model.matrix.multiple: Design matrix for multi-0.1variate models

Description

Use model.matrix.multiple after parse.formula to create a design matrix for multipleequation models.

Usage

```
model.matrix.multiple(object, data, shape = "compact", eqn = NULL, ...)
```

Arguments

object the list of formulas output from parse.formula

data a data frame created with model.frame.multiple

shape a character string specifying the shape of the outputed matrix. Available

options are

"compact" (default) the output matrix will be an $n \times v$, where v is the number of unique variables in all of the equations (including the intercept term)

"array" the output is an $n \times K \times J$ array where J is the total number of equations and K is the total number of parameters across all the equations. If a variable is not in a certain equation, it is observed as a vector of 0s.

"stacked" the output will be a $2n \times K$ matrix where K is the total number of parameters across all the equations.

eqn

a character string or a vector of character strings identifying the equations from which to construct the design matrix. The defaults to NULL, which only uses the systematic parameters (for which DepVar = TRUE in the

appropriate describe.model function)

additional arguments passed to model.matrix.default

Value

A design matrix or array, depending on the options chosen in shape, with appropriate terms attributes.

Author(s)

Kosuke Imai <\kimai@princeton.edu\>; Gary King <\king@harvard.edu\>; Olivia Lau <(olau@fas.harvard.edu)>; Ferdinand Alimadhi <(falimadhi@iq.harvard.edu)>

See Also

parse.par, parse.formula and the full Zelig manual at http://gking.harvard.edu/zelig

Examples

```
# Let's say that the name of the model is "bivariate.probit", and
# the corresponding describe function is describe.bivariate.probit(),
# which identifies mu1 and mu2 as systematic components, and an
# ancillary parameter rho, which may be parameterized, but is estimated
# as a scalar by default. Let par be the parameter vector (including
# parameters for rho), formulae a user-specified formula, and mydata
# the user specified data frame.
# Acceptable combinations of parse.par() and model.matrix() are as follows:
## Setting up
## Not run:
data(sanction)
formulae <- cbind(import, export) ~ coop + cost + target</pre>
fml <- parse.formula(formulae, model = "bivariate.probit")</pre>
D <- model.frame(fml, data = sanction)</pre>
terms <- attr(D, "terms")</pre>
## Intuitive option
Beta <- parse.par(par, terms, shape = "vector", eqn = c("mu1", "mu2"))</pre>
X <- model.matrix(fml, data = D, shape = "stacked", eqn = c("mu1", "mu2")</pre>
eta <- X
## Memory-efficient (compact) option (default)
Beta <- parse.par(par, terms, eqn = c("mu1", "mu2"))</pre>
X <- model.matrix(fml, data = D, eqn = c("mu1", "mu2"))</pre>
eta <- X
## Computationally-efficient (array) option
Beta <- parse.par(par, terms, shape = "vector", eqn = c("mu1", "mu2"))</pre>
X <- model.matrix(fml, data = D, shape = "array", eqn = c("mu1", "mu2"))</pre>
eta \leftarrow apply(X, 3, '
## End(Not run)
```