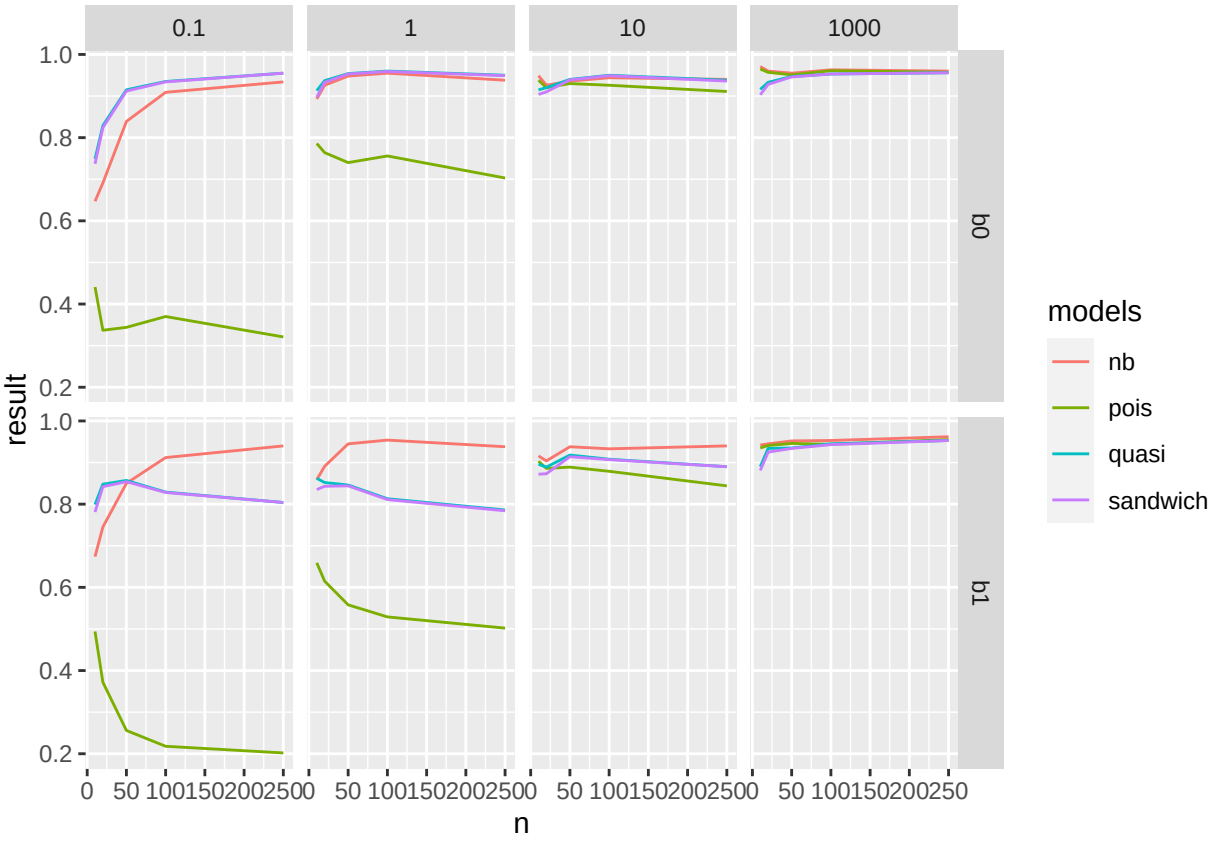
Stat 570 HW3

Dongyang Wang

2022-10-14

Q1

1



Based on the result, we can observe differences between the four models. The Poisson model performs worst with small b, but performs better as b is large such that the mean is approximately equal to the variance and meets model assumptions. The model performance gets worse for Poisson as n increases in cases where b is small, because more data points make the model go to the wrong direction even more. The quasi likelihood performs better than the Poisson model in small b cases, and display a similar trend in coverage with respect to n. This potentially is due to the fact that variance is quadratic in mean but the model assumes linearity. The NB model performs well overall, since it captures the correct trend in the data. The performance improves as n increases. The sandwich model performs well overall except in small sample sizes, more robust to variance mis-specifications.

Q2

a

Integrating over pdf gives cdf, and $h(y|\lambda) = \frac{p(y|\lambda)}{S(y|\lambda)} = \frac{\lambda e^{-\lambda y}}{e^{-\lambda y}} = \lambda$.

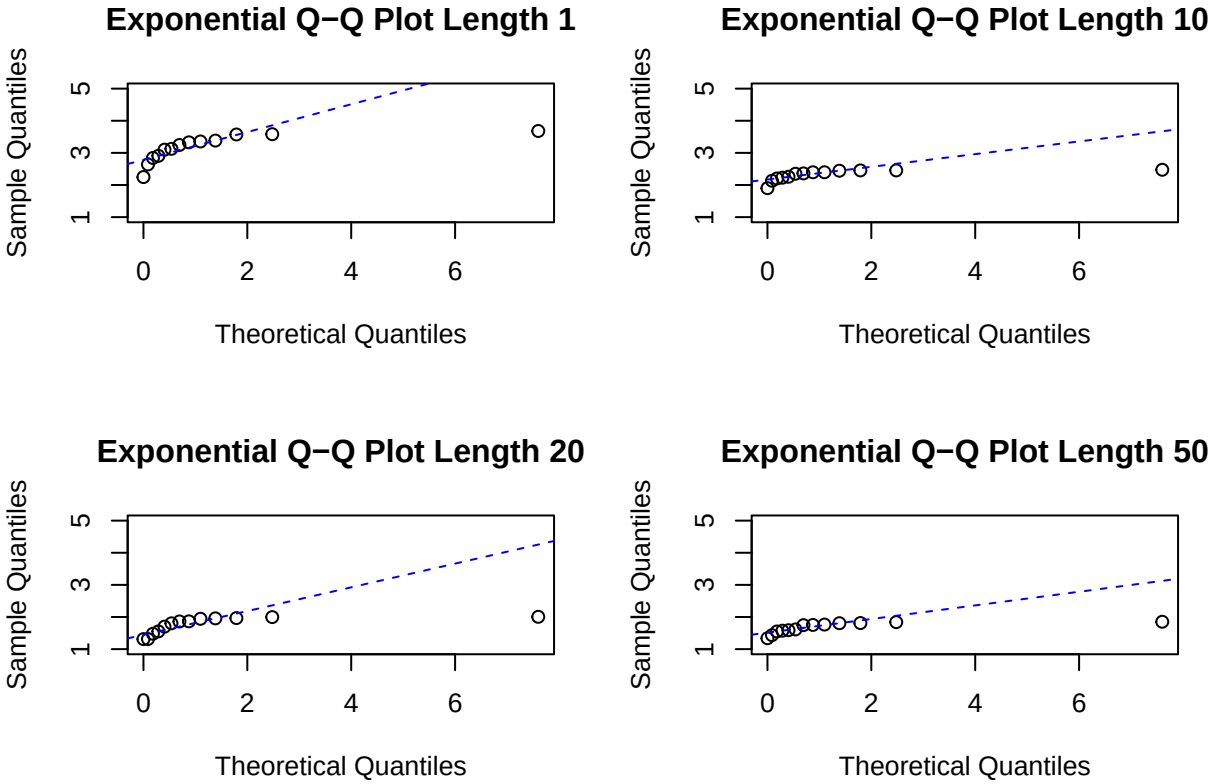
$l(\lambda|y) = n\log(\lambda) - \lambda\sum_{i=1}^{n} y_i$, taking derivative

$\hat{\lambda} = \frac{n}{\sum_{i=1}^{n} y_i} = \frac{1}{y}$ and

$I(\hat{\lambda}) = -E(-\frac{n}{\lambda^2}) = \frac{n}{\lambda^2}$

and therefore $\hat{\lambda}$ has variance $\frac{\lambda^2}{n}$.

b




```
## [1] 0.3170190 0.4325835 0.5712528 0.5998523
```

```
## [1] 0.08792526 0.11997708 0.15843702 0.16636911
```

The MLEs and standard errors are listed above. Based on the visualizations, the model seems not fitted very well since they do not align to the qqline.

c

$\hat{\alpha} = \frac{1}{n}\sum_{i=1}^{n} \frac{(Y_i - \hat{\mu})^2}{V(\hat{\mu})} = \frac{1}{n}\sum_{i=1}^{n} \frac{(Y_i - \bar{Y})^2}{\bar{Y}^2} = \frac{s^2}{\bar{Y}^2}$

```
## [1] 0.011421229 0.008549992 0.023068153 0.016329096
```

```
## [1] 0.016873235 0.005078492 0.021198850 0.009633385
```

```
## [1] 1.304445e-04 7.310236e-05 5.321397e-04 2.666394e-04
```

The alphas and quasi-variances are listed above. It turns out that the exponential model exceeds the dispersion of the sampling distribution.

d

Taking Score function as estimating function such that $G(\hat{\lambda}, Y_i) = \frac{1}{\hat{\lambda}} - y_i$, so we have $\hat{A} = E(\frac{1}{-\hat{\lambda}^2}) = -\bar{y}^2$ $\hat{B} = E((\frac{1}{\hat{\lambda}} - y_i)^2) = \frac{n-1}{n} s^2$ $var(\hat{\lambda}) = \frac{n-1}{n^2} \frac{s^2}{\bar{y}^4} = \frac{n-1}{n} var(quasi)$

```
## [1] 0.0001204103 0.0000674791 0.0004912058 0.0002461286
```

The above is sandwich variance. The following is a summary table.

```
##             length1    length10   length20    length50
## mle      0.31701905 0.432583522 0.57125280 0.599852344
## sds      0.08792526 0.119977082 0.15843702 0.166369106
## alpha    0.01687324 0.005078492 0.02119885 0.009633385
## quasi_sd 0.01142123 0.008549992 0.02306815 0.016329096
## sand_sd  0.01097316 0.008214566 0.02216316 0.015688488
```

e

$E(Y|\eta, \alpha) = \alpha\Gamma(1 + \frac{1}{\eta})$

$Var(Y|\eta, \alpha) = \alpha^2(\Gamma(1 + \frac{2}{\eta}) - \Gamma(1 + \frac{1}{\eta})^2)$

$h(Y|\eta, \alpha) = e^{-(\frac{y}{\alpha})^\eta}$

Weibull(1, α) is equivalent to exp($\frac{1}{\alpha}$)

f

No, the distribution does not follow exactly as an exponential distribution, although the transformation can be made when $\eta = 1$ as indicated in e. So the consistency for inference might be different from what we observe in previous questions.

g

Taking log of the likelihood function, $l(\eta, \alpha|y) = n\log(\eta) - n\eta\log(\alpha) + (\eta - 1)\sum_{i=1}^n \log(y_i) - \sum_{i=1}^n (\frac{y_i}{\alpha})^\eta$

$S_\eta(\eta, \alpha) = \frac{n}{\eta} - n\log(\alpha) + \sum_{i=1}^n \log(y_i) - \sum_{i=1}^n (\frac{y_i}{\alpha})^\eta \log(\frac{y_i}{\alpha})$

$S_\alpha(\eta, \alpha) = -\frac{n\eta}{\alpha} + \frac{\eta}{\alpha^{\eta+1}} \sum_{i=1}^n y_i^n$

Let 1 denote η and 2 denote α, $I_{11} = \frac{n}{\eta^2} + \sum_{i=1}^n (\log(\frac{y_i}{\alpha}))^2 (\frac{y_i}{\alpha})^\eta$

$I_{12} = I_{21} = \frac{n}{\alpha} - \frac{\eta}{\alpha} \sum_{i=1}^n (\frac{y_i}{\alpha})^\eta \log(\frac{y_i}{\alpha}) - \frac{1}{\alpha} \sum_{i=1}^n (\frac{y_i}{\alpha})^\eta$

$I_{22} = -\frac{n\eta}{\alpha} + \frac{\eta(\eta+1)}{\alpha^2} \sum_{i=1}^n (\frac{y_i}{\alpha})^\eta$

h

Setting two score functions to zero, we first obtain

$\alpha = (\frac{1}{n} \sum_{i=1}^n y_i^n)^{\frac{1}{\eta}}$

All we need to do is to solve

$\frac{n}{\eta} - n\log((\frac{1}{n} \sum_{i=1}^n y_i^n)^{\frac{1}{\eta}}) + \sum_{i=1}^n \log(y_i) - \sum_{i=1}^n (\frac{y_i}{(\frac{1}{n} \sum_{i=1}^n y_i^n)^{\frac{1}{\eta}}})^\eta \log(\frac{y_i}{(\frac{1}{n} \sum_{i=1}^n y_i^n)^{\frac{1}{\eta}}}) = 0$

i

```
##      length       eta   eta se    alpha   alpha se
## [1,]      1 10.334059 2.349637 3.318984 0.09344149
## [2,]     10 21.345255 5.054396 2.377346 0.03223122
```

```
## [3,]     20  9.635766 2.311027 1.852108 0.05574973
## [4,]     50 13.828583 3.212538 1.735277 0.03653187
```

Appendix

Q1

```
rm(list = ls())

library(MASS)
library(ggplot2)

set.seed(42)

beta_0 = 0.5
beta_1 = log(2.5)
b = c(0.1, 1, 10, 1000)
n = c(10, 20, 50, 100, 250)

data_gen <- function(b, n){
  x_i <- rnorm(n, 0, 1)
  mu_i <- exp(beta_0 + beta_1* x_i)

  theta_i <- rgamma(n, shape=b, rate=b)
  y_i <- sapply(mu_i*theta_i, function(mean_val) rpois(1, mean_val))

  return(data.frame(x = x_i, y = y_i))
}

fitting <- function(b, n, replications){

  # data
  df <- data_gen(b, n)

  # Poisson
  try_pois <- try({
  pois_model <- glm(y ~ x, data = df, family = "poisson")
  pois_ci <- c(pois_model$coef[1] + qnorm(0.025) * sqrt(vcov(pois_model)[1, 1]),
               pois_model$coef[1] + qnorm(0.975) * sqrt(vcov(pois_model)[1, 1]),
               pois_model$coef[2] + qnorm(0.025) * sqrt(vcov(pois_model)[2, 2]),
               pois_model$coef[2] + qnorm(0.975) * sqrt(vcov(pois_model)[2, 2]))
  pois_cover <- c(pois_ci[1] < beta_0 & beta_0 <  pois_ci[2],
                  pois_ci[3] < beta_1 & beta_1 < pois_ci[4])
  }, silent=TRUE)

  if (class(try_pois)=="try-error"){
    pois_cover <- c(FALSE, FALSE)
  }

  # NB
  try_nb <- try({
  nb_model <- glm.nb(y ~ x, data = df)
  nb_ci <- c(nb_model$coef[1] + qnorm(0.025) * sqrt(vcov(nb_model)[1, 1]),
             nb_model$coef[1] + qnorm(0.975) * sqrt(vcov(nb_model)[1, 1]),
```

```
              nb_model$coef[2] + qnorm(0.025) * sqrt(vcov(nb_model)[2, 2]),
              nb_model$coef[2] + qnorm(0.975) * sqrt(vcov(nb_model)[2, 2]))
  nb_cover <- c(nb_ci[1] < beta_0 & beta_0  < nb_ci[2],
                nb_ci[3] < beta_1 & beta_1 < nb_ci[4])
  }, silent=TRUE)

  if (class(try_nb)=="try-error"){
    nb_cover <- c(FALSE, FALSE)
  }

  # Quasi
  try_quasi <- try({
  quasi_model <- glm(y ~ x, data=df, family="quasipoisson")
  quasi_ci <- c(quasi_model$coef[1] + qnorm(0.025) * sqrt(vcov(quasi_model)[1, 1]),
                quasi_model$coef[1] + qnorm(0.975) * sqrt(vcov(quasi_model)[1, 1]),
                quasi_model$coef[2] + qnorm(0.025) * sqrt(vcov(quasi_model)[2, 2]),
                quasi_model$coef[2] + qnorm(0.975) * sqrt(vcov(quasi_model)[2, 2]))
  quasi_cover <- c(quasi_ci[1] < beta_0 & beta_0 < quasi_ci[2],
                   quasi_ci[3] < beta_1 & beta_1 < quasi_ci[4])
  }, silent=TRUE)

  if (class(try_quasi)=="try-error"){
    quasi_cover <- c(FALSE, FALSE)
  }

  # Sandwich

  try_sandwich <- try({

  sandcov = vcov(quasi_model) * ((n-1)/n)

  sandwich_ci <- c(pois_model$coef[1] + qnorm(0.025) * sqrt(sandcov[1, 1]),
                   pois_model$coef[1] + qnorm(0.975) * sqrt(sandcov[1, 1]),
                   pois_model$coef[2] + qnorm(0.025) * sqrt(sandcov[2, 2]),
                   pois_model$coef[2] + qnorm(0.975) * sqrt(sandcov[2, 2]))
  sandwich_cover <- c(sandwich_ci[1] < beta_0 & beta_0 < sandwich_ci[2],
                      sandwich_ci[3] < beta_1 & beta_1 < sandwich_ci[4]) },
  silent=TRUE)

  if (class(try_sandwich)=="try-error"){
    sandwich_cover <- c(FALSE, FALSE)
  }

  return(matrix(c(pois_cover, nb_cover, quasi_cover, sandwich_cover), nrow = 8, ncol = 1))
}

simulation <- function(b, n, replications = 1e3){

  result <- replicate(replications, fitting(b, n, replications))
  result_all <- apply(result, 1, mean)

  return(result_all)
}
```

```
create_df <- function(result, b, n){

  data = data.frame(matrix(c(1:8), nrow = 8))
  models <- c("pois", "pois", "nb", "nb", "quasi", "quasi", "sandwich", "sandwich")
  beta <- c("b0", "b1", "b0", "b1", "b0", "b1", "b0", "b1")

  data['result'] = result
  data['b'] = b
  data['n'] = n
  data['models'] = models
  data['beta'] = beta

  data = data[,-1]

  return(data)
}

data = data.frame()

for (i in b){
  for (j in n){

    result = simulation(b = i, n = j)
    data_temp = create_df(result, i, j)
    data = rbind(data, data_temp)
  }
}

ggplot(data=data, aes(x=n, y=result, col=models))+geom_line()
+ facet_grid(cols=vars(beta),rows=vars(b))
```

Q2

```
length1 <- c(2.247,2.640,2.842,2.908,3.099,3.126,3.245,3.328,3.355,
    3.383,3.572,3.581,3.681)
length10 <- c(1.901,2.132,2.203,2.228,2.257,2.35,2.361,2.396,2.397,
    2.445,2.454,2.454, 2.474)
length20 <- c(1.312,1.314,1.479,1.552,1.700,1.803,1.861,1.865,1.944,1.958,
    1.966,1.997,2.006)
length50 <- c(1.339,1.434,1.549,1.574,1.589,1.613,1.746,1.753,1.764, 1.807,
    1.812,1.840,1.852)

mle <- c(1/mean(length1), 1/mean(length10), 1/mean(length20),1/mean(length50))
sds <- sqrt(mle^2/13)

par(mfrow = c(2, 2))

qqplot(x=qexp(ppoints(1000)), y=length1 , ylim=c(1,5),
       main="Exponential Q-Q Plot Length 1",
       xlab="Theoretical Quantiles",ylab="Sample Quantiles")
qqline(length1, distribution=qexp,col="blue", lty=2)
```

```
qqplot(x=qexp(ppoints(1000)), y=length10 , ylim=c(1,5),
       main="Exponential Q-Q Plot Length 10",
       xlab="Theoretical Quantiles",ylab="Sample Quantiles")
qqline(length10, distribution=qexp,col="blue", lty=2)

qqplot(x=qexp(ppoints(1000)), y=length20 , ylim=c(1,5),
       main="Exponential Q-Q Plot Length 20",
       xlab="Theoretical Quantiles",ylab="Sample Quantiles")
qqline(length20, distribution=qexp,col="blue", lty=2)

qqplot(x=qexp(ppoints(1000)), y=length50 , ylim=c(1,5),
       main="Exponential Q-Q Plot Length 50",
       xlab="Theoretical Quantiles",ylab="Sample Quantiles")
qqline(length50, distribution=qexp,col="blue", lty=2)

y_bar <- c(mean(length1), mean(length10), mean(length20), mean(length50))
s2 <- c(sd(length1)^2, sd(length10)^2, sd(length20)^2, sd(length50)^2)
alpha <- s2/y_bar^2
quasi_var <- alpha * mle^2 / 13
sqrt(quasi_var)

alpha
quasi_var

sandwich_var = (13-1)/13 * quasi_var
sandwich_var

quasi_sd <- sqrt(quasi_var)
sand_sd <- sqrt(sandwich_var)
q <- rbind(mle, sds, alpha, quasi_sd, sand_sd)

colnames(q) <- c("length1", "length10", "length20", "length50")
q


weibull <- function(eta,data) {
  n <- length(data)
  alpha = ((1/n*sum(data^eta))^(1/eta))
  return(n/eta - n*log(alpha) + sum(log(data)) -
           sum(log(data/alpha)*(data/alpha)^eta))
}

eta1 <- uniroot(weibull, c(-100,100), data=length1)$root
eta10 <- uniroot(weibull, c(-100,100), data=length10)$root
eta20 <- uniroot(weibull, c(-100,100), data=length20)$root
eta50 <- uniroot(weibull, c(-100,100), data=length50)$root

alpha1 <- mean(length1^eta1)^(1/eta1)
alpha10 <- mean(length10^eta10)^(1/eta10)
alpha20 <-mean(length20^eta20)^(1/eta20)
alpha50 <- mean(length50^eta50)^(1/eta50)

var_gen <- function(y, eta, alpha){
```

```
  n <- length(y)

  var <- matrix(c(n/eta^2 + sum(log(y/alpha)^2*(y/alpha)^eta),
           n/alpha - eta/alpha*sum(log(y/alpha)*(y/alpha)^eta) -
             1/alpha*sum((y/alpha)^eta),
           n/alpha - eta/alpha*sum(log(y/alpha)*(y/alpha)^eta) -
             1/alpha*sum((y/alpha)^eta),
           eta*(eta+1)/alpha^(2+eta)*sum(y^eta) - eta*n/alpha^2 ),
         nrow = 2, ncol = 2)
  return(var)
}

res <- rbind(
  c(1,  eta1, sqrt(diag(solve(var_gen(length1,eta1,alpha1))))[1], alpha1,
    sqrt(diag(solve(var_gen(length1,eta1,alpha1))))[2]),
  c(10,  eta10, sqrt(diag(solve(var_gen(length10,eta10,alpha10))))[1], alpha10,
    sqrt(diag(solve(var_gen(length10,eta10,alpha10))))[2]),
  c(20,  eta20, sqrt(diag(solve(var_gen(length20,eta20,alpha20))))[1], alpha20,
    sqrt(diag(solve(var_gen(length20,eta20,alpha20))))[2]),
  c(50,  eta50, sqrt(diag(solve(var_gen(length50,eta50,alpha50))))[1], alpha50,
    sqrt(diag(solve(var_gen(length50,eta50,alpha50))))[2])

)


colnames(res) <- c("length", "eta", "eta se", "alpha", "alpha se")
res
```