Package ‘endorse’

February 19, 2015

Version 1.4.1

Date 2014-05-03

Title R Package for Analyzing Endorsement Experiments

Author Yuki Shiraito <shiraito@princeton.edu>, Kosuke Imai <kimai@princeton.edu>

Maintainer Yuki Shiraito <shiraito@princeton.edu>

Depends coda, utils

Description This R package implements the statistical model proposed by Bullock, Imai, and Shapiro (2011; Political Analysis) to analyze endorsement experiments. Endorsement experiments are a survey methodology for eliciting truthful responses to sensitive questions. This methodology is helpful when measuring support for socially sensitive political actors such as militant groups. The model is fitted with the Markov chain Monte Carlo algorithm and produces the output containing draws from the posterior distribution.

LazyLoad yes

LazyData yes

License GPL (>= 2)

URL http://imai.princeton.edu/software/endorse.html

NeedsCompilation yes

Repository CRAN

Date/Publication 2014-05-04 07:40:37

R topics documented:

endorse *Fitting the Measurement Model of Political Support via Markov Chain Monte Carlo*

Description

This function generates a sample from the posterior distribution of the measurement model of political support. Individual-level covariates may be included in the model. The details of the model are given under 'Details'. See also Bullock et al. (2011).

Usage

```
endorse(Y, data, data.village = NA, village = NA, treat = NA,
        na.strings = 99, identical.lambda = TRUE, covariates = FALSE,
        formula.indiv = NA, hierarchical = FALSE, formula.village = NA,
        x.start = 0, s.start = 0, beta.start = 1, tau.start = NA,
        lambda.start = 0, omega2.start = .1, theta.start = 0,
        phi2.start = .1, kappa.start = 0, psi2.start = 1,
        delta.start = 0, zeta.start = 0, rho2.start = 1, mu.beta = 0,
        mu.x = 0, mu.theta = 0, mu.kappa = 0, mu.delta = 0, mu.zeta = 0,
        precision.beta = 0.04, precision.x = 1, precision.theta = 0.04,
        precision.kappa = 0.04, precision.delta = 0.04,
        precision.zeta = 0.04, s0.omega2= 1, nu0.omega2 = 10,
        s0.phi2 = 1, nu0.phi2 = 10, s0.psi2 = 1, nu0.psi2 = 10,
        s0.sig2 = 1, nu0.sig2 = 400, s0.rho2 = 1, nu0.rho2 = 10,
        MCMC = 20000, burn = 1000, thin = 1,  mh = TRUE, prop = 0.001,
        x.sd = TRUE, tau.out = FALSE, s.out = FALSE, omega2.out = TRUE,
        phi2.out = TRUE, psi2.out = TRUE, verbose = TRUE,
        seed.store = FALSE, update = FALSE, update.start = NULL)
```

Arguments

`Y` a list of the variable names for the responses. It should take the following form: \ `list(Q1 = c("varnameQ1.1", ...), Q2 =    c("varnameQ2.1", ...), ...)`. If `treat` is `NA`, the first variable for each question should be the responses of the control observations while each of the other variables should correspond to each endorser. `treat` should be supplied if only one variable name is provided for a question in this argument.

`data` data frame containing the individual-level variables. The cases must be complete, i.e., no `NA`'s are allowed.

`data.village` data frame containing the village-level variables. The cases must be complete, i.e., no `NA`'s are allowed.

`village` character. The variable name of the village indicator in the individual-level data.

`treat` An optional matrix of non negative integers indicating the treatment status of each observation and each question. Rows are observations and columns are questions. 0 represents the control status while positive integers indicate treatment statuses. If `treat` is set to `NA`, the function generates the treatment matrix using `Y`. The default is `NA`.

`na.strings` a scalar or a vector indicating the values of the response variable that are to be interpreted as "Don't Know" or "Refused to Answer." The value should not be `NA` unless `treat` is provided, because `NA`'s are interpreted as the response to the question with another endorsement. Default is `99`.

`identical.lambda` logical. If `TRUE`, the model with a common lambda across questions will be fitted. The default is `TRUE`.

`covariates` logical. If `TRUE`, the model includes individual-level covariates. The default is `FALSE`.

`formula.indiv` a symbolic description specifying the individual level covariates for the support parameter and the ideal points. The formula should be one-sided, e.g. `~ Z1 + Z2`.

`hierarchical` logical. IF `TRUE`, the hierarchical model with village level predictors will be fitted. The default is `FALSE`.

`formula.village` a symbolic description specifying the village level covariates for the support parameter and the ideal points. The formula should be one-sided, e.g. `~ V1 + V2`.

`x.start` starting values for the ideal points vector x. If `x.start` is set to a scalar, the starting values for the ideal points of all respondents will be set to the scalar. If `x.start` is a vector of the same length as the number of observations, then this vector will be used as the starting values. The default is `0`.

`s.start` starting values for the support parameter, s_ijk. If `s.start` is set to a scalar, the starting values for the support parameter of all respondents and all questions will be the scalar. If `s.start` is set to a matrix, it should have the same number of rows as the number of observations and the same number of columns as the number of questions. Also, the value should be zero for the control condition. The default is `0`.

`beta.start` starting values for the question related parameters, α_j and β_j. If `beta.start` is set to a scalar, the starting values for the support parameter of all respondents and all questions will be the scalar. If `beta.start` is set to a matrix, the number of rows should be the number of questions and the number of columns should be 2. The first column will be the starting values for α_j and the second column will be the starting values for β_j. Since the parameter values are constrained to be positive, the starting values should be also positive. The default is `1`.

`tau.start` starting values for the cut points in the response model. If `NA`, the function generates the starting values so that each interval between the cut points is 0.5. If `tau.start` is set to a matrix, the number of rows should be the same as the number of questions and the number of columns should be the maximum value of the number of categories in the responses. The first cut point for each question should be set to 0 while the last one set to the previous cut point plus 1000. The default is `NA`.

`lambda.start` starting values for the coefficients in the support parameter model, λ_jk. If `lambda.start` is set to a scalar, the starting values for all coefficients will be the scalar. If `lambda.start` is set to a matrix, the number of rows should be the number of the individual level covariates (plus the number of villages, if the model is hierarchical), and the number of columns should be the number of endorsers (times the number of questions, if the model is with varying lambdas). The default is `0`.

`omega2.start` starting values for the variance of the support parameters, ω^2_{jk}. If `omega2.start` is set to a scalar, the starting values for $omega^2_{jk}$ will be the diagonal matrix with the diagonal elements set to the scalar. If `omega2.start` is set to a matrix, the number of rows should be the number of questions, while the number of columns should be the same as the number of endorsers. The default is `.1`.

`theta.start` starting values for the means of the λ_{jk} for each endorser. If `theta.start` is set to a scalar, the starting values for all parameters will be the scalar. If `theta.start` is set to a matrix, the number of rows should be the number of endorsers and the number of columns should be the dimension of covariates. The default is `0`.

`phi2.start` starting values for the covariance matrices of the coefficients of the support parameters, Φ_k. Φ_k is assumed to be a diagonal matrix. If `phi2.start` is set to a scalar, the starting values for all covariance matrices will be the same diagonal matrix with the diagonal elements set to the scalar. If `phi2.start` is set to a vector, the length should be the number of endorsers times the dimension of covariates. The default is `.1`.

`kappa.start` starting values for the coefficients on village level covariates in the support parameter model, κ_k. If `kappa.start` is set to a scalar, the starting values for all coefficients will be the scalar. If `kappa.start` is set to a matrix, the number of rows should be the number of the village level covariates, and the number of columns should be the number of endorsers (times the number of questions, if the varying-lambda model is fitted). The default is `0`.

`psi2.start` starting values for the variance of the village random intercepts in the support parameter model, ψ^2_k. If `psi2.start` is set to a scalar, the starting values for ψ^2_k will be the diagonal matrix with the diagonal elements set to the scalar. If `psi2.start` is set to a vector, its length should be the number of endorsers (times the number of questions, if the varying-lambda model is fitted). The default is `.1`.

`delta.start` starting values for the coefficients on individual level covariates in the ideal point model. Will be used only if `covariates = TRUE`. If `delta.start` is set to a scalar, the starting values for all coefficients will be the scalar. If `delta.start` is set to a vector, the length should be the dimension of covariates. The default is `0`.

`zeta.start` starting values for the coefficients on village level covariates in the ideal point model. Will be used only if `covariates = TRUE`. If `zeta.start` is set to a scalar, the starting values for all coefficients will be the scalar. If `zeta.start` is set to a vector, the length should be the dimension of covariates. The default is `0`.

`rho2.start` numeric. starting values for the variance of the village random intercepts in the ideal point model, ρ^2. The default is `1`.

mu.beta — the mean of the independent Normal prior on the question related parameters. Can be either a scalar or a matrix of dimension the number of questions times 2. The default is `0`.

mu.x — the mean of the independent Normal prior on the question related parameters. Can be either a scalar or a vector of the same length as the number of observations. The default is `0`.

mu.theta — the mean of the independent Normal prior on the mean of the coefficients in the support parameter model. Can be either a scalar or a vector of the same length as the dimension of covariates. The default is `0`.

mu.kappa — the mean of the independent Normal prior on the coefficients of village level covariates. Can be either a scalar or a matrix of dimension the number of covariates times the number of endorsers. The default is `0`.

mu.delta — the mean of the independent Normal prior on the the coefficients in the ideal point model. Can be either a scalar or a vector of the same length as the dimension of covariates. The default is `0`.

mu.zeta — the mean of the independent Normal prior on the the coefficients of village level covariates in the ideal point model. Can be either a scalar or a vector of the same length as the dimension of covariates. The default is `0`.

precision.beta — the precisions (inverse variances) of the independent Normal prior on the question related parameters. Can be either a scalar or a 2×2 diagonal matrix. The default is `0.04`.

precision.x — scalar. The known precision of the independent Normal distribution on the ideal points. The default is `1`.

precision.theta — the precisions of the independent Normal prior on the means of the coefficients in the support parameter model. Can be either a scalar or a vector of the same length as the dimension of covariates. The default is `0.04`.

precision.kappa — the precisions of the independent Normal prior on the coefficients of village level covariates in the support parameter model. Can be either a scalar or a vector of the same length as the dimension of covariates. The default is `0.04`.

precision.delta — the precisions of the independent Normal prior on the the coefficients in the ideal point model. Can be either a scalar or a square matrix of the same dimension as the dimension of covariates. The default is `0.04`.

precision.zeta — the precisions of the independent Normal prior on the the coefficients of village level covariates in the ideal point model. Can be either a scalar or a square matrix of the same dimension as the dimension of covariates. The default is `0.04`.

s0.omega2 — scalar. The scale of the independent scaled inverse- chi-squared prior for the variance parameter in the support parameter model. The default is `1`.

nu0.omega2 — scalar. The degrees of freedom of the independent scaled inverse-chi-squared prior for the variance parameter in the support parameter model. The default is `10`.

s0.phi2 — scalar. The scale of the independent scaled inverse-chi-squared prior for the variances of the coefficients in the support parameter model. The default is `1`.

`nu0.phi2`	scalar. The degrees of freedom of the independent scaled inverse-chi-squared prior for the variances of the coefficients in the support parameter model. The default is `10`.
`s0.psi2`	scalar. The scale of the independent scaled inverse-chi-squared prior for the variances of the village random intercepts in the support parameter model. The default is `1`.
`nu0.psi2`	scalar. The degrees of freedom of the independent scaled inverse-chi-squared prior for the variances of the village random intercepts in the support parameter model. The default is `10`.
`s0.sig2`	scalar. The scale of the independent scaled inverse-chi-squared prior for the variance parameter in the ideal point model. The default is `1`.
`nu0.sig2`	scalar. The degrees of freedom of the independent scaled inverse-chi-squared prior for the variance parameter in the ideal point model. The default is `400`.
`s0.rho2`	scalar. The scale of the independent scaled inverse-chi-squared prior for the variances of the village random intercepts in the ideal point model. The default is `1`.
`nu0.rho2`	scalar. The degrees of freedom of the independent scaled inverse-chi-squared prior for the variances of the village random intercepts in the ideal point model. The default is `10`.
`MCMC`	the number of iterations for the sampler. The default is `20000`.
`burn`	the number of burn-in iterations for the sampler. The default is `1000`.
`thin`	the thinning interval used in the simulation. The default is `1`.
`mh`	logical. If `TRUE`, the Metropolis-Hastings algorithm is used to sample the cut points in the response model. The default is `TRUE`.
`prop`	a positive number or a vector consisting of positive numbers. The length of the vector should be the same as the number of questions. This argument sets proposal variance for the Metropolis-Hastings algorithm in sampling the cut points of the response model. The default is `0.001`.
`x.sd`	logical. If `TRUE`, the standard deviation of the ideal points in each draw will be stored. If `FALSE`, a sample of the ideal points will be stored. *NOTE: Because storing a sample takes an enormous amount of memory, this option should be selected only if the chain is thinned heavily or the data have a small number of observations.*
`tau.out`	logical. A switch that determines whether or not to store the cut points in the response model. The default is `FALSE`.
`s.out`	logical. If `TRUE`, the support parameter for each respondent and each question will be stored. The default is `FALSE`. *NOTE: Because storing a sample takes an enormous amount of memory, this option should be selected only if the chain is thinned heavily or the data have a small number of observations.*
`omega2.out`	logical. If `TRUE`, the variannce parameter of the support parameter model will be stored. The default is `TRUE`.
`phi2.out`	logical. If `TRUE`, the variannce parameter of the model for the coefficients in the support parameter model will be stored. The default is `TRUE`.

`psi2.out`	logical. If `TRUE`, the variance of the village random intercepts in the support parameter model will be stored. The default is `TRUE`.
`verbose`	logical. A switch that determines whether or not to print the progress of the chain and Metropolis acceptance ratios for the cut points of the response model. The default is `TRUE`.
`seed.store`	logical. If `TRUE`, the seed will be stored in order to update the chain later. The default is `FALSE`.
`update`	logical. If `TURE`, the function is run to update a chain. The default is `FALSE`.
`update.start`	list. If the function is run to update a chain, the output object of the previous run should be supplied. The default is `NULL`.

Details

The model takes the following form:

Consider an endorsement experiment where we wish to measure the level of support for K political actors. In the survey, respondents are asked whether or not they support each of J policies chosen by researchers. Let Y_{ij} represent respondent i's answer to the survey question regarding policy j. Suppose that the response variable Y_{ij} is the ordered factor variable taking one of L_j levels, i.e., $Y_{ij} \in \{0, 1, \ldots, L_j - 1\}$ where $L_j > 1$. We assume that a greater value of Y_{ij} indicates a greater level of support for policy j. We denote an M dimensional vector of the observed characteristics of respondent i by Z_i.

In the experiment, we randomly assign one of K political actors as an endorser to respondent i's question regarding policy j and denote this treatment variable by $T_{ij} \in \{0, 1, \ldots, K\}$. We use $T_{ij} = 0$ to represent the control observations where no political endorsement is attached to the question. Alternatively, one may use the endorsement by a neutral actor as the control group.

The model for the response variable, Y_{ij}, is given by,

$$Y_{ij} = l \text{ if } \tau_l < Y^*_{ij} \le \tau_{l+1},$$

$$Y^*_{ij} \mid T_{ij} = k \sim \mathcal{N}(-\alpha_j + \beta_j(x_i + s_{ijk}),\ I)$$

where $l \in \{0, 1, \ldots, L_j\}, \tau_0 = -\infty < \tau_1 = 0 < \tau_2 < \ldots < \tau_{L_j} = \infty$. α_j and β_j are assumed to be positive.

The model for the support parameter, s_{ijk}, is given by if $T_{ij} \neq 0$,

$$s_{ijk} \sim \mathcal{N}(Z_i^T \lambda_{jk},\ \omega_{jk}^2)$$

with covariates, and

$$s_{ijk} \sim \mathcal{N}(\lambda_{jk},\ \omega_{jk}^2),$$

without covariates, for $j = 1, \ldots, J,\ k = 1, \ldots, K$, and if $T_{ij} = 0,\ s_{ijk} = 0$.

The λ's in the support parameter model are modeled in the following hierarchical manner,

$$\lambda_{jk} \sim \mathcal{N}(\theta_k,\ \Phi_k)$$

for $k = 1, \ldots, K$.

If you set `identical.lambda = FALSE` and `hierarchical = TRUE`, the model for s_{ijk} is if $T_{ij} \neq 0$,

$$s_{ijk} \sim \mathcal{N}(\lambda^0_{jk,village[i]} + Z_i^T \lambda_{jk},\ \omega_{jk}^2)$$

and

$$\lambda^0_{jk,village[i]} \sim \mathcal{N}(V^T_{village[i]}\kappa_{jk},\ \psi^2_{jk})$$

for $k = 1, \ldots, K$ and $j = 1, \ldots, J$. In addition, λ and κ are modeled in the following hierarchical manner,

$$\lambda^*_{jk} \sim \mathcal{N}(\theta_k,\ \Phi_k)$$

for $k = 1, \ldots, K$, where $\lambda^*_{jk} = (\lambda^T_{jk}, \kappa^T_{jk})^T$.

If you set `identical.lambda = TRUE` and `hierarchical = TRUE`, the model for s_{ijk} is if $T_{ij} \neq 0$,

$$s_{ijk} \sim \mathcal{N}(\lambda^0_{k,village[i]} + Z^T_i\lambda_k,\ \omega^2_k)$$

and

$$\lambda^0_{k,village[i]} \sim \mathcal{N}(V^T_{village[i]}\kappa_k,\ \psi^2_k)$$

for $k = 1, \ldots, K$.

If the covariates are included in the model, the model for the ideal points is given by

$$x_i \sim \mathcal{N}(Z^T_i\delta,\ \sigma^2_x)$$

for $i = 1, \ldots, N$ where σ^2_x is a known prior variance.

If you set `hierarchical = TRUE`, the model is

$$x_i \sim \mathcal{N}(\delta^0_{village[i]} + Z^T_i\delta,\ \sigma^2)$$

and

$$\delta^0_{village[i]} \sim \mathcal{N}(V^T_{village[i]}\zeta,\ \rho^2)$$

for $k = 1, \ldots, K$.

Finally, the following independent prior distributions are placed on unknown parameters,

$$\alpha_j \sim \mathcal{N}(\mu_\alpha,\ \sigma^2_\alpha)$$

for $j = 1, \ldots, J$,

$$\beta_j \sim \mathcal{TN}_{\beta_j>0}(\mu_\beta,\ \sigma^2_\beta)$$

for $j = 1, \ldots, J$,

$$\delta \sim \mathcal{N}(\mu_\delta,\ \Sigma_\delta),$$

$$\theta_k \sim \mathcal{N}(\mu_\theta,\ \Sigma_\theta)$$

for $k = 1, \ldots, K$,

$$\omega^2_{jk} \sim \text{Inv}-\chi^2(\nu^0_\omega,\ s^0_\omega)$$

for $j = 1, \ldots, J$ and $k = 1, \ldots, K$, and

$$\text{diag}(\Phi_k) \sim \text{Inv}-\chi^2(\nu^0_\Phi,\ s^0_\Phi)$$

for $k = 1, \ldots, K$, where Φ_k is assumed to be a diagonal matrix.

Value

A list containing the following elements:

beta	an mcmc object. A sample from the posterior distribution of α and β.
x	If `x.sd = TRUE`, a vector of the standard deviation of the ideal points in each draw. If `x.sd = FALSE`, an mcmc object that contains a sample from the posterior distribution of the ideal points.
s	If `s.out = TRUE`, an mcmc object that contains a sample from the posterior distribution of s_{ijk}. Variable names are `s(observation index)(question index)`.
delta	If `covariates = TRUE`, an mcmc object that contains a sample from the posterior distribution of δ.
tau	If `tau.out = TRUE`, an mcmc object that contains a sample from the posterior distribution of τ.
lambda	an mcmc object. A sample from the posterior distribution of λ. Variable names are: `\lambda(question index)(group index).(covariate index)`.
theta	an mcmc object. A sample from the posterior distribution of θ.
kappa	an mcmc object.
zeta	an mcmc object.

Note that the posterior sample of all parameters are NOT standardized. In making posterior inference, each parameter should be divided by the standard deviation of x (in the default setting, it is given as "x") or by σ^2 (in the default setting, it is given as "sigma2").

Also note that α and the intercept in δ (or, if the model is hierarchical, the intercept in ζ) are not identified. Instead,

$$-\alpha + \beta * \delta_0$$

or, if the model is hierarchical,

$$-\alpha + \beta * \zeta_0$$

is identified after either of the above standardization, where δ_0 and ζ_0 denote the intercepts.

Author(s)

Kosuke Imai, Department of Politics, Princeton University <kimai@Princeton.Edu>, http://imai.princeton.edu

Yuki Shiraito, Department of Politics, Princeton University <shiraito@Princeton.Edu>.

References

Bullock, Will, Kosuke Imai, and Jacob N. Shapiro. (2011) “Statistical Analysis of Endorsement Experiments: Measuring Support for Militant Groups in Pakistan,” *Political Analysis*, Vol. 19, No. 4 (Autumn), pp.363-384.

Examples

```
## Not run:
data(pakistan)

Y <- list(Q1 = c("Polio.a", "Polio.b", "Polio.c", "Polio.d", "Polio.e"),
          Q2 = c("FCR.a", "FCR.b", "FCR.c", "FCR.d", "FCR.e"),
          Q3 = c("Durand.a", "Durand.b", "Durand.c", "Durand.d",
                 "Durand.e"),
          Q4 = c("Curriculum.a", "Curriculum.b", "Curriculum.c",
                 "Curriculum.d", "Curriculum.e"))



## Varying-lambda non-hierarchical model without covariates
endorse.out <- endorse(Y = Y, data = pakistan, identical.lambda = FALSE,
                       covariates = FALSE, hierarchical = FALSE)

## Varying-lambda non-hierarchical model with covariates
indiv.covariates <- formula( ~ female + rural)
endorse.out <- endorse(Y = Y, data = pakistan, identical.lambda = FALSE,
                       covariates = TRUE, formula.indiv = indiv.covariates,
                       hierarchical = FALSE)

## Common-lambda non-hierarchical model with covariates
indiv.covariates <- formula( ~ female + rural)
endorse.out <- endorse(Y = Y, data = pakistan, identical.lambda = TRUE,
                       covariates = TRUE, formula.indiv = indiv.covariates,
                       hierarchical = FALSE)

## Varying-lambda hierarchical model without covariates
div.data <- data.frame(division = sort(unique(pakistan$division)))
div.formula <- formula(~ 1)
endorse.out <- endorse(Y = Y, data = pakistan, data.village = div.data,
                       village = "division", identical.lambda = FALSE,
                       covariates = FALSE, hierarchical = TRUE,
                       formula.village = div.formula)

## Varying-lambda hierarchical model with covariates
endorse.out <- endorse(Y = Y, data = pakistan, data.village = div.data,
                       village = "division", identical.lambda = FALSE,
                       covariates = TRUE,
                       formula.indiv = indiv.covariates,
                       hierarchical = TRUE,
                       formula.village = div.formula)

## Common-lambda hierarchical model without covariates
endorse.out <- endorse(Y = Y, data = pakistan, data.village = div.data,
                       village = "division", identical.lambda = TRUE,
                       covariates = FALSE, hierarchical = TRUE,
                       formula.village = div.formula)

## Common-lambda hierarchical model with covariates
```

```
endorse.out <- endorse(Y = Y, data = pakistan, data.village = div.data,
                       village = "division", identical.lambda = TRUE,
                       covariates = TRUE,
                       formula.indiv = indiv.covariates,
                       hierarchical = TRUE,
                       formula.village = div.formula)

## End(Not run)
```

---

endorse.plot *Descriptive Plot of Endorsement Experiment Data*

Description

This function creates a descriptive plot for a question in an endorsement experiment.

Usage

```
endorse.plot(Y, data, scale, dk = 98, ra = 99, yaxis = NULL, col.seq = NA)
```

Arguments

Y	a character vector. List of the variable names for the responses to a question. Each variable name corresponds to each treatment status.
data	data frame containing the variables.
scale	an integer. The scale of the responses. The function assumes that the responses are coded so that 1 indicates the lowest support while the integer specified in this argument represents the highest support.
dk	an integer indicating the value of the response variable that is to be interpreted as "Don't Know." Default is 98.
ra	an integer indicating the value of the response variable that is to be interpreted as "Refused." Default is 99.
yaxis	a character vector of the same length as Y. The argument will be used for the label of the horizontal axis. The order should be the same as Y.
col.seq	a vector of colors for the bars or bar components. By default, a gradation of gray where the darkest indicates the highest support level.

Value

A descriptive plot for the responses to a question.

Author(s)

Kosuke Imai, Department of Politics, Princeton University <kimai@Princeton.Edu>, http://imai.princeton.edu

Yuki Shiraito, Department of Politics, Princeton University <shiraito@Princeton.Edu>.

Examples

```
data(pakistan)

Y <- c("Polio.a", "Polio.b", "Polio.c", "Polio.d", "Polio.e")
yaxis <- c("Control", "Kashmir", "Afghan", "Al-Qaida", "Tanzeems")

endorse.plot(Y = Y, data = pakistan, scale = 5)
```

GeoCount *Counting Violent Incidents around Villages*

Description

This function calculates the number of violent incidents within a specified distance around specified points (villages).

Usage

```
GeoCount(x, y, distance, x.latitude = "latitude",
                      x.longitude = "longitude", y.latitude = "latitude",
                      y.longitude = "longitude")
```

Arguments

x	data frame containing the longitude and the latitude of villages.
y	data frame containing the longitude and the latitude of violent incidents.
distance	numeric. The distance from villages in kilometers.
x.latitude	character. The variable name for the latitude in x.
x.longitude	character. The variable name for the longitude in x.
y.latitude	character. The variable name for the latitude in y.
y.longitude	character. The variable name for the longitude in y.

Author(s)

Kosuke Imai, Department of Politics, Princeton University <kimai@Princeton.Edu>, http://imai.princeton.edu

Yuki Shiraito, Department of Politics, Princeton University <shiraito@Princeton.Edu>.

pakistan *Pakistan Survey Experiment on Support for Militant Groups*

Description

This data set is a subset of the data from the endorsement experiment conducted in Pakistan to study support for militant groups. The survey was implemented by Fair et al. (2009). It is also used by Bullock et al. (2011).

Usage

```
data(pakistan)
```

Format

A data frame containing 5212 observations. The variables are:

- `division`: division number.
- `edu`: education. 1 if "illiterate"; 2 if "primary"; 3 if "middle"; 4 if "matric"; 5 if "intermediate (f.a/f.sc)," "graduate (b.a/b.sc.)," or "professionals (m.a /or other professional degree)."
- `inc`: approximate monthly income. 1 if less than 3000 rupees; 2 if 3000 to 10,000 rupees; 3 if 10,001 to 15,000 rupees; 4 if more than 15,000 rupees.
- `female`: 0 if male; 1 if female
- `rural`: 0 if rural; 1 if urban
- `Polio.a-e`: support for World Health Organization's plan of universal polio vaccinations in Pakistan. 5 indicates the highest support while 1 indicates the lowest support.
- `FCR.a-e`: support for the reform of the Frontier Crimes Regulation (FCR) governing the tribal areas. 5 indicates the highest support while 1 indicates the lowest support.
- `Durand.a-e`: support for using peace jirgas to resolve disputes over the Afghan border, the Durand Line. 5 indicates the highest support while 1 indicates the lowest support.
- `Curriculum.a-e`: support for the Government of Pakistan's plan of curriculum reforms in religious schools or *madaris*. 5 indicates the highest support while 1 indicates the lowest support.

For the response variables, endorsers are:

- `varname.a`: control (no endorsement).
- `varname.b`: Pakistani militant groups in Kashmir.
- `varname.c`: Militants fighting in Afghanistan.
- `varname.d`: Al-Qaida.
- `varname.e`: Firqavarana Tanzeems.

Source

Bullock, Will, Kosuke Imai, and Jacob N. Shapiro. 2011. Replication data for: Statistical analysis of endorsement experiments: Measuring support for militant groups in Pakistan. hdl:1902.1/14840. The Dataverse Network.

References

Bullock, Will, Kosuke Imai, and Jacob N. Shapiro. (2011) "Statistical Analysis of Endorsement Experiments: Measuring Support for Militant Groups in Pakistan," *Political Analysis*, Vol. 19, No. 4 (Autumn), pp.363-384.

Fair, Christin C., Neil Malhotra, and Jacob N. Shapiro. (2009) "The Roots of Militancy: Explaining Support for Political Violence in Pakistan," Working Paper, Princeton University.

Index