Package 'shinyrecap'

July 29, 2026

Type Package

Title 'Shiny' User Interface for Multiple Source Capture Recapture Models

Version 0.2.0

Maintainer Ian E. Fellows <ian@fellstat.com>

Description Implements user interfaces for log-linear models, Bayesian model averaging and Bayesian Dirichlet process mixture models. See McIntyre, Fellows, Gutreuter and Hladik (2022) <doi:10.2196/32645>.

License MIT + file LICENCE

Imports Rcapture, shiny, shinycssloaders, ggplot2, reshape, dga, LCMCR, ipc, future, promises, coda, testthat,rhandsontable, shinyhelper, magrittr, HDInterval, shinyjs, rmarkdown, knitr

URL https://fellstat.github.io/shinyrecap/

Encoding UTF-8

RoxygenNote 7.3.2

NeedsCompilation no

Author Ian E. Fellows [aut, cre]

Repository CRAN

Date/Publication 2026-07-29 18:40:02 UTC

Contents

shinyrecap-package 2
disaggregate 2
extract_histories 3
formatGraphs 4
launchShinyPopSize 5
lcmcrSample 5
simulateCapture 6
simulateEstimates 7
simulateHeteroNormal 8
srhelp 8

Index **9**

`shinyrecap-package` *'Shiny' User Interface for Multiple Source Capture Recapture Models*

Description

Implements user interfaces for log-linear models, Bayesian model averaging and Bayesian Dirichlet process mixture models.

Author(s)

Ian E. Fellows <ian@fellstat.com>

`disaggregate` *Disaggregate data*

Description

Disaggregate data

Usage

```
disaggregate(dat, counts)
```

Arguments

`dat`	a data.frame
`counts`	frequency counts for each row

Value

A disaggregated data.frame

`extract_histories` *Convert Unique Event Identifier Event Data into Multiple Source Capture Recapture Histories*

Description

Convert Unique Event Identifier Event Data into Multiple Source Capture Recapture Histories

Usage

```
extract_histories(datasets, count_name, cov_names = c())
```

Arguments

`datasets`	An ordered list of capture event datasets. The first j-1 columns of the jth dataset should be a 0 or 1 indicator of beng captured in a previous event.
`count_name`	The name of the column holding the counts of the number of individuals.
`cov_names`	A character vector of column names for covariates to stratify by.

Value

A dataframe. The first length(datasets) columns of which contain all possible permutations of capture histories. The "__count__" column contains the number of observed cases. Other columns are the stratifying covariates.

Examples

```
# First sample
dat1 <- data.frame(
  count=c(55,43),
  c1=c("m","f"),
  stringsAsFactors = FALSE
)

# Second sample
dat2 <- data.frame(
  id1=c(1,0,1,0),
  count=c(10,11,5,12),
  c1=c("m","m","f","f"),
  stringsAsFactors = FALSE
)

# Third Sample
dat3 <- data.frame(
  id1=c(1,0,1,0,1,0,1,0),
  id2=c(1,1,1,1,0,0,0,0),
  count=c(2,3,4,3,2,30,4,30),
  c1=c("m","m","f","f","m","m","f","f"),
```

```
  stringsAsFactors = FALSE
)

# Fourth Sample
dat4 <- data.frame(
  id1=c(1,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0),
  id2=c(1,1,1,1,0,0,0,0,1,1,1,1,0,0,0,0),
  id3=c(1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0),
  count=c(1,1,1,1,0,1,0,0,1,1,1,1,1,0,1,10),
  c1=c("m","m","f","f","m","m","f","f","m","m","f","f","m","m","f","f"),
  stringsAsFactors = FALSE
)

# Combine 4 capture recatpure events together
datasets <- list(dat1,dat2,dat3,dat4)

# Get four sample capture recapture histories from unique event observations
extract_histories(datasets, "count")

# Stratify by a covariate
extract_histories(datasets, "count", cov_names="c1")

# Two sample capture recapture
extract_histories(datasets[1:2], "count", cov_names="c1")
```

`formatGraphs` *Format graphs*

Description

Format graphs

Usage

```
formatGraphs(graphs)
```

Arguments

`graphs`	the graphs

Value

String representations of the BMA graph structures

launchShinyPopSize *Launches the Shiny Application for Population Size*

Description

Launches the Shiny Application for Population Size

Usage

```
launchShinyPopSize(app = c("estimation", "power", "convert"))
```

Arguments

app	Which application to launch.

Details

The manual for this shiny application is located at https://fellstat.github.io/shinyrecap/

Value

A shiny app

lcmcrSample *Perform LCMCR sampling with a monitor function*

Description

Perform LCMCR sampling with a monitor function

Usage

```
lcmcrSample(
  object,
  burnin = 10000,
  samples = 1000,
  thinning = 10,
  clear_buffer = FALSE,
  output = TRUE,
  nMonitorBreaks = 100,
  monitorFunc = function(subs, tot) {
 }
)
```

Arguments

object	the samples
burnin	MCMC burn in
samples	number of samples
thinning	MCMC thinning
clear_buffer	buffer clear buffer of object
output	output progress
nMonitorBreaks	number of times to call the monitor function
monitorFunc	A function called nMonitorBreaks times taking the number of samples to be taken, and the total samples

Details

An edited version of `lcmCR_PostSampl`

Value

The posterior sample of N estimates

---

simulateCapture *Simulate Capture Re-capture with heterogeneity*

Description

Simulate Capture Re-capture with heterogeneity

Usage

```
simulateCapture(hetero, p)
```

Arguments

hetero	The heterogineity
p	A vector of capture event probabilities

Value

a matrix of simulated captures

Examples

```
het <- simulateHeteroNormal(1000, 1.1)
cap <- simulateCapture(het, p = c(.05,.1,.05,.1))
summary(cap)
```

`simulateEstimates` *Simulates capture re-capture estimates*

Description

Simulates capture re-capture estimates

Usage

```
simulateEstimates(
  nsim,
  N,
  p,
  htype = "None",
  heteroPerc = 1,
  monitorFunc = function(i) {
 }
)
```

Arguments

`nsim`	number of simulations
`N`	Population size
`p`	A vector of capture event probabilities
`htype`	The type of capture heterogeneity. Either "None" or "Normal"
`heteroPerc`	The increase in odds of capture for the perc 90th percentile most likely to be captured individuals, compared to the average individual.
`monitorFunc`	A function called after every iteration. Useful for monitoring simulation progress.

Value

a data.frame with the first row being the simulated estimates and the second the standard error

Examples

```
library(ggplot2)
# Simulate estimates from the Mt model with no population heterogeneity
ests <- simulateEstimates(15,500,c(.1,.1,.1))

# Simulate estimates from the Mth (Normal) model with no population heterogeneity.
ests2 <- simulateEstimates(20,500,c(.1,.1,.1), htype="Normal")

df <- data.frame(est = ests[[1]],type="Mt")
df <- rbind(df, data.frame(est = ests2[[1]],type="Mth (Normal)"))
qplot(x=est, color=type, data=df, geom="density") +
  geom_vline(xintercept=500,color="purple")
```

simulateHeteroNormal *simulate capture heterogineity*

Description

simulate capture heterogineity

Usage

```
simulateHeteroNormal(N, heteroPerc = 1, perc = 0.9)
```

Arguments

N	Population size
heteroPerc	The increase in odds of capture for the perc 90th percentile most likely to be captured individuals, compared to the average individual.
perc	The percentile to use.

Value

a random sample fnormal sample with sd=log(heteroPerc) / qnorm(perc)

Examples

```
het <- simulateHeteroNormal(100, 1.1)
hist(het)
```

srhelp *A simple wrapper for shinyhelper::helper*

Description

A simple wrapper for shinyhelper::helper

Usage

```
srhelp(x, content, ...)
```

Arguments

x	The shiny object to decorate
content	the name of the help
...	additional parameters for shinyhelper::helper

Value

A shinyhelper object

Index

disaggregate, 2

extract_histories, 3

formatGraphs, 4

launchShinyPopSize, 5
lcmcrSample, 5

shinyrecap-package, 2
simulateCapture, 6
simulateEstimates, 7
simulateHeteroNormal, 8
srhelp, 8