example

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.1.2
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(splines)
library(doParallel)
```

```
## Warning: package 'doParallel' was built under R version 4.1.2
```

```
## Loading required package: foreach
```

```
## Warning: package 'foreach' was built under R version 4.1.2
```

```
## Loading required package: iterators
```

```
## Warning: package 'iterators' was built under R version 4.1.2
```

```
## Loading required package: parallel
```

```
library(foreach)
library(tidyr)
```

```
## Warning: package 'tidyr' was built under R version 4.1.2
```

```
library(MASS)
```

```
## Warning: package 'MASS' was built under R version 4.1.2
```

```
##
## Attaching package: 'MASS'
```

```
## The following object is masked from 'package:dplyr':
##
##     select
```

```
library(nnet)
```

```
## Warning: package 'nnet' was built under R version 4.1.2
```

```
library(car)
```

```
## Loading required package: carData
```

```
## Warning: package 'carData' was built under R version 4.1.2
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:dplyr':
##
##     recode
```

```
delta <- 0.2
m <- 1000
n <- 100
B <- 50
dof = 5

l = 0.4
p = 0.2
r <- runif(m)
    alpha <- if_else(r < (1- p), 0,
                     if_else(r < (1- p/2), runif(m, -l-delta, -l), runif(m, l,l+delta))
    )
    r <- runif(m)
    beta <- if_else(r < (1- p), 0,
                    if_else(r < (1- p/2), runif(m, -l-delta, -l), runif(m, l,l+delta))
    )
    tp <- abs(alpha) > 0
    tn <- alpha == 0

    ## Generate X,Z


    Z <- matrix(rnorm(n*3), nrow = n)
    rho = 1
    softMax <- function(x){
    expx <- exp(rho* x)
    return(expx/sum(expx))
    }
    u = colMeans(Z)
    prob = softMax(u)
    X = sample(1:3, size = n, replace = T, prob = prob)
    X = as.factor(X)
    ##Generate Y
```

```
    Y = matrix(nrow = m, ncol = n)
    for(j in 1:m)
    {
      Y[j,] <- rpois(n, lambda = exp(alpha[j]*as.numeric(X) + beta[j]*(Z%*%c(1,0.5,0.2))))

    }

    j = 3
    Y = Y[j,]
fit2 = glm(Y ~ X, family = poisson(link = log), control = glm.control(maxit = 100))
fit1 = glm(Y ~ X + Z, family = poisson(link = log), control = glm.control(maxit = 100))
t2 = anova(fit2)$F[1]
t1 = anova(fit1)$F[1]
fit1
```

```
##
## Call:  glm(formula = Y ~ X + Z, family = poisson(link = log), control = glm.control(maxit = 100))
##
## Coefficients:
## (Intercept)           X2           X3           Z1           Z2           Z3
##     0.58097      0.30395      1.03805     -0.04642      0.03165     -0.08449
##
## Degrees of Freedom: 99 Total (i.e. Null);  94 Residual
## Null Deviance:       216.8
## Residual Deviance: 153.3     AIC: 419.3
```

```
anova(fit1)
```

```
## Analysis of Deviance Table
##
## Model: poisson, link: log
##
## Response: Y
##
## Terms added sequentially (first to last)
##
##
##      Df Deviance Resid. Df Resid. Dev
## NULL                    99     216.84
## X     2   60.254        97     156.59
## Z     3    3.324        94     153.27
```

```
anova(fit2)
```

```
## Analysis of Deviance Table
##
## Model: poisson, link: log
##
## Response: Y
##
## Terms added sequentially (first to last)
##
```

```
##
##      Df Deviance Resid. Df Resid. Dev
## NULL                    99     216.84
## X     2   60.254        97     156.59
```

```
t1
```

```
## NULL
```

```
t2
```

```
## NULL
```

Best I can do is use the Deviance.