Homework 3 - Report

3170105743 李政达

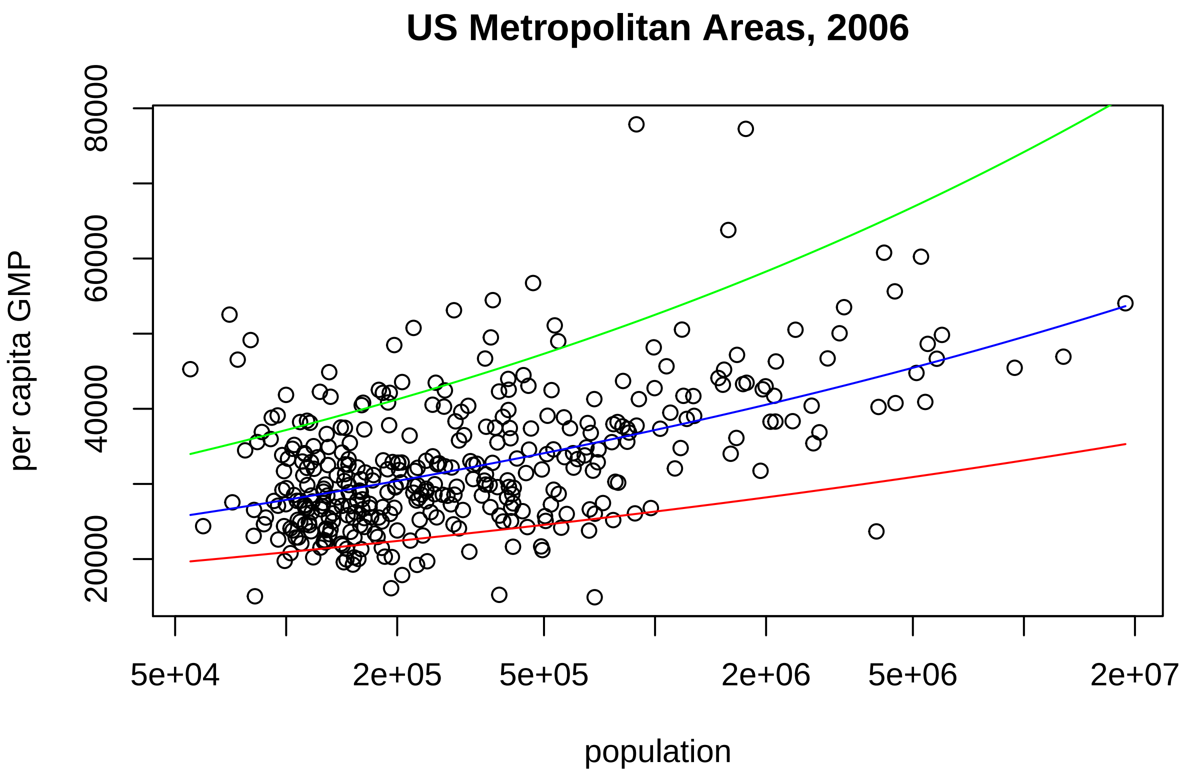
2020/7/8

You will estimate the power-law scaling model, and its uncertainty, using the data alluded to in lecture, available in the file `gmp.dat` from lecture, which contains data for 2006.

```
gmp <- read.table("../data/gmp.dat")
gmp$pop <- round(gmp$gmp / gmp$pcgmp)
```

1. First, plot the data as in lecture, with per capita GMP on the y-axis and population on the x-axis. Add the curve function with the default values provided in lecture. Add two more curves corresponding to $a = 0.1$ and $a = 0.15$; use the `col` option to give each curve a different color (of your choice).

```
plot(gmp$pcgmp ~ gmp$pop, log = "x",
     xlab = "population", ylab = "per capita GMP",
     main = "US Metropolitan Areas, 2006")
curve(6611*x^(1/8), add = TRUE, col = "blue")
curve(6611*x^(0.1), add = TRUE, col = "red")
curve(6611*x^(0.15), add = TRUE, col = "green")
```



2. Write a function, called `mse()`, which calculates the mean squared error of the model on a given data set. `mse()` should take three arguments: a numeric vector of length two, the first component standing for y_0 and the second for a; a numerical vector containing the values of N; and a numerical vector containing the values of Y. The function should return a single numerical value. The latter two arguments should have as the default values the columns `pop` and `pcgmp` (respectively) from the `gmp` data frame from lecture. Your function may not use `for()` or any other loop. Check that, with the default data, you get the following values.

```
> mse(c(6611,0.15))
[1] 207057513
> mse(c(5000,0.10))
[1] 298459915
```

```
mse <- function(x, N = gmp$pop, Y = gmp$pcgmp) {
  return(sum((Y - x[1]*N^x[2]) ^ 2) / length(N))
}
mse(c(6611, 0.15))
```

```
## [1] 207057513
```

```
mse(c(5000, 0.10))
```

```
## [1] 298459914
```

4. R has several built-in functions for optimization, which we will meet as we go through the course. One of the simplest is `nlm()`, or non-linear minimization. `nlm()` takes two required arguments: a function, and a starting value for that function. Run `nlm()` three times with your function `mse()` and three starting value pairs for y_0 and a as in

nlm(mse, c(y0=6611,a=1/8))

What do the quantities `minimum` and `estimate` represent? What values does it return for these?

```
nlm(mse, c(y0=6611,a=1/8))
```

```
## Warning in nlm(mse, c(y0 = 6611, a = 1/8)): NA/Inf replaced by maximum positive
## value

## Warning in nlm(mse, c(y0 = 6611, a = 1/8)): NA/Inf replaced by maximum positive
## value

## Warning in nlm(mse, c(y0 = 6611, a = 1/8)): NA/Inf replaced by maximum positive
## value

## Warning in nlm(mse, c(y0 = 6611, a = 1/8)): NA/Inf replaced by maximum positive
## value

## Warning in nlm(mse, c(y0 = 6611, a = 1/8)): NA/Inf replaced by maximum positive
## value

## Warning in nlm(mse, c(y0 = 6611, a = 1/8)): NA/Inf replaced by maximum positive
## value

## $minimum
## [1] 61857060
##
## $estimate
## [1] 6611.0000000    0.1263177
##
## $gradient
## [1] 50.048639 -9.976327
##
## $code
## [1] 2
##
## $iterations
## [1] 3
```

```
nlm(mse, c(y0=6611,a=0.1))
```

```
## Warning in nlm(mse, c(y0 = 6611, a = 0.1)): NA/Inf replaced by maximum positive
## value

## Warning in nlm(mse, c(y0 = 6611, a = 0.1)): NA/Inf replaced by maximum positive
## value

## Warning in nlm(mse, c(y0 = 6611, a = 0.1)): NA/Inf replaced by maximum positive
## value

## Warning in nlm(mse, c(y0 = 6611, a = 0.1)): NA/Inf replaced by maximum positive
## value

## Warning in nlm(mse, c(y0 = 6611, a = 0.1)): NA/Inf replaced by maximum positive
## value

## Warning in nlm(mse, c(y0 = 6611, a = 0.1)): NA/Inf replaced by maximum positive
```

```
## value
```

```
## $minimum
## [1] 61857060
##
## $estimate
## [1] 6611.0000003    0.1263177
##
## $gradient
## [1]   50.04683 -166.46087
##
## $code
## [1] 2
##
## $iterations
## [1] 6
```

```r
nlm(mse, c(y0=6611,a=0.15))
```

```
## $minimum
## [1] 61857060
##
## $estimate
## [1] 6610.9999997    0.1263182
##
## $gradient
## [1]   51.76354 -210.18952
##
## $code
## [1] 2
##
## $iterations
## [1] 7
```