

COMPoissonReg Demo

October 12, 2018

COM Poisson Regression

Generate some data.

```
library(COMPoissonReg, quietly = TRUE)
set.seed(1235)

n <- 400
x <- runif(n, 1, 4)
X <- model.matrix(~ x)
beta.true <- c(1, 0.5)
lambda.true <- exp(X %*% beta.true)
nu.true <- 0.75
y <- rcmp(n, lambda = lambda.true, nu = nu.true)
dat <- data.frame(y = y, x = x)
```

Fit the model. The package notices that we specify only an intercept for ν , and also displays an estimate for ν directly by using the appropriate transformation.

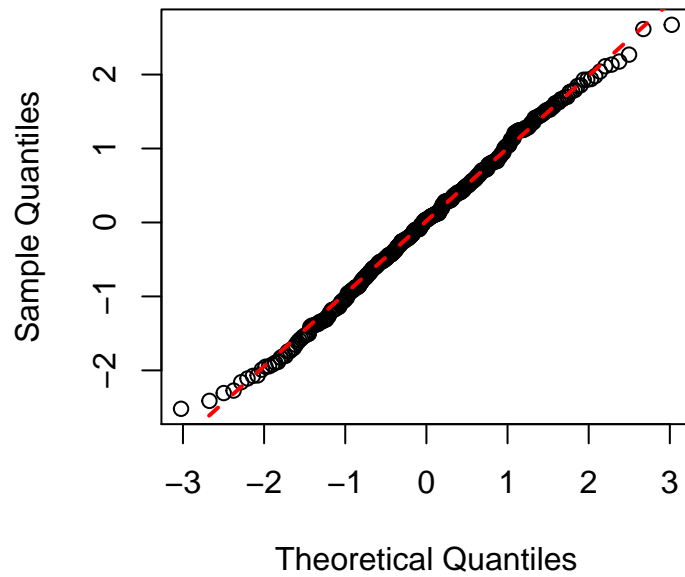
```
cmp.out <- glm.cmp(y ~ x, data = dat)
print(cmp.out)

## CMP coefficients
##           Estimate      SE z.value  p.value
## X:(Intercept)   1.0264 0.0879 11.6822 1.573e-31
## X:x              0.5225 0.0387 13.5175 1.233e-41
## S:(Intercept)  -0.2503 0.0726 -3.4462 0.0005684
## --
## Transformed intercept-only parameters
##      Estimate      SE
## nu    0.7785 0.0566
## --
## Chi-squared test for equidispersion
## X^2 = 13.0129, df = 1, p-value = 3.0935e-04
## --
## Elapsed Sec: 1.08   Sample size: 400   SEs via Hessian
## LogLik: -1224.2619   AIC: 2454.5238   BIC: 2466.4982
## Optimization Method: L-BFGS-B   Converged status: 0
## Message: CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH
```

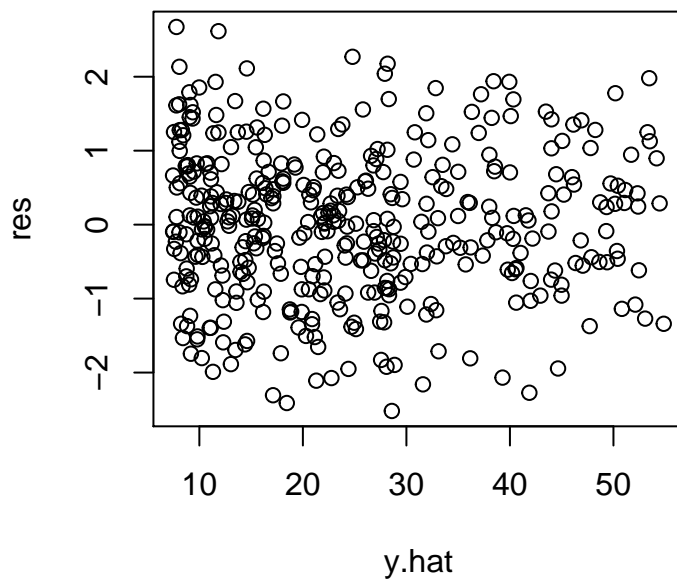
Plot the randomized quantile residuals.

```
y.hat <- predict(cmp.out)
res <- resid(cmp.out, type = "quantile")
qqnorm(res); qqline(res, lty = 2, col = "red", lwd = 2)
```

Normal Q-Q Plot



```
plot(y.hat, res)
```



Zero-Inflated COM Poisson Regression

Generate some data.

```
library(COMPoissonReg, quietly = TRUE)
set.seed(1235)
```

```
n <- 400
x <- runif(n, 1, 4)
X <- model.matrix(~ x)
S <- matrix(1, n, 1)
```

```

W <- model.matrix(~ x)
beta.true <- c(1, 2)
gamma.true <- 1
zeta.true <- c(0.05, -1)
lambda.true <- exp(X %*% beta.true)
nu.true <- exp(S %*% gamma.true)
p.true <- plogis(W %*% zeta.true)

y <- rzicmp(n, lambda = lambda.true, nu = nu.true, p = p.true)
dat <- data.frame(y = y, x = x)

```

Fit the model. The package notices that we specify only an intercept for ν , and also displays an estimate for ν directly by using the appropriate transformation.

```

options(COMPPoissonReg.optim.method = 'BFGS')
zicmp.out <- glm.cmp(y ~ x, formula.nu = ~ 1, formula.p = ~ x, data = dat)
print(zicmp.out)

```

```

## ZICMP coefficients
##              Estimate      SE z.value  p.value
## X:(Intercept)   1.1249 0.1508   7.4584 8.756e-14
## X:x              2.0405 0.1566  13.0265 8.646e-39
## S:(Intercept)   1.0324 0.0761  13.5646 6.498e-42
## W:(Intercept)   0.8494 0.5020   1.6921 0.09062
## W:x            -1.3670 0.2479  -5.5140 3.507e-08
## --
## Transformed intercept-only parameters
##      Estimate      SE
## nu      2.8077 0.2137
## --
## Chi-squared test for equidispersion
## X^2 = 384.5367, df = 1, p-value = 1.2802e-85
## --
## Elapsed Sec: 1.17   Sample size: 400   SEs via Hessian
## LogLik: -854.2464   AIC: 1718.4929   BIC: 1738.4502
## Optimization Method: BFGS

```

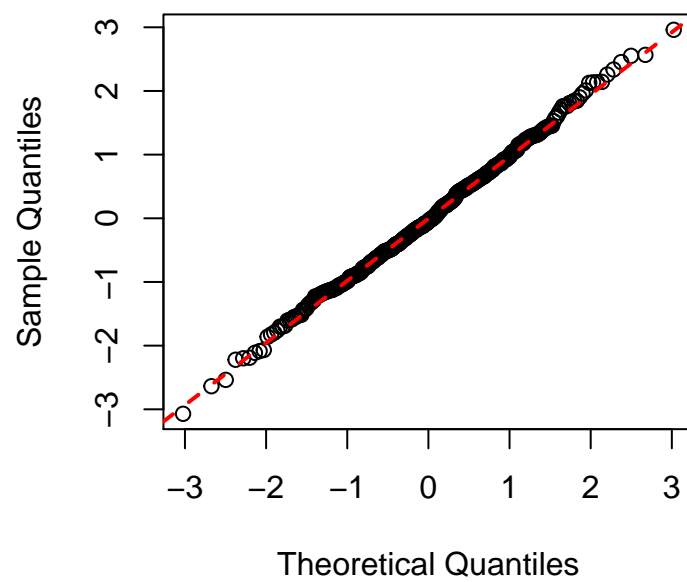
Plot the randomized quantile residuals.

```

y.hat <- predict(zicmp.out)
res <- resid(zicmp.out, type = "quantile")
qqnorm(res); qqline(res, lty = 2, col = "red", lwd = 2)

```

Normal Q-Q Plot



```
plot(y.hat, res)
```

