

Package ‘chords’

January 21, 2017

Type Package

Title Estimation in Respondent Driven Samples

Version 0.95.4

Date 2017-01-01

Author Jonathan Rosenblatt

Maintainer Jonathan Rosenblatt <johnros@bgu.ac.il>

Description Maximum likelihood estimation in respondent driven samples.

License GPL-2

Imports MASS, Matrix

Suggests testthat

NeedsCompilation no

Repository CRAN

Date/Publication 2017-01-21 23:29:35

R topics documented:

chords-package	2
brazil	2
Estimate.b.k	3
getTheta	7
initializeRdsObject	8
makeJackControl	10
makeRdsSample	10
thetaSmoothingNks	11
Index	13

chords-package

Population size estimation for respondent driven sampling

Description

Estimates population size and degree distribution in respondent driven samples (RDS).

Details

Maximum likelihood estimation of population size using the methodology in the reference. The fundamental idea is of modeling the sampling as an epidemic process. See [Estimate.b.k](#) for details.

Author(s)

Jonathan D. Rosenblatt <johnros@bgu.ac.il>

References

[1] Berchenko, Y., Rosenblatt J.D., and S.D.W. Frost. "Modeling and Analyzing Respondent Driven Sampling as a Counting Process." arXiv:1304.3505

brazil

Heavy Drug Users in Curitiba

Description

A respondent driven sample of heavy drug users in Curitiba.

Usage

```
data("brazil")
```

Format

A data frame with 303 observations on the following 8 variables.

MyUniID Subject's ID.

NS1 Subject's self reported degree.

refCoupNum Reference coupon no.

coup1 Supplied coupon.

coup2 Supplied coupon.

coup3 Supplied coupon.

interviewDt Time of interview. See details.

interviewDt2 Deprecated.

Details

The format of the data is essentially that of the RDS file format as specified in page 7 in the RDS Analysis tool manual: http://www.respondentdrivensampling.org/reports/RDSAT_7.1-Manual_2012-11-25.pdf.

The RDS format has been augmented with the time of interview (`interviewDt` variable) required for the methodology in [1].

The `interviewDt` variable encodes the time of interview. For the purpose of calling `Estimate.b.k` the scale and origin are imaterial. We thus use an arbitrary efficient encoding which might not adhere to the original scale.

For full details see the Source section.

Source

[1] Salganik, M.J., Fazito, D., Bertoni, N., Abdo, A.H., Mello, M.B., and Bastos, F.I. (2011). "Assessing Network Scale-up Estimates for Groups Most at Risk of HIV/AIDS: Evidence From a Multiple-Method Study of Heavy Drug Users in Curitiba, Brazil." *American Journal of Epidemiology*, 174(10): 1190-1196.

And <http://opr.princeton.edu/archive/nsum/>

References

[1] Berchenko, Y., Rosenblatt J.D., and S.D.W. Frost. "Modeling and Analyzing Respondent Driven Sampling as a Counting Process." arXiv:1304.3505

Estimate.b.k

RDS population size estimation

Description

Estimate population size from respondent driven samples (RDS) using maximum likelihood, and several variation. The underlying idea is that the sample spreads like an epidemic in the target population as described in the reference.

Usage

```
Estimate.b.k(rds.object, type = "mle", jack.control = NULL)
```

Arguments

- | | |
|-------------------------|--|
| <code>rds.object</code> | A object of class <code>rds-object</code> as constructed by <code>initializeRdsObject</code> or outputted by <code>Estimate.b.k</code> (depending on the type used). |
| <code>type</code> | A character vector with the type of estimation. Possible values: <ul style="list-style-type: none"> • mle Maximum likelihood. • integrated Integrated maximum likelihood. • observed Estimate with observed degrees. |

- **jeffreys** MAP estimation with Jeffreys prior.
- **parametric** Assume $\beta[k] := \beta * \theta^k$.
- **rescaling** Naive rescaling heuristic estimation.
- **leave-d-out** Leave-d-out resampling estimator.

jack.control A object of class jack.control as constructed by [makeJackControl](#).

Details

As of version 0.95, this function is the main workhorse of the chords package. Given an rds-class object, it will return population size estimates for each degree. Note that for the rescaling and parametric estimators, the input rds-object is expected to contain some initial estimate in the estimates slot.

See the reference for a description of the likelihood problem solved. Optimization is performed by noting that likelihood is coordinate-wise convex, thus amounts to a series of line-searches.

Value

An rds-class object with an updated estimates slot. The estimates slot is list with the following components:

call	The function call.
Nk.estimates	The estimated degree frequencies.
log.bk.estimates	The estimated sampling rates for each degree. In log scale.
convergence	0 if estimation of $N[k]$'s converged. Otherwise, 1 or -1, depending on the sign of the score function at the MLE.

References

[1] Berchenko, Y., Rosenblatt D.J., and S.D.W. Frost. "Modeling and Analyzing Respondent Driven Sampling as a Counting Process." arXiv:1304.3505,

See Also

[initializeRdsObject](#), [makeRdsSample](#), [getTheta](#).

Examples

```
# Preliminaries
data(brazil)
rds.object2<- initializeRdsObject(brazil)
see <- function(x) plot(x$estimates$Nk.estimates, type='h')

# Maximum likelihood
rds.object <- Estimate.b.k(rds.object = rds.object2 )
see(rds.object)

# View estimates:
plot(rds.object$estimates$Nk.estimates, type='h')
```

```

# Population size estimate:
sum(rds.object$estimates$Nk.estimates)
plot(rds.object$estimates$log.bk.estimates, type='h')

## Recover theta assuming  $b.k=b_0*k^\theta$ 
getTheta(rds.object)

# How many degrees were imputed?:
table(rds.object$estimates$convergence)

# Observed degree estimation:
rds.object.4 <- Estimate.b.k(rds.object = rds.object, type='observed')
see(rds.object.4)

# Naive rescaling
rds.object.5 <- Estimate.b.k(rds.object = rds.object, type='rescaling')
see(rds.object.5)

# Parametric rates
rds.object.6 <- Estimate.b.k(rds.object = rds.object,
                           type='parametric')
see(rds.object.6)
jack.control <- makeJackControl(3, 1e1)
rds.object.7 <- Estimate.b.k(rds.object = rds.object,
                           type='leave-d-out',
                           jack.control = jack.control)
see(rds.object.7)
rds.object.8 <- Estimate.b.k(rds.object = rds.object,
                           type='integrated',
                           jack.control = jack.control)
see(rds.object.8)
rds.object.9 <- Estimate.b.k(rds.object = rds.object,
                           type='jeffreys')
see(rds.object.9)

## Not run:
## Simulated data example:
dk <- c(2, 1e1) # unique degree classes
true.dks <- rep(0,max(dk)); true.dks[dk] <- dk
true.Nks <- rep(0,max(dk)); true.Nks[dk] <- 1e3
beta <- 1 #5e-6
theta <- 0.1
true.log.bks <- rep(-Inf, max(dk))
true.log.bks[dk] <- theta*log(beta*dk)
sample.length <- 4e2
nsims <- 1e2

simlist <- list()
for(i in 1:nsims){
  simlist[[i]] <- makeRdsSample(

```

```

    N.k =true.Nks ,
    b.k = exp(true.log.bks),
    sample.length = sample.length)
}

# Estimate betas and theta with chords:
llvec <- rep(NA,nsims)
bklist <- list()
for(i in 1:nsims){
  # i <- 2
  simlist[[i]] <- Estimate.b.k(rds.object = simlist[[i]])
  # llvec[i] <- simlist[[i]]$estimates$likelihood
  bklist[[i]] <- simlist[[i]]$estimates$log.bk.estimates
}
b1vec <- bklist
b2vec <- bklist

hist(b1vec)
abline(v=true.log.bks[2])
hist(b2vec)
abline(v=true.log.bks[10])

beta0vec <- rep(-Inf,nsims)
thetavec <- rep(-Inf,nsims)
nvec <- rep(-Inf,nsims)
converged <- rep(9999,nsims)

for(i in 1:nsims){
  # i <- 2
  nvec[i] <- sum(simlist[[i]]$estimates$Nk.estimates)
  converged[i] <- sum(simlist[[i]]$estimates$convergence, na.rm=TRUE)
  # tfit <- getTheta(simlist[[i]])
  # beta0vec[i] <- tfit$log.beta_0
  # thetavec[i] <- tfit$theta
}
summary(beta0vec)
summary(nvec)
# summary(thetavec)
# hist(thetavec)
# abline(v=theta)
hist(nvec)
abline(v=sum(true.Nks), col='red')
abline(v=median(nvec, na.rm = TRUE), lty=2)
table(converged)

# Try various re-estimations:
rds.object2 <- simlist[[which(is.infinite(nvec))[1]]]

rds.object <- Estimate.b.k(rds.object = rds.object2 )
see(rds.object)
rds.object$estimates$Nk.estimates

```

```

rds.object.5 <- Estimate.b.k(rds.object = rds.object, type='rescaling')
see(rds.object.5) # will not work. less than 2 converging estimates.
rds.object.5$estimates$Nk.estimates

rds.object.6 <- Estimate.b.k(rds.object = rds.object, type='parametric')
see(rds.object.6) # will not work. less than 2 converging estimates.

jack.control <- makeJackControl(3, 1e2)
rds.object.7 <- Estimate.b.k(rds.object = rds.object,
                           type='leave-d-out',
                           jack.control = jack.control)
see(rds.object.7)
rds.object.7$estimates$Nk.estimates

rds.object.8 <- Estimate.b.k(rds.object = rds.object, type='integrated')
see(rds.object.8)
rds.object.8$estimates$Nk.estimates

rds.object.9 <- Estimate.b.k(rds.object = rds.object, type='jeffreys')
see(rds.object.9)
rds.object.9$estimates$Nk.estimates

## End(Not run)

```

getTheta

Recover the "discoverability coefficient".

Description

Estimates the effect of the degree on the rate of sampling. Also known as the "coefficient of discoverability" in the oil-discovery literature [2]. Formally, we estimate θ and β_0 assuming that $\beta_k := \beta_0 * k^\theta$.

Usage

```
getTheta(rds.object, bin=1, robust=TRUE)
```

Arguments

rds.object	A rds-object with a estimates component as returned by Estimate.b.k
bin	Bin degree counts. See Note.
robust	Should β_0 and θ be recovered from β_k using a robust method (default) or not.

Value

A list including the following components:

log.beta_0	The log of β_0 in $\beta_k := \beta_0 * k^\theta$.
theta	θ in $\beta_k := \beta_0 * k^\theta$.

Note

If degree counts have been binned by `initializeRdsObject` (for variance reduction), the same value has to be supplied to `getTheta` for correct estimation.

References

- [1] Berchenko, Yakir, Jonathan Rosenblatt, and Simon D. W. Frost. "Modeling and Analyzing Respondent Driven Sampling as a Counting Process." arXiv:1304.3505. [HTTP://arXiv.org/abs/1304.3505](http://arXiv.org/abs/1304.3505).
 [2] Bloomfield, P., K.S. Deffeyes, B. Silverman, G.S. Watson, Y. Benjamini, and R.A. Stine. Volume and Area of Oil Fields and Their Impact on Order of Discovery, 1980. <http://www.osti.gov/scitech/servlets/purl/6037591>

See Also

[Estimate.b.k](#), [initializeRdsObject](#), [makeRdsSample](#)

`initializeRdsObject` *Construct a rds-object from a data.frame.*

Description

Given a data frame with the appropriate variables, initializes a rds-object with the components required by the [Estimate.b.k](#) function for estimation.

Usage

```
initializeRdsObject(rds.sample, bin=1L, seeds=1L)
```

Arguments

<code>rds.sample</code>	A data frame with required columns. See Details.
<code>bin</code>	The number of degrees fo bin together. See details.
<code>seeds</code>	The number of seed recruiters. See details.

Details

The essence of the function is in recovering the sampling snowball required by [Estimate.b.k](#). The function allows for recruiters to enter and exit the sampling snowball. The number of seed recruiters is typically not specified in an RDS file. The `seeds` argument is a workaround that allows to specify directly this number.

The `rds.sample` object is assumed to be a data frame with the following column names:

1. `MyUniID` An identifier of the sampling unit. [not required]
2. `NSI` The reported degree. [required]
3. `refCoupNum` The number of the referring coupon.
4. `coup1` The number of the 1st supplied coupon. NA if none. [required].
5. `coupX` The number of the Xth supplied coupon. NA if none. [not required]
6. `interviewDt` The time of the interview. In numeric representation from some origin. Ties are not allowed.

See [brazil](#) for a sample data.frame.

If the sample is short, stabilization of degree estimates can be achieved by binning degrees together. This can be done with the `bin` argument. Note however that the interpretation of the estimated degree counts is now different as the k 'th degree is actually the k 'th bin, which is only proportional to k . An exception is the function [getTheta](#) which also accepts a `bin` argument for proper estimation of *theta*.

Value

A list with the following components.

- `rds.sample` The input data frame. After ordering along time of arrival.
- `l.t` The sampling snowball. A list including the following items: `l.t` An integer of the count of the sampling individuals at the moments of recruitment. `degree.in` An integer with the degree of an added recruiter at the moments of recruitment. `degree.out` An integer with the degree of a removed recruiter at the moment of recruitment.
- `original.ordering` The order of the arrivals as was inputed in `rds.sample$interviewDt`
- `estimates` A placeholder for the future output of [Estimate.b.k](#)

References

[1] Berchenko, Y., Rosenblatt J.D., and S.D.W. Frost. "Modeling and Analyzing Respondent Driven Sampling as a Counting Process." arXiv:1304.3505

See Also

[Estimate.b.k](#), [makeRdsSample](#), [brazil](#)

makeJackControl *Construct a control object for delete-d estimation.*

Description

A utility function for using the delete-d option of the [Estimate.b.k](#) function.

Usage

```
makeJackControl(d, B)
```

Arguments

d The number of (random) arrivals in the sample to delete.
B The number of deleted samples to generate.

Value

A list with named control parameters.

References

Musa, John D., and A. Iannino. "Estimating the Total Number of Software Failures Using an Exponential Model." SIGSOFT Softw. Eng. Notes 16, no. 3 (July 1991): 80-84. doi:10.1145/127099.127123.

See Also

[Estimate.b.k](#)

makeRdsSample *Generate a synthetic (simulated) RDS sample.*

Description

Generates a sample from the sampling process assumed in the reference. Well, actually, only the sufficient statistics required by [Estimate.b.k](#) are returned.

Usage

```
makeRdsSample(N.k, b.k, sample.length)
```

Arguments

N.k An integer vector with the population frequency of each degree.
b.k A numeric vector of the sampling rates of each degree.
sample.length The length of the sample. Specified as the number of recruitees before termination.

Value

An object of class rds-object suitable for applying [Estimate.b.k](#).

Note

The simulator does not produce a whole RDS sample, but rather the sufficient statistics required for applying [Estimate.b.k](#).

References

[1] Berchenko, Y., Rosenblatt J.D., and S.D.W. Frost. "Modeling and Analyzing Respondent Driven Sampling as a Counting Process." arXiv:1304.3505

See Also

[Estimate.b.k](#)

Examples

```
# Generate data:
true.Nks <- rep(0,100); true.Nks[c(2,100)] <- 1000
theta <- 1e-1
true.log.bks <- rep(-Inf, 100);true.log.bks[c(2,100)] <- theta*log(c(2,100))
sample.length <- 1000L
rds.simulated.object <- makeRdsSample(
  N.k =true.Nks ,
  b.k = exp(true.log.bks),
  sample.length = sample.length)

# Estimate:
Estimate.b.k(rds.object = rds.simulated.object )
chords::compareNkEstimate(rds.simulated.object$estimates$Nk.estimates, true.Nks)
```

thetaSmoothingNks *Smooth estimates degree frequencies.*

Description

Smooths estimated N_k by assuming that $\beta_k = \beta_0 * k^\theta$.

Usage

```
thetaSmoothingNks(rds.object, bin=1)
```

Arguments

rds.object A rds-object class object as returned by [Estimate.b.k](#)
bin Number of degrees to bin together for estimation.

Value

A numeric vector of smoothed N_k values.

Author(s)

Jonathan D. Rosenblatt <john.ros.work@gmail.com>

See Also

[Estimate.b.k](#), [getTheta](#)

Examples

```
# See estimate.b.k()
```

Index

*Topic **datasets**

brazil, [2](#)

brazil, [2](#), [9](#)

chords (chords-package), [2](#)

chords-package, [2](#)

Estimate.b.k, [2](#), [3](#), [3](#), [7–12](#)

getTheta, [4](#), [7](#), [9](#), [12](#)

initializeRdsObject, [3](#), [4](#), [8](#), [8](#)

makeJackControl, [4](#), [10](#)

makeRdsSample, [4](#), [8](#), [9](#), [10](#)

thetaSmoothingNks, [11](#)