Package: ordinalRR (via r-universe)

October 27, 2026

Type Package

Title Analysis of Repeatability and Reproducibility Studies with Ordinal Measurements

Version 1.1

Date 2020-3-29

Author Ken Ryan <kjryan@mail.wvu.edu>

Maintainer Ken Ryan <kjryan@mail.wvu.edu>

Depends R (>= 2.10), rjags

Suggests graphics

LazyLoad Yes

Description Implements Bayesian data analyses of balanced repeatability and reproducibility studies with ordinal measurements. Model fitting is based on MCMC posterior sampling with 'rjags'. Function ordinalRR() directly carries out the model fitting, and this function has the flexibility to allow the user to specify key aspects of the model, e.g., fixed versus random effects. Functions for preprocessing data and for the numerical and graphical display of a fitted model are also provided. There are also functions for displaying the model at fixed (user-specified) parameters and for simulating a hypothetical data set at a fixed (user-specified) set of parameters for a random-effects rater population. For additional technical details, refer to Culp, Ryan, Chen, and Hamada (2018) and cite this Technometrics paper when referencing any aspect of this work. The demo of this package reproduces results from the Technometrics paper.

License GPL-2

NeedsCompilation no

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

Date/Publication 2020-03-30 08:00:04 UTC

RemoteUrl https://github.com/cran/ordinalRR

RemoteRef HEAD

RemoteSha b09380b4f953f36f115a5fb8dbe91a7e2020488c

Contents

compute.q . . . 2
density.ordinalRR . . . 3
followup . . . 4
hist.ordinalRR . . . 5
make.rater . . . 6
ordinalRR . . . 7
ordinalRR.control . . . 8
ordinalRR.sim . . . 9
preprocess . . . 11
print.ordinalRR . . . 12
summary.ordinalRR . . . 13

Index **15**

`compute.q` *Compute the probabilities for a single rater at a fixed part quality.*

Description

This function accepts an object of class 'rater' and the latent (scalar) value of a hypothetical part's quality. The probabilities of the H ordinal categories are outputted based on the De Mast-Van Wieringen Model.

Usage

```
compute.q(rater,x)
```

Arguments

`rater`	'rater' the rater's parameters. (Use make.rater() to construct this input.)
`x`	'scalar' latent quality of a fixed part.

Value

`vector`	a probability vector of length H

Author(s)

Ken Ryan

References

de Mast, J. and van Wieringen, W.N. (2010). "Modeling and Evaluating Repeatability and Reproducibility of Ordinal Classifications." *Technometrics*, 52(1), 94-106.

See Also

make.rater

Examples

```
(rater1=make.rater(1,c(-1.7,-0.5,1.6))) #3 cutpoints, so H=4
(prob=compute.q(rater1,.1)) #probabilities of 1,2,3,4 if part quality is x=.1
sum(prob) #should sum to one
```

`density.ordinalRR` *Plot densities of the latent part distributions.*

Description

This plots density estimates of the posterior distributions for repeatabilities, R&Rs, and proportions due to repeatability. This corresponds to Figure 5 from Culp, Ryan, Chen, and Hamada (2018). There will be a dark posterior predictive curve only if a random model was fit.

Usage

```
## S3 method for class 'ordinalRR'
density(x,plt.type=c("repeat", "rr", "prop", "all"),m=0, ...)
```

Arguments

`x`	'ordinalRR' output from function ordinalRR().
`plt.type`	'character' plot type (must be 'repeat', 'rr', 'prop', or 'all').
`m`	'nonnegative integer' relaxes the match criterion to within plus or minus m (see Equations (26) and (27) from Culp, Ryan, Chen, and Hamada, 2018).
`...`	mop up additional inputs.

Author(s)

Ken Ryan

References

Culp, S.L., Ryan, K.J., Chen, J., and Hamada, M.S. (2018). "Analysis of Repeatability and Reproducibility Studies with Ordinal Measurements." *Technometrics*, doi:10.1080/00401706.2018.1429317.

See Also

ordinalRR

`followup` *Followup data from experiment on soldered joints.*

Description

This is the followup data from the experiment analyzed in Culp, Ryan, Chen, and Hamada (2018). These data were first presented and described in de Mast and van Wieringen (2010). Thirty parts were rated on a 4-point ordinal scale of 1-4 by each of 3 raters in a balanced design of 2 repetitions per combination of part and rater.

Usage

```
data("followup")
```

Format

This data frame has 30 rows (i.e., one for each part) and the following 6 columns.

`Yi11` ordinal value assigned by rater 1 on repetition 1.

`Yi12` ordinal value assigned by rater 1 on repetition 2.

`Yi21` ordinal value assigned by rater 2 on repetition 1.

`Yi22` ordinal value assigned by rater 2 on repetition 2.

`Yi31` ordinal value assigned by rater 3 on repetition 1.

`Yi32` ordinal value assigned by rater 3 on repetition 2.

Author(s)

Ken Ryan

References

Culp, S.L., Ryan, K.J., Chen, J., and Hamada, M.S. (2018). "Analysis of Repeatability and Reproducibility Studies with Ordinal Measurements." *Technometrics*, doi:10.1080/00401706.2018.1429317.

de Mast, J. and van Wieringen, W.N. (2010). "Modeling and Evaluating Repeatability and Reproducibility of Ordinal Classifications." *Technometrics*, 52(1), 94-106.

`hist.ordinalRR` *Histogram for the latent part distributions from a Bayesian ordinal R&R analysis.*

Description

This accepts an input of class 'ordinalRR' and constructs a plot of overlaid density estimates for the latent part qualities X[i] for i=1,2,...,I,I+1. Part I+1 is a new/hypothetical part with the standard normal distribution. For an example, run this package's demo or see Figure 4 from Culp, Ryan, Chen, and Hamada (2018).

Usage

```
## S3 method for class 'ordinalRR'
hist(x,x.low=-4,x.high=4,col="grey",...)
```

Arguments

`x`	'ordinalRR' output from function ordinalRR().
`x.low`	'scalar' lower bound for density support.
`x.high`	'scalar' upper bound for density support.
`col`	'vector' colors for density curves (can be a vector of color names or numbers of length 1 or I+1).
`...`	mop up additional inputs.

Author(s)

Ken Ryan

References

Culp, S.L., Ryan, K.J., Chen, J., and Hamada, M.S. (2018). "Analysis of Repeatability and Reproducibility Studies with Ordinal Measurements". *Technometrics*, doi:10.1080/00401706.2018.1429317.

See Also

`ordinalRR`

Examples

```
demo(ordinalRR)
```

make.rater *Format the parameters for a single rater.*

Description

This function accepts fixed parameters for the De Mast-Van Wieringen Model of a single rater and formats them, so meaningful parametric functions can be displayed with the plot() function.

Usage

```
make.rater(alpha,cutpoints)
```

Arguments

alpha	'positive scalar' the rater's discrimination parameter.
cutpoints	'vector' H-1 (ordered values) corresponding to an H-point ordinal scale.

Value

rater	'rater' a list of rater parameters

Author(s)

Ken Ryan

References

de Mast, J. and van Wieringen, W.N. (2010). "Modeling and Evaluating Repeatability and Reproducibility of Ordinal Classifications." *Technometrics*, 52(1), 94-106.

See Also

compute.q

Examples

```
(rater1=make.rater(1,c(-1.7,-0.5,1.6))) #3 cutpoints, so H=4
(prob=compute.q(rater1,.1)) #probabilities of 1,2,3,4 if part quality is x=.1
sum(prob) #should sum to one
```

`ordinalRR` *Fit a Bayesian ordinal R&R model using JAGS.*

Description

This function can fit either the fixed- or random-effects model from Section 3 of Culp, Ryan, Chen, and Hamada (2018). The outputted class is of type 'ordinalRR', and there are S3 generic functions (e.g., plot and density) for class 'ordinalRR' that make graphs like those from the referenced paper. The user can also use the posterior sample for a customized Bayesian data analysis; see the value list for details on how the posterior sample from JAGS is outputted.

Usage

```
ordinalRR(x, random = TRUE, control = ordinalRR.control())
```

Arguments

`x`	'list' output from function preprocess() containing design parameters and the multinomial counts.
`random`	'Boolean' True for a random-effects model (Section 3.1 of Culp, Ryan, Chen, and Hamada, 2018); False for fixed effect (Section 3.2)
`control`	'list' contains random-effect hyperparameters for the prior and JAGS MCMC parameters.

Value

`post`	'mcmc.list' Bayesian model fit from JAGS including the posterior (i.e., a matrix with samplecontrol$rjag.B rows and one column for each parameter).
`x`	'mcmc' the columns X[1],...,X[I] from the posterior sample for the latent parts.
`a`	'mcmc' the columns alpha[1],...,alpha[J] from the posterior sample for the discrimination parameters.
`d`	'mcmc' the columns delta[1,1:(H-1)],...,delta[J,1:(H-1)] from the posterior sample for the cutpoints.
`mu.a`	'mcmc' column mu.alpha from the posterior sample for the mean of the normal log(alpha[J+1]) distribution (only outputed when random=TRUE).
`sigma.a`	'mcmc' column sigma.alpha from the posterior sample for the standard deviation of the normal log(alpha[J+1]) distribution (only outputed when random=TRUE).
`lambda`	'mcmc' columns lambda[1],...,lambda[H] from the posterior sample for the Dirichlet distribution used to induce a distribution on cutpoints delta[J+1,1:H] (only outputed when random=TRUE).
`...`	remaining internal outputs used for necessary processing.

Author(s)

Ken Ryan

References

Culp, S.L., Ryan, K.J., Chen, J., and Hamada, M.S. (2018). "Analysis of Repeatability and Reproducibility Studies with Ordinal Measurements." *Technometrics*, doi:10.1080/00401706.2018.1429317.

Plummer, M. (2016). "RJAGS: Bayesian Graphical Models using MCMC." R Package Version 4-6, https://CRAN.R-project.org/package=rjags.

Plummer, M. (2017). "JAGS: A Program for Analysis of Bayesian Graphical Models using Gibbs Sampling." Version 4.3.0, http://mcmc-jags.sourceforge.net.

See Also

preprocess ordinalRR.control

Examples

```
data(followup)
followup
x=preprocess(followup)
g.random<-ordinalRR(x)
g.fixed<-ordinalRR(x,random=FALSE)
```

---

ordinalRR.control *Set control parameters for a Bayesian ordinal R&R model.*

Description

The R-function 'ordinalRR.control' sets various control parameters for the prior if using the random-effects version of the model and for the MCMC. See Section 3.2 of Culp, Ryan, Chen, and Hamada (2018) for more on this prior. Default settings match this paper.

Usage

```
ordinalRR.control(mu.mu.alpha = 0.8, tau.mu.alpha = 0.4, mu.tau.alpha = 4,
tau.tau.alpha = 0.4, mu.lambda = 2, tau.lambda = 0.2, rjags.B = 10000L,
rjags.Burn = 1000L, rjags.n.chains = 1L, rjags.n.adapt = 5000L,r.seed=10L,rjags.seed=10L)
```

Arguments

mu.mu.alpha	'positive scalar' mean of the normal prior for mu.alpha.
tau.mu.alpha	'positive scalar' precision=1/variance of the normal prior for mu.alpha.
mu.tau.alpha	'positive scalar' mean of the log-normal prior for tau.alpha.
tau.tau.alpha	'positive scalar' precision of the log-normal prior for tau.alpha.
mu.lambda	'positive scalar' mean of the log-normal prior for the lambda.h.
tau.lambda	'positive scalar' precision of the log-normal prior for the lambda.h.
rjags.B	'positive integer' length of JAGS MCMC chain retained.
rjags.Burn	'positive integer' length of initial JAGS MCMC chain burnin discarded.
rjags.n.chains	'1' number of JAGS MCMC chains (currently only programmed to accept 1).
rjags.n.adapt	'positive integer' rjags n.adapt parameter within command jag.model().
r.seed	'positive integer' sets seed within R during function ordinalRR(). This is for predictive inference on a new rater which is only used with the random-effects model. This does not fix the JAGS seed for the MCMC.
rjags.seed	'positive integer' sets seed within JAGS for the posterior sample from function ordinalRR().

Author(s)

Ken Ryan

References

Culp, S.L., Ryan, K.J., Chen, J., and Hamada, M.S. (2018). "Analysis of Repeatability and Reproducibility Studies with Ordinal Measurements." *Technometrics*, doi:10.1080/00401706.2018.1429317.

Plummer, M. (2016). "RJAGS: Bayesian Graphical Models using MCMC." R Package Version 4-6, https://CRAN.R-project.org/package=rjags.

Plummer, M. (2017). "JAGS: A Program for Analysis of Bayesian Graphical Models using Gibbs Sampling." Version 4.3.0, http://mcmc-jags.sourceforge.net.

---

ordinalRR.sim *Simulate an ordinal R&R data set.*

Description

This function accepts design parameters and model parameters for the random-effects model from Section 3 of Culp, Ryan, Chen, and Hamada (2018) and outputs a simulated R&R data set in preprocessed form.

Usage

```
ordinalRR.sim(H=4L,I=30L,J=3L,K=2L,mu.a=2.6,sigma.a=.2,lambda=c(11,44,29,40),seed=10L)
```

Arguments

H	'positive integer' length of the H-point ordinal scale on {1,...,H}.
I	'positive integer' number of parts.
J	'positive integer' number of raters.
K	'positive integer' number of repetitions per rater.
mu.a	'scalar' mean of log(alpha_j).
sigma.a	'positive scalars' standard deviation of log(alpha_j).
lambda	'vector of length H of positive scalars' Dirichlet parameters used to induce a distribution on cutpoints.
seed	'positive integer' used to set R's seed for pseudo-random number generation.

Value

I	'positive integer' number of parts.
J	'positive integer' number of raters.
K	'positive integer' number of repetitions per rater.
H	'positive integer' length of the H-point ordinal scale on {1,...,H}.
x	'data.frame' containing J*K columns and entries from the H-point ordinal scale 1:H. Each part is a row, and there are blocks of K adjacent columns for the repetitions of each of J raters, e.g., rater 1's columns are the first K and rater J's columns are the last K.
R	'array' x is expanded into a 3-dimensional array (i.e., part 1:I, operator 1:J, ordinal value 1:H) with multinomial counts.
preprocess	'Boolean' will be TRUE if the data are ready for input into function ordinaRR() for Bayesian analysis with JAGS.

Author(s)

Ken Ryan

References

Culp, S.L., Ryan, K.J., Chen, J., and Hamada, M.S. (2018). "Analysis of Repeatability and Reproducibility Studies with Ordinal Measurements." *Technometrics*, doi:10.1080/00401706.2018.1429317.

See Also

preprocess

Examples

```
ordinalRR.sim()
```

`preprocess` *Format an ordinal R&R data frame into object required by function ordinalRR.*

Description

This function accepts a concise data frame containing the ordinal responses from an R&R study and expands it into the necessary multinomial counts, so it can be an input to function ordinalRR() for Bayesian data analysis.

Usage

```
preprocess(x, J=3, K=2, H=4)
```

Arguments

`x`	'data.frame' containing J*K columns and entries from the H-point ordinal scale 1:H. The required format is a row for each part and blocks of K adjacent columns for the repetitions of each of J raters, e.g., rater 1's columns are the first K and rater J's columns are the last K.
`J`	'positive integer' defining the number of raters.
`K`	'positive integer' defining the number of repetitions per rater.
`H`	'positive integer' defining the H-point ordinal scale on {1,...,H}.

Value

`I`	'positive integer' number of parts.
`J`	'positive integer' number of raters.
`K`	'positive integer' number of repetitions per rater.
`H`	'positive integer' length of the H-point ordinal scale on {1,...,H}.
`x`	'data.frame' containing J*K columns and entries from the H-point ordinal scale 1:H. Each part is a row, and there are blocks of K adjacent columns for the repetitions of each of J raters, e.g., rater 1's columns are the first K and rater J's columns are the last K.
`R`	'array' x is expanded into a 3-dimensional array (i.e., part 1:I, operator 1:J, ordinal value 1:H) with multinomial counts.
`preprocess`	'Boolean' will be TRUE if the data are ready for input into function ordinaRR() for Bayesian analysis with JAGS.

Author(s)

Ken Ryan

See Also

`followup`

Examples

```
data(followup)
followup
preprocess(followup)
```

`print.ordinalRR` *Print function for an object of class ordinalRR.*

Description

This standard print function displays: the function call to ordinalRR() that produced this object and descriptions of the ordinal R&R data and its Bayesian posterior sample from JAGS.

Usage

```
## S3 method for class 'ordinalRR'
print(x,...)
```

Arguments

x	'ordinalRR' output from function ordinalRR().
...	mop up additional inputs.

Author(s)

Ken Ryan

See Also

ordinalRR

Examples

```
data(followup)
followup
x=preprocess(followup)
g.random<-ordinalRR(x)
print(g.random)
```

`summary.ordinalRR` *Summarize an object of class ordinalRR.*

Description

This function displays: the original call to ordinalRR(), a summary of the ordinal repeatability and reproducibility (R&R) data and its Bayesian posterior sample from JAGS, a table with point estimates of repeatability and model parameters for each rater, and a table with point estimates of R&R for each pair of raters. The estimates for repeatability and R&R are the basic/simple statistics defined in Equations (24) and (25) of Culp, Ryan, Chen, and Hamada (2018). Estimated repeatability is the proportion of matches on all pairs of repetitions for a given rater on a given part across all parts. Estimated R&R is the proportion of matches on all pairs of repetitions for a given pair of raters on a given part across all parts. On the other hand, reported parameter estimates for the De Mast-Van Wieringen Model (i.e., discrimination parameter a and cutpoints d) are the posterior medians based on a Bayesian fixed- or random-effects extension from Culp, Ryan, Chen, and Hamada (2018).

Usage

```
## S3 method for class 'ordinalRR'
summary(object,decimals=1,...)
```

Arguments

`object`	'ordinalRR' output from function ordinalRR().
`decimals`	'positive integer' number of decimals to report model parameter estimates (repeatability and R&R are reported to two additional places).
`...`	mop up additional inputs.

Author(s)

Ken Ryan

References

Culp, S.L., Ryan, K.J., Chen, J., and Hamada, M.S. (2018). "Analysis of Repeatability and Reproducibility Studies with Ordinal Measurements." *Technometrics*, doi:10.1080/00401706.2018.1429317.

de Mast, J. and van Wieringen, W.N. (2010). "Modeling and Evaluating Repeatability and Reproducibility of Ordinal Classifications." *Technometrics*, 52(1), 94-106.

See Also

`ordinalRR`

Examples

```
data(followup)
followup
x=preprocess(followup)
g.random<-ordinalRR(x)
summary(g.random)
```

Index

∗ **classes**
    ordinalRR, 7
    ordinalRR.control, 8
∗ **datasets**
    followup, 4
∗ **methods**
    ordinalRR, 7
    ordinalRR.control, 8
∗ **models**
    ordinalRR, 7
    ordinalRR.control, 8

compute.q, 2, *6*

density.ordinalRR, 3

followup, 4, *11*

hist.ordinalRR, 5

make.rater, *3*, 6

ordinalRR, *3*, *5*, 7, *12, 13*
ordinalRR.control, 8, *8*
ordinalRR.sim, 9

preprocess, *8*, *10*, 11
print.ordinalRR, 12

summary.ordinalRR, 13