0.1 `parse.par`: Select and reshape parameter vectors

Description

The `parse.par` function reshapes parameter vectors for comfortability with the output matrix from `model.matrix.multiple`. Use `parse.par` to identify sets of parameters; for example, within optimization functions that require vector input, or within `qi` functions that take matrix input of all parameters as a lump.

Usage

```
parse.par(par, terms, shape = "matrix", eqn = NULL)
```

Arguments

`par`	the vector (or matrix) of parameters
`terms`	the terms from either `model.frame.multiple` or `model.matrix.multiple`
`shape`	a character string (either `"matrix"` or `"vector"`) that identifies the type of output structure
`eqn`	a character string (or strings) that identify the parameters that you would like to subset from the larger `par` structure

Value

A matrix or vector of the sub-setted (and reshaped) parameters for the specified parameters given in `"eqn"`. By default, `eqn = NULL`, such that all systematic components are selected. (Systematic components have `ExpVar = TRUE` in the appropriate `describe.model` function.)

If an ancillary parameter (for which `ExpVar = FALSE` in `describe.model`) is specified in `eqn`, it is always returned as a vector (ignoring `shape`). (Ancillary parameters are all parameters that have intercept only formulas.)

Author(s)

Kosuke Imai ¡⟨kimai@princeton.edu⟩¿; Gary King ¡⟨king@harvard.edu⟩¿; Olivia Lau ¡⟨olau@fas.harvard. Ferdinand Alimadhi ¡⟨falimadhi@iq.harvard.edu⟩¿

See Also

`model.matrix.multiple`, `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
fml <- parse.formula(formulae, model = "bivariate.probit")
D <- model.frame(fml, data = sanction)
terms <- attr(D, "terms")

## Intuitive option
Beta <- parse.par(par, terms, shape = "vector", eqn = c("mu1", "mu2"))
X <- model.matrix(fml, data = D, shape = "stacked", eqn = c("mu1", "mu2")
eta <- X

## Memory-efficient (compact) option (default)
Beta <- parse.par(par, terms, eqn = c("mu1", "mu2"))
X <- model.matrix(fml, data = D, eqn = c("mu1", "mu2"))
eta <- X

## Computationally-efficient (array) option
Beta <- parse.par(par, terms, shape = "vector", eqn = c("mu1", "mu2"))
X <- model.matrix(fml, data = D, shape = "array", eqn = c("mu1", "mu2"))
eta <- apply(X, 3, '
## End(Not run)
```