, which is not so surprised because we have easily tasted the flavor before the estimation of the sd even started through the crazy accuracy the method nlm process under our simple model.

8

```
gmp2013 <- read.table('data/gmp-2013.dat', header = T)
gmp2013$pop <- round(gmp2013$gmp/gmp2013$pcgmp)
```

```
defaultW <- getOption("warn")
options(warn = -1)
plm(c(6611, 0.125), X = gmp2013$pop, Y = gmp2013$pcgmp)
```

```
## $y0
## [1] 6611
##
## $a
## [1] 0.1433688
##
## $MSE
## [1] 135210524
```

```
plm.jackknife(c(6611, 0.125), X = gmp2013$pop, Y = gmp2013$pcgmp)
```

```
## y0.jackknife.sd  a.jackknife.sd
##    2.692652e-08    1.098548e-03
```

```
options(warn = defaultW)
```

The estimated values of the parameters change differently as the value of y0 didn't even changed while the value of 'a' from 0.125 to 0.143, indicating the ratio of gmp one person can process having gone up for a little. The plot of curves under corresponding parameters is showed below: