V-inflated Poisson count regression

Motivation

We want to model the distribution of groups size in parrot populations. These animals tend to form smaller or larger groups, but groups of size 2 are also often observed as a result of mating pairs.

The count distribution is characterized with a spike at 2, and by the absence of 0s due to group size being conditional on having >0 birds to consider it a group.

We start developing a general V-Inflated Poisson (VIP) model, then we add the >0 condition. Simulations are done to check the estimating procedure.

Maximum likelihood

Let Y be a random variable, and y are observations, V is the count value that has some extra probability mass ($V = 0$ is the ZIP model), $f(y; \lambda)$ is the Poisson density ($f(y; \lambda) = e^{-\lambda} \frac{\lambda^y}{y!}$).

The V-Inflated density can be written as $P(Y = y) = \phi I(Y = V) + (1 - \phi) f(y; \lambda)$ which is $\phi + (1 - \phi) f(V; \lambda)$ when $Y = V$ and $(1 - \phi) f(y; \lambda)$ otherwise.

Functions

```r
vip <-
function(Y, X, Z, V=0,
offsetx, offsetz, weights, linkz="logit",
truncate=FALSE, hessian=TRUE, method="Nelder-Mead", ...) {
    if (missing(Y))
        stop("C'mon, you must have some data?!")
    if (truncate && any(Y < 1))
        stop("Y must be >0 when truncate=TRUE")
    n <- length(Y)
    id0 <- Y == V
    id1 <- !id0
    if (missing(X)) {
        X <- matrix(1, n, 1)
        colnames(X) <- "(Intercept)"
    }
    if (missing(Z)) {
        Z <- matrix(1, n, 1)
        colnames(Z) <- "(Intercept)"
    }
    kx <- ncol(X)
    kz <- ncol(Z)
    if (missing(offsetx))
        offsetx <- 0
    if (missing(offsetz))
        offsetz <- 0
    if (missing(weights))
        weights <- rep(1, n)
    linkinvx <- poisson("log")$linkinv
    linkinvz <- binomial(linkz)$linkinv
    good.num.limit <- c(.Machine$double.xmin, .Machine$double.xmax)^(1/3)

    ## VIP model full likelihood
    nll_VIP_ML <- function(parms) {
        mu <- as.vector(linkinvx(X %*% parms[1:kx] + offsetx))
        phi <- as.vector(linkinvz(Z %*% parms[(kx + 1):(kx + kz)] + offsetz))
        loglik0 <- log(phi + (1 - phi) * dpois(V, lambda = mu, log = FALSE))
        loglik1 <- log(1 - phi) + dpois(Y, lambda = mu, log = TRUE)
        loglik <- sum(weights[id0] * loglik0[id0]) + sum(weights[id1] * loglik1[id1])
        if (!is.finite(loglik) || is.na(loglik))
            loglik <- -good.num.limit[2]
        -loglik
    }
    ## 0-truncated VIP model full likelihood
    nll_VIP_TR <- function(parms) {
        mu <- as.vector(linkinvx(X %*% parms[1:kx] + offsetx))
        phi <- as.vector(linkinvz(Z %*% parms[(kx + 1):(kx + kz)] + offsetz))
        loglik0 <- log(phi + (1 - phi) * dpois(V, lambda = mu, log = FALSE) / (1-exp(-mu)))
        loglik1 <- log((1 - phi) * dpois(Y, lambda = mu, log = FALSE) / (1-exp(-mu)))
        loglik <- sum(weights[id0] * loglik0[id0]) + sum(weights[id1] * loglik1[id1])
        if (!is.finite(loglik) || is.na(loglik))
            loglik <- -good.num.limit[2]
        -loglik
    }

    opt <- optim(rep(0, kx+kz),
        if (truncate) nll_VIP_TR else nll_VIP_ML,
        hessian=hessian, method=method, ...)
    par <- opt$par
    names(par) <- c(paste0("P_", colnames(X)), paste0("V_", colnames(Z)))
    vc <- if (hessian)
        solve(opt$hessian) else matrix(NA, length(par), length(par))
    dimnames(vc) <- list(names(par), names(par))
    out <- list(call=match.call(),
        coefficients=par, loglik=-opt$value, vcov=vc, nobs=n,
        truncate=truncate)
    class(out) <- "vip"
    out
}
vcov.vip <- function(object, ...) object$vcov
logLik.vip <- function (object, ...)
    structure(object$loglik, df = object$nobs - length(object$coef),
        nobs = object$nobs, class = "logLik")
summary.vip <- function (object, ...) {
    k <- length(object$coefficients)
    coefs <- coef(object)
    se <- sqrt(diag(vcov(object)))
    tstat <- coefs/se
    pval <- 2 * pnorm(-abs(tstat))
    coefs <- cbind(coefs, se, tstat, pval)
    colnames(coefs) <- c("Estimate", "Std. Error", "z value", "Pr(>|z|)")
```

```r
    coefs <- coefs[1:k, , drop = FALSE]
    rownames(coefs) <- names(coef(object))
    out <- list(call = object$call, coefficients=coefs, loglik = object$loglik,
        bic=BIC(object), truncate=object$truncate)
    class(out) <- "summary.vip"
    return(out)
}
print.summary.vip <- function (x, digits, ...)
{
    if (missing(digits))
        digits <- max(3, getOption("digits") - 3)
    cat("\nCall:", deparse(x$call,
        width.cutoff = floor(getOption("width") * 0.85)), "", sep = "\n")
    cat("V-Inflated", if (x$truncate) "(Zero-Truncated)" else "", "Poisson Model\n\n")
    cat(paste("Coefficients:\n", sep = ""))
    printCoefmat(x$coefficients, digits = digits, signif.legend = FALSE)
    if (!any(is.na(array(x$coefficients)))) {
        if (getOption("show.signif.stars") & any(x$coefficients[,4] < 0.1))
            cat("---\nSignif. codes: ", "0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1", "\n")
    }
    cat("\nLog-likelihood:", formatC(x$loglik, digits = digits),
        "\nBIC =", formatC(x$bic, digits = digits), "\n")
    cat("\n")
    invisible(x)
}
confint.vip <-
function (object, parm, level = 0.95, ...)
{
    cf <- coef(object)
    pnames <- names(cf)
    if (missing(parm)) {
        parm <- pnames
    } else {
        if (is.numeric(parm))
            parm <- pnames[parm]
    }
    a <- (1 - level)/2
    a <- c(a, 1 - a)
    pct <- paste(format(100 * a, trim = TRUE, scientific = FALSE, digits = 3), "%", sep="")
    ci <- array(NA, dim = c(length(parm), 2), dimnames = list(parm, pct))
    fac <- qnorm(a)
    ses <- sqrt(diag(vcov(object, model, type)))
    ci[] <- cf[parm] + ses[parm] %o% fac
    ci
}
```

Simple case

```r
set.seed(123)
n <- 1000
lam <- 2 # poisson mean, can be a vector of length n
phi <- 0.4 # V-inflation probability, can be a vector of length n
```

```r
V <- 2 # V is the count value, can be 0, 2, etc
y <- y0 <- rpois(n, lam)
a <- rbinom(n, 1, phi)
y[a > 0] <- V
table(Poisson=y0, Vinflated=y)
```

```
##        Vinflated
## Poisson   0   1   2   3   4   5   6   8
##       0  81   0  51   0   0   0   0   0
##       1   0 151 126   0   0   0   0   0
##       2   0   0 274   0   0   0   0   0
##       3   0   0  65 112   0   0   0   0
##       4   0   0  39   0  43   0   0   0
##       5   0   0  12   0   0  29   0   0
##       6   0   0   6   0   0   0   9   0
##       7   0   0   1   0   0   0   0   0
##       8   0   0   0   0   0   0   0   1
```

```r
mod <- vip(Y=y, V=2)
summary(mod)
```

```
##
## Call:
## vip(Y = y, V = 2)
##
## V-Inflated  Poisson Model
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## P_(Intercept)  0.70472    0.02909  24.224  < 2e-16 ***
## V_(Intercept) -0.33900    0.08824  -3.842 0.000122 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log-likelihood: -1345
## BIC =  9585
```

```r
cbind(True=c(log_lam=log(lam), logit_phi=qlogis(phi)),
      Est=coef(mod))
```

```
##                 True        Est
## log_lam    0.6931472  0.7047243
## logit_phi -0.4054651 -0.3389963
```

Covariates for the non-V part

```r
set.seed(123)
n <- 10000
x <- rnorm(n)
df <- data.frame(x=x)
X <- model.matrix(~x, df)
beta <- c(-0.5,-0.5) # Intercept and beta values for covariate
lam <- exp(X %*% beta) # poisson mean, can be a vector of length n
phi <- 0.4 # V-inflation probability, can be a vector of length n
```

```
V <- 2 # V is the count value, can be 0, 2, etc
y <- y0 <- rpois(n, lam)
a <- rbinom(n, 1, phi)
y[a > 0] <- V
table(Poisson=y0, Vinflated=y)
```

```
##        Vinflated
## Poisson    0    1    2    3    4    5    6    7    8
##       0 3182    0 2131    0    0    0    0    0    0
##       1    0 1981 1137    0    0    0    0    0    0
##       2    0    0 1088    0    0    0    0    0    0
##       3    0    0  118  226    0    0    0    0    0
##       4    0    0   40    0   57    0    0    0    0
##       5    0    0   14    0    0   17    0    0    0
##       6    0    0    1    0    0    0    3    0    0
##       7    0    0    2    0    0    0    0    1    0
##       8    0    0    1    0    0    0    0    0    1
```

```
mod <- vip(Y=y, X=X, V=2)
summary(mod)
```

```
##
## Call:
## vip(Y = y, X = X, V = 2)
##
## V-Inflated  Poisson Model
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## P_(Intercept) -0.45313    0.02037  -22.24   <2e-16 ***
## P_x           -0.49231    0.01664  -29.58   <2e-16 ***
## V_(Intercept) -0.48770    0.02483  -19.64   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log-likelihood: -1.133e+04
## BIC = 1.147e+05
```

```
cbind(True=c(beta=beta, logit_phi=qlogis(phi)),
      Est=coef(mod))
```

```
##                 True        Est
## beta1     -0.5000000 -0.4531273
## beta2     -0.5000000 -0.4923100
## logit_phi -0.4054651 -0.4876957
```

Methods

```
coef(mod)
```

```
## P_(Intercept)           P_x V_(Intercept)
##    -0.4531273    -0.4923100    -0.4876957
```

```
vcov(mod)
```

```
##                 P_(Intercept)           P_x V_(Intercept)
## P_(Intercept)  0.0004151059  1.815322e-04 -1.454395e-04
## P_x            0.0001815322  2.769780e-04 -5.339019e-05
## V_(Intercept) -0.0001454395 -5.339019e-05  6.165031e-04
```

```
summary(mod)
```

```
##
## Call:
## vip(Y = y, X = X, V = 2)
##
## V-Inflated  Poisson Model
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## P_(Intercept) -0.45313    0.02037  -22.24   <2e-16 ***
## P_x           -0.49231    0.01664  -29.58   <2e-16 ***
## V_(Intercept) -0.48770    0.02483  -19.64   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log-likelihood: -1.133e+04
## BIC = 1.147e+05
```

```
confint(mod)
```

```
##                     2.5%      97.5%
## P_(Intercept) -0.4930599 -0.4131947
## P_x           -0.5249290 -0.4596910
## V_(Intercept) -0.5363606 -0.4390308
```

```
nobs(mod)
```

```
## [1] 10000
```

```
logLik(mod)
```

```
## 'log Lik.' -11332.89 (df=9997)
```

```
AIC(mod)
```

```
## [1] 42659.77
```

```
BIC(mod)
```

```
## [1] 114741.5
```

Zero-truncated VIP

We can truncate counts to be larger than 0. We also need $V > 0$ (for $V = 0$ case, look into ZIP or conditional Poisson model). Conceptually, the V-Inflation follows the 0-truncation (because we cannot observe 0, real truncated distribution).

The 0-truncated PDF is $P(Y = y \mid Y > 0) = \frac{P(Y=y)}{1-P(Y=0)}$. The 0-truncated V-Inflated density is $P(Y = y \mid Y > 0, V > 0) = \phi I(Y = V) + (1 - \phi)\frac{f(y;\lambda)}{1-f(0;\lambda)}$. This can be achieved in the `vip` call by the argument `truncate=TRUE`.

Here we use covariates for both the V and non-V part.

```r
set.seed(1)
n <- 1000
x <- rnorm(n)
z <- runif(n, -1, 1)
df <- data.frame(x=x, z=z)
X <- model.matrix(~x, df)
Z <- model.matrix(~z, df)
beta <- c(-0.5, -0.5)
alpha <- c(0, 0.5)
lam <- exp(X %*% beta)
phi <- plogis(Z %*% alpha)
V <- 2 # V is the count value, cannot be 0
y <- y0 <- rpois(n, lam)
a <- rbinom(n, 1, phi)
keep <- y0>0
y <- y[keep] # conditioning (i.e. exclude 0s)
y0 <- y0[keep]
X <- X[keep,]
Z <- Z[keep,]
y[a[keep] > 0] <- V
table(Poisson=y0, Vinflated=y)
```

```
##        Vinflated
## Poisson   1   2   3   4   6
##       1 155 141   0   0   0
##       2   0 127   0   0   0
##       3   0  21  16   0   0
##       4   0   4   0   7   0
##       5   0   2   0   0   0
##       6   0   0   0   0   1
```

```r
mod <- vip(Y=y, X=X, Z=Z, V=2, truncate=TRUE)
summary(mod)
```

```
##
## Call:
## vip(Y = y, X = X, Z = Z, V = 2, truncate = TRUE)
##
## V-Inflated (Zero-Truncated) Poisson Model
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## P_(Intercept) -0.50814    0.14803  -3.433 0.000598 ***
## P_x           -0.57344    0.10170  -5.639 1.71e-08 ***
## V_(Intercept)  0.02131    0.12572   0.170 0.865387
## V_z            0.47041    0.20691   2.273 0.022999 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Log-likelihood: -384.1
## BIC =  3664
```

```r
cbind(True=c(beta=beta, alpha=alpha),
      Est=coef(mod))
```

```
##         True         Est
## beta1  -0.5 -0.50813933
## beta2  -0.5 -0.57343540
## alpha1  0.0  0.02131236
## alpha2  0.5  0.47040670
```