hw1

Jieun Shin

2022-10-08

1. 과산포의 표준오차 실험

시뮬레이션 데이터는 최민옥(2017)의 모의실험 디자인으로 한다. 과대산포를 허용하는 음이항 모형에서 반응변수를 생성하고 이를 포아송회귀모형 및 음이항 회귀모형에서 추정한 후, 회귀계수 추정량 및 회귀계수 추정량의 추정량의 표준오차를 계산하기로 한다. 또한 회귀계수에 대한 가설검정을 실시한다.

- $E(Y_i) = \mu_i = \exp(\beta_0 + \beta_1 x_i)$와 α (산포모수)를 기반으로 함.
- 설명변수 x_i는 $\text{Unif}(0, 1)$에서 생성함.
- 회귀계수 β_0와 β_1의 참값은 각각 1.2와 0.5로 설정함. 이를 기반으로 μ_i의 값을 계산함.
- 반응변수 Y_i는 $NB(\mu_i, \alpha)$에서 생성함. 이때 산포모수의 값은 산포모수의 효과를 알아보기 위하여 0에서 1까지 0.1단위로 움직임.
- 표본 수 n은 200과 500을 사용함.
- 각 모수의 값에서 총 1000번의 반복을 실시함. 각 반복에서는 포아송과 음이항 회귀모형에서의 회귀계수에 대한 추정 및 가설검정을 실시하였음.

```
rm(list=ls())
# 시뮬레이션 데이터 생성
n = 200
p = 2
beta = c(1.2, 0.5)      # true coefficient

N_rep = 1000            # 반복 수
result = list()         # tau별로 저장할 공간
tau_grid = seq(0.1, 1, length.out = 10)

t = 0
for(tau in tau_grid){
  t = t + 1
  ps_theta = matrix(0, p, N_rep)  # 반복별로 theta_hat을 저장할 공간 (pois)
  ps_wald = matrix(0, 3, N_rep)     # 반복별로 wald를 저장할 공간 (pois)
  nb_theta = matrix(0, p+1, N_rep)  # 반복별로 theta_hat을 저장할 공간 (nbr)
  nb_wald = matrix(0, 3, N_rep)     # 반복별로 wald를 저장할 공간 (nbr)

    for(r in 1:N_rep){
      set.seed(r)
      x = runif(n, 0, 1)
      design_x = cbind(rep(1, n), x)  # design matrix
      mu = exp(design_x %*% beta)
      y = rnbinom(n, mu = mu, size = 1/tau)

      # fit poisson regression
```

```
      psr_fit = summary(glm(y ~ x, family = poisson()))
      ps_theta[,r] = c(psr_fit$coefficients[,1])                # (beta0, beta1) 추정
      mu_hat = exp(design_x %*% ps_theta[,r])

      ps_wald[1,r] = psr_fit$coefficients[2,2]              # beta1 standard error estimate
      ps_wald[2,r] = psr_fit$coefficients[2,3]              # wald statistics
      CI = ps_theta[2, r] + c(-1, 1) * 1.96 * ps_wald[1,r]  # if beta1 is rejected
      ps_wald[3,r] = CI[1] <= beta[2] & beta[2] <= CI[2]    # If beta under H0 is in CI, then H0 is no

      # fit negative binomial regression
      nbr_fit = summary(glm.nb(y ~ x))

      nb_theta[,r] = c(nbr_fit$coefficients[,1], 1/nbr_fit$theta) # (beta0, beta1, tau) 추정
      mu_hat = exp(design_x %*% nb_theta[1:2, r])

      nb_wald[1,r] = nbr_fit$coefficients[2,2]              # beta1 standard error estimate
      nb_wald[2,r] = nbr_fit$coefficients[2,3]              # wald statistics
      CI = nb_theta[2, r] + c(-1, 1) * 1.96 * nb_wald[1,r]  # if beta1 is rejected
      nb_wald[3,r] = CI[1] <= beta[2] & beta[2] <= CI[2]    # If beta under H0 is in CI, then H0 is no
    }

  out = list()
  out$tau = tau
  out$ps_theta = ps_theta
  out$ps_wald = ps_wald
  out$nb_theta = nb_theta
  out$nb_wald = nb_wald
  result[[t]] = out
}
```

```
## Warning in theta.ml(Y, mu, sum(w), w, limit = control$maxit, trace =
## control$trace > : iteration limit reached

## Warning in theta.ml(Y, mu, sum(w), w, limit = control$maxit, trace =
## control$trace > : iteration limit reached

## Warning in theta.ml(Y, mu, sum(w), w, limit = control$maxit, trace =
## control$trace > : iteration limit reached

## Warning in theta.ml(Y, mu, sum(w), w, limit = control$maxit, trace =
## control$trace > : iteration limit reached

## Warning in theta.ml(Y, mu, sum(w), w, limit = control$maxit, trace =
## control$trace > : iteration limit reached

## Warning in theta.ml(Y, mu, sum(w), w, limit = control$maxit, trace =
## control$trace > : iteration limit reached
```

```
# 음이항 회귀모형 fit결과: tau = 0.1 일때 1000개 중 5개의 추정치 보기
tem = result[[1]]$nb_theta[,1:5]
rownames(tem) = c("beta0_hat", "beta1_hat", "tau_hat")
colnames(tem) = paste0("iter", 1:5)
tem
```

```
##               iter1     iter2     iter3     iter4     iter5
```

```
## beta0_hat 1.2417427 1.2107332 1.1802949 0.9545122 1.1156051
## beta1_hat 0.3923176 0.4658423 0.5965548 0.7620163 0.5630438
## tau_hat   0.1397265 0.1322678 0.1189517 0.1433956 0.0955906
```

```
# 음이항 회귀모형 fit결과: tau = 0.1 일때 1000개 중 5개의 wald 통계량과 기각여부 보기
tem = result[[1]]$nb_wald[,1:5]
tem[3,] = ifelse(1, "not reject", "reject")
colnames(tem) = paste0("iter", 1:5)
rownames(tem) = c("beta1_var_hat", "wald", "if.rejected")
tem
```

```
##                iter1              iter2               iter3
## beta1_var_hat "0.16194605561965" "0.141433347373997" "0.151477710738149"
## wald          "2.4225204584562"  "3.29372292474517"  "3.93823479467975"
## if.rejected   "not reject"       "not reject"        "not reject"
##                iter4              iter5
## beta1_var_hat "0.15452298810824" "0.143376108599999"
## wald          "4.93141081777171" "3.9270404527536"
## if.rejected   "not reject"       "not reject"
```

```
rm(tem)
```

이번에는 전체 결과를 표로 정리해보자. 여기서는 τ별 1000개 β_1의 추청치 평균, bias, mse만 출력하였다. 여기서 포아송회귀, 음이항 회귀모형에서 모두 β_1의 bias가 매우 작으므로 불편추정치라고 할 수 있다. 따라서 회귀계수 추정에는 문제가 없다.

다음으로 표준오차의 평균 및 $H_0 : \beta_1 = 0.5$에 대한 검정 결과를 보자. 포아송회귀에서는 과대산포가 커질수록 회귀계수의 표준오차의 평균에 큰 변화가 없는 반면 음이항 회귀모형에서 회귀계수의 표준오차는 커지고 있다. 이로부터 포아송모형은 회귀계수를 과소추정하는 것을 알 수 있다. 이어서 검정 결과를 살펴보면 음이항 회귀에서 추정된 유의수준은 명목 유의수준 5%를 어느정도 유지하고 있지만 포아송회귀의 경우 과대추정하는 것을 확인할 수 있다. 따라서 데이터에서 과대산포를 무시하면 추정에는 문제가 없지만 표준오차 추정에는 문제가 발생하는 것을 알 수 있다.

```
ps_beta1_mean = sapply(1:length(tau_grid), function(r) mean(result[[r]]$ps_theta[2,]))  # tau별 1000개 b
ps_beta1_bias = sapply(1:length(tau_grid), function(r) mean(result[[r]]$ps_theta[2,]) - 0.5)  # tau별 10
ps_beta1_mse = sapply(1:length(tau_grid), function(r) mean((result[[r]]$ps_theta[2,] - 0.5)^2))  # tau별

nb_beta1_mean = sapply(1:length(tau_grid), function(r) mean(result[[r]]$nb_theta[2,]))  # tau별 1000개 b
nb_beta1_bias = sapply(1:length(tau_grid), function(r) mean(result[[r]]$nb_theta[2,]) - 0.5)  # tau별 10
nb_beta1_mse = sapply(1:length(tau_grid), function(r) mean((result[[r]]$nb_theta[2,] - 0.5)^2))  # tau별

view1 = data.frame("tau" = tau_grid,
                   "ps_beta1_mean"= ps_beta1_mean,
                   "ps_beta1_bias" = ps_beta1_bias,
                   "ps_beta1_mse"= ps_beta1_mse,
                   "nb_beta1_mean"= nb_beta1_mean,
                   "nb_beta1_bias" = nb_beta1_bias,
                   "nb_beta1_mse"= nb_beta1_mse
                   )
view1
```

```
##    tau ps_beta1_mean ps_beta1_bias ps_beta1_mse nb_beta1_mean nb_beta1_bias
## 1  0.1     0.4913434 -0.0086566251   0.01985694     0.4916500 -0.0083499913
## 2  0.2     0.4898730 -0.0101269862   0.02607720     0.4896715 -0.0103285089
## 3  0.3     0.5021637  0.0021636940   0.03540845     0.5029156  0.0029155813
## 4  0.4     0.4871652 -0.0128348098   0.04082516     0.4884451 -0.0115549182
## 5  0.5     0.4896710 -0.0103290401   0.04521122     0.4904683 -0.0095316572
```

```
## 6  0.6     0.4844980 -0.0155019869   0.04929874     0.4857151 -0.0142848721
## 7  0.7     0.4912303 -0.0087697079   0.05599574     0.4922318 -0.0077682434
## 8  0.8     0.5005065  0.0005064652   0.06467434     0.5021803  0.0021802765
## 9  0.9     0.4850902 -0.0149098438   0.07599463     0.4872160 -0.0127839895
## 10 1.0     0.4997086 -0.0002914272   0.07340177     0.5008680  0.0008680216
##    nb_beta1_mse
## 1    0.01989858
## 2    0.02598928
## 3    0.03524901
## 4    0.04049798
## 5    0.04446456
## 6    0.04893750
## 7    0.05595319
## 8    0.06502399
## 9    0.07650263
## 10   0.07358609
```

```
# standard error table
ps_beta1_se = sapply(1:length(tau_grid), function(r) mean(result[[r]]$ps_wald[1,]))
ps_beta1_rej = sapply(1:length(tau_grid), function(r) 1-mean(result[[r]]$ps_wald[3,]))

nb_beta1_se = sapply(1:length(tau_grid), function(r) mean(result[[r]]$nb_wald[1,]))
nb_beta1_rej = sapply(1:length(tau_grid), function(r) 1-mean(result[[r]]$nb_wald[3,]))

view2 = data.frame("tau" = tau_grid,
                   "ps_beta1_se"= ps_beta1_se,
                   "ps_beta1_rej" = ps_beta1_rej,
                   "nb_beta1_se"= nb_beta1_se,
                   "nb_beta1_rej" = nb_beta1_rej
                   )
view2
```

```
##    tau ps_beta1_se ps_beta1_rej nb_beta1_se nb_beta1_rej
## 1  0.1   0.1192905        0.102   0.1418746        0.054
## 2  0.2   0.1193339        0.143   0.1615055        0.045
## 3  0.3   0.1193756        0.213   0.1793026        0.066
## 4  0.4   0.1194074        0.248   0.1953117        0.058
## 5  0.5   0.1196290        0.284   0.2098598        0.045
## 6  0.6   0.1196783        0.302   0.2239737        0.053
## 7  0.7   0.1196637        0.325   0.2370437        0.054
## 8  0.8   0.1196332        0.349   0.2495102        0.051
## 9  0.9   0.1195817        0.411   0.2607202        0.061
## 10 1.0   0.1197429        0.388   0.2719820        0.049
```

2. 과대산포에 대한 모의실험

과대산포를 허용하는 음이항모형에서 반응변수를 생성하고, 유의수준 0.05에서 과대산포에 대한 3가지 검정(LR, Wald, Score test)를 실시하고 기각 및 채택여부를 파악한 후, 추정된 유의수준과 검정력을 계산해보자. 이를 통해 3가지 검정의 소표본 성질을 알아보고자 한다.