Package 'lllcrc'

July 2, 2014

Type Package

Title Local Log-linear Models for Capture-Recapture

Version 1.1

Date 2013-12-16

Author Zach Kurtz

Maintainer Zach Kurtz <zkurtz@gmail.com>

Description Applies local log-linear capture-recapture models (LLLMs) a for closed populations, as described in the ongoing doctoral thesis work of Zachary Kurtz in the department of statistics at Carnegie Mellon University. The method is relevant when there are 3-4 capture occasions,with auxiliary covariates available for all capture occasions. As part of estimating the number of missing population units, the method estimates the ``rate of missingness" as it varies over the covariate space. In addition,user-friendly functions are provided to recreate (approximately) the method of Zwane and Van der Heijden (2004), which applied the VGAM package in a way that is closely related to LLLMs.

License GPL-2

Imports data.table, combinat, VGAM, plyr

NeedsCompilation no

Repository CRAN

Date/Publication 2014-03-12 07:21:07

R topics documented:

`lllcrc-package` *Local Log-linear Models for Capture-Recapture*

Description

Applies local log-linear capture-recapture models (LLLMs) a for closed populations, as described in the ongoing doctoral thesis work of Zachary Kurtz in the department of statistics at Carnegie Mellon University. The method is relevant when there are 3-4 capture occasions, with auxiliary covariates available for all capture occasions. As part of estimating the number of missing population units, the method estimates the "rate of missingness" as it varies over the covariate space. In addition, user-friendly functions are provided to recreate (approximately) the method of Zwane and van der Heijden (2004), which applied the `VGAM` package in a way that is closely related to LLLMs.

Details

Package:	lllcrc
Type:	Package
Version:	1.1
Date:	2013-12-16
License:	GPL-2 <http://cran.r-project.org/web/licenses/GPL-2>

Author(s)

Zach Kurtz Maintainer: Zach <zkurtz@gmail.com>

References

Kurtz ZT (2013). "Smooth Post-Stratification for Multiple Capture-Recapture." *arXiv preprint arXiv:1302.0890*.

Examples

```
#########################################################################
## Simulate a dataset much like the French multiple sclerosis data
## (El Adssi et. al., 2012)
set.seed(100)
pop = poptop(k = 3, n=5000, heters = "french.1", covs = "age.sex.zip")
dt = formatdata(x = pop, y = "y", cont.vars = c("age"), categ.vars = c("sex", "zip"))
# Notice the new variable notation:
attributes(dt)$var.keys


#########################################################################
## Implement basic log-linear models (LLM)
```

```
llmod = flat.IC(pop, rasch = FALSE, ic = "AICc", adjust = TRUE)
boot.list = list(n.reps = 25, est = llmod$pred, pop = pop, ic = "AICc", rasch = FALSE,
adjust = TRUE, averaging = FALSE)
llmod.boots = llcrc.flat.boots(boot.list)
hist(llmod.boots); abline(v = llmod$pred, lwd = 3, col = "red")

#########################################################################
## Use local log-linear models (LLLM)
# LLL Step 1: Use your excellent researcher intuition to set a bandwidth for each dimension:
x.covs = names(dt)[substr(names(dt), 1,1) == "x"]
bw.key = data.frame(bw = matrix(NA, nrow = length(x.covs), ncol = 1))
row.names(bw.key) = x.covs
bw.key[1,1] = 5
bw.key[2:7,1] = sqrt(2)
bw.key = as.matrix(bw.key)
# LLL Step 2: Obtain point estimates and bootstrap the variance for LLLMs:
#  (For more-thorough results, increase the number n.reps of bootstrap replications)
lmod = lllcrc(dat = dt, kfrac = 0.2, bw = bw.key, ic = "AICc", round.vars = c("x.con.1"),
rounding.scale = 20, boot.control = list(n.reps = 10, seed = 13))
summary(lmod)
# View rates of missingness by category
rates.by.category(x=lmod, reference.levels = c("x.dis.1.F","x.dis.2.a"))
# View a pointwise C.I. for the rate of missingness across age in a chosen category:
ci = extract.CI(mod = lmod, probs = c(0.025, 0.975), cont.var = "x.con.1",
selection = c("x.dis.1.F", "x.dis.2.a"))
plot(ci$x, ci$pi0, ylim = c(0,max(ci$upper)), bty = "n", xlab = "age", type = "l",
lwd = 2, ylab = "Rate of missingness", main = "LLLM")
lines(ci$x, ci$lower, lty = 2)
lines(ci$x, ci$upper, lty = 2)
mtext("females in zip code 57")
# Alternatively, view the relative capture pattern frequencies in a stacked form,
# along with the confidence interval:
plot(lmod, selection = c("x.dis.1.F", "x.dis.2.a"), main = "LLLM")

## Not run:
#########################################################################
## Use a VGAM for CRC
# For better results, increase the number n.reps of bootstrap replications
vg = vgam.crc(dat = dt, sdf = 4, round.vars = c("x.con.1"), rounding.scale = 10,
boot.control = list(n.reps=10, seed = 4))
summary(vg)
# View rates of missingness by category
rates.by.category(x=vg, reference.levels = c("x.dis.1.F","x.dis.2.a"))
# View a pointwise C.I. for the rate of missingness across age in a chosen category:
ci = extract.CI(mod = vg, probs = c(0.025, 0.975), cont.var = "x.con.1",
selection = c("x.dis.1.F", "x.dis.2.a"))
plot(ci$x, ci$pi0, ylim = c(0,max(ci$upper)), bty = "n", xlab = "age", type = "l",
lwd = 2, ylab = "Rate of missingness", main = "VGAM for CRC")
lines(ci$x, ci$lower, lty = 2)
lines(ci$x, ci$upper, lty = 2)
mtext("females in zip code 57")
# Alternatively, view the relative capture pattern frequencies in a stacked form,
# along with the confidence interval:
```

```
plot(vg, selection = c("x.dis.1.F", "x.dis.2.a"), main = "VGAM for CRC")

## End(Not run)
```

`age.sex.zip` *Simulate CRC data with age, sex, and zip code*

Description

Generate a fake data set with covariates age, sex, and zip code.

Usage

```
age.sex.zip(n)
```

Arguments

`n`	The number of population units.

Details

This simulation is vaguely inspired by the French dataset on multiple sclerosis prevalence (El Adssi et. al., 2012). It is used to create a three-list capture-recapture system with unit-level independence between lists. Capture probabilities are constant (for simplicity!) across lists, but not across population units. The heterogeneity is a function of the covariates: age, sex, and zip code. We code zip here as 'a', 'b', 'c', 'd'.

Value

`pop`	A data.frame containing the covariates and capture histories associated with each observed unit.

Author(s)

Zach Kurtz

`AICc.vgam` *Compute the AICc for a VGAM model*

Description

The AICc, or Akaike Information Criterion with small sample correction, is computed for a VGAM model.

Usage

```
AICc.vgam(mod)
```

Arguments

`mod`	A VGAM model object

Details

See references. Burnham and Anderson (2002). This correction to the AIC is derived under some assumption of normality, so it's validity may depend on there being large counts for each capture pattern so that the multivariate normal approximation to the multinomial is good.

The AIC methods for VGAM purport to have the AICc option for the `vglm`, but this did not seem to exist for the `vgam` as of October 2013.

Value

`AICc`	The value of the AICc

Author(s)

Zach Kurtz

References

http://stats.stackexchange.com/questions/69723/two-different-formulas-for-aicc

apply.ic.fit *Select an LLLM at each point*

Description

Select LLLMs for each row of the input data.

Usage

```
apply.ic.fit(ydens, models, ess, mct, ic, cell.adj, averaging, loud = TRUE)
```

Arguments

ydens	A matrix with 2^k-1 columns, one for each capture pattern. Each row sums to 1; these are empirical capture pattern probabilities.
models	A list of character vectors, with each vector containing column names from the associated log-linear design matrix. For example, see the output of make.hierarchical.term.sets().
ess	A vector of effective sample sizes, one for each row of ydens.
mct	The number of population units that were observed for each row of ydens.
ic	The chosen information criterion. Currently implemented: "AIC", "AICc", "BIC", "BICpi".
cell.adj	Logical: TRUE means that the cell adjustment of Evans and Bonet (1995) is applied.
averaging	Logical: TRUE means that the information criterion weights are used to do model averaging, locally.
loud	Logical: TRUE means that the progress is noted by printing the number of the row of ydens currently being processed.

Details

See Kurtz (2013). Each row of ydens corresponds to a covariate vector, and contains a local average of multinomial capture pattern outcomes across nearby points. apply.ic.fit applies the function ic.fit at each row. The vector of local effective sample sizes is crucial, and is specified in the ess argument.

Value

lll	An object of class "lllcrc"

Author(s)

Zach Kurtz

References

Kurtz (2013)

apply.local.ml *Fit LLLMs*

Description

Fit a previously selected log-linear model at each covariate vector.

Usage

```
apply.local.ml(dens, ml.meth = NULL, predictors = NULL,
  design.style = "standard")
```

Arguments

dens	Same as ydens in apply.ic.fit.
ml.meth	A model specification, in case predictors is NOT specified. Options include "indep" and "indep.mono".
predictors	A character vector of predictors of the form "c1", "c2" for main effects, or "c12" for an interaction.
design.style	The kind of model design; Rasch or not Rasch

Details

This function is closely related to apply.ic.fit, with the difference being that the user supplies the log-linear model to be applied locally (instead of using an information criterion to select a potentially different model at each point).

Value

A vector of estimated rates of missingness corresponding to each point.

Author(s)

Zach Kurtz

as.num *Conversion to numeric*

Description

Simply an abbreviation for as.numeric(as.character())

Usage

```
as.num(x)
```

Arguments

x	Something to be converted to numeric

Author(s)

Zach Kurtz

---

captures *Simulating captures*

Description

Simulating captures as part of a CRC experiment.

Usage

```
captures(n, pop, k, heters, covs)
```

Arguments

n	The population size.
pop	The data.frame with a basic template for the population.
k	The number of lists.
heters	A character string to denote a specific heterogeneity structure. Currently only french.1 is implemented.
covs	A character string to denote the covariates to be included. Currently only NULL and age.sex.zip are implemented.

Details

Given (possibly) some covariates specified by `covs` and a heterogeneity function specified by `heters`, conduct a capture-recapture experiment, and return the capture-recapture data.

Value

A data.frame with capture-recapture data

Author(s)

Zach Kurtz

`construct.vgam` *Make a VGAM model*

Description

A wrapper function to fit a CRC model using the vgam command in the VGAM package.

Usage

```
construct.vgam(sdf, constr.cols = NULL, constraints, dat)
```

Arguments

`sdf`	Nonlinear degrees of freedom. See VGAM documentation for more on what this means.
`constr.cols`	A vector of column indices.
`constraints`	A transformed log-linear design matrix. The submatrix indicated by `constr.cols` is the desired argument to the VGAM command to impose the corresponding log-linear model on the data.
`dat`	A CRC dataset in standard form (i.e., as output of `format.data`)

Details

Even as a wrapper function, this still primarily a workhorse function for `vgam.crc`.

Value

A `vgam` model object

Author(s)

Zach Kurtz

References

See references for `vgam.crc`.

extract.CI *Use bootstrap output to get CI*

Description

Given bootstrap output of functions such as `lllcrc` or `vgam.crc`, produce confidence intervals corresponding to a selection of covariate vectors.

Usage

```
extract.CI(mod, probs = c(0.025, 0.975), cont.var = "x.con.1",
  selection = NULL)
```

Arguments

`mod`	The object that is output by a `lllcrc` or `vgam.crc`.
`probs`	A vector c(a,b), with 0<a<b<1, marking the upper and lower probability bounds of the desired confidence interval
`cont.var`	The name of one continuous covariate to serve as the primary index of the ordered set of confidence intervals (one CI for each covariate vector)
`selection`	A set of categorical covariate names. CIs will be produced only for covariate vectors that have a one in every position corresponding to the elements of `selection`.

Value

A data frame that includes `cont.var`, the corresponding point estimate of the rate of missingness `pi0`, and the lower and upper bounds of the associated confidence intervals.

Author(s)

Zach Kurtz

flat.IC *Select an LLM*

Description

Without using covariates (i.e., with capture probabilities assumed flat over the covariate space), select the best log-linear model for the marginal contingency table of capture pattern counts.

Usage

```
flat.IC(pop, models = make.hierarchical.term.sets(k = attributes(dt)$k),
  rasch = FALSE, ic = "AICc", adjust = FALSE, averaging = FALSE)
```

Arguments

pop	A data.frame containing CRC data as output of formatdata.
models	A list of models – or an expression that returns a list of models – to be considered in local model search. The default is NULL, and in this case make.hierarchical.term.sets(k = attributes(dat)$k) is called to generate all hierarchical models that include all main effects.
rasch	Logical: Should the Rasch model (most basic version, Darroch et. al. 1993) be considered, in addition to standard models? FALSE by default.
ic	Character string specifying the information criterion to use for model selection. Currently AIC, AICc, BIC, and BICpi are implemented.
adjust	Logical: Should we adjust the cells as in Evans and Bonett (1995)?
averaging	Logical: Should we use model averaging based on the information criterion scores?

Value

pred	The point estimate of the population size
form	The log-linear terms in the chosen model

Author(s)

Zach Kurtz

References

Fienberg SE (1972). "The Multiple Recapture Census for Closed Populations and Incomplete 2^k Contingency Tables." *Biometrika*, **59**(3), pp. 591.

---

flat.log.linear *Fit an LLM*

Description

Fit a log-linear model. This is a wrapper function for our own variant of the glm function, pirls.

Usage

```
flat.log.linear(pop, model.terms, rasch = FALSE)
```

Arguments

pop	The CRC data as a data frame.
model.terms	The columns of the standard design matrix to include in the model. For example, "c1", "c2" for main effects, and "c12" for interactions.
rasch	Logical: Is this the Rasch model?

Details

Maximum likelihood estimation is used, conditioning on the observed population as if it were the full population.

Value

A vector of log-linear coefficients. The first coefficient is the intercept, and the rest correspond (in order) with the `model.terms` argument

Author(s)

Zach Kurtz

`formatdata` *Format the CRC data*

Description

Put a CRC dataset into a standard form that is understood by functions like `lllcrc`.

Usage

```
formatdata(x, y, cont.vars = NULL, categ.vars = NULL)
```

Arguments

`x`	The data frame of CRC data, such as the output of the simulation function `poptop`.
`y`	A character string indicating which column of `x` contains the capture patterns.
`cont.vars`	A character vector of variable names for continuous variables.
`categ.vars`	A character vector of variable names for categorical variables.

Value

A data frame that is an expanded version of the argument `x`. Variables are renamed using an "x.con...." or "x.dis...." for continuous and discrete variables, respectively. Each discrete/categorical variable with L levels is expanded into L separate binary variables. Several attributes are included to describe the data format.

Author(s)

Zach Kurtz

`french.1` *A fake dataset, french.1*

Description

Generate a fake dataset, for simulation purposes, that is somewhat similar to the data on multiple sclerosis in the Lorraine Region of France as in El Adssi et. al. (2012).

Usage

```
french.1(pop, n, k)
```

Arguments

`pop`	A data frame containing a template for the population.
`n`	The size of the target population.
`k`	The number of lists.

Details

This function is the guts for a capture-recapture experiment that is specified by the `heters` argument in the function `poptop`; with a minor revision ot the function `captures`, you could create your own guts function for a very different capture-recapture experiment that sits within the same coding framework.

Value

A data frame, which gets passed back up to the main function `poptop` for a bit of cleaning.

Author(s)

Zach Kurtz

References

El Adssi H, Debouverie M and Guillemin F (2012). "Estimating the prevalence and incidence of multiple sclerosis in the Lorraine region, France, by the capture-Recapture method." *Multiple Sclerosis Journal*, **18**(9), pp. 1244-1250.

`get.IC` *Compute an information criterion*

Description

Given the fitted parameter values for a log-linear model, compute an information criterion.

Usage

```
get.IC(predictors, ddat, ic, beta)
```

Arguments

`predictors`	A character vector of predictors of the form "c1", "c2" for main effects, or "c12" for an interaction. The predictors to be used in a log-linear model. For example, "c1", "c2" for main effects, or "c12" for an interaction.
`ddat`	A data frame that is the design matrix for a log-linear model.
`ic`	The information criterion to be computed. Currently the AIC, AICc, BIC, BICpi are implemented.
`beta`	The vector of log-linear coefficients that were previously estimated.

Details

Computes the conditional multinomial likelihood and uses it to compute the specified information criterion

Value

The value of the information criterion

Author(s)

Zach Kurtz

References

Thesis of Zach Kurtz (2014), Carnegie Mellon University, Statistics

Anderson DR and Burnham KP (1999). "Understanding information criteria for selection among capture-recapture or ring recovery models." *Bird Study*, **46**(S1), pp. S14-S21.

ic.all *Compute an IC for several LLMs*

Description

Given several sets of log-linear terms, compute the IC corresponding to each model.

Usage

```
ic.all(models, ddat, ic, normalized = normalized)
```

Arguments

models	A list of character vectors, with each vector containing column names from the associated log-linear design matrix. For example, see the output of make.hierarchical.term.sets().
ddat	The log-linear design matrix.
ic	The information criterion, such as AIC, AICc, BIC, or BICpi.
normalized	Logical: TRUE means that beta0 will be adjusted so that the log-linear model corresponds to cell probabilities instead of expected cell counts.

Value

A matrix with as many rows as there are entries in models. The columns contain the point estimates of the population size, the information criterion scores, and the information criterion weights for all the models, which sum to one

Author(s)

Zach Kurtz

ic.fit *Select and fit an LLM*

Description

Use an information criterion to select a local log-linear model

Usage

```
ic.fit(densi, models, N, ic, averaging = averaging, normalized = TRUE,
  rasch = FALSE)
```

Arguments

densi	A matrix with one row and 2^k-1 column containing cell counts or empirical cell probabilities corresponding to all the possible capture patterns.
models	A list of character vectors, with each vector containing column names from the associated log-linear design matrix. For example, see the output of make.hierarchical.term.sets().
N	If you multiply densi by N and then sum over the resulting vector, you should get the effective sample size.
ic	The information criterion, such as AIC, AICc, BIC, or BICpi.
averaging	Logical: TRUE means that we use information criterion scores to do model averaging.
normalized	Logical: TRUE means that beta0 will be adjusted so that the log-linear model corresponds to cell probabilities instead of expected cell counts.
rasch	Logical: TRUE means that the Rasch model is a candidate.

Details

Just like flat.IC except that it is designed to take in a local average instead of a full capture-recapture dataset

Value

pred	Estimated rate of missingness for the selected model
form	Formula of the selected model

Author(s)

Zach Kurtz

---

ic.wghts *Information criterion model weights*

Description

Compute model weights according to the information criterion scores of each model.

Usage

```
ic.wghts(scores)
```

Arguments

scores	The information criterion scores.

Details

The formula is quite simple: Identify the smallest (best) score among the various models. Subtract this minimum value from all of the scores, and call the resulting set of scores s. Compute exp(-0.5 s) for all the scores, and normalize the resulting vector to obtain the vector of model weights

Value

A vector of weights, which can be interpreted (loosely) as the relative desireability of the models corresponding to the weights

Author(s)

Zach Kurtz

References

Burnham and Anderson (2002)

---

init.pop *Set up a fake population*

Description

Simply initialized a data frame to represent the CRC data in a simulation experiment.

Usage

```
init.pop(n, covs = NULL)
```

Arguments

n	The population size.
covs	A string specifying a function for generating covariate values. By default, there are no covariates. The only non-default method at present is `french.1`.

Details

This is just a workhorse function, typically called by `poptop`.

Value

A data.frame representing a population, possibly with covariates. If there are no covariates, it is just a single column labeled 'id'.

Author(s)

Zach Kurtz

initialize.u.vec *Initialize log-linear parameters*

Description

A tool for setting up the simulations of `llm.sim`.

Usage

```
initialize.u.vec(k)
```

Arguments

`k`	The number of lists to be modeled

Value

A vector of log-linear parameters, all initialized to zero, corresponding to the columns of the most general design matrix (but no Rasch terms).

Author(s)

Zach Kurtz

llcrc.flat.boots *Bootstrapping LLMs*

Description

For a log-linear model, bootstrap estimates of the corresponding estimates in a way that incorporates the uncertainty due to model selection.

Usage

```
llcrc.flat.boots(boot.list)
```

Arguments

`boot.list`	A list of control arguments to direct the bootstrap procedure. See example.

Details

Implements the nonparametric bootstrap of Norris and Pollock (1996), 'Method 2'. The results can be used to create empirical confidence intervals.

Value

A vector of bootstrap estimates.

Author(s)

Zach Kurtz

References

Norris JL and Pollock KH (1996). "Including model uncertainty in estimating variances in multiple capture studies." *Environmental and Ecological Statistics*, **3**, pp. 235-244.

`lllcrc` *Local log-linear models (LLLMs) for capture-recapture (CRC)*

Description

Fits local log-linear models. Each distinct covariate vector gets its own model. To reduce the number of models, some rounding of continuous covariates is done first.

Usage

```
lllcrc(dat, kfrac, models = NULL, ic = "BICpi", bw = NULL,
  averaging = FALSE, cell.adj = TRUE, round.vars = NULL,
  rounding.scale = 0.01, boot.control = NULL)
```

Arguments

`dat`	Capture-recapture data, as output of `formatdata`
`kfrac`	The approximate fraction of the data that is included in the support of the kernel for the local averages.
`models`	A list of models – or an expression that returns a list of models – to be considered in local model search. The default is `NULL`, and in this case `make.hierarchical.term.sets(k = attributes(dat)$k)` is called to generate all hierarchical models that include all main effects.
`ic`	The information criterion for selection of local log-linear models. The default, BICpi, appears in Hook and Regal (1997).
`bw`	A single-column matrix with rownames that match the covariate names in `dat`. The values in the column are scalars that are used in constructing distances between covariate vectors. Raw differences are divided by the corresponding scalars before being squared in the context of a Euclidean metric. Defaults to a column of 1's.
`averaging`	Logical: Should model averaging be done for each local model?
`cell.adj`	Logical: Whether to adjust the cells as in Evans and Bonett (1994). TRUE by default.

round.vars	See micro.post.stratify, which is called within lllcrc.
rounding.scale	See micro.post.stratify, which is called within lllcrc.
boot.control	A list of control parameters for bootstrapping the sampling distribution of the estimator(s). By default, there is no bootstrapping.

Details

The key implementation of the thesis of Kurtz 2013, Carnegie Mellon University

Value

est	A point estimate of the population size
llform	The set of log-linear terms
dat	The output of function micro.post.stratify, with estimated local rates of missingness appended as an extra column labeled pi0. In addition, mct (multinomial cell count) gives the number of observed units with that distinct covariate vector, and cpi0 (cumulative number missing) gives the the product of pi0 with mct, such that summing over this vectorized product is exactly the Horvitz-Thompson style sum in capture recapture.
ess	The local effective sample sizes that are based on the local averaging weights and used as eta_i in local model selection
hpi	The matrix of local averages
...	The output is of class lllcrc and has attributes cont.x and conteg.x, which relate the continuous and categorical variables in the model

Author(s)

Zach Kurtz

References

Kurtz ZT (2013). "Smooth Post-Stratification for Multiple Capture-Recapture." *arXiv preprint arXiv:1302.0890*.

Anderson DR and Burnham KP (1999). "Understanding information criteria for selection among capture-recapture or ring recovery models." *Bird Study*, **46**(S1), pp. S14-S21.

Fienberg SE (1972). "The Multiple Recapture Census for Closed Populations and Incomplete 2^k Contingency Tables." *Biometrika*, **59**(3), pp. 591.

Evans MA and Bonett DG (1994). "Bias Reduction for Multiple-Recapture Estimators of Closed Population Size." *Biometrics*, **50**(2), pp. 388-395.

`lllcrc.boots` *Bootstrap for LLLMs*

Description

Implement a single instance of resampling process to simulate a replicate of the `lllcrc` model.

Usage

```
lllcrc.boots(boot.list)
```

Arguments

`boot.list`	A list of control parameters for the bootstrapping process. See the example under `lllcrc-package`

Details

`vgam.crc` calls this function for each bootstrap replicate. Many replicates together can be used for empirical confidence interval estimation. The bootstrap is much like that described by Zwane 2003, but more like Method 2 in Norris and Polluck 1996, as we repeat the process of selecting log-linear terms for each bootstrap iteration, and the multinomial sampling probabilities are based on raw local estimates instead of post-log-linear modeling estimates.

Value

A list containing two vectors. The first vector, `cpi0`, gives the estimated number of missing units at each point; the second, `mct`, gives the number of units observed at each point. (Dividing the first vector by the second gives an estimate rate of missingness)

Author(s)

Zach Kurtz

References

Norris JL and Pollock KH (1996). "Including model uncertainty in estimating variances in multiple capture studies." *Environmental and Ecological Statistics*, **3**, pp. 235-244.

`llm.sim` *Simulate basic log-linear CRC experiments*

Description

Replicate and summarize the generation and log-linear analysis of data sets that are consistent with arbitrary log-linear models

Usage

```
llm.sim(n.grid = c(100, 300, 900, 2700), n.reps = 100, u.vec, p0 = NULL,
  models = NULL, ic = "BICpi", cell.adj = TRUE, averaging = FALSE,
  fixed.sample.size = FALSE)
```

Arguments

`n.grid`	A vector of positive integers, by default `c(100,300,900,2700)`. Each integer is the number of population units that are observed in a corresponding collection of simulations.
`n.reps`	The number of replicates for each integer in `n.grid`, i.e., for each population size of interest.
`u.vec`	A vector of log-linear parameters, excluding the intercept term. The length of the vector and the order of its terms must correspond to the column names of the design matrix produced by `make.design.matrix(k)`, where `k` is the number of lists.
`p0`	Optional: a number in `(0,1)`, the fraction of the population that is to be undetected. See details.
`models`	See `lllcrc`
`ic`	See `lllcrc`
`cell.adj`	See `lllcrc`
`averaging`	`lllcrc`
`fixed.sample.size`	Logical: If `TRUE`, the simulations fix the number of units that are detected, defining the true population size such that the number of units detected is equal to its expectation. If `FALSE`, the observed population size is variable, such that the integers in `n.grid` indicate only the expectations of the corresponding simulation sizes.

Details

`u.vec`, together with the constraint that the multinomial probabilities sum to 1, uniquely determines the unspecified intercept term. Specifying `p0` overdetermines the intercept term. We rectify this overspecification by adjusting all main effects by the same additive adjustment `a`, where the unique value of `a` is approximated with numerical methods.

Once the log-linear terms are fully specified, we perform multinomial draws to simulate a CRC experiment. We include the zero cell in the multinomial draw only if `fixed.sample.size = TRUE`.

On each replicate, the data log-linear model search according to the parameters `models`, `ic`, `cell.adj`, and `averaging` produces an estimate of the missing cell. The main matrix `res` of simulation results stores the ratios of the estimated missing cell over the 'true' missing cell.

Value

A list of class `llsim`, for "log-linear simulations". The list contains the set of multinomial capture pattern probabilities `p`, the matrix `res` of simulation results, and many of the arguments to the `llm.sim`.

Author(s)

Zach Kurtz

Examples

```
## Not run:
## A basic simulation with four lists.
# Begin by specifying the vector of log-linear parameters.
# The parameters must match the design matrix:
names(make.design.matrix(k=4))
u.vec = initialize.u.vec(k=4)
u.vec[5:10] = 2
## Run the simulation with an adjustment to the main effects in
# u.vec such that the probability of nondetection is 0.5.
sim = llm.sim(n.grid = c(100,300,900,2700), n.reps = 10, u.vec,
p0 = 0.5, ic = "BIC", cell.adj = FALSE)
# View the results
plot(sim)

## End(Not run)
```

---

local.ml *Maximum likelihood estimation for fixed LLLMs*

Description

A workhorse function for `apply.local.ml`. For a fixed log-linear model, get best fit using maximum local likelihood estimation.

Usage

```
local.ml(pdat, ml.meth = NULL, predictors = NULL, k)
```

Arguments

pdat	A design matrix with cell counts (possibly fractional) included in the column named "c".
ml.meth	A model specification such as "indep" or "indep.mono".
predictors	A character vector of predictors of the form "c1", "c2" for main effects, or "c12" for an interaction.
k	Number of lists.

Details

Specify exactly one of the two arguments `predictors`, `ml.meth`

Author(s)

Zach Kurtz

References

Fienberg SE (1972). "The Multiple Recapture Census for Closed Populations and Incomplete 2^k Contingency Tables." *Biometrika*, **59**(3), pp. 591.

Cormack RM (1989). "Log-linear models for capture-recapture." *Biometrics*, pp. 395-413.

---

make.design.matrix *Construct standard LLM design matrix.*

Description

Makes a design matrix for a (local) log-linear model

Usage

```
make.design.matrix(k = 3, order.max = k - 1, rasch = FALSE)
```

Arguments

k	The number of lists
order.max	The maximum number of lists to include in interaction terms
rasch	Logical: if TRUE, include a column for the square of the number of captures

Value

A design matrix as a data frame

Author(s)

Zach Kurtz

`make.hierarchical.term.sets`

Generate a universe of hierarchical models.

Description

Creates a list of all hierarchical log-linear models that include all main effects. Optionally, may includes the Rasch model if there are exactly 3 lists.

Usage

```
make.hierarchical.term.sets(k = 3, rasch = FALSE)
```

Arguments

`k`	The number of lists; must be in the set 2,3,4,5.
`rasch`	If TRUE, include the Rasch model (simplest Rasch model in Darroch et. al. 1993).

Value

A list of character vectors

Author(s)

Zach Kurtz

`make.patterns.template`

Template for capture-pattern counts

Description

Make a matrix with one row and as many columns as there are observeable capture patterns, with column names corresponding to the possible capture patterns.

Usage

```
make.patterns.template(k = 3)
```

Arguments

`k`	The number of lists

Details

By default, the matrix gets populated with uniform value summing to 1 across all the entries.

Author(s)

Zach Kurtz

`micro.post.stratify` *Collapse CRC data through micro post-stratification*

Description

Rounding continuous covariates creates "micro-post-strata" and therefore tends to reduce the number of distinct covariate vectors. After rounding, the data is collapsed so that there is exactly one row for each distinct covariate vector, and a column called `mct` (for multinomial cell count) is appended with that contains the number of records corresponding to each row.

Usage

```
micro.post.stratify(dat, round.vars = NULL, rounding.scale = NULL)
```

Arguments

`dat`	The data in a matrix form
`round.vars`	A vector of names of variables to be rounded for the purpose of collapsing the data.
`rounding.scale`	A vector of scalars that determines how much each corresponding variable in `round.vars` is to be rounded. For example, the first variable `round.vars[1]` will be divided by `rounding.scale[1]`, then rounded to the nearest whole number, and then multiplied by `rounding.scale[1]`. The net effect is to round to the nearest multiple of `rounding.scale[1]`.

Details

Continuous variables will be divided by `rounding.scale`, then rounded to the nearest whole number, and then multiplied by `rounding.scale`. The net effect is to round to the nearest multiple of `rounding.scale`

Value

Another matrix, just like the input `dat` except that there are fewer rows, data values are rounded, and there is a new column `mct`, which gives the number of data points corresponding to each row.

Author(s)

Zach Kurtz

odd.evens *Determine the even-ness of capture patterns*

Description

Return a list containing the indices of capture patterns with an even number of captures and the indices that of capture patterns with an odd number of captures

Usage

```
odd.evens(Y)
```

Arguments

Y	A vector of capture patterns, such as the output of the function patterns.possible

Author(s)

Zach Kurtz

patterns *Collapse capture events into capture patterns (strings)*

Description

For any capture-recapture data matrix with columns such as c1, c2 ... as binary indicators of capture or noncapture, running patterns on the data appends a new column y that binds c1, c2, ... together as a column of capture patterns.

Usage

```
patterns(pop, k)
```

Arguments

pop	Capture-recapture data, at an intermediate form in the inner workings of poptop
k	The number of lists

Author(s)

Zach Kurtz

patterns.possible *Generate all observable capture patterns*

Description

Returns a vector of binary strings representing all possible capture patterns excluding complete noncapture

Usage

```
patterns.possible(k)
```

Arguments

k	The number of lists

Author(s)

Zach Kurtz

pirls *Maximum likelihood for log-linear coefficients*

Description

A simplified version of `glm` that does only parameter estimation. This attempts to mimic the IRLS routine invoked by `glm`, without returning extra “baggage" such as standard errors.

Usage

```
pirls(predictors, data, epsilon = 1e-08, iter.max = 25, normalized = TRUE)
```

Arguments

predictors	The columns of the standard design matrix to include in the model. For example, "c1", "c2" for main effects, and "c12" for interactions.
data	A design matrix with cell counts included in the column named "c". Must be a matrix (not a data frame)!
epsilon	Convergence tolerance, intended to play the same role as `epsilon` the control parameters for `glm.fit`.
iter.max	The maximum number of IRLS iterations. A warning appears if this maximum is ever reached.
normalized	Logical: If TRUE, include a normalization step after coefficient estimation, which resets the value of the intercept so that the sum of predicted values is exactly 1

Details

The main purpose of `pirls` is to obtain speed, for the special circumstance in which one must fit gazillions of Poisson regression models, where the only quantity of interested is the point estimates of the regression coefficients. Matrix inversion is one of the most time consuming steps in the function, and the overall speed can be improved by about 20 percent by modifying the source code to replace the `solve()` command with `.Internal(La_solve())`.

Value

The vector of estimated log-linear coefficients. The first coefficient is the intercept, and the remaining ones correspond to the `predictors` argument, in that order

Author(s)

Zach Kurtz

`plot.lllcrc` *Plot LLLMs*

Description

Generates stacked relative frequency diagrams for local log-linear or VGAM capture recapture models

Usage

```
## S3 method for class 'lllcrc'
plot(x, cont.var = "x.con.1", selection = NULL,
  main = "Set the main argument for the title", xlab = "x",
  label.offset = 0.01, text.size = 0.7, x.range = NULL, subtitle = NULL,
  padj.adj = -0.3, lr = 1, lr.global = NULL, ylim = NULL, ...)

## S3 method for class 'vgam.crc'
plot(x, cont.var = "x.con.1", selection = NULL,
  main = "Set the main argument for the title", xlab = "x",
  label.offset = 0.01, text.size = 0.7, x.range = NULL, subtitle = NULL,
  padj.adj = -0.3, lr = 1, lr.global = NULL, ylim = NULL, ...)
```

Arguments

`x`	Output of the function for LLLMs, `lllcrc`, or the CRC wrapper function for VGAM, `vgam.crc`
`cont.var`	The name (in the form `x.dis ...` – see `format.data`) of a continuous variable that will be used for the x-axis of the plot. By default, it is `x.con.1`, the first continuous variable
`selection`	See the same argument in `extract.CI`

main	Plot title
xlab	x-axis label
label.offset	Controls the distance between the capture-pattern labels and corresponding curves
text.size	Controls text size of capture pattern labels
x.range	The range of the primary continuous variable that is included in the plot
subtitle	An optional subtitle
padj.adj	Controls the vertical positioning of the subtitle if it is defined
lr	Has value 1 or -1; controls the left-right (or right-left) alternating pattern of the capture pattern labels
lr.global	NULL by default; if specified, it must be -1 or 1, causing all capture pattern labels to appear on the left or the right, without alternating
ylim	Optional argument of the form c(a,b), where a and b are numbers.
...	Additional parameters to be passed into plot

Details

The capture pattern relative frequencies are plotted in a stacked form, summing to 1 over each vertical cross-section. On top, the rate of missingness is plotted for the all-zero capture pattern, along with 95 percent confidence curves if a boots entry is included in x, the first argument

Value

Returns nothing, but makes a plot if you're paying attention

Author(s)

Zach Kurtz

---

plot.llsim *Plot the output of* llm.sim

Description

Plot the output of llm.sim

Usage

```
## S3 method for class 'llsim'
plot(x, y.top = 2, probs = c(0.25, 0.75), main = NULL,
  ...)
```

Arguments

x	An object of class llsim
y.top	The upper bound of the plotting window
probs	The interval width, in terms of quantiles
main	Plot title
...	Additional parameters to be passed into plot

Author(s)

Zach Kurtz

---

pop.to.counts *Put CRC data into LLM vector*

Description

Essentially, this does table(y), where y is a vector of capture patterns, with the exception that here every capture pattern that is observable (in principle) is included, with a possibly zero count.

Usage

```
pop.to.counts(y)
```

Arguments

y	A vector of capture patterns

Author(s)

Zach Kurtz

---

poptop *Simulate a CRC experiment*

Description

Generate fake capture-recapture datasets to be used in simulation experiments

Usage

```
poptop(k = 2, n = 100, heters = NULL, covs = NULL,
  remove.zeros = TRUE)
```

Arguments

k	The number of lists to be generated
n	The true size of the [fake] population
heters	Optional: a string that is the name of some function for defining a heterogeneity structure, possibly in terms of covariates. Currently the only non-null option implemented is french.1, although a user could easily create additional options
covs	Optional: a string that is the name of some function for defining covariates for each population unit. Currently the only non-null option implemented is age.sex.zip
remove.zeros	Logical, TRUE by default. If FALSE, keep the all-zero capture patterns in the dataset to be returned

Value

A data frame containing fake capture recapture data

Author(s)

Zach Kurtz

---

rates.by.category *Display estimated rates of missingness by category*

Description

A wrapper for summarize.by.factors. A table is generated to display the estimated average rate of missingness for every point in the space of categorical covariates. Reference levels may be selected for each category to reduce the size of the table

Usage

```
rates.by.category(x, reference.levels = c())
```

Arguments

x	The output of lllcrc or vgam.crc
reference.levels	A vector of variable names, where the names are in the form "x.dis...." as in the output of format.data. For example, if sex is a variable, two columns x.dis.1.M and x.dis.1.F may be included in the table unless one of them is specified as a reference level

Value

A data frame

Author(s)

Zach Kurtz

---

resample.captures *Tool for bootstrapping*

Description

Used by `vgam.crc.boots` and `lllcrc.boots`. See these functions for details.

Usage

```
resample.captures(boot.list)
```

Arguments

`boot.list`	See `lllcrc.boots` or `vgam.crc.boots`.

Author(s)

Zach Kurtz

References

Zwane E and Heijden Pvd (2003). "Implementing the parametric bootstrap in capture-recapture models with continuous covariates." *Statistics & Probability Letters*, **65**, pp. 121-125.

---

saturated.local *Use odd-even formula to fit saturated LLM*

Description

See the odd-even formula in Fienberg 1972, or a localized version of it in Zwane 2004.

Usage

```
saturated.local(dens)
```

Arguments

`dens`	Matrix with column names corresponding to the possible capture patterns

Details

The function is vectorized – it gets applied to each row of the input matrix, resulting in a vector of estimated rates of missingness or the total number missing

Author(s)

Zach Kurtz

References

Fienberg SE (1972). "The Multiple Recapture Census for Closed Populations and Incomplete 2^k Contingency Tables." *Biometrika*, **59**(3), pp. 591.

---

smooth.patterns *Local averaging for LLLMs*

Description

A nearest-neighbors procedure is used in conjunction with the Epanechnikov kernel to define a kernel smooth of multinomial outcomes across the covariate space

Usage

```
smooth.patterns(dat, kfrac, bw)
```

Arguments

dat	The capture-recapture data in the form that is returned by formatdata or micro.post.stratify.
kfrac	The approximate fraction of the data that is included in the support of the kernel for the local averages.
bw	A matrix a single column, with rownames that match the covariate names in dat. The values in the column are scalars that are used in constructing distances between covariate vectors. Raw differences are divided by the corresponding scalars before being squared in the context of a Euclidean metric.

Details

See Kurtz 2013, Chapter on multiple sclerosis

Value

A list containing the original data (dat), the smoothed data (hpi), and the effective sample sizes (ess) for each local average, or row, in the smoothed data

Author(s)

Zach Kurtz

References

Kurtz 2013

stackydens *Stack local capture pattern frequencies for plotting*

Description

Used in the plot functions

Usage

```
stackydens(ydens)
```

Arguments

ydens	A matrix or data.frame with as many columns as there are capture patterns. Each row is the relative frequency (possibly non-integer) of each pattern.

Author(s)

Zach Kurtz

string.to.array *Put LLM vector into a LLM design matrix*

Description

Given the output of pop.to.counts, i.e., a 1-row dataframe with as many columns as there are observable capture patterns, create a design matrix for log-linear modeling

Usage

```
string.to.array(densi, rasch = FALSE)
```

Arguments

densi	The capture pattern counts
rasch	Logical: if TRUE, the square of the number of captures is included in the design matrix as the column ck2

Value

A design matrix

Author(s)

Zach Kurtz

summarize.by.factors *Summarize LLLM by factor*

Description

Compute the number of observations and the average estimated rate of missingness for each combination of a set of categorical covariates

Usage

```
summarize.by.factors(dat, vars)
```

Arguments

dat	The dat object in the output of lllcrc or vgam.crc
vars	A vector of variable names, where the names are in the form "x.dis...." as in the output of format.data

Value

A data frame

Author(s)

Zach Kurtz

summary.lllcrc *Summary of LLLM or VGAM CRC analysis*

Description

Provides some description of a local log-linear modelling object

Usage

```
## S3 method for class 'lllcrc'
summary(object, ...)

## S3 method for class 'vgam.crc'
summary(object, ...)
```

Arguments

object	The output of lllcrc, the LLLM function, or vgam.crc.
...	Optional arguments

Author(s)

Zach Kurtz

`vgam.crc` *Build a VGAM CRC model*

Description

A high-level function to fit a VGAM CRC model to standardized data (the output of format.data).

Usage

```
vgam.crc(dat, models = make.hierarchical.term.sets(k = attributes(dt)$k), sdf,
  llform = NULL, round.vars = NULL, rounding.scale = NULL,
  boot.control = NULL)
```

Arguments

`dat`	The CRC data, as output of `format.data`
`models`	A list of models – or an expression that returns a list of models – to be considered in local model search. Run the default, `make.hierarchical.term.sets(k = attributes(dat)$k)`, to see an example.
`sdf`	A vector, with length corresponding to the number of continuous predictor variables, that states the desired effective degrees of freedom for the corresponding smooth spline in VGAM.
`llform`	A character vector of predictors of the form "c1", "c2" for main effects, or "c12" for an interaction. By default, the function `AICc.vgam` is used to select the best set of terms among the candidate sets that are proposed by the function `make.hierarchical.term.sets`.
`round.vars`	See `micro.post.stratify`, which is called within `vgam.crc`.
`rounding.scale`	See `micro.post.stratify`, which is called within `vgam.crc`.
`boot.control`	A list of control parameters for bootstrapping the sampling distribution of the estimator(s). By default, there is no bootstrapping.

Details

Implements, approximately, the method of Zwane (2004). Serves mainly as a user-friendly interface to the VGAM package.

Value

est	A point estimate of the population size
llform	The set of log-linear terms
dat	The output of function micro.post.stratify, with estimated local rates of missingness appended as an extra column labeled pi0. In addition, mct (multinomial cell count) gives the number of observed units with that distinct covariate vector, and cpi0 (cumulative number missing) gives the the product of pi0 with mct, such that summing over this vectorized product is exactly the Horvitz-Thompson style sum in capture recapture.
aic	The AICc for the chosen VGAM, as computed by function AICc.vgam
mod	The VGAM model object; see the vgam function in package VGAM for details
...	The output is of class vgam.crc and has attributes cont.x and conteg.x, which relate the continuous and categorical variables in the model

Author(s)

Zach Kurtz

References

Zwane E and Heijden Pvd (2003). "Implementing the parametric bootstrap in capture-recapture models with continuous covariates." *Statistics & Probability Letters*, **65**, pp. 121-125.

Zwane E and Heijden Pvd (2004). "Semiparametric models for capture-recapture studies with covariates." *Computational Statistics & Data Analysis*, **47**, pp. 729-743.

---

vgam.crc.boots *Bootstrapping for a VGAM CRC model*

Description

Implement a single instance of resampling process to simulate a replicate of the vgam.crc model.

Usage

```
vgam.crc.boots(boot.list)
```

Arguments

boot.list	A list of control parameters for the bootstrapping process. See the example under lllcrc-package

Details

vgam.crc calls this function for each bootstrap replicate. Many replicates together can be used for empirical confidence interval estimation. The bootstrap is much like that described by Zwane 2003, but more like Method 2 in Norris and Polluck 1996, as we repeat the process of selecting log-linear terms for each bootstrap iteration, and the multinomial sampling probabilities are based on raw local estimates instead of post-log-linear modeling estimates.

Value

A list containing two vectors. The first vector, cpi0, gives the estimated number of missing units at each point; the second, mct, gives the number of units observed at each point. (Dividing the first vector by the second gives an estimate rate of missingness)

Author(s)

Zach Kurtz

References

Norris JL and Pollock KH (1996). "Including model uncertainty in estimating variances in multiple capture studies." *Environmental and Ecological Statistics*, **3**, pp. 235-244.

---

y.string.to.y.glm *Capture patterns to design matrix*

Description

Tally the multinomial counts for a vector of capture patterns, and put the result into design matrix

Usage

```
y.string.to.y.glm(y)
```

Arguments

y	A vector of capture patterns (each capture pattern is a single string of binary digits)

Value

A design matrix, including capture pattern counts, that is ready for log-linear analysis

Author(s)

Zach Kurtz

zglm *Maximum likelihood for log-linear coefficients*

Description

A simplified version of `glm` that does only parameter estimation

Usage

```
zglm(predictors, data, normalized = TRUE, precision = 1000)
```

Arguments

`predictors`	The columns of the standard design matrix to include in the model. For example, "c1", "c2" for main effects, and "c12" for interactions.
`data`	A design matrix with cell counts included
`normalized`	Logical: If TRUE, include a normalization step after coefficient estimation, which resets the value of the intercept so that the sum of predicted values is exactly 1
`precision`	Controls the precision of the coefficient estimates. A higher number is less precise. 1 corresponds to machine epsilon.

Details

Maximize the Poisson likelihood using BFGS in `optim()`.

Value

The vector of estimated log-linear coefficients. The first coefficient is the intercept, and the remaining ones correspond to the `predictors` argument, in that order

Author(s)

Zach Kurtz

Index