Package: detect (via r-universe)

September 23, 2024

Type Package

Title Analyzing Wildlife Data with Detection Error

Version 0.5-0

Date 2024-05-21

Author Peter Solymos [cre, aut]
(<https://orcid.org/0000-0001-7337-1740>), Monica Moreno [aut], Subhash R. Lele [aut]

Maintainer Peter Solymos <psolymos@gmail.com>

Depends R (>= 2.13.0), Formula, stats4, pbapply

Imports Matrix

Suggests dclone, dcmle

Description Models for analyzing site occupancy and count data models with detection error, including single-visit based models (Lele et al. 2012 <doi:10.1093/jpe/rtr042>, Moreno et al. 2010 <doi:10.1890/09-1073.1>, Solymos et al. 2012 <doi:10.1002/env.1149>, Denes et al. 2016 <doi:10.1111/1365-2664.12818>), conditional distance sampling and time-removal models (Solymos et al. 2013 <doi:10.1111/2041-210X.12106>, Solymos et al. 2018 <doi:10.1650/CONDOR-18-32.1>). Package development was supported by the Alberta Biodiversity Monitoring Institute and the Boreal Avian Modelling Project.

License GPL-2

URL https://github.com/psolymos/detect

BugReports https://github.com/psolymos/detect/issues

LazyLoad yes

LazyData true

Repository https://psolymos.r-universe.dev

RemoteUrl https://github.com/psolymos/detect

RemoteRef HEAD

RemoteSha 9e5fd779e3288a7620b3796bb0a56e423c961bdc

Contents

`detect-package` *Analyzing Wildlife Data with Detection Error*

Description

Models for analyzing site occupancy and count data models with detection error, including single-visit based models, conditional distance sampling and time-removal models. Package development was supported by the Alberta Biodiversity Monitoring Institute and the Boreal Avian Modelling Project.

Details

`svocc`: single visit occupancy model (Lele et al. 2012, Moreno et al. 2010).

`svabu`: single visit abundance model based on conditional maximum likelihood (Solymos et al. 2012, Solymos and Lele 2016, Denes et al. 2016).

`cmulti`: conditional multinomial maximum likelihood estimation for removal and (point count) distance sampling, efficient and flexible setup for varying methodologies (Solymos et al. 2013, Solymos et al. 2018).

Author(s)

Peter Solymos, Monica Moreno, Subhash R Lele

Maintainer: Peter Solymos <solymos@ualberta.ca>

References

Denes, F., Solymos, P., Lele, S. R., Silveira, L. & Beissinger, S. 2017. Biome scale signatures of land use change on raptor abundance: insights from single-visit detection-based models. *Journal of Applied Ecology*, **54**, 1268–1278. <doi:10.1111/1365-2664.12818>

Lele, S.R., Moreno, M. and Bayne, E. 2012. Dealing with detection error in site occupancy surveys: What can we do with a single survey? *Journal of Plant Ecology*, **5(1)**, 22–31. <doi:10.1093/jpe/rtr042>

Moreno, M. and Lele, S. R. 2010. Improved estimation of site occupancy using penalized likelihood. *Ecology*, **91**, 341–346. <doi:10.1890/09-1073.1>

Solymos, P., Lele, S. R. and Bayne, E. 2012. Conditional likelihood approach for analyzing single visit abundance survey data in the presence of zero inflation and detection error. *Environmetrics*, **23**, 197–205. <doi:10.1002/env.1149>

Solymos, P., Matsuoka, S. M., Bayne, E. M., Lele, S. R., Fontaine, P., Cumming, S. G., Stralberg, D., Schmiegelow, F. K. A. & Song, S. J., 2013. Calibrating indices of avian density from non-standardized survey data: making the most of a messy situation. *Methods in Ecology and Evolution*, **4**, 1047–1058. <doi:10.1111/2041-210X.12106>

Solymos, P., Lele, S. R. 2016. Revisiting resource selection probability functions and single-visit methods: clarification and extensions. *Methods in Ecology and Evolution*, **7**, 196–205. <doi:10.1111/2041-210X.12432>

Solymos, P., Matsuoka, S. M., Cumming, S. G., Stralberg, D., Fontaine, P., Schmiegelow, F. K. A., Song, S. J., and Bayne, E. M., 2018. Evaluating time-removal models for estimating availability of boreal birds during point-count surveys: sample size requirements and model complexity. *Condor*, **120**, 765–786. <doi:10.1650/CONDOR-18-32.1>

Supporting info, including a tutorial for the QPAD method: https://github.com/psolymos/QPAD/tree/master/inst/doc/v2

`AUC` *AUC ROC plot for fitted models*

Description

Area under the receiver-operator (ROC) curve (AUC), and ROC plot methods for fitted models.

Usage

```
AUC(object, ...)
rocplot(x, ...)
```

Arguments

`object`, `x`	a fitted model object
`...`	other arguments

Value

AUC returns AUC value for a model, or a data frame with values for more models.

rocplot returns the values used for the plot invisibly, and as a side effect it draws a graph.

Author(s)

Peter Solymos and Monica Moreno

---

bootstrap *Do bootstrap and extract bootstrap results*

Description

Do bootstrap and extract bootstrap results.

Usage

```
bootstrap(object, ...)
extractBOOT(object, ...)
```

Arguments

object	a fitted model object
...	other arguments

Value

bootstrap performs bootstrap.

extractBOOT is used to extract bootstrap results.

Author(s)

Peter Solymos

cmulti *Conditional Multinomial Maximum Likelihood Estimation*

Description

Conditional Multinomial Maximum Likelihood Estimation for different sampling methodologies.

Usage

```
cmulti(formula, data, type = c("rem", "mix", "dis", "fmix"),
    inits = NULL, method = "Nelder-Mead", ...)
cmulti.fit(Y, D, X=NULL, type=c("rem", "mix", "dis", "fmix"),
    inits=NULL, method="Nelder-Mead", ...)

cmulti2.fit(Y, D1, D2, X1=NULL, X2=NULL,
    inits=NULL, method="Nelder-Mead", ...)

## S3 method for class 'cmulti'
fitted(object, ...)
## S3 method for class 'cmulti'
model.frame(formula, ...)
## S3 method for class 'cmulti'
model.matrix(object, ...)
## S3 method for class 'cmulti'
predict(object, newdata = NULL,
    type = c("link", "response"), ...)
```

Arguments

formula	formula, LHS takes 2 matrices in the form of Y | D, RHS is either 1 or some covariates, see Examples.
data	data.
type	character, one of "rem" (removal sampling, homogeneous singing rates), "mix" and "fmix" (removal sampling, heterogeneous singing rates, "mix" implies that 'phi' is constant but 'c' can vary; "fmix" implies that 'c' is constant but 'phi' can vary), "dis" (distance sampling, half-normal detection function for point counts, circular area). For the predict method it is the type of prediction required; the default is on the scale of the linear predictors; the alternative "response" is on the scale of the response variable.
Y	this contains the cell counts. cmulti.fit requires that Y is a matrix (observations x intervals), dimensions and pattern in NAs must match that of D. cmulti2.fit requires that Y is a 3-dimensional array (observations x time intervals x distance intervals), dimensions and pattern in NAs must match that of D1 and D2.
D, D1, D2	design matrices, that describe the interval endpoints for the sampling methodology, dimensions must match dimensions of Y.

X, X1, X2	design matrices, X is the matrix with covariates for the removal/distance sampling parameters. X1 is the matrix with covariates for the removal, X2 is the matrix with covariates for the distance sampling parameters.
inits	optional initial values.
method	method for optim.
object	fitted model object.
newdata	optionally, a data frame in which to look for variables with which to predict. If omitted, the fitted linear predictors are used.
...	additional options for optim.

Details

Conditional Multinomial Maximum Likelihood Estimation for different sampling methodologies.

Value

An object of class 'cmulti'.

Author(s)

Peter Solymos

References

Solymos, P., Matsuoka, S. M., Bayne, E. M., Lele, S. R., Fontaine, P., Cumming, S. G., Stralberg, D., Schmiegelow, F. K. A. & Song, S. J., 2013. Calibrating indices of avian density from non-standardized survey data: making the most of a messy situation. *Methods in Ecology and Evolution*, **4**, 1047–1058. <doi:10.1111/2041-210X.12106>

Solymos, P., Matsuoka, S. M., Cumming, S. G., Stralberg, D., Fontaine, P., Schmiegelow, F. K. A., Song, S. J., and Bayne, E. M., 2018. Evaluating time-removal models for estimating availability of boreal birds during point-count surveys: sample size requirements and model complexity. *Condor*, **120**, 765–786. <doi:10.1650/CONDOR-18-32.1>

Supporting info, including a tutorial for the QPAD method: https://github.com/psolymos/QPAD/tree/master/inst/doc/v2

Examples

```
simfun1 <- function(n = 10, phi = 0.1, c=1, tau=0.8, type="rem") {
    if (type=="dis") {
        Dparts <- matrix(c(0.5, 1, NA,
                    0.5, 1, Inf,
                    1, Inf, NA), 3, 3, byrow=TRUE)
        D <- Dparts[sample.int(3, n, replace=TRUE),]
        CP <- 1-exp(-(D/tau)^2)
    } else {
        Dparts <- matrix(c(5, 10, NA,
                    3, 5, 10,
                    3, 5, NA), 3, 3, byrow=TRUE)
```

```
        D <- Dparts[sample.int(3, n, replace=TRUE),]
        CP <- 1-c*exp(-D*phi)
    }
    k <- ncol(D)
    P <- CP - cbind(0, CP[, -k, drop=FALSE])
    Psum <- rowSums(P, na.rm=TRUE)
    PPsum <- P / Psum
    Pok <- !is.na(PPsum)
    N <- rpois(n, 10)
    Y <- matrix(NA, ncol(PPsum), nrow(PPsum))
    Ypre <- sapply(1:n, function(i) rmultinom(1, N, PPsum[i,Pok[i,]]))
    Y[t(Pok)] <- unlist(Ypre)
    Y <- t(Y)
    list(Y=Y, D=D)
}

n <- 200
x <- rnorm(n)
X <- cbind(1, x)

## removal, constant
vv <- simfun1(n=n, phi=exp(-1.5))
m1 <- cmulti(vv$Y | vv$D ~ 1, type="rem")
coef(m1)
## mixture, constant (mix and fmix are identical)
vv <- simfun1(n=n, phi=exp(-1.5), c=plogis(0.8))
m2 <- cmulti(vv$Y | vv$D ~ 1, type="mix")
coef(m2)
m2f <- cmulti(vv$Y | vv$D ~ 1, type="fmix")
coef(m2f)
## dist, constant
vv <- simfun1(n=n, tau=exp(-0.2), type="dis")
m3 <- cmulti(vv$Y | vv$D ~ 1, type="dis")
coef(m3)

## removal, not constant
log.phi <- crossprod(t(X), c(-2,-1))
vv <- simfun1(n=n, phi=exp(cbind(log.phi, log.phi, log.phi)))
m1 <- cmulti(vv$Y | vv$D ~ x, type="rem")
coef(m1)
## mixture, fixed phi, varying c
logit.c <- crossprod(t(X), c(-2,1))
vv <- simfun1(n=n, phi=exp(-1.5), c=plogis(cbind(logit.c, logit.c, logit.c)))
m2 <- cmulti(vv$Y | vv$D ~ x, type="mix")
coef(m2)
## mixture, varying phi, fixed c
log.phi <- crossprod(t(X), c(-2,-1))
vv <- simfun1(n=n, phi=exp(cbind(log.phi, log.phi, log.phi)), c=plogis(0.8))
m2f <- cmulti(vv$Y | vv$D ~ x, type="fmix")
coef(m2f)
## dist, not constant
log.tau <- crossprod(t(X), c(-0.5,-0.2))
vv <- simfun1(n=n, tau=exp(cbind(log.tau, log.tau, log.tau)), type="dis")
```

```
m3 <- cmulti(vv$Y | vv$D ~ x, type="dis")
coef(m3)

summary(m3)
coef(m3)
vcov(m3)
AIC(m3)
confint(m3)
logLik(m3)

## fitted values
plot(exp(log.tau), fitted(m3))

## prediction for new locations (type = 'rem')
ndf <- data.frame(x=seq(-1, 1, by=0.1))
summary(predict(m1, newdata=ndf, type="link"))
summary(pr1 <- predict(m1, newdata=ndf, type="response"))
## turing singing rates into probabilities requires total duration
## 5 minutes used here
psing <- 1-exp(-5*pr1)
plot(ndf$x, psing, type="l", ylim=c(0,1))

## prediction for new locations (type = 'dis')
summary(predict(m3, newdata=ndf, type="link"))
summary(pr3 <- predict(m3, newdata=ndf, type="response"))
## turing EDR into probabilities requires finite truncation distances
## r=0.5 used here (50 m)
r <- 0.5
pdet <- pr3^2*(1-exp(-r^2/pr3^2))/r^2
plot(ndf$x, pdet, type="l", ylim=c(0,1))

## joint removal-distance estimation
## is not different from 2 orthogonal estimations

simfun12 <- function(n = 10, phi = 0.1, c=1, tau=0.8, type="rem") {
    Flat <- function(x, DIM, dur=TRUE) {
        x <- array(x, DIM)
        if (!dur) {
            x <- aperm(x,c(1,3,2))
        }
        dim(x) <- c(DIM[1], DIM[2]*DIM[3])
        x
    }
    Dparts1 <- matrix(c(5, 10, NA,
                        3, 5, 10,
                        3, 5, NA), 3, 3, byrow=TRUE)
    D1 <- Dparts1[sample.int(3, n, replace=TRUE),]
    CP1 <- 1-c*exp(-D1*phi)
    Dparts2 <- matrix(c(0.5, 1, NA,
                        0.5, 1, Inf,
                        1, Inf, NA), 3, 3, byrow=TRUE)
    D2 <- Dparts2[sample.int(3, n, replace=TRUE),]
    CP2 <- 1-exp(-(D2/tau)^2)
```

```
    k1 <- ncol(D1)
    k2 <- ncol(D2)
    DIM <- c(n, k1, k2)
    P1 <- CP1 - cbind(0, CP1[, -k1, drop=FALSE])
    P2 <- CP2 - cbind(0, CP2[, -k2, drop=FALSE])
    Psum1 <- rowSums(P1, na.rm=TRUE)
    Psum2 <- rowSums(P2, na.rm=TRUE)
    Pflat <- Flat(P1, DIM, dur=TRUE) * Flat(P2, DIM, dur=FALSE)
    PsumFlat <- Psum1 * Psum2
    PPsumFlat <- Pflat / PsumFlat
    PokFlat <- !is.na(PPsumFlat)
    N <- rpois(n, 10)
    Yflat <- matrix(NA, ncol(PPsumFlat), nrow(PPsumFlat))
    YpreFlat <- sapply(1:n, function(i) rmultinom(1, N, PPsumFlat[i,PokFlat[i,]]))
    Yflat[t(PokFlat)] <- unlist(YpreFlat)
    Yflat <- t(Yflat)
    Y <- array(Yflat, DIM)
    k1 <- dim(Y)[2]
    k2 <- dim(Y)[3]
    Y1 <- t(sapply(1:n, function(i) {
        count <- rowSums(Y[i,,], na.rm=TRUE)
        nas <- rowSums(is.na(Y[i,,]))
        count[nas == k2] <- NA
        count
    }))
    Y2 <- t(sapply(1:n, function(i) {
        count <- colSums(Y[i,,], na.rm=TRUE)
        nas <- colSums(is.na(Y[i,,]))
        count[nas == k2] <- NA
        count
    }))
    list(Y=Y, D1=D1, D2=D2, Y1=Y1, Y2=Y2)
}

## removal and distance, constant
vv <- simfun12(n=n, phi=exp(-1.5), tau=exp(-0.2))
res <- cmulti2.fit(vv$Y, vv$D1, vv$D2)
res1 <- cmulti.fit(vv$Y1, vv$D1, NULL, "rem")
res2 <- cmulti.fit(vv$Y2, vv$D2, NULL, "dis")
## points estimates are identical
cbind(res$coef, c(res1$coef, res2$coef))
## standard errors are identical
cbind(sqrt(diag(res$vcov)),
    c(sqrt(diag(res1$vcov)),sqrt(diag(res2$vcov))))

## removal and distance, not constant
vv <- simfun12(n=n,
    phi=exp(cbind(log.phi, log.phi, log.phi)),
    tau=exp(cbind(log.tau, log.tau, log.tau)))
res <- cmulti2.fit(vv$Y, vv$D1, vv$D2, X1=X, X2=X)
res1 <- cmulti.fit(vv$Y1, vv$D1, X, "rem")
res2 <- cmulti.fit(vv$Y2, vv$D2, X, "dis")
```

```
## points estimates are identical
cbind(res$coef, c(res1$coef, res2$coef))
## standard errors are identical
cbind(sqrt(diag(res$vcov)),
    c(sqrt(diag(res1$vcov)),sqrt(diag(res2$vcov))))
```

`convertEDT` *Conversion between truncated and unlimited effective detection distance (EDR)*

Description

Conversion between truncated and unlimited effective detection distance (EDR).

Usage

```
convertEDR(edr, r, truncated=FALSE)
```

Arguments

`edr`	effective detection distance. In same units as `r`.
`r`	truncation distance (radius of point count). In same units as `edr`.
`truncated`	logical, see Details.

Details

`truncated = FALSE` means that `edr` is unlimited EDR, and the function returns the truncated EDR given `r`.

`truncated = TRUE` means that `edr` is truncated EDR given `r`, and the function returns the unlimited EDR.

Value

A numeric vector with converted EDR values.

Author(s)

Peter Solymos

References

Matsuoka, S. M., Bayne, E. M., Solymos, P., Fontaine, P., Cumming, S. G., Schmiegelow, F. K. A., & Song, S. A., 2012. Using binomial distance-sampling models to estimate the effective detection radius of point-counts surveys across boreal Canada. *Auk*, **129**, 268–282. <doi:10.1525/auk.2012.11190>

Solymos, P., Matsuoka, S. M., Bayne, E. M., Lele, S. R., Fontaine, P., Cumming, S. G., Stralberg, D., Schmiegelow, F. K. A. & Song, S. J., 2013. Calibrating indices of avian density from non-standardized survey data: making the most of a messy situation. *Methods in Ecology and Evolution*, **4**, 1047–1058. <doi:10.1111/2041-210X.12106>

Supporting info, including a tutorial for the above paper: https://github.com/psolymos/QPAD/tree/master/inst/doc/v2

Examples

```
convertEDR(1, 0.5, truncated=FALSE)
## should be close to 1
convertEDR(convertEDR(1, 0.5, truncated=FALSE), 0.5, truncated=TRUE)
```

`databu` *Simulated example for abundance model*

Description

Simulated example for abundance model, see code below.

Usage

```
data(databu)
```

Format

A data frame with 1000 observations on the following 11 variables.

`N` true counts

`Y` observed counts

`x1` random variables used as covariates

`x2` random variables used as covariates

`x3` random variables used as covariates

`x4` random variables used as covariates

`x5` random variables used as covariates

`x6` random variables used as covariates

`p` probability of detection

`lambda` mean of the linear predictor

`A` occupancy

`phi` zero inflation probabilities

Details

This simulated example corresponds to the Binomial - ZIP model implemented in the function `svabu`.

Source

Simulated example.

References

Solymos, P., Lele, S. R. and Bayne, E. 2012. Conditional likelihood approach for analyzing single visit abundance survey data in the presence of zero inflation and detection error. *Environmetrics*, **23**, 197–205. <doi:10.1002/env.1149>

Examples

```
data(databu)
str(databu)
## Not run:
## simulation
n <- 1000
set.seed(1234)
x1 <- runif(n,0,1)
x2 <- rnorm(n,0,1)
x3 <- runif(n,-1,1)
x4 <- runif(n,-1,1)
x5 <- rbinom(n,1,0.6)
x6 <- rbinom(n,1,0.4)
x7 <- rnorm(n,0,1)
X <- model.matrix(~ x1 + x5)
Z <- model.matrix(~ x2 + x5)
Q <- model.matrix(~ x7)
beta <- c(2,-0.8,0.5)
theta <- c(1, 2, -0.5)
phi <- 0.3
p <- drop(binomial("logit")$linkinv(Z %*% theta))
lambda <- drop(exp(X %*% beta))
A <- rbinom(n, 1, 1-phi)
N <- rpois(n, lambda * A)
Y <- rbinom(n, N, p)
databu <- data.frame(N=N, Y=Y, x1, x2, x3, x4, x5, x6, p=p, lambda=lambda, A, phi)

## End(Not run)
```

---

datocc *Simulated example for occupancy model*

Description

Simulated example for occupancy model, see code below.

Usage

```
data(datocc)
```

Format

A data frame with 1000 observations on the following 6 variables.

`Y` true occupancy

`W` observations

`x1` random variables used as covariates

`x2` random variables used as covariates

`x3` random variables used as covariates

`x4` random variables used as covariates

`p.occ` probability of occurrence

`p.det` probability of detection

Details

This simulated example corresponds to the ZI Binomial model implemented in the function svocc.

Source

Simulated example.

References

Lele, S.R., Moreno, M. and Bayne, E. (2012) Dealing with detection error in site occupancy surveys: What can we do with a single survey? *Journal of Plant Ecology*, **5(1)**, 22–31. <doi:10.1093/jpe/rtr042>

Examples

```
data(datocc)
str(datocc)
## Not run:
## simulation
n <- 1000
set.seed(1234)
x1 <- runif(n, -1, 1)
x2 <- as.factor(rbinom(n, 1, 0.5))
x3 <- rnorm(n)
x4 <- rnorm(n)
beta <- c(0.6, 0.5)
theta <- c(0.4, -0.5, 0.3)
X <- model.matrix(~ x1)
Z <- model.matrix(~ x1 + x3)
mu <- drop(X %*% beta)
nu <- drop(Z %*% theta)
p.occ <- binomial("cloglog")$linkinv(mu)
p.det <- binomial("logit")$linkinv(nu)
Y <- rbinom(n, 1, p.occ)
W <- rbinom(n, 1, Y * p.det)
datocc <- data.frame(Y, W, x1, x2, x3, x4, p.occ, p.det)

## End(Not run)
```

`hbootindex` *Hierarchical bootstrap indices*

Description

Generates hierarchical bootstrap indices.

Usage

```
hbootindex(groups, strata, B = 199)
```

Arguments

`groups`	group membership vector.
`strata`	strata, optional.
`B`	number of bootstrap iterations.

Details

Resampling with replacement with weights proportional to the number of observations in each of the group level (unique values in `groups`).

Values of `groups` within levels (unique values) of `strata` are resampled independently of other `strata` levels.

Value

A matrix with bootstrapped indices, number of columns is `B` + 1. The column is a resample without replacement (random subsets can be selected without further reshuffling). Other elements contain indices according to rules described in Details section (these also randomly reshuffled).

Author(s)

Peter Solymos

Examples

```
## equal group sizes
groups <- rep(1:4, each=5)
strata <- rep(1:2, each=10)
hbootindex(groups, strata, 3)

## unequal group sizes
groups <- groups[-c(5,9,10,11)]
strata <- strata[-c(5,9,10,11)]
hbootindex(groups, strata, 3)
```

load_BAM_QPAD *Load BAM QPAD parameter estimates and support functions*

Description

Load BAM QPAD parameter estimates and support functions.

Usage

```
load_BAM_QPAD(version)
```

Arguments

version	version of the BAM QPAD estimates. List of selection is provided if missing.

Details

The load_BAM_QPAD function in the 'detect' package is deprecated. Use the load_BAM_QPAD function 'QPAD' package instead. See https://github.com/psolymos/QPAD for more information.

Value

It returns a message.

Author(s)

Peter Solymos

References

Solymos, P., Matsuoka, S. M., Bayne, E. M., Lele, S. R., Fontaine, P., Cumming, S. G., Stralberg, D., Schmiegelow, F. K. A. & Song, S. J., 2013. Calibrating indices of avian density from non-standardized survey data: making the most of a messy situation. *Methods in Ecology and Evolution*, **4**, 1047–1058. <doi:10.1111/2041-210X.12106>

Supporting info, including a tutorial for the above paper: https://github.com/psolymos/QPAD/tree/master/inst/doc/v2

oven *Ovenbird abundances*

Description

Ovenbird abundances from BBS

Usage

```
data(oven)
```

Format

A data frame with 891 observations on the following 11 variables.

`count` observations

`route` route id

`stop` stop id within route

`pforest` proportion of forest

`pdecid` proportion of deciduous forest

`pagri` proportion of agricultural areas

`long` longitude

`lat` latitude

`observ` observer, a factor with levels `ARS` `DW` `RDW` `SVW`

`julian` Julian day

`timeday` time of day

Source

BBS, Erin Bayne (Univ. Alberta), unpublished data set used in Solymos et al. 2012.

References

Solymos, P., Lele, S. R. and Bayne, E. 2012. Conditional likelihood approach for analyzing single visit abundance survey data in the presence of zero inflation and detection error. *Environmetrics*, **23**, 197–205. <doi:10.1002/env.1149>

Examples

```
data(oven)
str(oven)
```

svabu *Single visit N-mixture abundance models*

Description

Binomial-Poisson, Binomial-NegBin, Binomial-ZIP, and Binomial-ZINB models with single visit.

Usage

```
svabu(formula, data, zeroinfl = TRUE, area = 1, N.max = NULL,
    inits, link.det = "logit", link.zif = "logit",
    model = TRUE, x = FALSE, distr = c("P", "NB"), ...)

svabu.fit(Y, X, Z, Q = NULL, zeroinfl = TRUE, area = 1, N.max = NULL,
    inits, link.det = "logit", link.zif = "logit", ...)
svabu_nb.fit(Y, X, Z, Q = NULL, zeroinfl = TRUE, area = 1, N.max = NULL,
    inits, link.det = "logit", link.zif = "logit", ...)

zif(x)
is.present(object, ...)
predictMCMC(object, ...)
svabu.step(object, model, trace = 1, steps = 1000,
    criter = c("AIC", "BIC"), test = FALSE, k = 2, control, ...)
```

Arguments

formula	formula of the form y ~ x | z, where y is a vector of observations, x is the set of covariates for the occurrence model, z is the set of covariates for the detection model. x can further expanded as x1 + zif(x2) into terms for the nonzero count data part (x1) and the zero inflation component (zif(x2)) using the zif special.
Y, X, Z, Q	vector of observation, design matrix for abundance model, design matrix for detection and design matrix for zero inflation model
data	data
area	area
N.max	maximum of true count values (for calculating the integral)
zeroinfl	logical, if the Binomial-ZIP model should be fitted
inits	initial values used by link{optim}
link.det, link.zif	link function for the detection and zero inflation parts of the model
model	a logical value indicating whether model frame should be included as a component of the returned value, or true state or detection model
x	logical values indicating whether the response vector and model matrix used in the fitting process should be returned as components of the returned value. For the function zif it is any object to be returned.

object	a fitted object.
trace	info returned during the procedure
steps	max number of steps
criter	criterion to be minimized (cAUC=1-AUC)
test	logical, if decrease in deviance should be tested
k	penalty to be used with AIC
control	controls for optimization, if missing taken from object
distr	character, abundance distribution: "P" for Poisson, "NB" for Negative Binomial.
...	other arguments passed to the functions

Details

See Examples.

The right hand side of the formula must contain at least one continuous (i.e. non discrete/categorical) covariate. This is the necessary condition for the single-visit method to be valid and parameters to be identifiable. See References for more detailed description.

The Binomial-Poisson model is the single visit special case of the *N*-mixture model proposed by Royle (2004) and explained in Solymos et a. (2012) and Solymos and Lele (2016).

Value

An object of class 'svabu'.

Author(s)

Peter Solymos and Subhash Lele

References

Royle, J. A. 2004. *N*-Mixture Models for Estimating Population Size from Spatially Replicated Counts. *Biometrics*, **60(1)**, 108–115. <doi:10.1111/j.0006-341X.2004.00142.x>

Solymos, P., Lele, S. R. and Bayne, E. 2012. Conditional likelihood approach for analyzing single visit abundance survey data in the presence of zero inflation and detection error. *Environmetrics*, **23**, 197–205. <doi:10.1002/env.1149>

Solymos, P., Lele, S. R. 2016. Revisiting resource selection probability functions and single-visit methods: clarification and extensions. *Methods in Ecology and Evolution*, **7**, 196–205. <doi:10.1111/2041-210X.12432>

Denes, F., Solymos, P., Lele, S. R., Silveira, L. & Beissinger, S. 2017. Biome scale signatures of land use change on raptor abundance: insights from single-visit detection-based models. *Journal of Applied Ecology*, **54**, 1268–1278. <doi:10.1111/1365-2664.12818>

Examples

```
data(databu)

## fit BZIP and BP models
m00 <- svabu(Y ~ x1 + x5 | x2 + x5, databu[1:200,])

## print method
m00
## summary: CMLE
summary(m00)
## coef
coef(m00)
coef(m00, model="sta") ## state (abundance)
coef(m00, model="det") ## detection
coef(m00, model="zif") ## zero inflation (this is part of the 'true state'!)

## Not run:
## Diagnostics and model comparison

m01 <- svabu(Y ~ x1 + x5 | x2 + x5, databu[1:200,], zeroinfl=FALSE)
## compare estimates (note, zero inflation is on the logit scale!)
cbind(truth=c(2,-0.8,0.5, 1,2,-0.5, plogis(0.3)),
"B-ZIP"=coef(m00), "B-P"=c(coef(m01), NA))

## fitted
plot(fitted(m00), fitted(m01))
abline(0,1)

## compare models
AIC(m00, m01)
BIC(m00, m01)
logLik(m00)
logLik(m01)
## diagnostic plot
plot(m00)
plot(m01)

## Bootstrap

## non parametric bootstrap
## - initial values are the estimates
m02 <- bootstrap(m00, B=25)
attr(m02, "bootstrap")
extractBOOT(m02)
summary(m02)
summary(m02, type="cmle")
summary(m02, type="boot")
## vcov
vcov(m02, type="cmle")
vcov(m02, type="boot")
vcov(m02, model="sta")
vcov(m02, model="det")
```

```
## confint
confint(m02, type="cmle") ## Wald-type
confint(m02, type="boot") ## quantile based
## parametric bootstrap
simulate(m00, 5)
m03 <- bootstrap(m00, B=5, type="param")
extractBOOT(m03)
summary(m03)

## Model selection

m04 <- svabu(Y ~ x1 + x5 | x2 + x5 + x3, databu[1:200,], phi.boot=0)
m05 <- drop1(m04, model="det")
m05
m06 <- svabu.step(m04, model="det")
summary(m06)
m07 <- update(m04, . ~ . | . - x3)
m07

## Controls

m00$control
getOption("detect.optim.control")
getOption("detect.optim.method")
options("detect.optim.method"="BFGS")
m08 <- svabu(Y ~ x1 + x5 | x2 + x5, databu[1:100,])
m08$control ## but original optim method is retained during model selection and bootstrap
## fitted models can be used to provide initial values
options("detect.optim.method"="Nelder-Mead")
m09 <- svabu(Y ~ x1 + x5 | x2 + x5, databu[1:100,], inits=coef(m08))

## Ovenbirds dataset

data(oven)
ovenc <- oven
ovenc[, c(4:8,10:11)][] <- lapply(ovenc[, c(4:8,10:11)], scale)
moven <- svabu(count ~ pforest | observ + pforest + julian + timeday, ovenc)
summary(moven)
drop1(moven, model="det")
moven2 <- update(moven, . ~ . | . - timeday)
summary(moven2)
moven3 <- update(moven2, . ~ . | ., zeroinfl=FALSE)
summary(moven3)
BIC(moven, moven2, moven3)

## End(Not run)
```

---

svocc *ZI Binomial model with single visit*

Description

ZI Binomial model with single visit

Usage

```
svocc(formula, data, link.sta = "cloglog", link.det = "logit",
    penalized = FALSE, method = c("optim", "dc"), inits,
    model = TRUE, x = FALSE, ...)
svocc.fit(Y, X, Z, link.sta = "cloglog", link.det = "logit",
    penalized = FALSE, auc = FALSE, method = c("optim", "dc"),
    inits, ...)

extractMLE(object, ...)
svocc.step(object, model, trace = 1, steps = 1000,
    criter = c("AIC", "BIC", "cAUC"), test = FALSE, k = 2,
    control, ...)
```

Arguments

formula	formula of the form y ~ x | z, where y is a vector of observations, x is the set of covariates for the occurrence model, z is the set of covariates for the detection model
Y, X, Z	vector of observation, design matrix for occurrence model, and design matrix for detection model
data	data
link.sta, link.det	link function for the occurrence (true state) and detection model
penalized	logical, if penalized likelihood estimate should be computed
method	optimization or data cloning to be used as optimization
inits	initial values
model	a logical value indicating whether model frame should be included as a component of the returned value, or true state or detection model
x	logical values indicating whether the response vector and model matrix used in the fitting process should be returned as components of the returned value
auc	logical, if AUC should be calculated
object	a fitted model object
trace	info returned during the procedure
steps	max number of steps
criter	criterion to be minimized (cAUC=1-AUC)
test	logical, if decrease in deviance should be tested
k	penalty to be used with AIC
control	controls for optimization, if missing taken from object
...	other arguments passed to the functions

Details

See Examples.

The right hand side of the formula must contain at least one continuous (i.e. non discrete/categorical) covariate. This is the necessary condition for the single-visit method to be valid and parameters to be identifiable. See References for more detailed description.

Value

An object of class 'svocc'.

Author(s)

Peter Solymos and Monica Moreno

References

Lele, S.R., Moreno, M. and Bayne, E. 2012. Dealing with detection error in site occupancy surveys: What can we do with a single survey? *Journal of Plant Ecology*, **5(1)**, 22–31. <doi:10.1093/jpe/rtr042>

Moreno, M. and Lele, S. R. 2010. Improved estimation of site occupancy using penalized likelihood. *Ecology*, **91**, 341–346. <doi:10.1890/09-1073.1>

Solymos, P., Lele, S. R. 2016. Revisiting resource selection probability functions and single-visit methods: clarification and extensions. *Methods in Ecology and Evolution*, **7**, 196–205. <doi:10.1111/2041-210X.12432>

Examples

```
data(datocc)
## MLE
m00 <- svocc(W ~ x1 | x1 + x3, datocc)
## PMLE
m01 <- svocc(W ~ x1 | x1 + x3, datocc, penalized=TRUE)

## print
m00
## summary
summary(m00)
## coefficients
coef(m00)
## state (occupancy) model estimates
coef(m00, "sta")
## detection model estimates
coef(m00, "det")
## compare estimates
cbind(truth=c(0.6, 0.5, 0.4, -0.5, 0.3),
mle=coef(m00), pmle=coef(m01))

## AIC, BIC
AIC(m00)
BIC(m00)
## log-likelihood
```

```
logLik(m00)
## variance-covariance matrix
vcov(m00)
vcov(m00, model="sta")
vcov(m00, model="det")
## confidence intervals
confint(m00)
confint(m00, model="sta")
confint(m00, model="det")

## fitted values
## (conditional probability of occurrence given detection history:
## if W=1, fitted=1,
## if W=0, fitted=(phi*(1-delta)) / ((1-delta) + phi * (1-delta))
summary(fitted(m00))
## estimated probabilities: (phi*(1-delta)) / ((1-delta) + phi * (1-delta))
summary(m00$estimated.probabilities)
## probability of occurrence (phi)
summary(m00$occurrence.probabilities)
## probability of detection (delta)
summary(m00$detection.probabilities)

## Not run:
## model selection
m02 <- svocc(W ~ x1 | x3 + x4, datocc)
m03 <- drop1(m02, model="det")
## dropping one term at a time, resulting change in AIC
m03
## updating the model
m04 <- update(m02, . ~ . | . - x4)
m04
## automatic model selection
## part of the model (sta/det) must be specified
m05 <- svocc.step(m02, model="det")
summary(m05)

## nonparametric bootstrap
m06 <- bootstrap(m01, B=25)
attr(m06, "bootstrap")
extractBOOT(m06)
summary(m06, type="mle")
summary(m06, type="pmle") ## no SEs! PMLE!!!
summary(m06, type="boot")
## vcov
#vcov(m06, type="mle") ## this does not work with PMLE
vcov(m06, type="boot") ## this works
## confint
confint(m06, type="boot") ## quantile based

## parametric bootstrap
## sthis is how observations are simulated
head(simulate(m01, 5))
m07 <- bootstrap(m01, B=25, type="param")
```

```
extractBOOT(m07)
summary(m07)

data(oven)
ovenc <- oven
ovenc[, c(4:8,10:11)][] <- lapply(ovenc[, c(4:8,10:11)], scale)
ovenc$count01 <- ifelse(ovenc$count > 0, 1, 0)
moven <- svocc(count01 ~ pforest | julian + timeday, ovenc)
summary(moven)
drop1(moven, model="det")
moven2 <- update(moven, . ~ . | . - timeday)
summary(moven)

BIC(moven, moven2)
AUC(moven, moven2)
rocplot(moven)
rocplot(moven2, col=2, add=TRUE)

## End(Not run)
```

Index