Project for P-spline and Multilevel

Choi TaeYoung

2020-08-06

Contents

1 필요한 패키지

2 데이터

- Y data : Y데이터의 경우 134주(2018년 1월 ~ 2020년 7월)동안의 카카오로 "혼자여행"을 검색한 횟수를 지역별로 나타냄
- X data : X데이터의 경우 17개의 지역별 인구수

3 데이터 정리 및 Goodness of fit test를 통한 적절한 모델 찾기

- X, Y 데이터 모두 리스트화를 거쳤다. Y데이터가 허들모델이라는 가정으로 각 행마다 0이 얼마나 포함되어 있는지 알아보았다.
- 그 결과 모든 행이 0을 적어도 하나의 값 이상 포함했다.

```
x_list <- x_pop %>% as.data.frame() %>% unlist() %>% as.list()

#How many zero in y_list?
y_list <- obs_y[-1,-1] %>% t() %>% as.data.frame()
y_zero <- NULL
for(m in 1:135){
  zero <- NULL
  zero <- length(which(y_list[m] == 0))/17
  y_zero <- rbind(y_zero,zero)

  zero_count <- length(which(y_zero > 0))
  zero_where <- which(y_zero > 0)
  zero_count
  zero_where
}
zero_count
```

```
## [1] 135
```

4 Multilevel 모델에 적용

- 논문의 방법인 EM알고리즘을 통해 multilevel spline 방법으로 최적의 μ 벡터를 찾았다.

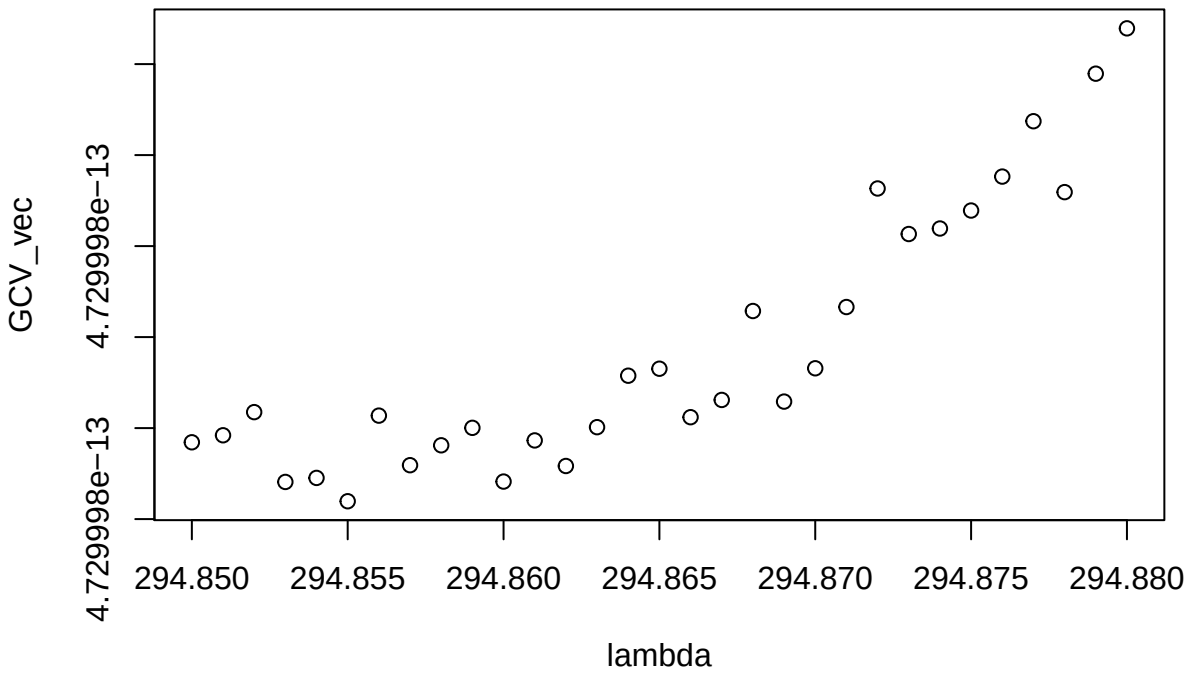
```
x_list <- x_pop %>% as.data.frame() %>% unlist() %>% as.list()
y_list <- obs_y[-1,-1] %>% t() %>% as.data.frame()
#multilevel

  #beta_hat_vector 구하기

  grain_out <- NULL
  J=135
  beta_hat <- NULL
  for(m in 1:135){
    result2_out <- NULL
    results2 <- glm(unlist(y_list[m]) ~ unlist(x_list), maxit=2000)
      kth_beta_hat <- coef(results2)[2]
      kth_var <- diag(vcov(results2))[2]
      grain_out <- list(kth_beta_hat, kth_var)
      grain_out
    beta_hat <- rbind(beta_hat,grain_out)
  }
```

- p-spline 기법을 활용하여 새롭게 짠 코드로 리얼데이터에 적용

```
# lambda <- c(1e-09,1e-08,1e-07,1e-06,1e-05,1e-04,1e-03,1e-02,1, 10,100,1000,10000)
# GCV_vec <- NULL
#
n <- length(unlist(beta_hat[,1]))
IK <- unlist(beta_hat[,1])[-c(1, n)]
EK <- unlist(beta_hat[,1])[c(1, n)]
B = GetBSpline(unlist(beta_hat[,1]), deg = 3, IK, EK)
#
# for(i in 1:length(lambda)){
# EM_out <- main_EM_p(beta_hat_vec = unlist(beta_hat[,1]), V = diag(unlist(beta_hat[,2])),
#                     B=GetBSpline(unlist(beta_hat[,1]), deg = 3, IK, EK), D=GetDiffMatrix(dim(B)[1], 2
# GCV_vec <- rbind(GCV_vec,EM_out$GCV)
# }
#
# lambda[which.min(GCV_vec)]
```

- lambda[which.min(GCV_vec)]을 실행할 때, 100이 나온다.
- 그래서 100근처에서 GCV벡터를 더 찾아보기로 한다.

```
lambda <- seq(294.85,294.88, by=.001)
GCV_vec <- NULL

for(i in 1:length(lambda)){
EM_out <- main_EM_p(beta_hat_vec = unlist(beta_hat[,1]), V = diag(unlist(beta_hat[,2])),
                    B=GetBSpline(unlist(beta_hat[,1]), deg = 3, IK, EK), D=GetDiffMatrix(dim(B)[1], 2),
GCV_vec <- rbind(GCV_vec,EM_out$GCV)
}


plot(lambda, GCV_vec)
```



- 최적의 GCV_vec로 EM_out구하기 <- mu_hat 구함

```
lambda[which.min(GCV_vec)]
```

```
## [1] 294.855
```

```
EM_out <- main_EM_p(beta_hat_vec = unlist(beta_hat[,1]), V = diag(unlist(beta_hat[,2])),
                    B=GetBSpline(unlist(beta_hat[,1]), deg = 3, IK, EK), D=GetDiffMatrix(dim(B)[1], 2),
tail(EM_out$mu)
```

```
##                 [,1]
## [130,] -3.055785e-07
## [131,]  3.102098e-06
## [132,]  1.053962e-06
## [133,]  1.676608e-06
## [134,]  1.583844e-06
## [135,]  7.936623e-07
```

- Multilevel과 성능을 비교하기위해서 Naive한 방법으로 구해보자.
- Naive기법 역시 P-spline으로 코드를 짠 후 실행했다.

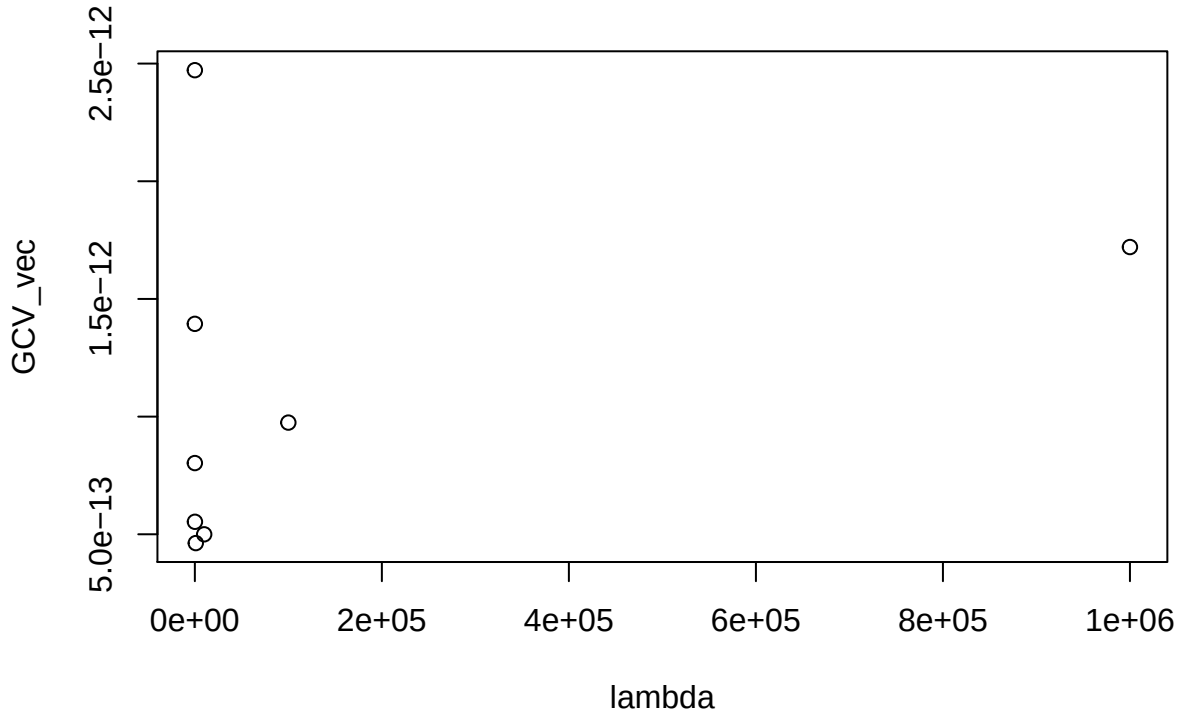
```
#naive
GCV_vec <- NULL
lambda <- c(0.1,1,10,100,1000, 10^4, 10^5, 10^6)
for(i in 1:length(lambda)){
  naive_out <- naive_ss_p(beta_hat_vec = unlist(beta_hat[,1]), B=GetBSpline(unlist(beta_hat[,1]), deg =
  GCV_vec <- rbind(GCV_vec,naive_out$GCV)
}

lambda[which.min(GCV_vec)]
```

```
## [1] 1000
```

4.1 Naive's GCV vector 찾기

- Naive 역시 비슷한 방법으로 풀어나간다.

```
# lambda <- seq(1870, 1910, by=1)
# GCV_vec <- NULL
# for(i in 1:length(lambda)){
#   naive_out <- naive_ss_p(beta_hat_vec = unlist(beta_hat[,1]), lambda = lambda[i],
#                           B=GetBSpline(unlist(beta_hat[,1]), deg = 3, IK, EK), D=GetDiffMatrix(dim(B)[
#   GCV_vec <- rbind(GCV_vec,naive_out$GCV)
# }
plot(lambda, GCV_vec)
```



```r
lambda[which.min(GCV_vec)]
```

```
## [1] 1000
```

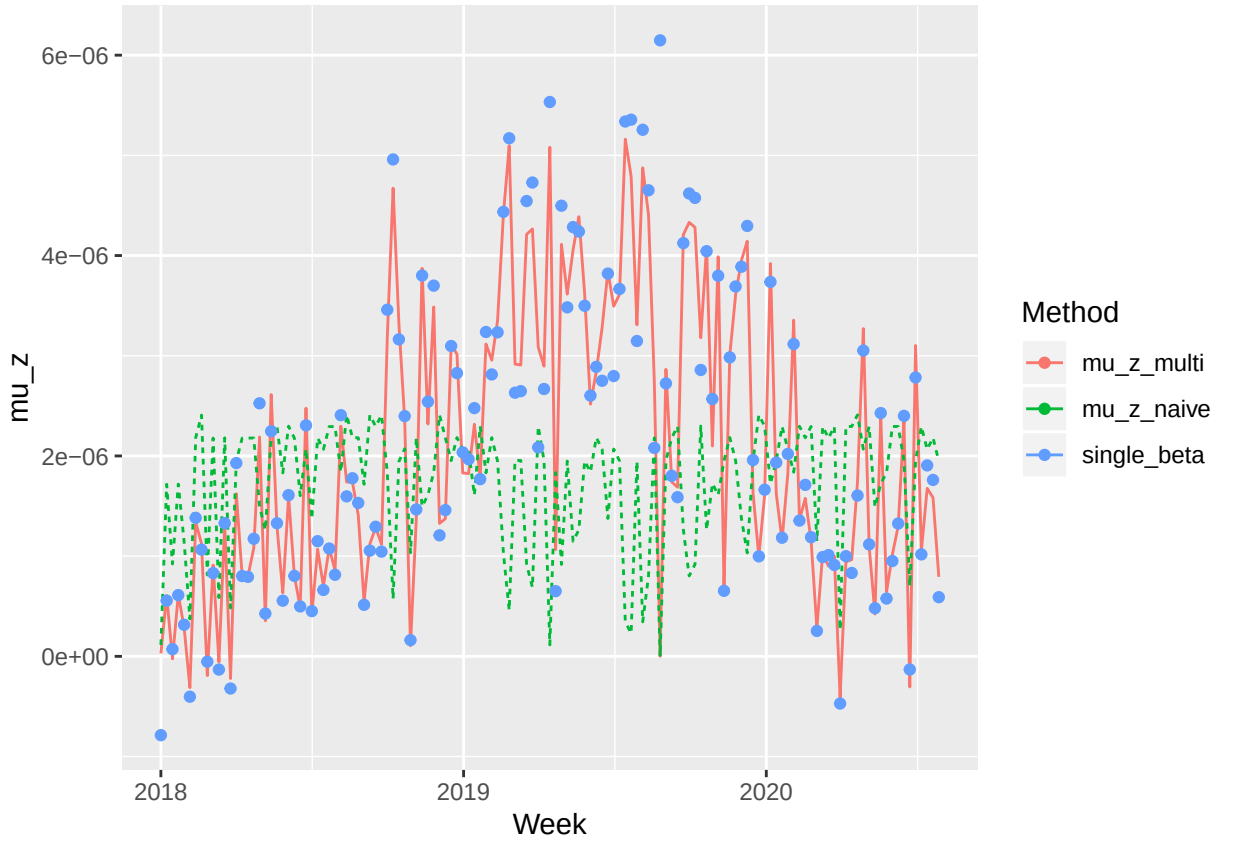
```r
naive_out <- naive_ss_p(beta_hat_vec = unlist(beta_hat[,1]), B=GetBSpline(unlist(beta_hat[,1]), deg = 3


tail(naive_out$mu)
```

```
##                [,1]
## [130,] 7.018662e-07
## [131,] 1.953920e-06
## [132,] 2.293678e-06
## [133,] 2.065588e-06
## [134,] 2.181511e-06
## [135,] 1.948507e-06
```

5 그래프

```r
# hat_all
single_beta <- unlist(beta_hat[,1]) %>% as.vector()
mu_z_naive <- naive_out$mu %>% as.vector()
mu_z_multi <- EM_out$mu %>% as.vector()

hat_all <- cbind(mu_z_naive,mu_z_multi,single_beta) %>% as.data.frame

test_mon <- fread("S:\\Teo\\OneDrive - 성균관대학교\\soltr_kakao_y.csv")
test_mon <- test_mon[-1,1]
hat_all <- cbind(test_mon,hat_all)
hat_all <- rename(hat_all, Week = V1)

hat_all$Week <- parse_date_time(hat_all$Week, "ymd")
hat_all$Week <- as.Date(hat_all$Week, format="%Y-%m-%d")
hat_all <- as.data.frame(hat_all)
hat_all <- hat_all %>% mutate_if(is.character,parse_number)

# gather 사용
df1 <- gather(hat_all[, c("Week", "mu_z_naive", "mu_z_multi")],
              key = "Method", value = "mu_z", -Week)

df2 <- cbind(test_mon,single_beta)
df2 <- rename(df2, Week =V1)
df2$Week <- parse_date_time(df2$Week, "ymd")
df2$Week <- as.Date(df2$Week, format="%Y-%m-%d")
df2 <- as.data.frame(df2)
df2 <- df2 %>% mutate_if(is.character,parse_number)


g <- ggplot(df1) +
  geom_line(aes(x = Week, y = mu_z, color = Method, linetype = Method)) +
  geom_point(data=df2, aes(x = Week, y = single_beta, color = "single_beta")) +
  guides(linetype = "none") +
  scale_color_discrete(name = "Method")
```

g


6e−06
4e−06
2e−06
0e+00
mu_z
2018
2019
2020
Week
Method
mu_z_multi
mu_z_naive
single_beta