Package ‘BGGM’

February 23, 2024

Type Package

Title Bayesian Gaussian Graphical Models

Version 2.1.1

Date 2024-02-22

Description Fit Bayesian Gaussian graphical models. The methods are separated into two Bayesian approaches for inference: hypothesis testing and estimation. There are extensions for confirmatory hypothesis testing, comparing Gaussian graphical models, and node wise predictability. These methods were recently introduced in the Gaussian graphical model literature, including
Williams (2019) <doi:10.31234/osf.io/x8dpr>,
Williams and Mulder (2019) <doi:10.31234/osf.io/ypxd8>,
Williams, Rast, Pericchi, and Mulder (2019) <doi:10.31234/osf.io/yt386>.

Depends R (>= 4.3.0)

License GPL-2

Imports BFpack (>= 1.2.3), GGally (>= 1.4.0), ggplot2 (>= 3.2.1), ggridges (>= 0.5.1), grDevices, MASS (>= 7.3-51.5), methods, mvnfast (>= 0.2.5), network (>= 1.15), reshape (>= 0.8.8), Rcpp (>= 1.0.4.6), Rdpack (>= 0.11-1), sna (>= 2.5), stats, utils,

Suggests abind (>= 1.4-5), assortnet (>= 0.12), networktools (>= 1.3.0), mice (>= 3.8.0), psych, knitr, rmarkdown, testthat (>= 3.0.0)

Encoding UTF-8

LazyData true

VignetteBuilder knitr

RoxygenNote 7.3.1

LinkingTo Rcpp, RcppArmadillo, RcppDist, RcppProgress

RdMacros Rdpack

BugReports `https://github.com/donaldRwilliams/BGGM/issues`

Config/testthat/edition 3

NeedsCompilation yes

Author Donald Williams [aut],
Joris Mulder [aut],
Philippe Rast [aut, cre]

Maintainer Philippe Rast <`rast.ph@gmail.com`>

Repository CRAN

Date/Publication 2024-02-23 08:00:06 UTC

R topics documented:

`asd_ocd` *Data: Autism and Obssesive Compulsive Disorder*

Description

A correlation matrix with 17 variables in total (autsim: 9; OCD: 8). The sample size was 213.

Usage

```
data("asd_ocd")
```

Format

A correlation matrix including 17 variables. These data were measured on a 4 level likert scale.

Details

Autism:

- `CI` Circumscribed interests
- `UP` Unusual preoccupations
- `RO` Repetitive use of objects or interests in parts of objects
- `CR` Compulsions and/or rituals
- `CI` Unusual sensory interests
- `SM` Complex mannerisms or stereotyped body movements
- `SU` Stereotyped utterances/delayed echolalia
- `NIL` Neologisms and/or idiosyncratic language
- `VR` Verbal rituals

OCD

- `CD` Concern with things touched due to dirt/bacteria
- `TB` Thoughts of doing something bad around others
- `CT` Continual thoughts that do not go away
- `HP` Belief that someone/higher power put reoccurring thoughts in their head
- `CW` Continual washing
- `CCh` Continual checking CntCheck
- `CC` Continual counting/repeating
- `RD` Repeatedly do things until it feels good or just right

References

Jones, P. J., Ma, R., & McNally, R. J. (2019). Bridge centrality: A network approach to understanding comorbidity. Multivariate behavioral research, 1-15.

Ruzzano, L., Borsboom, D., & Geurts, H. M. (2015). Repetitive behaviors in autism and obsessive-compulsive disorder: New perspectives from a network analysis. Journal of Autism and Developmental Disorders, 45(1), 192-202. doi:10.1007/s10803-014-2204-9

Examples

```
data("asd_ocd")

# generate continuous
Y <- MASS::mvrnorm(n = 213,
                   mu = rep(0, 17),
                   Sigma = asd_ocd,
                   empirical = TRUE)
```

`bfi` *Data: 25 Personality items representing 5 factors*

Description

This dataset and the corresponding documentation was taken from the **psych** package. We refer users to that package for further details (Revelle 2019).

Usage

```
data("bfi")
```

Format

A data frame with 25 variables and 2800 observations (including missing values)

Details

- `A1` Am indifferent to the feelings of others. (q_146)
- `A2` Inquire about others' well-being. (q_1162)
- `A3` Know how to comfort others. (q_1206)
- `A4` Love children. (q_1364)
- `A5` Make people feel at ease. (q_1419)
- `C1` Am exacting in my work. (q_124)
- `C2` Continue until everything is perfect. (q_530)
- `C3` Do things according to a plan. (q_619)

- `C4` Do things in a half-way manner. (q_626)
- `C5` Waste my time. (q_1949)
- `E1` Don't talk a lot. (q_712)
- `E2` Find it difficult to approach others. (q_901)
- `E3` Know how to captivate people. (q_1205)
- `E4` Make friends easily. (q_1410)
- `E5` Take charge. (q_1768)
- `N1` Get angry easily. (q_952)
- `N2` Get irritated easily. (q_974)
- `N3` Have frequent mood swings. (q_1099)
- `N4` Often feel blue. (q_1479)
- `N5` Panic easily. (q_1505)
- `o1` Am full of ideas. (q_128)
- `o2` Avoid difficult reading material.(q_316)
- `o3` Carry the conversation to a higher level. (q_492)
- `o4` Spend time reflecting on things. (q_1738)
- `o5` Will not probe deeply into a subject. (q_1964)
- `gender` Males = 1, Females =2
- `education` 1 = HS, 2 = finished HS, 3 = some college, 4 = college graduate 5 = graduate degree

References

Revelle W (2019). *psych: Procedures for Psychological, Psychometric, and Personality Research*. Northwestern University, Evanston, Illinois. R package version 1.9.12, `https://CRAN.R-project.org/package=psych`.

`bggm_missing` *GGM: Missing Data*

Description

Estimation and exploratory hypothesis testing with missing data.

Usage

```
bggm_missing(x, iter = 2000, method = "estimate", ...)
```

Arguments

`x`	An object of class `mid` `mice`.
`iter`	Number of iterations for each imputed dataset (posterior samples; defaults to 2000).
`method`	Character string. Which method should be used (default set to `estimate`)? The current options are `"estimate"` and `"explore"`.
`...`	Additional arguments passed to either `estimate` or `explore`.

Value

An object of class `estimate` or `explore`.

Note

Currently, **BGGM** is compatible with the package `mice` for handling the missing data. This is accomplished by fitting a model for each imputed dataset (i.e., more than one to account for uncertainty in the imputation step) and then pooling the estimates.

In a future version, an additional option will be added that allows for imputing the missing values during model fitting. This option will be incorporated directly into the `estimate` or `explore` functions, such that `bggm_missing` will always support missing data with `mice`.

Support:

There is limited support for missing data. As of version `2.0.0`, it is possible to determine the graphical structure with either `estimate` or `explore`, in addition to plotting the graph with `plot.select`. All data types *are* currently supported.

Memory Warning: A model is fitted for each imputed dataset. This results in a potentially large object.

Examples

```
# note: iter = 250 for demonstrative purposes

# need this package
library(mice, warn.conflicts = FALSE)

# data
Y <- ptsd[,1:5]

# matrix for indices
mat <- matrix(0, nrow = 221, ncol = 5)

# indices
indices <- which(mat == 0, arr.ind = TRUE)

# 50 NAs
Y[indices[sample(1:nrow(indices), 50),]] <- NA

# impute
```

```
x <- mice(Y, m = 5, print = FALSE)

#########################
#######   copula    #####
#########################
# rank based parital correlations

# estimate the model
fit_est <-  bggm_missing(x,
                         method = "estimate",
                         type =  "mixed",
                         iter = 250,
                         progress = FALSE)

# select edge set
E <- select(fit_est)

# plot E
plt_E <- plot(E)$plt

plt_E
```

`coef.estimate` *Compute Regression Parameters for* `estimate` *Objects*

Description

There is a direct correspondence between the inverse covariance matrix and multiple regression (Kwan 2014; Stephens 1998). This readily allows for converting the GGM parameters to regression coefficients. All data types are supported.

Usage

```
## S3 method for class 'estimate'
coef(object, iter = NULL, progress = TRUE, ...)
```

Arguments

`object`	An Object of class `estimate`
`iter`	Number of iterations (posterior samples; defaults to the number in the object).
`progress`	Logical. Should a progress bar be included (defaults to `TRUE`) ?
`...`	Currently ignored.

Value

An object of class `coef`, containting two lists.

- `betas` A list of length p, each containing a p - 1 by `iter` matrix of posterior samples
- `object` An object of class `estimate` (the fitted model).

References

Kwan CC (2014). "A regression-based interpretation of the inverse of the sample covariance matrix." *Spreadsheets in Education*, **7**(1), 4613.

Stephens G (1998). "On the Inverse of the Covariance Matrix in Portfolio Analysis." *The Journal of Finance*, **53**(5), 1821–1827.

Examples

```
# note: iter = 250 for demonstrative purposes

#########################
### example 1: binary ###
#########################
# data
Y <- women_math[1:500, ]

# fit model
fit <- estimate(Y, type = "binary",
                iter = 250,
                progress = FALSE)

# summarize the partial correlations
reg <- coef(fit, progress = FALSE)

# summary
summ <- summary(reg)

summm
```

`coef.explore` *Compute Regression Parameters for* `explore` *Objects*

Description

There is a direct correspondence between the inverse covariance matrix and multiple regression (Kwan 2014; Stephens 1998). This readily allows for converting the GGM parameters to regression coefficients. All data types are supported.

Usage

```
## S3 method for class 'explore'
coef(object, iter = NULL, progress = TRUE, ...)
```

Arguments

`object`	An Object of class `explore`.
`iter`	Number of iterations (posterior samples; defaults to the number in the object).
`progress`	Logical. Should a progress bar be included (defaults to `TRUE`) ?
`...`	Currently ignored.

Value

An object of class `coef`, containting two lists.

- `betas` A list of length p, each containing a p - 1 by `iter` matrix of posterior samples
- `object` An object of class `explore` (the fitted model).

References

Kwan CC (2014). "A regression-based interpretation of the inverse of the sample covariance matrix." *Spreadsheets in Education*, **7**(1), 4613.

Stephens G (1998). "On the Inverse of the Covariance Matrix in Portfolio Analysis." *The Journal of Finance*, **53**(5), 1821–1827.

Examples

```
# note: iter = 250 for demonstrative purposes

# data
Y <- ptsd[,1:4]

##########################
### example 1: ordinal ###
##########################

# fit model (note + 1, due to zeros)
fit <- explore(Y + 1,
               type = "ordinal",
               iter = 250,
               progress = FALSE)

# summarize the partial correlations
reg <- coef(fit, progress = FALSE)

# summary
summ <- summary(reg)

summ
```

`confirm` *GGM: Confirmatory Hypothesis Testing*

Description

Confirmatory hypothesis testing in GGMs. Hypotheses are expressed as equality and/or ineqaulity contraints on the partial correlations of interest. Here the focus is *not* on determining the graph (see `explore`) but testing specific hypotheses related to the conditional (in)dependence structure. These methods were introduced in Williams and Mulder (2019).

Usage

```
confirm(
  Y,
  hypothesis,
  prior_sd = 0.25,
  formula = NULL,
  type = "continuous",
  mixed_type = NULL,
  iter = 25000,
  progress = TRUE,
  impute = TRUE,
  seed = 1,
  ...
)
```

Arguments

`Y`	Matrix (or data frame) of dimensions n (observations) by p (variables).
`hypothesis`	Character string. The hypothesis (or hypotheses) to be tested. See details.
`prior_sd`	Numeric. Scale of the prior distribution, approximately the standard deviation of a beta distribution (defaults to 0.25).
`formula`	An object of class `formula`. This allows for including control variables in the model (e.g.,, `~ gender * education`).
`type`	Character string. Which type of data for **Y** ? The options include `continuous`, `binary`, `ordinal`, or `mixed`. See the note for further details.
`mixed_type`	Numeric vector of length p. An indicator for which varibles should be treated as ranks. (1 for rank and 0 to assume normality). The default is currently (dev version) to treat all integer variables as ranks when `type = "mixed"` and `NULL` otherwise. See note for further details.
`iter`	Number of iterations (posterior samples; defaults to 25,000).
`progress`	Logical. Should a progress bar be included (defaults to `TRUE`) ?
`impute`	Logicial. Should the missing values (`NA`) be imputed during model fitting (defaults to `TRUE`) ?
`seed`	An integer for the random seed.
`...`	Currently ignored.

Details

The hypotheses can be written either with the respective column names or numbers. For example, `1--2` denotes the relation between the variables in column 1 and 2. Note that these must correspond to the upper triangular elements of the correlation matrix. This is accomplished by ensuring that the first number is smaller than the second number. This also applies when using column names (i.e,, in reference to the column number).

One Hypothesis:

To test whether some relations are larger than others, while others are expected to be equal, this can be writting as

- `hyp <- c(1--2 > 1--3 = 1--4 > 0)`,

where there is an addition additional contraint that all effects are expected to be positive. This is then compared to the complement.

More Than One Hypothesis:

The above hypothesis can also be compared to, say, a null model by using ";" to seperate the hypotheses, for example,

- `hyp <- c(1--2 > 1--3 = 1--4 > 0; 1--2 = 1--3 = 1--4 = 0)`.

Any number of hypotheses can be compared this way.

Using "&"

It is also possible to include `&`. This allows for testing one constraint **and** another contraint as one hypothesis.

- `hyp <- c("A1--A2 > A1--A2 & A1--A3 = A1--A3")`

Of course, it is then possible to include additional hypotheses by separating them with ";". Note also that the column names were used in this example (e.g., `A1--A2` is the relation between those nodes).

Testing Sums

It might also be interesting to test the sum of partial correlations. For example, that the sum of specific relations is larger than the sum of other relations. This can be written as

- `hyp <- c("A1--A2 + A1--A3 > A1--A4 + A1--A5; A1--A2 + A1--A3 = A1--A4 + A1--A5")`

Potential Delays:

There is a chance for a potentially long delay from the time the progress bar finishes to when the function is done running. This occurs when the hypotheses require further sampling to be tested, for example, when grouping relations `c("(A1--A2, A1--A3) > (A1--A4, A1--A5)"`. This is not an error.

Controlling for Variables:

When controlling for variables, it is assumed that `Y` includes *only* the nodes in the GGM and the control variables. Internally, `only` the predictors that are included in `formula` are removed from `Y`. This is not behavior of, say, `lm`, but was adopted to ensure users do not have to write out each variable that should be included in the GGM. An example is provided below.

Mixed Type:

The term "mixed" is somewhat of a misnomer, because the method can be used for data including *only* continuous or *only* discrete variables (Hoff 2007). This is based on the ranked likelihood which requires sampling the ranks for each variable (i.e., the data is not merely transformed to ranks). This is computationally expensive when there are many levels. For example, with continuous data, there are as many ranks as data points!

The option `mixed_type` allows the user to determine which variable should be treated as ranks and the "emprical" distribution is used otherwise. This is accomplished by specifying an indicator vector of length p. A one indicates to use the ranks, whereas a zero indicates to "ignore" that variable. By default all integer variables are handled as ranks.

Dealing with Errors:

An error is most likely to arise when `type = "ordinal"`. The are two common errors (although still rare):

- The first is due to sampling the thresholds, especially when the data is heavily skewed. This can result in an ill-defined matrix. If this occurs, we recommend to first try decreasing `prior_sd` (i.e., a more informative prior). If that does not work, then change the data type to `type = mixed` which then estimates a copula GGM (this method can be used for data containing **only** ordinal variable). This should work without a problem.
- The second is due to how the ordinal data are categorized. For example, if the error states that the index is out of bounds, this indicates that the first category is a zero. This is not allowed, as the first category must be one. This is addressed by adding one (e.g., `Y + 1`) to the data matrix.

Value

The returned object of class `confirm` contains a lot of information that is used for printing and plotting the results. For users of **BGGM**, the following are the useful objects:

- `out_hyp_prob` Posterior hypothesis probabilities.
- `info` An object of class `BF` from the R package **BFpack**.

Note

"Default" Prior:

In Bayesian statistics, a default Bayes factor needs to have several properties. I refer interested users to section 2.2 in Dablander et al. (2020). In Williams and Mulder (2019), some of these propteries were investigated (e.g., model selection consistency). That said, we would not consider this a "default" or "automatic" Bayes factor and thus we encourage users to perform sensitivity analyses by varying the scale of the prior distribution.

Furthermore, it is important to note there is no "correct" prior and, also, there is no need to entertain the possibility of a "true" model. Rather, the Bayes factor can be interpreted as which hypothesis best (relative to each other) predicts the observed data (Section 3.2 in Kass and Raftery 1995).

Interpretation of Conditional (In)dependence Models for Latent Data:

See `BGGM-package` for details about interpreting GGMs based on latent data (i.e, all data types besides `"continuous"`)

References

Dablander F, Bergh Dvd, Ly A, Wagenmakers E (2020). "Default Bayes Factors for Testing the (In) equality of Several Population Variances." *arXiv preprint arXiv:2003.06278*.

Hoff PD (2007). "Extending the rank likelihood for semiparametric copula estimation." *The Annals of Applied Statistics*, **1**(1), 265–283. doi:10.1214/07AOAS107.

Kass RE, Raftery AE (1995). "Bayes Factors." *Journal of the American Statistical Association*, **90**(430), 773–795.

Williams DR, Mulder J (2019). "Bayesian Hypothesis Testing for Gaussian Graphical Models: Conditional Independence and Order Constraints." *PsyArXiv*. doi:10.31234/osf.io/ypxd8.

Examples

```
# note: iter = 250 for demonstrative purposes

##########################
### example 1: cheating ##
##########################
# Here a true hypothesis is tested,
# which shows the method works nicely
# (peeked at partials beforehand)

# data
Y <- BGGM::bfi[,1:10]

hypothesis <- c("A1--A2 < A1--A3 < A1--A4 = A1--A5")

# test cheat
test_cheat <-  confirm(Y = Y,
                       type = "continuous",
                       hypothesis  = hypothesis,
                       iter = 250,
                       progress = FALSE)

# print (probabilty of nearly 1 !)
test_cheat
```

`constrained_posterior` *Constrained Posterior Distribution*

Description

Compute the posterior distribution with off-diagonal elements of the precision matrix constrained to zero.

Usage

```
constrained_posterior(
  object,
  adj,
  method = "direct",
  iter = 5000,
  progress = TRUE,
  ...
)
```

Arguments

`object`	An object of class `estimate` or `explore`
`adj`	A `p` by `p` adjacency matrix. The zero entries denote the elements that should be constrained to zero.
`method`	Character string. Which method should be used ? Defaults to the "direct sampler" (i.e., `method = "direct"`) described in page 122, section 2.4, Lenkoski (2013). The other option is a Metropolis-Hastings algorithm (`MH`). See details.
`iter`	Number of iterations (posterior samples; defaults to 5000).
`progress`	Logical. Should a progress bar be included (defaults to `TRUE`) ?
`...`	Currently ignored.

Value

An object of class `contrained`, including

- `precision_mean` The posterior mean for the precision matrix.
- `pcor_mean` The posterior mean for the precision matrix.
- `precision_samps` A 3d array of dimension `p` by `p` by `iter` including the sampled precision matrices.
- `pcor_samps` A 3d array of dimension `p` by `p` by `iter` including sampled partial correlations matrices.

References

Lenkoski A (2013). “A direct sampler for G-Wishart variates.” *Stat*, **2**(1), 119–128.

Examples

```
# data
Y <- bfi[,1:10]

# sample posterior
fit <- estimate(Y, iter = 100)
```

```
# select graph
sel <- select(fit)

# constrained posterior
post <- constrained_posterior(object = fit,
                              adj = sel$adj,
                              iter = 100,
                              progress = FALSE)
```

`convergence` *MCMC Convergence*

Description

Monitor convergence of the MCMC algorithms.

Usage

```
convergence(object, param = NULL, type = "trace", print_names = FALSE)
```

Arguments

`object`	An object of class `estimate` or `explore`
`param`	Character string. Names of parameters for which to monitor MCMC convergence.
`type`	Character string. Which type of convergence plot ? The current options are `trace` (default) and `acf`.
`print_names`	Logical. Should the parameter names be printed (defaults to `FALSE`)? This can be used to first determine the parameter names to specify in `type`.

Value

A list of `ggplot` objects.

Note

An overview of MCMC diagnostics can be found here.

Examples

```
# note: iter = 250 for demonstrative purposes

# data
Y <- ptsd[,1:5]
```

```
#########################
###### continuous #######
#########################
fit <- estimate(Y, iter = 250,
                progress = FALSE)

# print names first
convergence(fit, print_names = TRUE)

# trace plots
convergence(fit, type = "trace",
            param = c("B1--B2", "B1--B3"))[[1]]

# acf plots
convergence(fit, type = "acf",
            param = c("B1--B2", "B1--B3"))[[1]]
```

`csws` *Data: Contingencies of Self-Worth Scale (CSWS)*

Description

A dataset containing items from the Contingencies of Self-Worth Scale (CSWS) scale. There are 35 variables and 680 observations

Usage

```
data("csws")
```

Format

A data frame with 35 variables and 680 observations (7 point Likert scale)

Details

- `1` When I think I look attractive, I feel good about myself
- `2` My self-worth is based on God's love
- `3` I feel worthwhile when I perform better than others on a task or skill.
- `4` My self-esteem is unrelated to how I feel about the way my body looks.
- `5` Doing something I know is wrong makes me lose my self-respect
- `6` I don't care if other people have a negative opinion about me.
- `7` Knowing that my family members love me makes me feel good about myself.
- `8` I feel worthwhile when I have God's love.
- `9` I can't respect myself if others don't respect me.
- `10` My self-worth is not influenced by the quality of my relationships with my family members.

- `11` Whenever I follow my moral principles, my sense of self-respect gets a boost.
- `12` Knowing that I am better than others on a task raises my self-esteem.
- `13` My opinion about myself isn't tied to how well I do in school.
- `14` I couldn't respect myself if I didn't live up to a moral code.
- `15` I don't care what other people think of me.
- `16` When my family members are proud of me, my sense of self-worth increases.
- `17` My self-esteem is influenced by how attractive I think my face or facial features are.
- `18` My self-esteem would suffer if I didn't have God's love.
- `19` Doing well in school gives me a sense of selfrespect.
- `20` Doing better than others gives me a sense of self-respect.
- `21` My sense of self-worth suffers whenever I think I don't look good.
- `22` I feel better about myself when I know I'm doing well academically.
- `23` What others think of me has no effect on what I think about myself.
- `24` When I don't feel loved by my family, my selfesteem goes down.
- `25` My self-worth is affected by how well I do when I am competing with others.
- `26` My self-esteem goes up when I feel that God loves me.
- `27` My self-esteem is influenced by my academic performance.
- `28` My self-esteem would suffer if I did something unethical.
- `29` It is important to my self-respect that I have a family that cares about me.
- `30` My self-esteem does not depend on whether or not I feel attractive.
- `31` When I think that I'm disobeying God, I feel bad about myself.
- `32` My self-worth is influenced by how well I do on competitive tasks.
- `33` I feel bad about myself whenever my academic performance is lacking.
- `34` My self-esteem depends on whether or not I follow my moral/ethical principles.
- `35` My self-esteem depends on the opinions others hold of me.
- `gender` "M" (male) or "F" (female)

Note

There are seven domains

FAMILY SUPPORT: items 7, 10, 16, 24, and 29.

COMPETITION: items 3, 12, 20, 25, and 32.

APPEARANCE: items 1, 4, 17, 21, and 30.

GOD'S LOVE: items 2, 8, 18, 26, and 31.

ACADEMIC COMPETENCE: items 13, 19, 22, 27, and 33.

VIRTUE: items 5, 11, 14, 28, and 34.

APPROVAL FROM OTHERS: items: 6, 9, 15, 23, and 35.

References

Briganti, G., Fried, E. I., & Linkowski, P. (2019). Network analysis of Contingencies of Self-Worth Scale in 680 university students. Psychiatry research, 272, 252-257.

Examples

```
data("csws")
```

`depression_anxiety_t1` *Data: Depression and Anxiety (Time 1)*

Description

A data frame containing 403 observations (n = 403) and 16 variables (p = 16) measured on the 4-point likert scale (depression: 9; anxiety: 7).

Usage

```
data("depression_anxiety_t1")
```

Format

A data frame containing 403 observations (n = 7466) and 16 variables (p = 16) measured on the 4-point likert scale.

Details

Depression:

- `PHQ1` Little interest or pleasure in doing things?
- `PHQ2` Feeling down, depressed, or hopeless?
- `PHQ3` Trouble falling or staying asleep, or sleeping too much?
- `PHQ4` Feeling tired or having little energy?
- `PHQ5` Poor appetite or overeating?
- `PHQ6` Feeling bad about yourself — or that you are a failure or have let yourself or your family down?
- `PHQ7` Trouble concentrating on things, such as reading the newspaper or watching television?
- `PHQ8` Moving or speaking so slowly that other people could have noticed? Or so fidgety or restless that you have been moving a lot more than usual?
- `PHQ9` Thoughts that you would be better off dead, or thoughts of hurting yourself in some way?

Anxiety

- `GAD1` Feeling nervous, anxious, or on edge
- `GAD2` Not being able to stop or control worrying
- `GAD3` Worrying too much about different things
- `GAD4` Trouble relaxing
- `GAD5` Being so restless that it's hard to sit still
- `GAD6` Becoming easily annoyed or irritable
- `GAD7` Feeling afraid as if something awful might happen

References

Forbes, M. K., Baillie, A. J., & Schniering, C. A. (2016). A structural equation modeling analysis of the relationships between depression,anxiety, and sexual problems over time. The Journal of Sex Research, 53(8), 942-954.

Forbes, M. K., Wright, A. G., Markon, K. E., & Krueger, R. F. (2019). Quantifying the reliability and replicability of psychopathology network characteristics. Multivariate behavioral research, 1-19.

Jones, P. J., Williams, D. R., & McNally, R. J. (2019). Sampling variability is not nonreplication: a Bayesian reanalysis of Forbes, Wright, Markon, & Krueger.

Examples

```
data("depression_anxiety_t1")
labels<- c("interest", "down", "sleep",
            "tired", "appetite", "selfest",
           "concen", "psychmtr", "suicid",
           "nervous", "unctrworry", "worrylot",
           "relax", "restless", "irritable", "awful")
```

`depression_anxiety_t2` *Data: Depression and Anxiety (Time 2)*

Description

A data frame containing 403 observations (n = 403) and 16 variables (p = 16) measured on the 4-point likert scale (depression: 9; anxiety: 7).

Usage

```
data("depression_anxiety_t2")
```

Format

A data frame containing 403 observations (n = 7466) and 16 variables (p = 16) measured on the 4-point likert scale.

Details

Depression:

- `PHQ1` Little interest or pleasure in doing things?
- `PHQ2` Feeling down, depressed, or hopeless?
- `PHQ3` Trouble falling or staying asleep, or sleeping too much?
- `PHQ4` Feeling tired or having little energy?
- `PHQ5` Poor appetite or overeating?
- `PHQ6` Feeling bad about yourself — or that you are a failure or have let yourself or your family down?
- `PHQ7` Trouble concentrating on things, such as reading the newspaper or watching television?
- `PHQ8` Moving or speaking so slowly that other people could have noticed? Or so fidgety or restless that you have been moving a lot more than usual?
- `PHQ9` Thoughts that you would be better off dead, or thoughts of hurting yourself in some way?

Anxiety

- `GAD1` Feeling nervous, anxious, or on edge
- `GAD2` Not being able to stop or control worrying
- `GAD3` Worrying too much about different things
- `GAD4` Trouble relaxing
- `GAD5` Being so restless that it's hard to sit still
- `GAD6` Becoming easily annoyed or irritable
- `GAD7` Feeling afraid as if something awful might happen

References

Forbes, M. K., Baillie, A. J., & Schniering, C. A. (2016). A structural equation modeling analysis of the relationships between depression,anxiety, and sexual problems over time. The Journal of Sex Research, 53(8), 942-954.

Forbes, M. K., Wright, A. G., Markon, K. E., & Krueger, R. F. (2019). Quantifying the reliability and replicability of psychopathology network characteristics. Multivariate behavioral research, 1-19.

Jones, P. J., Williams, D. R., & McNally, R. J. (2019). Sampling variability is not nonreplication: a Bayesian reanalysis of Forbes, Wright, Markon, & Krueger.

Examples

```
data("depression_anxiety_t2")
labels<- c("interest", "down", "sleep",
            "tired", "appetite", "selfest",
           "concen", "psychmtr", "suicid",
           "nervous", "unctrworry", "worrylot",
           "relax", "restless", "irritable", "awful")
```

`estimate` *GGM: Estimation*

Description

Estimate the conditional (in)dependence with either an analytic solution or efficiently sampling from the posterior distribution. These methods were introduced in Williams (2018). The graph is selected with `select.estimate` and then plotted with `plot.select`.

Usage

```
estimate(
  Y,
  formula = NULL,
  type = "continuous",
  mixed_type = NULL,
  analytic = FALSE,
  prior_sd = 0.25,
  iter = 5000,
  impute = FALSE,
  progress = TRUE,
  seed = 1,
  ...
)
```

Arguments

`Y`	Matrix (or data frame) of dimensions *n* (observations) by *p* (variables).
`formula`	An object of class `formula`. This allows for including control variables in the model (i.e., `~ gender`). See the note for further details.
`type`	Character string. Which type of data for `Y` ? The options include `continuous`, `binary`, `ordinal`, or `mixed`. Note that mixed can be used for data with only ordinal variables. See the note for further details.
`mixed_type`	Numeric vector. An indicator of length *p* for which variables should be treated as ranks. (1 for rank and 0 to assume normality). The default is currently to treat all integer variables as ranks when `type = "mixed"` and `NULL` otherwise. See note for further details.
`analytic`	Logical. Should the analytic solution be computed (default is `FALSE`)?
`prior_sd`	Scale of the prior distribution, approximately the standard deviation of a beta distribution (defaults to 0.50).
`iter`	Number of iterations (posterior samples; defaults to 5000).
`impute`	Logical. Should the missing values (`NA`) be imputed during model fitting (defaults to `TRUE`) ?
`progress`	Logical. Should a progress bar be included (defaults to `TRUE`) ?
`seed`	An integer for the random seed.
`...`	Currently ignored.

Details

The default is to draw samples from the posterior distribution (`analytic = FALSE`). The samples are required for computing edge differences (see `ggm_compare_estimate`), Bayesian R2 introduced in Gelman et al. (2019) (see `predictability`), etc. If the goal is to *only* determine the non-zero effects, this can be accomplished by setting `analytic = TRUE`. This is particularly useful when a fast solution is needed (see the examples in `ggm_compare_ppc`)

Controlling for Variables:

When controlling for variables, it is assumed that `Y` includes *only* the nodes in the GGM and the control variables. Internally, `only` the predictors that are included in `formula` are removed from `Y`. This is not behavior of, say, `lm`, but was adopted to ensure users do not have to write out each variable that should be included in the GGM. An example is provided below.

Mixed Type:

The term "mixed" is somewhat of a misnomer, because the method can be used for data including *only* continuous or *only* discrete variables. This is based on the ranked likelihood which requires sampling the ranks for each variable (i.e., the data is not merely transformed to ranks). This is computationally expensive when there are many levels. For example, with continuous data, there are as many ranks as data points!

The option `mixed_type` allows the user to determine which variable should be treated as ranks and the "emprical" distribution is used otherwise (Hoff 2007). This is accomplished by specifying an indicator vector of length p. A one indicates to use the ranks, whereas a zero indicates to "ignore" that variable. By default all integer variables are treated as ranks.

Dealing with Errors:

An error is most likely to arise when `type = "ordinal"`. The are two common errors (although still rare):

- The first is due to sampling the thresholds, especially when the data is heavily skewed. This can result in an ill-defined matrix. If this occurs, we recommend to first try decreasing `prior_sd` (i.e., a more informative prior). If that does not work, then change the data type to `type = mixed` which then estimates a copula GGM (this method can be used for data containing **only** ordinal variable). This should work without a problem.
- The second is due to how the ordinal data are categorized. For example, if the error states that the index is out of bounds, this indicates that the first category is a zero. This is not allowed, as the first category must be one. This is addressed by adding one (e.g., `Y + 1`) to the data matrix.

Imputing Missing Values:

Missing values are imputed with the approach described in Hoff (2009). The basic idea is to impute the missing values with the respective posterior pedictive distribution, given the observed data, as the model is being estimated. Note that the default is `TRUE`, but this ignored when there are no missing values. If set to `FALSE`, and there are missing values, list-wise deletion is performed with `na.omit`.

Value

The returned object of class `estimate` contains a lot of information that is used for printing and plotting the results. For users of **BGGM**, the following are the useful objects:

- `pcor_mat` Partial correltion matrix (posterior mean).
- `post_samp` An object containing the posterior samples.

Note

Posterior Uncertainty:

A key feature of **BGGM** is that there is a posterior distribution for each partial correlation. This readily allows for visiualizing uncertainty in the estimates. This feature works with all data types and is accomplished by plotting the summary of the `estimate` object (i.e., `plot(summary(fit))`). Several examples are provided below.

Interpretation of Conditional (In)dependence Models for Latent Data:

See `BGGM-package` for details about interpreting GGMs based on latent data (i.e, all data types besides `"continuous"`)

References

Gelman A, Goodrich B, Gabry J, Vehtari A (2019). "R-squared for Bayesian Regression Models." *American Statistician*, **73**(3), 307–309. ISSN 15372731, doi:10.1080/00031305.2018.1549100.

Hoff PD (2007). "Extending the rank likelihood for semiparametric copula estimation." *The Annals of Applied Statistics*, **1**(1), 265–283. doi:10.1214/07AOAS107.

Hoff PD (2009). *A first course in Bayesian statistical methods*, volume 580. Springer.

Williams DR (2018). "Bayesian Estimation for Gaussian Graphical Models: Structure Learning, Predictability, and Network Comparisons." *arXiv*. doi:10.31234/OSF.IO/X8DPR.

Examples

```
# note: iter = 250 for demonstrative purposes

#########################################
### example 1: continuous and ordinal ###
#########################################
# data
Y <- ptsd

# continuous

# fit model
fit <- estimate(Y, type = "continuous",
                iter = 250)

# summarize the partial correlations
summ <- summary(fit)

# plot the summary
plt_summ <- plot(summary(fit))
```

```
# select the graph
E <- select(fit)

# plot the selected graph
plt_E <- plot(select(fit))


# ordinal

# fit model (note + 1, due to zeros)
fit <- estimate(Y + 1, type = "ordinal",
                iter = 250)

# summarize the partial correlations
summ <- summary(fit)

# plot the summary
plt <- plot(summary(fit))

# select the graph
E <- select(fit)

# plot the selected graph
plt_E <- plot(select(fit))

##################################
## example 2: analytic solution ##
##################################
# (only continuous)

# data
Y <- ptsd

# fit model
fit <- estimate(Y, analytic = TRUE)

# summarize the partial correlations
summ <- summary(fit)

# plot summary
plt_summ <- plot(summary(fit))

# select graph
E <- select(fit)

# plot the selected graph
plt_E <- plot(select(fit))
```

`explore` *GGM: Exploratory Hypothesis Testing*

Description

Learn the conditional (in)dependence structure with the Bayes factor using the matrix-F prior distribution (Mulder and Pericchi 2018). These methods were introduced in Williams and Mulder (2019). The graph is selected with `select.explore` and then plotted with `plot.select`.

Usage

```
explore(
  Y,
  formula = NULL,
  type = "continuous",
  mixed_type = NULL,
  analytic = FALSE,
  prior_sd = 0.25,
  iter = 5000,
  progress = TRUE,
  impute = FALSE,
  seed = 1,
  ...
)
```

Arguments

`Y`	Matrix (or data frame) of dimensions *n* (observations) by *p* (variables).
`formula`	An object of class `formula`. This allows for including control variables in the model (i.e., ~ `gender`).
`type`	Character string. Which type of data for `Y` ? The options include `continuous`, `binary`, `ordinal`, or `mixed` (semi-parametric copula). See the note for further details.
`mixed_type`	Numeric vector. An indicator of length p for which varibles should be treated as ranks. (1 for rank and 0 to assume normality). The default is to treat all integer variables as ranks when `type = "mixed"` and `NULL` otherwise. See note for further details.
`analytic`	Logical. Should the analytic solution be computed (default is `FALSE`)? (currently not implemented)
`prior_sd`	Scale of the prior distribution, approximately the standard deviation of a beta distribution (defaults to 0.25).
`iter`	Number of iterations (posterior samples; defaults to 5000).
`progress`	Logical. Should a progress bar be included (defaults to `TRUE`) ?
`impute`	Logicial. Should the missing values (`NA`) be imputed during model fitting (defaults to `TRUE`) ?

`seed`	An integer for the random seed.
`...`	Currently ignored (leave empty).

Details

Controlling for Variables:

When controlling for variables, it is assumed that `Y` includes *only* the nodes in the GGM and the control variables. Internally, `only` the predictors that are included in `formula` are removed from `Y`. This is not behavior of, say, `lm`, but was adopted to ensure users do not have to write out each variable that should be included in the GGM. An example is provided below.

Mixed Type:

The term "mixed" is somewhat of a misnomer, because the method can be used for data including *only* continuous or *only* discrete variables. This is based on the ranked likelihood which requires sampling the ranks for each variable (i.e., the data is not merely transformed to ranks). This is computationally expensive when there are many levels. For example, with continuous data, there are as many ranks as data points!

The option `mixed_type` allows the user to determine which variable should be treated as ranks and the "emprical" distribution is used otherwise. This is accomplished by specifying an indicator vector of length p. A one indicates to use the ranks, whereas a zero indicates to "ignore" that variable. By default all integer variables are handled as ranks.

Dealing with Errors:

An error is most likely to arise when `type = "ordinal"`. The are two common errors (although still rare):

- The first is due to sampling the thresholds, especially when the data is heavily skewed. This can result in an ill-defined matrix. If this occurs, we recommend to first try decreasing `prior_sd` (i.e., a more informative prior). If that does not work, then change the data type to `type = mixed` which then estimates a copula GGM (this method can be used for data containing **only** ordinal variable). This should work without a problem.
- The second is due to how the ordinal data are categorized. For example, if the error states that the index is out of bounds, this indicates that the first category is a zero. This is not allowed, as the first category must be one. This is addressed by adding one (e.g., `Y + 1`) to the data matrix.

Imputing Missing Values:

Missing values are imputed with the approach described in Hoff (2009). The basic idea is to impute the missing values with the respective posterior pedictive distribution, given the observed data, as the model is being estimated. Note that the default is `TRUE`, but this ignored when there are no missing values. If set to `FALSE`, and there are missing values, list-wise deletion is performed with `na.omit`.

Value

The returned object of class `explore` contains a lot of information that is used for printing and plotting the results. For users of **BGGM**, the following are the useful objects:

- `pcor_mat` partial correltion matrix (posterior mean).
- `post_samp` an object containing the posterior samples.

Note

Posterior Uncertainty:

A key feature of **BGGM** is that there is a posterior distribution for each partial correlation. This readily allows for visiualizing uncertainty in the estimates. This feature works with all data types and is accomplished by plotting the summary of the `explore` object (i.e., `plot(summary(fit))`). Note that in contrast to `estimate` (credible intervals), the posterior standard deviation is plotted for `explore` objects.

"Default" Prior:

In Bayesian statistics, a default Bayes factor needs to have several properties. I refer interested users to section 2.2 in Dablander et al. (2020). In Williams and Mulder (2019), some of these propteries were investigated including model selection consistency. That said, we would not consider this a "default" (or "automatic") Bayes factor and thus we encourage users to perform sensitivity analyses by varying the scale of the prior distribution.

Furthermore, it is important to note there is no "correct" prior and, also, there is no need to entertain the possibility of a "true" model. Rather, the Bayes factor can be interpreted as which hypothesis best (**relative** to each other) predicts the observed data (Section 3.2 in Kass and Raftery 1995).

Interpretation of Conditional (In)dependence Models for Latent Data:

See `BGGM-package` for details about interpreting GGMs based on latent data (i.e, all data types besides `"continuous"`)

References

Dablander F, Bergh Dvd, Ly A, Wagenmakers E (2020). “Default Bayes Factors for Testing the (In) equality of Several Population Variances.” *arXiv preprint arXiv:2003.06278*.

Hoff PD (2009). *A first course in Bayesian statistical methods*, volume 580. Springer.

Kass RE, Raftery AE (1995). “Bayes Factors.” *Journal of the American Statistical Association*, **90**(430), 773–795.

Mulder J, Pericchi L (2018). “The Matrix-F Prior for Estimating and Testing Covariance Matrices.” *Bayesian Analysis*, 1–22. ISSN 19316690, doi:10.1214/17BA1092.

Williams DR, Mulder J (2019). “Bayesian Hypothesis Testing for Gaussian Graphical Models: Conditional Independence and Order Constraints.” *PsyArXiv*. doi:10.31234/osf.io/ypxd8.

Examples

```
# note: iter = 250 for demonstrative purposes

###########################
### example 1:  binary ####
###########################
Y <- women_math[1:500,]

# fit model
```

```
fit <- explore(Y, type = "binary",
                iter = 250,
                progress = FALSE)

# summarize the partial correlations
summ <- summary(fit)

# plot the summary
plt_summ <- plot(summary(fit))

# select the graph
E <- select(fit)

# plot the selected graph
plt_E <- plot(E)

plt_E$plt_alt
```

`fisher_r_to_z` *Fisher Z Transformation*

Description

Tranform correlations to Fisher's Z

Usage

```
fisher_r_to_z(r)
```

Arguments

`r`	correlation (can be a vector)

Value

Fisher Z transformed correlation(s)

Examples

```
fisher_r_to_z(0.5)
```

`fisher_z_to_r` *Fisher Z Back Transformation*

Description

Back tranform Fisher's Z to correlations

Usage

```
fisher_z_to_r(z)
```

Arguments

`z`	Fisher Z

Value

Correlation (s) (backtransformed)

Examples

```
fisher_z_to_r(0.5)
```

`gen_net` *Simulate a Partial Correlation Matrix*

Description

Simulate a Partial Correlation Matrix

Usage

```
gen_net(p = 20, edge_prob = 0.3, lb = 0.05, ub = 0.3)
```

Arguments

`p`	number of variables (nodes)
`edge_prob`	connectivity
`lb`	lower bound for the partial correlations
`ub`	upper bound for the partial correlations

Value

A list containing the following:

- **pcor**: Partial correlation matrix, encoding the conditional (in)dependence structure.
- **cors**: Correlation matrix.
- **adj**: Adjacency matrix.
- **trys**: Number of attempts to obtain a positive definite matrix.

Note

The function checks for a valid matrix (positive definite), but sometimes this will still fail. For example, for larger `p`, to have large partial correlations this requires a sparse GGM (accomplished by setting `edge_prob` to a small value).

Examples

```
true_net <- gen_net(p = 10)
```

`gen_ordinal` *Generate Ordinal and Binary data*

Description

Generate Multivariate Ordinal and Binary data.

Usage

```
gen_ordinal(n, p, levels = 2, cor_mat, empirical = FALSE)
```

Arguments

`n`	Number of observations (n).
`p`	Number of variables (p).
`levels`	Number of categories (defaults to 2; binary data).
`cor_mat`	A p by p matrix including the true correlation structure.
`empirical`	Logical. If true, `cor_mat` specifies the empirical not population covariance matrix.

Value

A n by p data matrix.

Note

In order to allow users to enjoy the functionality of **BGGM**, we had to make minor changes to the function `rmvord_naiv` from the `R` package **orddata** (Leisch et al. 2010). All rights to, and credit for, the function `rmvord_naiv` belong to the authors of that package.

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version. This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details. A copy of the GNU General Public License is available online.

References

Leisch F, Kaiser AWS, Hornik K (2010). *orddata: Generation of Artificial Ordinal and Binary Data*. R package version 0.1/r4, https://R-Forge.R-project.org/projects/orddata/.

Examples

```
#################################
######### example 1 ############
#################################

main <-  ptsd_cor1[1:5,1:5]
p <- ncol(main)

pcors <- -(cov2cor(solve(main)) -diag(p))
diag(pcors) <- 1
pcors <- ifelse(abs(pcors) < 0.05, 0, pcors)

inv <-  -pcors
diag(inv) <- 1
cors <- cov2cor( solve(inv))

# example data
Y <- BGGM::gen_ordinal(n = 500, p = 5,
                       levels = 2,
                       cor_mat = cors,
                       empirical = FALSE)



#################################
######### example 2 ############
#################################
# empirical = TRUE

Y <-  gen_ordinal(n = 500,
                  p = 16,
                  levels = 5,
                  cor_mat = ptsd_cor1,
                  empirical = TRUE)
```

`ggm_compare_confirm` *GGM Compare: Confirmatory Hypothesis Testing*

Description

Confirmatory hypothesis testing for comparing GGMs. Hypotheses are expressed as equality and/or ineqaulity contraints on the partial correlations of interest. Here the focus is *not* on determining the graph (see `explore`) but testing specific hypotheses related to the conditional (in)dependence structure. These methods were introduced in Williams and Mulder (2019) and in Williams et al. (2020)

Usage

```
ggm_compare_confirm(
  ...,
  hypothesis,
  formula = NULL,
  type = "continuous",
  mixed_type = NULL,
  prior_sd = 0.25,
  iter = 25000,
  impute = TRUE,
  progress = TRUE,
  seed = 1
)
```

Arguments

`...`	At least two matrices (or data frame) of dimensions *n* (observations) by *p* (nodes).
`hypothesis`	Character string. The hypothesis (or hypotheses) to be tested. See notes for futher details.
`formula`	an object of class `formula`. This allows for including control variables in the model (i.e., `~ gender`).
`type`	Character string. Which type of data for `Y` ? The options include `continuous`, `binary`, `ordinal`, or `mixed`. Note that mixed can be used for data with only ordinal variables. See the note for further details.
`mixed_type`	numeric vector. An indicator of length p for which varibles should be treated as ranks. (1 for rank and 0 to assume normality). The default is currently (dev version) to treat all integer variables as ranks when `type = "mixed"` and `NULL` otherwise. See note for further details.
`prior_sd`	Numeric. The scale of the prior distribution (centered at zero), in reference to a beta distribtuion (defaults to 0.25).
`iter`	Number of iterations (posterior samples; defaults to 25,000).

`impute`	Logical. Should the missing values (`NA`) be imputed during model fitting (defaults to `TRUE`) ?
`progress`	Logical. Should a progress bar be included (defaults to `TRUE`) ?
`seed`	An integer for the random seed.

Details

The hypotheses can be written either with the respective column names or numbers. For example, `g1_1--2` denotes the relation between the variables in column 1 and 2 for group 1. The `g1_` is required and the only difference from `confirm` (one group). Note that these must correspond to the upper triangular elements of the correlation matrix. This is accomplished by ensuring that the first number is smaller than the second number. This also applies when using column names (i.e,, in reference to the column number).

One Hypothesis:

To test whether a relation in larger in one group, while both are expected to be positive, this can be written as

- `hyp <- c(g1_1--2 > g2_1--2 > 0)`

This is then compared to the complement.

More Than One Hypothesis:

The above hypothesis can also be compared to, say, a null model by using ";" to seperate the hypotheses, for example,

- `hyp <- c(g1_1--2 > g2_1--2 > 0; g1_1--2 = g2_1--2 = 0)`.

Any number of hypotheses can be compared this way.

Using "&"

It is also possible to include `&`. This allows for testing one constraint **and** another contraint as one hypothesis.

- `hyp <- c("g1_A1--A2 > g2_A1--A2 & g1_A1--A3 = g2_A1--A3")`

Of course, it is then possible to include additional hypotheses by separating them with ";".

Testing Sums

It might also be interesting to test the sum of partial correlations. For example, that the sum of specific relations in one group is larger than the sum in another group.

- `hyp <- c("g1_A1--A2 + g1_A1--A3 > g2_A1--A2 + g2_A1--A3; g1_A1--A2 + g1_A1--A3 = g2_A1--A2 + g2_A1--A3")`

Potential Delays:

There is a chance for a potentially long delay from the time the progress bar finishes to when the function is done running. This occurs when the hypotheses require further sampling to be tested, for example, when grouping relations `c("(g1_A1--A2, g2_A2--A3) > (g2_A1--A2, g2_A2--A3)"`. This is not an error.

Controlling for Variables:

When controlling for variables, it is assumed that `Y` includes *only* the nodes in the GGM and the control variables. Internally, `only` the predictors that are included in `formula` are removed from `Y`. This is not behavior of, say, `lm`, but was adopted to ensure users do not have to write out each variable that should be included in the GGM. An example is provided below.

Mixed Type:

The term "mixed" is somewhat of a misnomer, because the method can be used for data including *only* continuous or *only* discrete variables (Hoff 2007). This is based on the ranked likelihood which requires sampling the ranks for each variable (i.e., the data is not merely transformed to ranks). This is computationally expensive when there are many levels. For example, with continuous data, there are as many ranks as data points!

The option `mixed_type` allows the user to determine which variable should be treated as ranks and the "emprical" distribution is used otherwise. This is accomplished by specifying an indicator vector of length p. A one indicates to use the ranks, whereas a zero indicates to "ignore" that variable. By default all integer variables are handled as ranks.

Dealing with Errors:

An error is most likely to arise when `type = "ordinal"`. The are two common errors (although still rare):

- The first is due to sampling the thresholds, especially when the data is heavily skewed. This can result in an ill-defined matrix. If this occurs, we recommend to first try decreasing `prior_sd` (i.e., a more informative prior). If that does not work, then change the data type to `type = mixed` which then estimates a copula GGM (this method can be used for data containing **only** ordinal variable). This should work without a problem.
- The second is due to how the ordinal data are categorized. For example, if the error states that the index is out of bounds, this indicates that the first category is a zero. This is not allowed, as the first category must be one. This is addressed by adding one (e.g., `Y + 1`) to the data matrix.

Imputing Missing Values:

Missing values are imputed with the approach described in Hoff (2009). The basic idea is to impute the missing values with the respective posterior pedictive distribution, given the observed data, as the model is being estimated. Note that the default is `TRUE`, but this ignored when there are no missing values. If set to `FALSE`, and there are missing values, list-wise deletion is performed with `na.omit`.

Value

The returned object of class `confirm` contains a lot of information that is used for printing and plotting the results. For users of **BGGM**, the following are the useful objects:

- `out_hyp_prob` Posterior hypothesis probabilities.
- `info` An object of class `BF` from the R package **BFpack** (Mulder et al. 2019)

Note

"Default" Prior:

In Bayesian statistics, a default Bayes factor needs to have several properties. I refer interested users to section 2.2 in Dablander et al. (2020). In Williams and Mulder (2019), some of these propteries were investigated (e.g., model selection consistency). That said, we would not consider this a "default" or "automatic" Bayes factor and thus we encourage users to perform sensitivity analyses by varying the scale of the prior distribution (`prior_sd`).

Furthermore, it is important to note there is no "correct" prior and, also, there is no need to entertain the possibility of a "true" model. Rather, the Bayes factor can be interpreted as which hypothesis best (relative to each other) predicts the observed data (Section 3.2 in Kass and Raftery 1995).

Interpretation of Conditional (In)dependence Models for Latent Data:

See `BGGM-package` for details about interpreting GGMs based on latent data (i.e, all data types besides `"continuous"`)

References

Dablander F, Bergh Dvd, Ly A, Wagenmakers E (2020). “Default Bayes Factors for Testing the (In) equality of Several Population Variances.” *arXiv preprint arXiv:2003.06278*.

Hoff PD (2007). “Extending the rank likelihood for semiparametric copula estimation.” *The Annals of Applied Statistics*, **1**(1), 265–283. doi:10.1214/07AOAS107.

Hoff PD (2009). *A first course in Bayesian statistical methods*, volume 580. Springer.

Kass RE, Raftery AE (1995). “Bayes Factors.” *Journal of the American Statistical Association*, **90**(430), 773–795.

Mulder J, Gu X, Olsson-Collentine A, Tomarken A, Böing-Messing F, Hoijtink H, Meijerink M, Williams DR, Menke J, Fox J, others (2019). “BFpack: Flexible Bayes Factor Testing of Scientific Theories in R.” *arXiv preprint arXiv:1911.07728*.

Williams DR, Mulder J (2019). “Bayesian Hypothesis Testing for Gaussian Graphical Models: Conditional Independence and Order Constraints.” *PsyArXiv*. doi:10.31234/osf.io/ypxd8.

Williams DR, Rast P, Pericchi LR, Mulder J (2020). “Comparing Gaussian graphical models with the posterior predictive distribution and Bayesian model selection.” *Psychological Methods*. doi:10.1037/met0000254.

Examples

```
# note: iter = 250 for demonstrative purposes

# data
Y <- bfi

###############################
#### example 1: continuous ####
```

```
###############################

# males
Ymale   <- subset(Y, gender == 1,
                  select = -c(education,
                              gender))[,1:5]


# females
Yfemale <- subset(Y, gender == 2,
                     select = -c(education,
                                 gender))[,1:5]

 # exhaustive
 hypothesis <- c("g1_A1--A2 >  g2_A1--A2;
                  g1_A1--A2 <  g2_A1--A2;
                  g1_A1--A2 =  g2_A1--A2")

# test hyp
test <- ggm_compare_confirm(Ymale,  Yfemale,
                            hypothesis = hypothesis,
                            iter = 250,
                            progress = FALSE)

# print (evidence not strong)
test

#########################################
#### example 2: sensitivity to prior ####
#########################################
# continued from example 1

# decrease prior SD
test <- ggm_compare_confirm(Ymale,
                            Yfemale,
                            prior_sd = 0.1,
                            hypothesis = hypothesis,
                            iter = 250,
                            progress = FALSE)

# print
test

# indecrease prior SD
test <- ggm_compare_confirm(Ymale,
                            Yfemale,
                            prior_sd = 0.5,
                            hypothesis = hypothesis,
                            iter = 250,
                            progress = FALSE)

# print
test
```

```
################################
#### example 3: mixed data #####
################################

hypothesis <- c("g1_A1--A2 >  g2_A1--A2;
                 g1_A1--A2 <  g2_A1--A2;
                 g1_A1--A2 =  g2_A1--A2")

# test (1000 for example)
test <- ggm_compare_confirm(Ymale,
                            Yfemale,
                            type = "mixed",
                            hypothesis = hypothesis,
                            iter = 250,
                            progress = FALSE)

# print
test

##############################
##### example 4: control #####
##############################
# control for education

# data
Y <- bfi

# males
Ymale   <- subset(Y, gender == 1,
                  select = -c(gender))[,c(1:5, 26)]

# females
Yfemale <- subset(Y, gender == 2,
                  select = -c(gender))[,c(1:5, 26)]

# test
test <- ggm_compare_confirm(Ymale,
                             Yfemale,
                             formula = ~ education,
                             hypothesis = hypothesis,
                             iter = 250,
                             progress = FALSE)
# print
test


#####################################
##### example 5: many relations #####
#####################################

# data
Y <- bfi
```

```
hypothesis <- c("g1_A1--A2 > g2_A1--A2 & g1_A1--A3 = g2_A1--A3;
                 g1_A1--A2 = g2_A1--A2 & g1_A1--A3 = g2_A1--A3;
                 g1_A1--A2 = g2_A1--A2 = g1_A1--A3 = g2_A1--A3")

Ymale   <- subset(Y, gender == 1,
                  select = -c(education,
                              gender))[,1:5]


# females
Yfemale <- subset(Y, gender == 2,
                     select = -c(education,
                                 gender))[,1:5]

test <- ggm_compare_confirm(Ymale,
                            Yfemale,
                             hypothesis = hypothesis,
                             iter = 250,
                             progress = FALSE)

# print
test
```

`ggm_compare_estimate` *GGM Compare: Estimate*

Description

Compare partial correlations that are estimated from any number of groups. This method works for continuous, binary, ordinal, and mixed data (a combination of categorical and continuous variables). The approach (i.e., a difference between posterior distributions) was described in Williams (2018).

Usage

```
ggm_compare_estimate(
  ...,
  formula = NULL,
  type = "continuous",
  mixed_type = NULL,
  analytic = FALSE,
  prior_sd = 0.5,
  iter = 5000,
  impute = TRUE,
  progress = TRUE,
  seed = 1
)
```

Arguments

`...`	Matrices (or data frames) of dimensions *n* (observations) by *p* (variables). Requires at least two.
`formula`	An object of class `formula`. This allows for including control variables in the model (i.e., ~ `gender`). See the note for further details.
`type`	Character string. Which type of data for **Y** ? The options include `continuous`, `binary`, `ordinal`, or `continuous`. See the note for further details.
`mixed_type`	Numeric vector. An indicator of length *p* for which varibles should be treated as ranks. (1 for rank and 0 to use the 'empirical' or observed distribution). The default is currently to treat all integer variables as ranks when `type = "mixed"` and `NULL` otherwise. See note for further details.
`analytic`	Logical. Should the analytic solution be computed (default is `FALSE`)? This is only available for continous data. Note that if `type = "mixed"` and `analytic = TRUE`, the data will automatically be treated as continuous.
`prior_sd`	The scale of the prior distribution (centered at zero), in reference to a beta distribtuion (defaults to 0.50). See note for further details.
`iter`	Number of iterations (posterior samples; defaults to 5000).
`impute`	Logicial. Should the missing values (`NA`) be imputed during model fitting (defaults to `TRUE`) ?
`progress`	Logical. Should a progress bar be included (defaults to `TRUE`) ?
`seed`	An integer for the random seed.

Details

This function can be used to compare the partial correlations for any number of groups. This is accomplished with pairwise comparisons for each relation. In the case of three groups, for example, group 1 and group 2 are compared, then group 1 and group 3 are compared, and then group 2 and group 3 are compared. There is a full distibution for each difference that can be summarized (i.e., `summary.ggm_compare_estimate`) and then visualized (i.e., `plot.summary.ggm_compare_estimate`). The graph of difference is selected with `select.ggm_compare_estimate`).

Controlling for Variables:

When controlling for variables, it is assumed that `Y` includes *only* the nodes in the GGM and the control variables. Internally, `only` the predictors that are included in `formula` are removed from `Y`. This is not behavior of, say, `lm`, but was adopted to ensure users do not have to write out each variable that should be included in the GGM. An example is provided below.

Mixed Type:

The term "mixed" is somewhat of a misnomer, because the method can be used for data including *only* continuous or *only* discrete variables. This is based on the ranked likelihood which requires sampling the ranks for each variable (i.e., the data is not merely transformed to ranks). This is computationally expensive when there are many levels. For example, with continuous data, there are as many ranks as data points!

The option `mixed_type` allows the user to determine which variable should be treated as ranks and the "emprical" distribution is used otherwise. This is accomplished by specifying an indicator vector of length p. A one indicates to use the ranks, whereas a zero indicates to "ignore" that variable. By default all integer variables are handled as ranks.

Dealing with Errors:

An error is most likely to arise when `type = "ordinal"`. The are two common errors (although still rare):

- The first is due to sampling the thresholds, especially when the data is heavily skewed. This can result in an ill-defined matrix. If this occurs, we recommend to first try decreasing `prior_sd` (i.e., a more informative prior). If that does not work, then change the data type to `type = mixed` which then estimates a copula GGM (this method can be used for data containing **only** ordinal variable). This should work without a problem.
- The second is due to how the ordinal data are categorized. For example, if the error states that the index is out of bounds, this indicates that the first category is a zero. This is not allowed, as the first category must be one. This is addressed by adding one (e.g., `Y + 1`) to the data matrix.

Imputing Missing Values:

Missing values are imputed with the approach described in Hoff (2009). The basic idea is to impute the missing values with the respective posterior pedictive distribution, given the observed data, as the model is being estimated. Note that the default is `TRUE`, but this ignored when there are no missing values. If set to `FALSE`, and there are missing values, list-wise deletion is performed with `na.omit`.

Value

A list of class `ggm_compare_estimate` containing:

- `pcor_diffs` partial correlation differences (posterior distribution)
- `p` number of variable
- `info` list containing information about each group (e.g., sample size, etc.)
- `iter` number of posterior samples
- `call` `match.call`

Note

Mixed Data:

The mixed data approach was introduced in Hoff (2007) (our paper describing an extension to Bayesian hypothesis testing if forthcoming). This is a semi-paramateric copula model based on the ranked likelihood. This is computationally expensive when treating continuous data as ranks. The current default is to treat only integer data as ranks. This should of course be adjusted for continous data that is skewed. This can be accomplished with the argument `mixed_type`. A `1` in the numeric vector of length $p$indicates to treat that respective node as a rank (corresponding to the column number) and a zero indicates to use the observed (or "emprical") data.

It is also important to note that `type = "mixed"` is not restricted to mixed data (containing a combination of categorical and continuous): all the nodes can be ordinal or continuous (but again this will take some time).

Interpretation of Conditional (In)dependence Models for Latent Data:

See `BGGM-package` for details about interpreting GGMs based on latent data (i.e, all data types besides `"continuous"`)

Additional GGM Compare Methods

Bayesian hypothesis testing is implemented in `ggm_compare_explore` and `ggm_compare_confirm` (Williams and Mulder 2019). The latter allows for confirmatory hypothesis testing. An approach based on a posterior predictive check is implemented in `ggm_compare_ppc` (Williams et al. 2020). This provides a 'global' test for comparing the entire GGM and a 'nodewise' test for comparing each variable in the network Williams (2018).

References

Hoff PD (2007). "Extending the rank likelihood for semiparametric copula estimation." *The Annals of Applied Statistics*, **1**(1), 265–283. doi:10.1214/07AOAS107.

Hoff PD (2009). *A first course in Bayesian statistical methods*, volume 580. Springer.

Williams DR (2018). "Bayesian Estimation for Gaussian Graphical Models: Structure Learning, Predictability, and Network Comparisons." *arXiv*. doi:10.31234/OSF.IO/X8DPR.

Williams DR, Mulder J (2019). "Bayesian Hypothesis Testing for Gaussian Graphical Models: Conditional Independence and Order Constraints." *PsyArXiv*. doi:10.31234/osf.io/ypxd8.

Williams DR, Rast P, Pericchi LR, Mulder J (2020). "Comparing Gaussian graphical models with the posterior predictive distribution and Bayesian model selection." *Psychological Methods*. doi:10.1037/met0000254.

Examples

```
# note: iter = 250 for demonstrative purposes

# data
Y <- bfi

# males and females
Ymale <- subset(Y, gender == 1,
                   select = -c(gender,
                               education))[,1:10]

Yfemale <- subset(Y, gender == 2,
                     select = -c(gender,
                                 education))[,1:10]

# fit model
fit <- ggm_compare_estimate(Ymale,  Yfemale,
                           type = "ordinal",
                           iter = 250,
                           prior_sd = 0.25,
                           progress = FALSE)
```

```
###########################
### example 2: analytic ###
###########################
# only continuous

# fit model
fit <- ggm_compare_estimate(Ymale, Yfemale,
                            analytic = TRUE)

# summary
summ <- summary(fit)

# plot summary
plt_summ <- plot(summary(fit))

# select
E <- select(fit)

# plot select
plt_E <- plot(select(fit))
```

`ggm_compare_explore` *GGM Compare: Exploratory Hypothesis Testing*

Description

Compare Gaussian graphical models with exploratory hypothesis testing using the matrix-F prior distribution (Mulder and Pericchi 2018). A test for each partial correlation in the model for any number of groups. This provides evidence for the null hypothesis of no difference and the alternative hypothesis of difference. With more than two groups, the test is for *all* groups simultaneously (i.e., the relation is the same or different in all groups). This method was introduced in Williams et al. (2020). For confirmatory hypothesis testing see `confirm_groups`.

Usage

```
ggm_compare_explore(
  ...,
  formula = NULL,
  type = "continuous",
  mixed_type = NULL,
  analytic = FALSE,
  prior_sd = 0.2,
  iter = 5000,
  progress = TRUE,
  seed = 1
)
```

Arguments

`...`	At least two matrices (or data frame) of dimensions *n* (observations) by *p* (variables).
`formula`	An object of class `formula`. This allows for including control variables in the model (i.e., `~ gender`).
`type`	Character string. Which type of data for `Y` ? The options include `continuous`, `binary`, or `ordinal`. See the note for further details.
`mixed_type`	Numeric vector. An indicator of length p for which varibles should be treated as ranks. (1 for rank and 0 to assume normality). The default is currently (dev version) to treat all integer variables as ranks when `type = "mixed"` and `NULL` otherwise. See note for further details.
`analytic`	logical. Should the analytic solution be computed (default is `FALSE`) ? See note for details.
`prior_sd`	Numeric. The scale of the prior distribution (centered at zero), in reference to a beta distribtuion. The ‘default‘ is 0.20. See note for further details.
`iter`	number of iterations (posterior samples; defaults to 5000).
`progress`	Logical. Should a progress bar be included (defaults to `TRUE`) ?
`seed`	An integer for the random seed.

Details

Controlling for Variables:

When controlling for variables, it is assumed that `Y` includes *only* the nodes in the GGM and the control variables. Internally, `only` the predictors that are included in `formula` are removed from `Y`. This is not behavior of, say, `lm`, but was adopted to ensure users do not have to write out each variable that should be included in the GGM. An example is provided below.

Mixed Type:

The term "mixed" is somewhat of a misnomer, because the method can be used for data including *only* continuous or *only* discrete variables. This is based on the ranked likelihood which requires sampling the ranks for each variable (i.e., the data is not merely transformed to ranks). This is computationally expensive when there are many levels. For example, with continuous data, there are as many ranks as data points!

The option `mixed_type` allows the user to determine which variable should be treated as ranks and the "emprical" distribution is used otherwise. This is accomplished by specifying an indicator vector of length *p*. A one indicates to use the ranks, whereas a zero indicates to "ignore" that variable. By default all integer variables are handled as ranks.

Dealing with Errors:

An error is most likely to arise when `type = "ordinal"`. The are two common errors (although still rare):

- The first is due to sampling the thresholds, especially when the data is heavily skewed. This can result in an ill-defined matrix. If this occurs, we recommend to first try decreasing `prior_sd` (i.e., a more informative prior). If that does not work, then change the data type to `type = mixed` which then estimates a copula GGM (this method can be used for data containing **only** ordinal variable). This should work without a problem.

- The second is due to how the ordinal data are categorized. For example, if the error states that the index is out of bounds, this indicates that the first category is a zero. This is not allowed, as the first category must be one. This is addressed by adding one (e.g., `Y + 1`) to the data matrix.

Value

The returned object of class `ggm_compare_explore` contains a lot of information that is used for printing and plotting the results. For users of **BGGM**, the following are the useful objects:

- `BF_01` A p by p matrix including the Bayes factor for the null hypothesis.
- `pcor_diff` A p by p matrix including the difference in partial correlations (only for two groups).
- `samp` A list containing the fitted models (of class `explore`) for each group.

Note

"Default" Prior:

In Bayesian statistics, a default Bayes factor needs to have several properties. I refer interested users to section 2.2 in Dablander et al. (2020). In Williams and Mulder (2019), some of these propteries were investigated, such model selection consistency. That said, we would not consider this a "default" Bayes factor and thus we encourage users to perform sensitivity analyses by varying the scale of the prior distribution.

Furthermore, it is important to note there is no "correct" prior and, also, there is no need to entertain the possibility of a "true" model. Rather, the Bayes factor can be interpreted as which hypothesis best (relative to each other) predicts the observed data (Section 3.2 in Kass and Raftery 1995).

Interpretation of Conditional (In)dependence Models for Latent Data:

See `BGGM-package` for details about interpreting GGMs based on latent data (i.e, all data types besides `"continuous"`)

References

Dablander F, Bergh Dvd, Ly A, Wagenmakers E (2020). “Default Bayes Factors for Testing the (In) equality of Several Population Variances.” *arXiv preprint arXiv:2003.06278*.

Kass RE, Raftery AE (1995). “Bayes Factors.” *Journal of the American Statistical Association*, **90**(430), 773–795.

Mulder J, Pericchi L (2018). “The Matrix-F Prior for Estimating and Testing Covariance Matrices.” *Bayesian Analysis*, 1–22. ISSN 19316690, doi:10.1214/17BA1092.

Williams DR, Mulder J (2019). “Bayesian Hypothesis Testing for Gaussian Graphical Models: Conditional Independence and Order Constraints.” *PsyArXiv*. doi:10.31234/osf.io/ypxd8.

Williams DR, Rast P, Pericchi LR, Mulder J (2020). “Comparing Gaussian graphical models with the posterior predictive distribution and Bayesian model selection.” *Psychological Methods*. doi:10.1037/met0000254.

Examples

```
# note: iter = 250 for demonstrative purposes

# data
Y <- bfi

# males and females
Ymale <- subset(Y, gender == 1,
                   select = -c(gender,
                               education))[,1:10]

Yfemale <- subset(Y, gender == 2,
                     select = -c(gender,
                                 education))[,1:10]

##########################
### example 1: ordinal ###
##########################

# fit model
fit <- ggm_compare_explore(Ymale,  Yfemale,
                           type = "ordinal",
                           iter = 250,
                           progress = FALSE)
# summary
summ <- summary(fit)

# edge set
E <- select(fit)
```

`ggm_compare_ppc` *GGM Compare: Posterior Predictive Check*

Description

Compare GGMs with a posterior predicitve check (Gelman et al. 1996). This method was introduced in Williams et al. (2020). Currently, there is a `global` (the entire GGM) and a `nodewise` test. The default is to compare GGMs with respect to the posterior predictive distribution of Kullback Leibler divergence and the sum of squared errors. It is also possible to compare the GGMs with a user defined test-statistic.

Usage

```
ggm_compare_ppc(
  ...,
```

```
  test = "global",
  iter = 5000,
  FUN = NULL,
  custom_obs = NULL,
  loss = TRUE,
  progress = TRUE
)
```

Arguments

`...`	At least two matrices (or data frames) of dimensions *n* (observations) by *p* (variables).
`test`	Which test should be performed (defaults to `"global"`) ? The options include `global` and `nodewise`.
`iter`	Number of replicated datasets used to construct the predictivie distribution (defaults to 5000).
`FUN`	An optional function for comparing GGMs that returns a number. See **Details**.
`custom_obs`	Number corresponding to the observed score for comparing the GGMs. This is required if a function is provided in `FUN`. See **Details**.
`loss`	Logical. If a function is provided, is the measure a "loss function" (i.e., a large score is bad thing). This determines how the *p*-value is computed. See **Details**.
`progress`	Logical. Should a progress bar be included (defaults to `TRUE`) ?

Details

The `FUN` argument allows for a user defined test-statisic (the measure used to compare the GGMs). The function must include only two agruments, each of which corresponds to a dataset. For example, `f <- function(Yg1, Yg2)`, where each Y is dataset of dimensions *n* by *p*. The groups are then compare within the function, returning a single number. An example is provided below.

Further, when using a custom function care must be taken when specifying the argument `loss`. We recommended to visualize the results with `plot` to ensure the *p*-value was computed in the right direction.

Value

The returned object of class `ggm_compare_ppc` contains a lot of information that is used for printing and plotting the results. For users of **BGGM**, the following are the useful objects:

`test = "global"`

- `ppp_jsd` posterior predictive *p*-values (JSD).
- `ppp_sse` posterior predictive *p*-values (SSE).
- `predictive_jsd` list containing the posterior predictive distributions (JSD).
- `predictive_sse` list containing the posterior predictive distributions (SSE).
- `obs_jsd` list containing the observed error (JSD).
- `obs_sse` list containing the observed error (SSE).

`test = "nodewise"`

- `ppp_jsd` posterior predictive *p*-values (JSD).
- `predictive_jsd` list containing the posterior predictive distributions (JSD).
- `obs_jsd` list containing the observed error (JSD).

`FUN = f()`

- `ppp_custom` posterior predictive *p*-values (custom).
- `predictive_custom` posterior predictive distributions (custom).
- `obs_custom` observed error (custom).

Note

Interpretation:

The primary test-statistic is symmetric KL-divergence that is termed Jensen-Shannon divergence (JSD). This is in essence a likelihood ratio that provides the "distance" between two multivariate normal distributions. The basic idea is to (1) compute the posterior predictive distribution, assuming group equality (the null model). This provides the error that we would expect to see under the null model; (2) compute JSD for the observed groups; and (3) compare the observed JSD to the posterior predictive distribution, from which a posterior predictive *p*-value is computed.

For the `global` check, the sum of squared error is also provided. This is computed from the partial correlation matrices and it is analagous to the strength test in van Borkulo et al. (2017). The `nodewise` test compares the posterior predictive distribution for each node. This is based on the correspondence between the inverse covariance matrix and multiple regresssion (Kwan 2014; Stephens 1998).

If the null model is `not` rejected, note that this does `not` provide evidence for equality! Further, if the null model is rejected, this means that the assumption of group equality is not tenable–the groups are different.

Alternative Methods:

There are several methods in **BGGM** for comparing groups. See `ggm_compare_estimate` (posterior differences for the partial correlations), `ggm_compare_explore` (exploratory hypothesis testing), and `ggm_compare_confirm` (confirmatory hypothesis testing).

References

Gelman A, Meng X, Stern H (1996). “Posterior predictive assessment of model fitness via realized discrepancies.” *Statistica sinica*, 733–760.

Kwan CC (2014). “A regression-based interpretation of the inverse of the sample covariance matrix.” *Spreadsheets in Education*, **7**(1), 4613.

Stephens G (1998). “On the Inverse of the Covariance Matrix in Portfolio Analysis.” *The Journal of Finance*, **53**(5), 1821–1827.

Williams DR, Rast P, Pericchi LR, Mulder J (2020). “Comparing Gaussian graphical models with the posterior predictive distribution and Bayesian model selection.” *Psychological Methods*.

doi:10.1037/met0000254.

van Borkulo CD, Boschloo L, Kossakowski J, Tio P, Schoevers RA, Borsboom D, Waldorp LJ (2017). “Comparing network structures on three aspects: A permutation test.” *Manuscript submitted for publication*.

Examples

```


# note: iter = 250 for demonstrative purposes

# data
Y <- bfi

#############################
######### global ############
#############################


# males
Ym <- subset(Y, gender == 1,
             select = - c(gender, education))

# females

Yf <- subset(Y, gender == 2,
             select = - c(gender, education))


global_test <- ggm_compare_ppc(Ym, Yf,
                               iter = 250)

global_test


#############################
###### custom function ######
#############################
# example 1

# maximum difference van Borkulo et al. (2017)

f <- function(Yg1, Yg2){

# remove NA
x <- na.omit(Yg1)
y <- na.omit(Yg2)

# nodes
p <- ncol(Yg1)
```

```
# identity matrix
I_p <- diag(p)

# partial correlations

pcor_1 <- -(cov2cor(solve(cor(x))) - I_p)
pcor_2 <- -(cov2cor(solve(cor(y))) - I_p)

# max difference
max(abs((pcor_1[upper.tri(I_p)] - pcor_2[upper.tri(I_p)])))

}

# observed difference
obs <- f(Ym, Yf)

global_max <- ggm_compare_ppc(Ym, Yf,
                              iter = 250,
                              FUN = f,
                              custom_obs = obs,
                              progress = FALSE)

global_max


# example 2
# Hamming distance (squared error for adjacency)

f <- function(Yg1, Yg2){

# remove NA
x <- na.omit(Yg1)
y <- na.omit(Yg2)

# nodes
p <- ncol(x)

# identity matrix
I_p <- diag(p)

fit1 <-  estimate(x, analytic = TRUE)
fit2 <-  estimate(y, analytic = TRUE)

sel1 <- select(fit1)
sel2 <- select(fit2)

sum((sel1$adj[upper.tri(I_p)] - sel2$adj[upper.tri(I_p)])^2)

}

# observed difference
obs <- f(Ym, Yf)
```

```
global_hd <- ggm_compare_ppc(Ym, Yf,
                             iter = 250,
                             FUN = f,
                             custom_obs  = obs,
                             progress = FALSE)

global_hd


#############################
########  nodewise ##########
#############################

nodewise <- ggm_compare_ppc(Ym, Yf, iter = 250,
                            test = "nodewise")

nodewise
```

`gss` *Data: 1994 General Social Survey*

Description

A data frame containing 1002 rows and 7 variables measured on various scales, including binary and ordered cateogrical (with varying numbers of categories). There are also missing values in each variable

- `Inc` Income of the respondent in 1000s of dollars, binned into 21 ordered categories.
- `DEG` Highest degree ever obtained (none, HS, Associates, Bachelors, or Graduate)
- `CHILD` Number of children ever had.
- `PINC` Financial status of respondent's parents when respondent was 16 (on a 5-point scale).
- `PDEG` Maximum of mother's and father's highest degree
- `PCHILD` Number of siblings of the respondent plus one
- `AGE` Age of the respondent in years.

Usage

```
data("gss")
```

Format

A data frame containing 1190 observations (n = 1190) and 6 variables (p = 6) measured on the binary scale (Fowlkes et al. 1988). The variable descriptions were copied from section 4, Hoff (2007)

References

Fowlkes EB, Freeny AE, Landwehr JM (1988). "Evaluating logistic models for large contingency tables." *Journal of the American Statistical Association*, **83**(403), 611–622. doi:10.1080/01621459.1988.10478640.

Hoff PD (2007). "Extending the rank likelihood for semiparametric copula estimation." *The Annals of Applied Statistics*, **1**(1), 265–283. doi:10.1214/07AOAS107.

Examples

```
data("gss")
```

`ifit` *Data: ifit Intensive Longitudinal Data*

Description

A data frame containing 8 variables and nearly 200 observations. There are two subjects, each of which provided data every data for over 90 days. Six variables are from the PANAS scale (positive and negative affect), the daily number of steps, and the subject id.

- `id` Subject id
- `interested`
- `disinterested`
- `excited`
- `upset`
- `strong`
- `stressed`
- `steps` steps recorded by a fit bit

Usage

```
data("ifit")
```

Format

A data frame containing 197 observations and 8 variables. The data have been used in (O'Laughlin et al. 2020) and (Williams et al. 2019)

References

O'Laughlin KD, Liu S, Ferrer E (2020). "Use of Composites in Analysis of Individual Time Series: Implications for Person-Specific Dynamic Parameters." *Multivariate Behavioral Research*, 1–18. doi:10.1080/00273171.2020.1716673.

Williams DR, Liu S, Martin SR, Rast P (2019). "Bayesian Multivariate Mixed-Effects Location Scale Modeling of Longitudinal Relations among Affective Traits, States, and Physical Activity." *PsyArXiv*. doi:10.31234/osf.io/4kfjp.

Examples

```
data("ifit")
```

`impute_data` *Obtain Imputed Datasets*

Description

Impute missing values, assuming a multivariate normal distribution, with the posterior predictive distribution. For binary, ordinal, and mixed (a combination of discrete and continuous) data, the values are first imputed for the latent data and then converted to the original scale.

Usage

```
impute_data(
  Y,
  type = "continuous",
  lambda = NULL,
  mixed_type = NULL,
  iter = 1000,
  progress = TRUE
)
```

Arguments

`Y`	Matrix (or data frame) of dimensions *n* (observations) by *p* (variables).
`type`	Character string. Which type of data for `Y` ? The options include `continuous`, `binary`, `ordinal`, or `mixed`. Note that mixed can be used for data with only ordinal variables. See the note for further details.
`lambda`	Numeric. A regularization parameter, which defaults to p + 2. A larger value results in more shrinkage.
`mixed_type`	Numeric vector. An indicator of length *p* for which variables should be treated as ranks. (1 for rank and 0 to assume the observed marginal distribution). The default is currently to treat all integer variables as ranks when `type = "mixed"` and `NULL` otherwise. See note for further details.
`iter`	Number of iterations (posterior samples; defaults to 1000).
`progress`	Logical. Should a progress bar be included (defaults to `TRUE`) ?

Details

Missing values are imputed with the approach described in Hoff (2009). The basic idea is to impute the missing values with the respective posterior pedictive distribution, given the observed data, as the model is being estimated. Note that the default is `TRUE`, but this ignored when there are no missing values. If set to `FALSE`, and there are missing values, list-wise deletion is performed with `na.omit`.

Value

An object of class `mvn_imputation`:

- `imputed_datasets` An array including the imputed datasets.

References

Hoff PD (2009). *A first course in Bayesian statistical methods*, volume 580. Springer.

Examples

```
# obs
n <- 5000

# n missing
n_missing <- 1000

# variables
p <- 16

# data
Y <- MASS::mvrnorm(n, rep(0, p), ptsd_cor1)

# for checking
Ymain <- Y

# all possible indices
indices <- which(matrix(0, n, p) == 0,
                 arr.ind = TRUE)

# random sample of 1000 missing values
na_indices <- indices[sample(5:nrow(indices),
                             size = n_missing,
                             replace = FALSE),]

# fill with NA
Y[na_indices] <- NA

# missing = 1
Y_miss <- ifelse(is.na(Y), 1, 0)

# true values (to check)
true <- unlist(sapply(1:p, function(x)
        Ymain[which(Y_miss[,x] == 1),x] ))

# impute
fit_missing <- impute_data(Y, progress = FALSE, iter = 250)

# impute
fit_missing <- impute_data(Y,
                           progress = TRUE,
```

```
iter = 250)
```

`iri` *Data: Interpersonal Reactivity Index (IRI)*

Description

A dataset containing items from the Interpersonal Reactivity Index (IRI; an empathy measure). There are 28 variables and 1973 observations

Usage

```
data("iri")
```

Format

A data frame with 28 variables and 1973 observations (5 point Likert scale)

Details

- `1` I daydream and fantasize, with some regularity, about things that might happen to me.
- `2` I often have tender, concerned feelings for people less fortunate than me.
- `3` I sometimes find it difficult to see things from the "other guy's" point of view.
- `4` Sometimes I don't feel very sorry for other people when they are having problems.
- `5` I really get involved with the feelings of the characters in a novel.
- `6` In emergency situations, I feel apprehensive and ill-at-ease.
- `7` I am usually objective when I watch a movie or play, and I don't often get completely caught up in it.
- `8` I try to look at everybody's side of a disagreement before I make a decision.
- `9` When I see someone being taken advantage of, I feel kind of protective towards them.
- `10` I sometimes feel helpless when I am in the middle of a very emotional situation.
- `11` I sometimes try to understand my friends better by imagining how things look from their perspective
- `12` Becoming extremely involved in a good book or movie is somewhat rare for me.
- `13` When I see someone get hurt, I tend to remain calm.
- `14` Other people's misfortunes do not usually disturb me a great deal.
- `15` If I'm sure I'm right about something, I don't waste much time listening to other people's arguments.
- `16` After seeing a play or movie, I have felt as though I were one of the characters.
- `17` Being in a tense emotional situation scares me.

- `18` When I see someone being treated unfairly, I sometimes don't feel very much pity for them.
- `19` I am usually pretty effective in dealing with emergencies.
- `20` I am often quite touched by things that I see happen.
- `21` I believe that there are two sides to every question and try to look at them both.
- `22` I would describe myself as a pretty soft-hearted person.
- `23` When I watch a good movie, I can very easily put myself in the place of a leading character
- `24` I tend to lose control during emergencies.
- `25` When I'm upset at someone, I usually try to "put myself in his shoes" for a while.
- `26` When I am reading an interesting story or novel, I imagine how I would feel if the events in the story were happening to me.
- `27` When I see someone who badly needs help in an emergency, I go to pieces.
- `28` Before criticizing somebody, I try to imagine how I would feel if I were in their place.
- `gender` "M" (male) or "F" (female)

Note

There are four domains

Fantasy: items 1, 5, 7, 12, 16, 23, 26

Perspective taking: items 3, 8, 11, 15, 21, 25, 28

Empathic concern: items 2, 4, 9, 14, 18, 20, 22

Personal distress: items 6, 10, 13, 17, 19, 24, 27,

References

Briganti, G., Kempenaers, C., Braun, S., Fried, E. I., & Linkowski, P. (2018). Network analysis of empathy items from the interpersonal reactivity index in 1973 young adults. Psychiatry research, 265, 87-92.

Examples

```
data("iri")
```

`map` *Maximum A Posteriori Precision Matrix*

Description

Maximum A Posteriori Precision Matrix

Usage

```
map(Y)
```

Arguments

`Y`	Matrix (or data frame) of dimensions *n* (observations) by *p* (variables).

Value

An object of class `map`, including the precision matrix, partial correlation matrix, and regression parameters.

Examples

```
Y <- BGGM::bfi[, 1:5]

# map
map <- map(Y)
map
```

`pcor_mat` *Extract the Partial Correlation Matrix*

Description

Extract the partial correlation matrix (posterior mean) from `estimate`, `explore`, `ggm_compare_estimate`, and `ggm_compare_explore` objects. It is also possible to extract the partial correlation differences for `ggm_compare_estimate` and `ggm_compare_explore` objects.

Usage

```
pcor_mat(object, difference = FALSE, ...)
```

Arguments

`object`	A model estimated with **BGGM**. All classes are supported, assuming there is matrix to be extracted.
`difference`	Logical. Should the difference be returned (defaults to `FALSE`) ? Note that this assumes there is a difference (e.g., an object of class `ggm_compare_estimate`) and ignored otherwise.
`...`	Currently ignored.

Value

The estimated partial correlation matrix.

Examples

```
# note: iter = 250 for demonstrative purposes

# data
Y <- ptsd[,1:5] + 1

# ordinal
fit <- estimate(Y, type = "ordinal",
                iter = 250,
                progress = FALSE)

pcor_mat(fit)
```

`pcor_sum` *Partial Correlation Sum*

Description

Compute and test partial correlation sums either within or between GGMs (e.g., different groups), resulting in a posterior distribution.

Usage

```
pcor_sum(..., iter = NULL, relations)
```

Arguments

`...`	An object of class `estimate`. This can be either one or two fitted objects.
`iter`	Number of iterations (posterior samples; defaults to the number in the object).
`relations`	Character string. Which partial correlations should be summed?

Details

Some care must be taken when writing the string for `partial_sum`. Below are several examples

Just a Sum: Perhaps a sum is of interest, and not necessarily the difference of two sums. This can be written as

- `partial_sum <- c("A1--A2 + A1--A3 + A1--A4")`

which will sum those relations.

Comparing Sums: When comparing sums, each must be seperated by "`;`". For example,

- `partial_sum <- c("A1--A2 + A1--A3; A1--A2 + A1--A4")` which will sum both and compute the difference. Note that there cannot be more than two sums, such that `c("A1--A2 + A1--A3; A1--A2 + A1--A4; A1--A2 + A1--A5")` will result in an error.

Comparing Groups:

When more than one fitted object is suppled to `object` it is assumed that the groups should be compared for the same sum. Hence, in this case, only the sum needs to be written.

- `partial_sum <- c("A1--A2 + A1--A3 + A1--A4")`

The above results in that sum being computed for each group and then compared.

Value

An object of class `posterior_sum`, including the sum and possibly the difference for two sums.

Examples

```
# data
Y <- bfi

# males
Y_males <- subset(Y, gender == 1, select = -c(education, gender))[,1:5]

# females
Y_females <- subset(Y, gender == 2, select = -c(education, gender))[,1:5]

# males
fit_males <- estimate(Y_males, seed = 1,
                      progress = FALSE)

# fit females
fit_females <- estimate(Y_females, seed = 2,
                        progress = FALSE)


sums <- pcor_sum(fit_males,
                 fit_females,
                 relations = "A1--A2 + A1--A3")
# print
sums

# plot difference
plot(sums)[[3]]
```

`pcor_to_cor` *Compute Correlations from the Partial Correlations*

Description

Convert the partial correlation matrices into correlation matrices. To our knowledge, this is the only Bayesian implementation in `R` that can estiamte Pearson's, tetrachoric (binary), polychoric (ordinal with more than two cateogries), and rank based correlation coefficients.

Usage

```
pcor_to_cor(object, iter = NULL)
```

Arguments

`object`	An object of class `estimate` or `explore`
`iter`	numeric. How many iterations (i.e., posterior samples) should be used ? The default uses all of the samples, but note that this can take a long time with large matrices.

Value

- `R` An array including the correlation matrices (of dimensions *p* by *p* by *iter*)
- `R_mean` Posterior mean of the correlations (of dimensions *p* by *p*)

Note

The 'default' prior distributions are specified for partial correlations in particular. This means that the implied prior distribution will not be the same for the correlations.

Examples

```
# note: iter = 250 for demonstrative purposes

# data
Y <- BGGM::ptsd

#########################
###### continuous #######
#########################

# estimate the model
fit <- estimate(Y, iter = 250,
                progress = FALSE)

# compute correlations
cors <- pcor_to_cor(fit)
```

```
#########################
###### ordinal  #########
#########################

# first level must be 1 !
Y <- Y + 1

# estimate the model
fit <- estimate(Y, type =  "ordinal",
                iter = 250,
                progress = FALSE)

# compute correlations
cors <- pcor_to_cor(fit)


#########################
#######   mixed    ######
#########################

# rank based correlations

# estimate the model
fit <- estimate(Y, type =  "mixed",
                iter = 250,
                progress = FALSE)

# compute correlations
cors <- pcor_to_cor(fit)
```

`plot.confirm` *Plot* `confirm` *objects*

Description

Plot the posterior hypothesis probabilities as a pie chart, with each slice corresponding the probability of a given hypothesis.

Usage

```
## S3 method for class 'confirm'
plot(x, ...)
```

Arguments

`x`	An object of class `confirm`
`...`	Currently ignored.

Value

A `ggplot` object.

Examples

```
#####################################
##### example 1: many relations #####
#####################################

# data
Y <- bfi

hypothesis <- c("g1_A1--A2 > g2_A1--A2 & g1_A1--A3 = g2_A1--A3;
                 g1_A1--A2 = g2_A1--A2 & g1_A1--A3 = g2_A1--A3;
                 g1_A1--A2 = g2_A1--A2 = g1_A1--A3 = g2_A1--A3")

Ymale   <- subset(Y, gender == 1,
                  select = -c(education,
                              gender))[,1:5]


# females
Yfemale <- subset(Y, gender == 2,
                     select = -c(education,
                                 gender))[,1:5]

test <- ggm_compare_confirm(Ymale,
                            Yfemale,
                            hypothesis = hypothesis,
                            iter = 250,
                            progress = FALSE)


# plot
plot(test)
```

`plot.ggm_compare_ppc` *Plot* `ggm_compare_ppc` *Objects*

Description

Plot the predictive check with `ggridges`

Usage

```
## S3 method for class 'ggm_compare_ppc'
plot(
  x,
  critical = 0.05,
  col_noncritical = "#84e184A0",
  col_critical = "red",
  point_size = 2,
  ...
)
```

Arguments

`x`	An object of class `ggm_compare_ppc`
`critical`	Numeric. The 'significance' level (defaults to `0.05`).
`col_noncritical`	Character string. Fill color for the non-critical region (defaults to `"#84e184A0"`).
`col_critical`	Character string. Fill color for the critical region (defaults to `"red"`).
`point_size`	Numeric. The point size for the observed score (defaults to `2`).
`...`	Currently ignored.

Value

An object (or list of objects) of class `ggplot`.

Note

See ggridges for many examples.

See Also

`ggm_compare_ppc`

Examples

```
# data
Y <- bfi

#############################
######### global ############
#############################
# males
Ym <- subset(Y, gender == 1,
             select = - c(gender, education))

# females

Yf <- subset(Y, gender == 2,
```

```
            select = - c(gender, education))


global_test <- ggm_compare_ppc(Ym, Yf,
                               iter = 250,
                               progress = FALSE)

plot(global_test)
```

`plot.pcor_sum` *Plot* `pcor_sum` *Object*

Description

Plot `pcor_sum` Object

Usage

```
## S3 method for class 'pcor_sum'
plot(x, fill = "#CC79A7", ...)
```

Arguments

`x`	An object of class `posterior_sum`
`fill`	Character string. What fill for the histogram (defaults to colorblind "pink")?
`...`	Currently ignored.

Value

A list of `ggplot` objects

Note

Examples:

See Also

pcor_sum

`plot.predictability` *Plot* `predictability` *Objects*

Description

Plot `predictability` Objects

Usage

```
## S3 method for class 'predictability'
plot(
  x,
  type = "error_bar",
  cred = 0.95,
  alpha = 0.5,
  scale = 1,
  width = 0,
  size = 1,
  color = "blue",
  ...
)
```

Arguments

`x`	An object of class `predictability`
`type`	Character string. Which type of plot ? The options are `"error_bar"` or `"ridgeline"` (defaults to `"error_bar"`).
`cred`	Numeric. The credible interval width for summarizing the posterior distributions (defaults to 0.95; must be between 0 and 1).
`alpha`	Numeric. Transparancey of the ridges
`scale`	Numeric. This controls the overlap of densities for `type = "ridgeline"` (defaults to 1).
`width`	Numeric. The width of error bar ends (defaults to `0`) for `type = "error_bar"`.
`size`	Numeric. The size for the points (defaults to `2`) for `type = "error_bar"`.
`color`	Character string. What color for the point (`type = "error_bar"`) or tail region (`type = "ridgeline"`) ? Defaults to `"blue"`.
`...`	Currently ignored.

Value

An object of class `ggplot`.

Examples

```
Y <- ptsd[,1:5]

fit <- explore(Y, iter = 250,
               progress = FALSE)

r2 <- predictability(fit, iter = 250,
                     progress = FALSE)

plot(r2)
```

`plot.roll_your_own` *Plot* `roll_your_own` *Objects*

Description

Plot `roll_your_own` Objects

Usage

```
## S3 method for class 'roll_your_own'
plot(x, fill = "#CC79A7", alpha = 0.5, ...)
```

Arguments

`x`	An object of class `roll_your_own`
`fill`	Character string specifying the color for the ridges.
`alpha`	Numeric. Transparancey of the ridges
`...`	Currently ignored

Value

An object of class `ggplot`

Examples

```
####################################
###### example 1: assortment #######
####################################
# assortment
library(assortnet)

Y <- BGGM::bfi[,1:10]
membership <- c(rep("a", 5), rep("c", 5))
```

```
# fit model
fit <- estimate(Y = Y, iter = 250,
                progress = FALSE)

# membership
membership <- c(rep("a", 5), rep("c", 5))

# define function
f <- function(x,...){
 assortment.discrete(x, ...)$r
}

net_stat <- roll_your_own(object = fit,
                          FUN = f,
                          types = membership,
                          weighted = TRUE,
                          SE = FALSE, M = 1,
                          progress = FALSE)

# plot
plot(net_stat)
```

`plot.select` *Network Plot for* `select` *Objects*

Description

Visualize the conditional (in)dependence structure.

Usage

```
## S3 method for class 'select'
plot(
  x,
  layout = "circle",
  pos_col = "#009E73",
  neg_col = "#D55E00",
  node_size = 10,
  edge_magnify = 1,
  groups = NULL,
  palette = "Set3",
  ...
)
```

Arguments

`x`	An object of class `select`.
`layout`	Character string. Which graph layout (defaults is `circle`) ? See gplot.layout.
`pos_col`	Character string. Color for the positive edges (defaults to `green`).
`neg_col`	Character string. Color for the negative edges (defaults to `green`).
`node_size`	Numeric. The size of the nodes (defaults to `10`).
`edge_magnify`	Numeric. A value that is multiplied by the edge weights. This increases (> 1) or decrease (< 1) the line widths (defaults to 1).
`groups`	A character string of length p (the number of nodes in the model). This indicates groups of nodes that should be the same color (e.g., "clusters" or "communities").
`palette`	A character string sepcifying the palette for the `groups`. (default is `Set3`). See palette options here.
`...`	Additional options passed to ggnet2

Value

An object (or list of objects) of class `ggplot` that can then be further customized.

Note

A more extensive example of a custom plot is provided here

Examples

```
#########################
### example 1: one ggm ##
#########################

# data
Y <- bfi[,1:25]

# estimate
fit <- estimate(Y, iter = 250,
                progress = FALSE)

# "communities"
comm <- substring(colnames(Y), 1, 1)

# edge set
E <- select(fit)

# plot edge set
plt_E <- plot(E, edge_magnify = 5,
              palette = "Set1",
              groups = comm)
```

```
#############################
### example 2: ggm compare ##
#############################
# compare males vs. females

# data
Y <- bfi[,1:26]

Ym <- subset(Y, gender == 1,
             select = -gender)

Yf <- subset(Y, gender == 2,
              select = -gender)

# estimate
fit <- ggm_compare_estimate(Ym, Yf, iter = 250,
                            progress = FALSE)

# "communities"
comm <- substring(colnames(Ym), 1, 1)

# edge set
E <- select(fit)

# plot edge set
plt_E <- plot(E, edge_magnify = 5,
              palette = "Set1",
              groups = comm)
```

`plot.summary.estimate` *Plot* `summary.estimate` *Objects*

Description

Visualize the posterior distributions for each partial correlation.

Usage

```
## S3 method for class 'summary.estimate'
plot(x, color = "black", size = 2, width = 0, ...)
```

Arguments

`x`	An object of class `summary.estimate`
`color`	Character string. The color for the error bars. (defaults to `"black"`).

`size`	Numeric. The size for the points (defaults to 2).
`width`	Numeric. The width of error bar ends (defaults to `0`).
`...`	Currently ignored

Value

A `ggplot` object.

See Also

`estimate`

Examples

```
# data
Y <- ptsd[,1:5]

fit <- estimate(Y, iter = 250,
                progress = FALSE)


plot(summary(fit))
```

`plot.summary.explore` *Plot* `summary.explore` *Objects*

Description

Visualize the posterior distributions for each partial correlation.

Usage

```
## S3 method for class 'summary.explore'
plot(x, color = "black", size = 2, width = 0, ...)
```

Arguments

`x`	An object of class `summary.explore`
`color`	Character string. The color for the error bars. (defaults to `"black"`).
`size`	Numeric. The size for the points (defaults to 2).
`width`	Numeric. The width of error bar ends (defaults to `0`).
`...`	Currently ignored

Value

A `ggplot` object

See Also

`explore`

Examples

```
# note: iter = 250 for demonstrative purposes

Y <- ptsd[,1:5]

fit <- explore(Y, iter = 250,
               progress = FALSE)

plt <- plot(summary(fit))

plt
```

`plot.summary.ggm_compare_estimate`
Plot `summary.ggm_compare_estimate` *Objects*

Description

Visualize the posterior distribution differences.

Usage

```
## S3 method for class 'summary.ggm_compare_estimate'
plot(x, color = "black", size = 2, width = 0, ...)
```

Arguments

`x`	An object of class `ggm_compare_estimate`.
`color`	Character string. The color of the points (defaults to `"black"`).
`size`	Numeric. The size of the points (defaults to 2).
`width`	Numeric. The width of error bar ends (defaults to `0`).
`...`	Currently ignored.

Value

An object of class `ggplot`

See Also

ggm_compare_estimate

Examples

```
# note: iter = 250 for demonstrative purposes
# data
Y <- bfi

# males and females
Ymale <- subset(Y, gender == 1,
                select = -c(gender,
                            education))[,1:5]

Yfemale <- subset(Y, gender == 2,
                  select = -c(gender,
                              education))[,1:5]

# fit model
fit <- ggm_compare_estimate(Ymale,  Yfemale,
                            type = "ordinal",
                            iter = 250,
                            prior_sd = 0.25,
                            progress = FALSE)

plot(summary(fit))
```

`plot.summary.ggm_compare_explore`
Plot `summary.ggm_compare_explore` *Objects*

Description

Visualize the posterior hypothesis probabilities.

Usage

```
## S3 method for class 'summary.ggm_compare_explore'
plot(x, size = 2, color = "black", ...)
```

Arguments

`x`	An object of class `summary.ggm_compare_explore`
`size`	Numeric. The size of the points (defaults to 2).
`color`	Character string. The color of the points (defaults to `"black"`).
`...`	Currently ignored.

Value

A `ggplot` object

See Also

`ggm_compare_explore`

Examples

```
# note: iter = 250 for demonstrative purposes

# data
Y <- bfi

# males and females
Ymale <- subset(Y, gender == 1,
                   select = -c(gender,
                               education))[,1:10]

Yfemale <- subset(Y, gender == 2,
                     select = -c(gender,
                                 education))[,1:10]

##########################
### example 1: ordinal ###
##########################

# fit model
fit <- ggm_compare_explore(Ymale,  Yfemale,
                           type = "ordinal",
                           iter = 250,
                           progress = FALSE)
# summary
summ <- summary(fit)

plot(summ)
```

`plot.summary.select.explore`

Plot `summary.select.explore` *Objects*

Description

Visualize the posterior hypothesis probabilities.

Usage

```
## S3 method for class 'summary.select.explore'
plot(x, size = 2, color = "black", ...)
```

Arguments

`x`	An object of class `summary.select.explore`
`size`	Numeric. The size for the points (defaults to 2).
`color`	Character string. The Color for the points
`...`	Currently ignored

Value

A `ggplot` object

Examples

```
#  data
Y <- bfi[,1:10]

# fit model
fit <- explore(Y, iter = 250,
               progress = FALSE)

# edge set
E <- select(fit,
            alternative = "exhaustive")

plot(summary(E))
```

`plot.summary.var_estimate` *Plot* `summary.var_estimate` *Objects*

Description

Visualize the posterior distributions of each partial correlation and regression coefficient.

Usage

```
## S3 method for class 'summary.var_estimate'
plot(x, color = "black", size = 2, width = 0, param = "all", order = TRUE, ...)
```

Arguments

`x`	An object of class `summary.var_estimate`
`color`	Character string. The color for the error bars. (defaults to `"black"`).
`size`	Numeric. The size for the points (defaults to `2`).
`width`	Numeric. The width of error bar ends (defaults to `0`).
`param`	Character string. Which parameters should be plotted ? The options are `pcor`, `beta`, or `all` (default).
`order`	Logical. Should the relations be ordered by size (defaults to `TRUE`) ?
`...`	Currently ignored

Value

A list of `ggplot` objects.

Examples

```
# data
Y <- subset(ifit, id == 1)[,-1]

# fit model with alias (var_estimate also works)
fit <- var_estimate(Y, progress = FALSE)

plts <- plot(summary(fit))
plts$pcor_plt
```

`plot_prior` *Plot: Prior Distribution*

Description

Visualize the implied prior distribution for the partial correlations. This is particularly useful for the Bayesian hypothesis testing methods.

Usage

```
plot_prior(prior_sd = 0.2, iter = 5000)
```

Arguments

`prior_sd`	Scale of the prior distribution, approximately the standard deviation of a beta distribution (defaults to 0.25).
`iter`	Number of iterations (prior samples; defaults to 5000).

Value

A `ggplot` object.

Examples

```
# note: iter = 250 for demonstrative purposes

plot_prior(prior_sd = 0.25, iter = 250)
```

`posterior_predict` *Posterior Predictive Distribution*

Description

Draw samples from the posterior predictive distribution.

Usage

```
posterior_predict(object, iter = 1000, progress = TRUE)
```

Arguments

`object`	An object of class `estimate` or `explore`
`iter`	Numeric. Number of samples from the predictive distribution
`progress`	Logical. Should a progress bar be included (defaults to `TRUE`)

Value

A 3D array containing the predicted datasets

Note

Currently only implemented for `type = "mixed"`, `type = "ordinal"`, and `type = "binary"`. Note the term mixed is confusing, in that it can be used with only, say, ordinal data. In this case, reestimate the model with `type = "mixed"` until all data types are supported.

Examples

```
Y <- gss

fit <- estimate(as.matrix(Y),
                impute = TRUE,
               iter = 150, type = "mixed")

yrep <- posterior_predict(fit, iter = 100)
```

`posterior_samples` *Extract Posterior Samples*

Description

Extract posterior samples for all parameters.

Usage

```
posterior_samples(object, ...)
```

Arguments

`object`	an object of class `estimate` or `explore`.
`...`	currently ignored.

Value

A matrix of posterior samples for the partial correlation. Note that if controlling for variables (e.g., formula ~ age), the matrix also includes the coefficients from each multivariate regression.

Examples

```
# note: iter = 250 for demonstrative purposes

########################################
### example 1: control  with formula ###
########################################
# (the following works with all data types)

# controlling for gender
Y <- bfi

# to control for only gender
# (remove education)
Y <- subset(Y, select = - education)

# fit model
fit <- estimate(Y, formula = ~ gender,
                iter = 250)

# note regression coefficients
samps <- posterior_samples(fit)

hist(samps[,1])
```

`precision` *Precision Matrix Posterior Distribution*

Description

Transform the sampled correlation matrices to precision matrices (i.e., inverse covariance matrices).

Usage

```
precision(object, progress = TRUE)
```

Arguments

`object`	An object of class `estimate`.
`progress`	Logical. Should a progress bar be included (defaults to `TRUE`) ?

Value

- `precision_mean` The mean of the precision matrix (`p` by `p` matrix).
- `precision` 3d array of dimensions `p` by `p` by `iter` including **unconstrained** (i.e., from th full graph) precision matrices.

Note

The estimated precision matrix is the inverse of the **correlation** matrix.

Examples

```
# data
Y <- ptsd

# fit model
fit <- estimate(Y)

# precision matrix
Theta <- precision(fit)
```

`predict.estimate` *Model Predictions for* `estimate` *Objects*

Description

Model Predictions for `estimate` Objects

Usage

```
## S3 method for class 'estimate'
predict(
  object,
  newdata = NULL,
  summary = TRUE,
  cred = 0.95,
  iter = NULL,
  progress = TRUE,
  ...
)
```

Arguments

`object`	object of class `estimate`
`newdata`	an optional data frame for obtaining predictions (e.g., on test data)
`summary`	summarize the posterior samples (defaults to `TRUE`).
`cred`	credible interval used for summarizing
`iter`	number of posterior samples (defaults to all in the object).
`progress`	Logical. Should a progress bar be included (defaults to `TRUE`) ?
`...`	currently ignored

Value

`summary` = `TRUE`: 3D array of dimensions n (observations), 4 (posterior summary), p (number of nodes). `summary` = `FALSE`: list containing predictions for each variable

Examples

```

# # data
Y <- ptsd

fit <- estimate(Y, iter = 250,
                progress = FALSE)

pred <- predict(fit,
                progress = FALSE)
```

`predict.explore` *Model Predictions for* `explore` *Objects*

Description

Model Predictions for `explore` Objects

Usage

```
## S3 method for class 'explore'
predict(
  object,
  newdata = NULL,
  summary = TRUE,
  cred = 0.95,
  iter = NULL,
  progress = TRUE,
  ...
)
```

Arguments

`object`	object of class `explore`
`newdata`	an optional data frame for obtaining predictions (e.g., on test data)
`summary`	summarize the posterior samples (defaults to `TRUE`).
`cred`	credible interval used for summarizing
`iter`	number of posterior samples (defaults to all in the object).
`progress`	Logical. Should a progress bar be included (defaults to `TRUE`) ?
`...`	currently ignored

Value

`summary = TRUE`: 3D array of dimensions n (observations), 4 (posterior summary), p (number of nodes). `summary = FALSE`: list containing predictions for each variable

Examples

```

# data
Y <- ptsd

# fit model
fit <- explore(Y, iter = 250,
               progress = FALSE)

# predict
```

```
pred <- predict(fit,
                progress = FALSE)
```

`predict.var_estimate` *Model Predictions for* `var_estimate` *Objects*

Description

Model Predictions for `var_estimate` Objects

Usage

```
## S3 method for class 'var_estimate'
predict(object, summary = TRUE, cred = 0.95, iter = NULL, progress = TRUE, ...)
```

Arguments

`object`	object of class `var_estimate`
`summary`	summarize the posterior samples (defaults to `TRUE`).
`cred`	credible interval used for summarizing
`iter`	number of posterior samples (defaults to all in the object).
`progress`	Logical. Should a progress bar be included (defaults to `TRUE`) ?
`...`	Currently ignored

Value

The predicted values for each regression model.

Examples

```
# data
Y <- subset(ifit, id == 1)[,-1]

# fit model with alias (var_estimate also works)
fit <- var_estimate(Y, progress = FALSE)

# fitted values
pred <- predict(fit, progress = FALSE)

# predicted values (1st outcome)
pred[,,1]
```

`predictability` *Predictability: Bayesian Variance Explained (R2)*

Description

Compute nodewise predictability or Bayesian variance explained (R2 Gelman et al. 2019). In the context of GGMs, this method was described in Williams (2018).

Usage

```
predictability(
  object,
  select = FALSE,
  cred = 0.95,
  BF_cut = 3,
  iter = NULL,
  progress = TRUE,
  ...
)
```

Arguments

`object`	object of class `estimate` or `explore`
`select`	logical. Should the graph be selected ? The default is currently `FALSE`.
`cred`	numeric. credible interval between 0 and 1 (default is 0.95) that is used for selecting the graph.
`BF_cut`	numeric. evidentiary threshold (default is 3).
`iter`	interger. iterations (posterior samples) used for computing R2.
`progress`	Logical. Should a progress bar be included (defaults to `TRUE`) ?
`...`	currently ignored.

Value

An object of classes `bayes_R2` and `metric`, including

- `scores` A list containing the posterior samples of R2. The is one element for each node.

Note

Binary and Ordinal Data:

R2 is computed from the latent data.

Mixed Data:

The mixed data approach is somewhat ad-hoc see for example p. 277 in Hoff (2007). This is becaue uncertainty in the ranks is not incorporated, which means that variance explained is computed from the 'empirical' *CDF*.

Model Selection:

Currently the default to include all nodes in the model when computing R2. This can be changed (i.e., `select = TRUE`), which then sets those edges not detected to zero. This is accomplished by subsetting the correlation matrix according to each neighborhood of relations.

References

Gelman A, Goodrich B, Gabry J, Vehtari A (2019). "R-squared for Bayesian Regression Models." *American Statistician*, **73**(3), 307–309. ISSN 15372731, doi:10.1080/00031305.2018.1549100.

Hoff PD (2007). "Extending the rank likelihood for semiparametric copula estimation." *The Annals of Applied Statistics*, **1**(1), 265–283. doi:10.1214/07AOAS107.

Williams DR (2018). "Bayesian Estimation for Gaussian Graphical Models: Structure Learning, Predictability, and Network Comparisons." *arXiv*. doi:10.31234/OSF.IO/X8DPR.

Examples

```
# data
Y <- ptsd[,1:5]

fit <- estimate(Y, iter = 250, progress = FALSE)

r2 <- predictability(fit, select = TRUE,
                     iter = 250, progress = FALSE)

# summary
r2
```

`predicted_probability` *Predicted Probabilities*

Description

Compute the predicted probabilities for discrete data, with the possibility of conditional predictive probabilities (i.e., at fixed values of other nodes)

Usage

```
predicted_probability(object, outcome, Y, ...)
```

Arguments

`object`	An object of class `posterior_predict`
`outcome`	Character string. Node for which the probabilities are computed.
`Y`	Matrix (or data frame) of dimensions *n* (observations) by *p* (variables). This must include the column names.
`...`	Compute conditional probabilities by specifying a column name in `Y` (besides the `outcome`) and a fixed value. This can include any number of nodes. See example below. Leave this blank to compute unconditional probabilities for `outcome`.

Value

A list containing a matrix with the computed probabilities (a row for each predictive sample and a column for each category).

Note

There are no checks that the conditional probability exists, i.e., suppose you wish to condition on, say, B3 = 2 and B4 = 1, yet there is no instance in which B3 is 2 AND B4 is 1. This will result in an uninformative error.

Examples

```
Y <- ptsd
fit <- estimate(as.matrix(Y), iter = 150, type = "mixed")

pred <- posterior_predict(fit, iter = 100)

prob <- predicted_probability(pred,
                              Y = Y,
                              outcome = "B3",
                              B4 = 0,
                              B5 = 0)
```

`print.BGGM` *Print method for* `BGGM` *objects*

Description

Mainly used to avoid a plethora of different print functions that overcrowded the documentation in previous versions of **BGGM**.

Usage

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

Arguments

`x`	An object of class `BGGM`
`...`	currently ignored

`prior_belief_ggm` *Prior Belief Gaussian Graphical Model*

Description

Incorporate prior information into the estimation of the conditional dependence structure. This prior information is expressed as the prior odds that each relation should be included in the graph.

Usage

```
prior_belief_ggm(Y, prior_ggm, post_odds_cut = 3, ...)
```

Arguments

`Y`	Matrix (or data frame) of dimensions n (observations) by p (variables/nodes).
`prior_ggm`	Matrix of dimensions p by p, encoding the prior odds for including each relation in the graph (see '`Details`')
`post_odds_cut`	Numeric. Threshold for including an edge (defaults to 3). Note `post_odds` refers to posterior odds.
`...`	Additional arguments passed to `explore`.

Details

Technically, the prior odds is not for including an edge in the graph, but for (H1)/p(H0), where H1 captures the hypothesized edge size and H0 is the null model (see Williams2019_bf). Accordingly, setting an entry in `prior_ggm` to, say, 10, encodes a prior belief that H1 is 10 times more likely than H0. Further, setting an entry in `prior_ggm` to 1 results in equal prior odds (the default in `select.explore`).

Value

An object including:

- **adj**: Adjacency matrix
- **post_prob**: Posterior probability for the alternative hypothesis.

Examples

```
# Assume perfect prior information
# synthetic ggm
p <- 20
main <- gen_net()

# prior odds 10:1, assuming graph is known
prior_ggm <- ifelse(main$adj == 1, 10, 1)

# generate data
y <- MASS::mvrnorm(n = 200,
                   mu = rep(0, 20),
                   Sigma = main$cors)

# prior est
prior_est <- prior_belief_ggm(Y = y,
                              prior_ggm = prior_ggm,
                              progress = FALSE)

# check scores
BGGM:::performance(Estimate = prior_est$adj,
                   True = main$adj)

# default in BGGM
default_est <- select(explore(y, progress = FALSE))

# check scores
BGGM:::performance(Estimate = default_est$Adj_10,
                   True = main$adj)
```

`prior_belief_var` *Prior Belief Graphical VAR*

Description

Prior Belief Graphical VAR

Usage

```
prior_belief_var(
  Y,
  prior_temporal = NULL,
  post_odds_cut = 3,
  est_ggm = TRUE,
  prior_ggm = NULL,
  progress = TRUE,
```

```
  ...
)
```

Arguments

`Y`	Matrix (or data frame) of dimensions *n* (observations) by *p* (variables/nodes).
`prior_temporal`	Matrix of dimensions *p* by *p*, encoding the prior odds for including each relation in the temporal graph (see '`Details`'). If null a matrix of 1's is used, resulting in equal prior odds.
`post_odds_cut`	Numeric. Threshold for including an edge (defaults to 3). Note `post_odds` refers to posterior odds.
`est_ggm`	Logical. Should the contemporaneous network be estimated (defaults to `TRUE`)?
`prior_ggm`	Matrix of dimensions *p* by *p*, encoding the prior odds for including each relation in the graph (see '`Details`'). If null a matrix of 1's is used, resulting in equal prior odds.
`progress`	Logical. Should a progress bar be included (defaults to `TRUE`) ?
`...`	Additional arguments passed to `explore`. Ignored if `prior_ggm = FALSE`.

Details

Technically, the prior odds is not for including an edge in the graph, but for (H1)/p(H0), where H1 captures the hypothesized edge size and H0 is the null model (see Williams2019_bf). Accordingly, setting an entry in `prior_ggm` to, say, 10, encodes a prior belief that H1 is 10 times more likely than H0. Further, setting an entry in `prior_ggm` or `prior_var` to 1 results in equal prior odds (the default in `select.explore`).

Value

An object including (`est_ggm = FALSE`):

- **adj**: Adjacency matrix
- **post_prob**: Posterior probability for the alternative hypothesis.

An object including (`est_ggm = TRUE`):

- **adj_temporal**: Adjacency matrix for the temporal network.
- **post_prob_temporal**: Posterior probability for the alternative hypothesis (temporal edge)
- **adj_ggm**: Adjacency matrix for the contemporaneous network (ggm).
- post_prob_ggm: Posterior probability for the alternative hypothesis (contemporaneous edge)

Note

The returned matrices are formatted with the rows indicating the outcome and the columns the predictor. Hence, adj_temporal[1,4] is the temporal relation of node 4 predicting node 1. This follows the convention of the **vars** package (i.e., `Acoef`).

Further, in order to compute the Bayes factor the data is standardized (mean = 0 and standard deviation = 1).

Examples

```
# affect data from 1 person
# (real data)
y <- na.omit(subset(ifit, id == 1)[,2:7])
p <- ncol(y)

# random prior graph
# (dont do this in practice!!)
prior_var = matrix(sample(c(1,10),
                   size = p^2, replace = TRUE),
                   nrow = p, ncol = p)

# fit model
fit <- prior_belief_var(y,
                        prior_temporal = prior_var,
                        post_odds_cut = 3)
```

`ptsd` *Data: Post-Traumatic Stress Disorder*

Description

A dataset containing items that measure Post-traumatic stress disorder symptoms (Armour et al. 2017). There are 20 variables (*p*) and 221 observations (*n*).

Usage

`data("ptsd")`

Format

A dataframe with 221 rows and 20 variables

Details

- Intrusive Thoughts
- Nightmares
- Flashbacks
- Emotional cue reactivity
- Psychological cue reactivity
- Avoidance of thoughts
- Avoidance of reminders
- Trauma-related amnesia
- Negative beliefs

- Negative trauma-related emotions
- Loss of interest
- Detachment
- Restricted affect
- Irritability/anger
- Self-destructive/reckless behavior
- Hypervigilance
- Exaggerated startle response
- Difficulty concentrating
- Sleep disturbance

References

Armour C, Fried EI, Deserno MK, Tsai J, Pietrzak RH (2017). “A network analysis of DSM-5 posttraumatic stress disorder symptoms and correlates in US military veterans.” *Journal of anxiety disorders*, **45**, 49–59. doi:10.31234/osf.io/p69m7.

`ptsd_cor1` *Data: Post-Traumatic Stress Disorder (Sample # 1)*

Description

A correlation matrix that includes 16 variables. The correlation matrix was estimated from 526 individuals (Fried et al. 2018).

Format

A correlation matrix with 16 variables

Details

- Intrusive Thoughts
- Nightmares
- Flashbacks
- Physiological/psychological reactivity
- Avoidance of thoughts
- Avoidance of situations
- Amnesia
- Disinterest in activities
- Feeling detached
- Emotional numbing
- Foreshortened future

- Sleep problems
- Irritability
- Concentration problems
- Hypervigilance
- Startle response

References

Fried EI, Eidhof MB, Palic S, Costantini G, Huisman-van Dijk HM, Bockting CL, Engelhard I, Armour C, Nielsen AB, Karstoft K (2018). "Replicability and generalizability of posttraumatic stress disorder (PTSD) networks: a cross-cultural multisite study of PTSD symptoms in four trauma patient samples." *Clinical Psychological Science*, **6**(3), 335–351.

Examples

```
data(ptsd_cor1)

Y <- MASS::mvrnorm(n = 526,
                   mu = rep(0, 16),
                   Sigma = ptsd_cor1,
                   empirical = TRUE)
```

`ptsd_cor2` *Data: Post-Traumatic Stress Disorder (Sample # 2)*

Description

A correlation matrix that includes 16 variables. The correlation matrix was estimated from 365 individuals (Fried et al. 2018).

Format

A correlation matrix with 16 variables

Details

- Intrusive Thoughts
- Nightmares
- Flashbacks
- Physiological/psychological reactivity
- Avoidance of thoughts
- Avoidance of situations
- Amnesia

- Disinterest in activities
- Feeling detached
- Emotional numbing
- Foreshortened future
- Sleep problems
- Irritability
- Concentration problems
- Hypervigilance
- Startle response

References

Fried EI, Eidhof MB, Palic S, Costantini G, Huisman-van Dijk HM, Bockting CL, Engelhard I, Armour C, Nielsen AB, Karstoft K (2018). “Replicability and generalizability of posttraumatic stress disorder (PTSD) networks: a cross-cultural multisite study of PTSD symptoms in four trauma patient samples.” *Clinical Psychological Science*, **6**(3), 335–351.

Examples

```
data(ptsd_cor2)
Y <- MASS::mvrnorm(n = 365,
                   mu = rep(0, 16),
                   Sigma = ptsd_cor2,
                   empirical = TRUE)
```

`ptsd_cor3` *Data: Post-Traumatic Stress Disorder (Sample # 3)*

Description

A correlation matrix that includes 16 variables. The correlation matrix was estimated from 926 individuals (Fried et al. 2018).

Format

A correlation matrix with 16 variables

Details

- Intrusive Thoughts
- Nightmares
- Flashbacks
- Physiological/psychological reactivity
- Avoidance of thoughts

- Avoidance of situations
- Amnesia
- Disinterest in activities
- Feeling detached
- Emotional numbing
- Foreshortened future
- Sleep problems
- Irritability
- Concentration problems
- Hypervigilance
- Startle response

References

Fried EI, Eidhof MB, Palic S, Costantini G, Huisman-van Dijk HM, Bockting CL, Engelhard I, Armour C, Nielsen AB, Karstoft K (2018). "Replicability and generalizability of posttraumatic stress disorder (PTSD) networks: a cross-cultural multisite study of PTSD symptoms in four trauma patient samples." *Clinical Psychological Science*, **6**(3), 335–351.

Examples

```
data(ptsd_cor3)
Y <- MASS::mvrnorm(n = 926,
                   mu = rep(0, 16),
                   Sigma = ptsd_cor3,
                   empirical = TRUE)
```

`ptsd_cor4` *Data: Post-Traumatic Stress Disorder (Sample # 4)*

Description

A correlation matrix that includes 16 variables. The correlation matrix was estimated from 965 individuals (Fried et al. 2018).

Format

A correlation matrix with 16 variables

Details

- Intrusive Thoughts
- Nightmares
- Flashbacks
- Physiological/psychological reactivity
- Avoidance of thoughts
- Avoidance of situations
- Amnesia
- Disinterest in activities
- Feeling detached
- Emotional numbing
- Foreshortened future
- Sleep problems
- Irritability
- Concentration problems
- Hypervigilance
- Startle response

References

Fried EI, Eidhof MB, Palic S, Costantini G, Huisman-van Dijk HM, Bockting CL, Engelhard I, Armour C, Nielsen AB, Karstoft K (2018). "Replicability and generalizability of posttraumatic stress disorder (PTSD) networks: a cross-cultural multisite study of PTSD symptoms in four trauma patient samples." *Clinical Psychological Science*, **6**(3), 335–351.

Examples

```
data(ptsd_cor4)
Y <- MASS::mvrnorm(n = 965,
                   mu = rep(0, 16),
                   Sigma = ptsd_cor4,
                   empirical = TRUE)
```

`regression_summary` *Summarary Method for Multivariate or Univarate Regression*

Description

Summarary Method for Multivariate or Univarate Regression

Usage

```
regression_summary(object, cred = 0.95, ...)
```

Arguments

`object`	An object of class `estimate`
`cred`	Numeric. The credible interval width for summarizing the posterior distributions (defaults to 0.95; must be between 0 and 1).
`...`	Currently ignored

Value

A list of length *p* including the summaries for each regression.

Examples

```
# note: iter = 250 for demonstrative purposes

# data
Y <- bfi

Y <- subset(Y, select = c("E5", "N5",
                          "gender", "education"))


fit_mv_ordinal <- estimate(Y, formula = ~ gender + as.factor(education),
                           type = "ordinal",
                           iter = 250,
                           progress = FALSE)

regression_summary(fit_mv_ordinal)
```

`roll_your_own` *Compute Custom Network Statistics*

Description

This function allows for computing custom network statistics for weighted adjacency matrices (partial correlations). The statistics are computed for each of the sampled matrices, resulting in a distribution.

Usage

```
roll_your_own(
  object,
  FUN,
  iter = NULL,
  select = FALSE,
  cred = 0.95,
  progress = TRUE,
```

```
  ...
)
```

Arguments

`object`	An object of class `estimate`.
`FUN`	A custom function for computing the statistic. The first argument must be a partial correlation matrix.
`iter`	Number of iterations (posterior samples; defaults to the number in the object).
`select`	Logical. Should the graph be selected ? The default is currently `FALSE`.
`cred`	Numeric. Credible interval between 0 and 1 (default is 0.95) that is used for selecting the graph.
`progress`	Logical. Should a progress bar be included (defaults to `TRUE`) ?
`...`	Arguments passed to the function.

Details

The user has complete control of this function. Hence, care must be taken as to what `FUN` returns and in what format. The function should return a single number (one for the entire GGM) or a vector (one for each node). This ensures that the print and `plot.roll_your_own` will work.

When `select = TRUE`, the graph is selected and then the network statistics are computed based on the weigthed adjacency matrix. This is accomplished internally by multiplying each of the sampled partial correlation matrices by the adjacency matrix.

Value

An object defined by `FUN`.

Examples

```
####################################
###### example 1: assortment #######
####################################
# assortment
library(assortnet)

Y <- BGGM::bfi[,1:10]
membership <- c(rep("a", 5), rep("c", 5))

# fit model
fit <- estimate(Y = Y, iter = 250,
                progress = FALSE)

# membership
membership <- c(rep("a", 5), rep("c", 5))

# define function
f <- function(x,...){
```

```
 assortment.discrete(x, ...)$r
}


net_stat <- roll_your_own(object = fit,
                          FUN = f,
                          types = membership,
                          weighted = TRUE,
                          SE = FALSE, M = 1,
                          progress = FALSE)

# print
net_stat


############################################
###### example 2: expected influence #######
############################################
# expected influence from this package
library(networktools)

# data
Y <- depression

# fit model
fit <- estimate(Y = Y, iter = 250)

# define function
f <- function(x,...){
     expectedInf(x,...)$step1
}

# compute
net_stat <- roll_your_own(object = fit,
                          FUN = f,
                          progress = FALSE)

#######################################
### example 3: mixed data & bridge ####
#######################################
# bridge from this package
library(networktools)

# data
Y <- ptsd[,1:7]

fit <- estimate(Y,
                type = "mixed",
                iter = 250)

# clusters
communities <- substring(colnames(Y), 1, 1)
```

```
# function is slow
f <- function(x, ...){
 bridge(x, ...)$`Bridge Strength`
}

net_stat <- roll_your_own(fit,
                          FUN = f,
                          select = TRUE,
                          communities = communities,
                          progress = FALSE)
```

`rsa` *Data: Resilience Scale of Adults (RSA)*

Description

A dataset containing items from the Resilience Scale of Adults (RSA). There are 33 items and 675 observations

Usage

```
data("rsa")
```

Format

A data frame with 28 variables and 1973 observations (5 point Likert scale)

Details

- `1` My plans for the future are
- `2` When something unforeseen happens
- `3` My family understanding of what is important in life is
- `4` I feel that my future looks
- `5` My goals
- `6` I can discuss personal issues with
- `7` I feel
- `8` I enjoy being
- `9` Those who are good at encouraging are
- `10` The bonds among my friends
- `11` My personal problems
- `12` When a family member experiences a crisis/emergency
- `13` My family is characterised by

- `14` To be flexible in social settings
- `15` I get support from
- `16` In difficult periods my family
- `17` My judgements and decisions
- `18` New friendships are something
- `19` When needed, I have
- `20` I am at my best when I
- `21` Meeting new people is
- `22` When I am with others
- `23` When I start on new things/projects
- `24` Facing other people, our family acts
- `25` Belief in myself
- `26` For me, thinking of good topics of conversation is
- `27` My close friends/family members
- `28` I am good at
- `29` In my family, we like to
- `30` Rules and regular routines
- `31` In difficult periods I have a tendency to
- `32` My goals for the future are
- `33` Events in my life that I cannot influence
- `gender` "M" (male) or "F" (female)

Note

There are 6 domains

Planned future: items 1, 4, 5, 32

Perception of self: items 2, 11, 17, 25, 31, 33

Family cohesion: items 3, 7, 13, 16, 24, 29

Social resources: items 6, 9, 10, 12, 15, 19, 27

Social Competence: items 8, 14, 18, 21, 22, 26,

Structured style: items 23, 28, 30

References

Briganti, G., & Linkowski, P. (2019). Item and domain network structures of the Resilience Scale for Adults in 675 university students. Epidemiology and psychiatric sciences, 1-9.

Examples

```
data("rsa")
```

`Sachs` *Data: Sachs Network*

Description

Protein expression in human immune system cells

Usage

```
data("Sachs")
```

Format

A data frame containing 7466 cells (n = 7466) and flow cytometry measurements of 11 (p = 11) phosphorylated proteins and phospholipids

@references Sachs, K., Gifford, D., Jaakkola, T., Sorger, P., & Lauffenburger, D. A. (2002). Bayesian network approach to cell signaling pathway modeling. Sci. STKE, 2002(148), pe38-pe38.

Examples

```
data("Sachs")
```

`select` *S3* `select` *method*

Description

S3 select method

Usage

```
select(object, ...)
```

Arguments

`object`	object of class `estimate` or`explore`
`...`	not currently used

Value

`select` works with the following methods:

- `select.estimate`
- `select.explore`
- `select.ggm_compare_estimate`

`select.estimate` *Graph Selection for* `estimate` *Objects*

Description

Provides the selected graph based on credible intervals for the partial correlations that did not contain zero (Williams 2018).

Usage

```
## S3 method for class 'estimate'
select(object, cred = 0.95, alternative = "two.sided", ...)
```

Arguments

`object`	An object of class `estimate.default`.
`cred`	Numeric. The credible interval width for selecting the graph (defaults to 0.95; must be between 0 and 1).
`alternative`	A character string specifying the alternative hypothesis. It must be one of "two.sided" (default), "greater" or "less". See note for futher details.
`...`	Currently ignored.

Details

This package was built for the social-behavioral sciences in particular. In these applications, there is strong theory that expects *all* effects to be positive. This is known as a "positive manifold" and this notion has a rich tradition in psychometrics. Hence, this can be incorporated into the graph with `alternative = "greater"`. This results in the estimated structure including only positive edges.

Value

The returned object of class `select.estimate` contains a lot of information that is used for printing and plotting the results. For users of **BGGM**, the following are the useful objects:

- `pcor_adj` Selected partial correlation matrix (weighted adjacency).
- `adj` Adjacency matrix for the selected edges
- `object` An object of class `estimate` (the fitted model).

References

Williams DR (2018). “Bayesian Estimation for Gaussian Graphical Models: Structure Learning, Predictability, and Network Comparisons.” *arXiv*. doi:10.31234/OSF.IO/X8DPR.

See Also

`estimate` and `ggm_compare_estimate` for several examples.

Examples

```
# note: iter = 250 for demonstrative purposes

# data
Y <- bfi[,1:10]

# estimate
fit <- estimate(Y, iter = 250,
                progress = FALSE)


# select edge set
E <- select(fit)
```

`select.explore` *Graph selection for* `explore` *Objects*

Description

Provides the selected graph based on the Bayes factor (Williams and Mulder 2019).

Usage

```
## S3 method for class 'explore'
select(object, BF_cut = 3, alternative = "two.sided", ...)
```

Arguments

`object`	An object of class `explore.default`
`BF_cut`	Numeric. Threshold for including an edge (defaults to 3).
`alternative`	A character string specifying the alternative hypothesis. It must be one of "two.sided" (default), "greater", "less", or "exhaustive". See note for further details.
`...`	Currently ignored.

Details

Exhaustive provides the posterior hypothesis probabilities for a positive, negative, or null relation (see Table 3 in Williams and Mulder 2019).

Value

The returned object of class `select.explore` contains a lot of information that is used for printing and plotting the results. For users of **BGGM**, the following are the useful objects:

`alternative = "two.sided"`

- `pcor_mat_zero` Selected partial correlation matrix (weighted adjacency).
- `pcor_mat` Partial correlation matrix (posterior mean).
- `Adj_10` Adjacency matrix for the selected edges.
- `Adj_01` Adjacency matrix for which there was evidence for the null hypothesis.

`alternative = "greater"` and `"less"`

- `pcor_mat_zero` Selected partial correlation matrix (weighted adjacency).
- `pcor_mat` Partial correlation matrix (posterior mean).
- `Adj_20` Adjacency matrix for the selected edges.
- `Adj_02` Adjacency matrix for which there was evidence for the null hypothesis (see note).

`alternative = "exhaustive"`

- `post_prob` A data frame that included the posterior hypothesis probabilities.
- `neg_mat` Adjacency matrix for which there was evidence for negative edges.
- `pos_mat` Adjacency matrix for which there was evidence for positive edges.
- `neg_mat` Adjacency matrix for which there was evidence for the null hypothesis (see note).
- `pcor_mat` Partial correlation matrix (posterior mean). The weighted adjacency matrices can be computed by multiplying `pcor_mat` with an adjacency matrix.

Note

Care must be taken with the options `alternative = "less"` and `alternative = "greater"`. This is because the full parameter space is not included, such, for `alternative = "greater"`, there can be evidence for the "null" when the relation is negative. This inference is correct: the null model better predicted the data than the positive model. But note this is relative and does **not** provide absolute evidence for the null hypothesis.

References

Williams DR, Mulder J (2019). “Bayesian Hypothesis Testing for Gaussian Graphical Models: Conditional Independence and Order Constraints.” *PsyArXiv*. doi:10.31234/osf.io/ypxd8.

See Also

`explore` and `ggm_compare_explore` for several examples.

Examples

```
#################
### example 1 ###
#################

#  data
Y <- bfi[,1:10]

# fit model
fit <- explore(Y, progress = FALSE)

# edge set
E <- select(fit,
            alternative = "exhaustive")
```

`select.ggm_compare_estimate`

Graph Selection for `ggm_compare_estimate` *Objects*

Description

Provides the selected graph (of differences) based on credible intervals for the partial correlations that did not contain zero (Williams 2018).

Usage

```
## S3 method for class 'ggm_compare_estimate'
select(object, cred = 0.95, ...)
```

Arguments

`object`	An object of class `estimate.default`.
`cred`	Numeric. The credible interval width for selecting the graph (defaults to 0.95; must be between 0 and 1).
`...`	not currently used

Value

The returned object of class `select.ggm_compare_estimate` contains a lot of information that is used for printing and plotting the results. For users of **BGGM**, the following are the useful objects:

- `mean_diff` A list of matrices for each group comparsion (partial correlation differences).
- `pcor_adj` A list of weighted adjacency matrices for each group comparsion.
- `adj` A list of adjacency matrices for each group comparsion.

Examples

```
# note: iter = 250 for demonstrative purposes
##################
### example 1: ###
##################
# data
Y <- bfi

# males and females
Ymale <- subset(Y, gender == 1,
               select = -c(gender,
                           education))

Yfemale <- subset(Y, gender == 2,
                  select = -c(gender,
                              education))

# fit model
fit <- ggm_compare_estimate(Ymale, Yfemale,
                            type = "continuous",
                            iter = 250,
                            progress = FALSE)


E <- select(fit)
```

`select.ggm_compare_explore` *Graph selection for* `ggm_compare_explore` *Objects*

Description

Provides the selected graph (of differences) based on the Bayes factor (Williams et al. 2020).

Usage

```
## S3 method for class 'ggm_compare_explore'
select(object, BF_cut = 3, ...)
```

Arguments

`object`	An object of class `ggm_compare_explore`.
`BF_cut`	Numeric. Threshold for including an edge (defaults to 3).
`...`	Currently ignored.

Value

The returned object of class `select.ggm_compare_explore` contains a lot of information that is used for printing and plotting the results. For users of **BGGM**, the following are the useful objects:

- `adj_10` Adjacency matrix for which there was evidence for a difference.
- `adj_10` Adjacency matrix for which there was evidence for a null relation
- `pcor_mat_10` Selected partial correlation matrix (weighted adjacency; only for two groups).

See Also

`explore` and `ggm_compare_explore` for several examples.

Examples

```
##################
### example 1: ###
##################
# data
Y <- bfi

# males and females
Ymale <- subset(Y, gender == 1,
                   select = -c(gender,
                               education))[,1:10]

Yfemale <- subset(Y, gender == 2,
                     select = -c(gender,
                                 education))[,1:10]

# fit model
fit <- ggm_compare_explore(Ymale, Yfemale,
                           iter = 250,
                           type = "continuous",
                           progress = FALSE)


E <- select(fit, post_prob = 0.50)
```

`select.var_estimate` *Graph Selection for* `var.estimate` *Object*

Description

Graph Selection for `var.estimate` Object

Usage

```
## S3 method for class 'var_estimate'
select(object, cred = 0.95, alternative = "two.sided", ...)
```

Arguments

`object`	An object of class `VAR.estimate`.
`cred`	Numeric. The credible interval width for selecting the graph (defaults to 0.95; must be between 0 and 1).
`alternative`	A character string specifying the alternative hypothesis. It must be one of "two.sided" (default), "greater" or "less". See note for futher details.
`...`	Currently ignored.

Value

An object of class `select.var_estimate`, including

- pcor_adj Adjacency matrix for the partial correlations.
- beta_adj Adjacency matrix for the regression coefficients.
- pcor_weighted_adj Weighted adjacency matrix for the partial correlations.
- beta_weighted_adj Weighted adjacency matrix for the regression coefficients.
- `pcor_mu` Partial correlation matrix (posterior mean).
- `beta_mu` A matrix including the regression coefficients (posterior mean).

Examples

```
# data
Y <- subset(ifit, id == 1)[,-1]

# fit model with alias (var_estimate also works)
fit <- var_estimate(Y, progress = FALSE)

# select graphs
select(fit, cred = 0.95)
```

`summary.coef` *Summarize* `coef` *Objects*

Description

Summarize regression parameters with the posterior mean, standard deviation, and credible interval.

Usage

```
## S3 method for class 'coef'
summary(object, cred = 0.95, ...)
```

Arguments

`object`	An object of class `coef`.
`cred`	Numeric. The credible interval width for summarizing the posterior distributions (defaults to 0.95; must be between 0 and 1).
`...`	Currently ignored

Value

A list of length p including the summaries for each multiple regression.

Note

See `coef.estimate` and `coef.explore` for examples.

`summary.estimate` *Summary method for* `estimate.default` *objects*

Description

Summarize the posterior distribution of each partial correlation with the posterior mean and standard deviation.

Usage

```
## S3 method for class 'estimate'
summary(object, col_names = TRUE, cred = 0.95, ...)
```

Arguments

`object`	An object of class `estimate`
`col_names`	Logical. Should the summary include the column names (default is `TRUE`)? Setting to `FALSE` includes the column numbers (e.g., `1--2`).
`cred`	Numeric. The credible interval width for summarizing the posterior distributions (defaults to 0.95; must be between 0 and 1).
`...`	Currently ignored.

Value

A dataframe containing the summarized posterior distributions.

See Also

`estimate`

Examples

```
# data
Y <- ptsd[,1:5]

fit <- estimate(Y, iter = 250,
                progress = FALSE)

summary(fit)
```

`summary.explore` *Summary Method for* `explore.default` *Objects*

Description

Summarize the posterior distribution for each partial correlation with the posterior mean and standard deviation.

Usage

```
## S3 method for class 'explore'
summary(object, col_names = TRUE, ...)
```

Arguments

`object`	An object of class `estimate`
`col_names`	Logical. Should the summary include the column names (default is `TRUE`)? Setting to `FALSE` includes the column numbers (e.g., `1--2`).
`...`	Currently ignored

Value

A dataframe containing the summarized posterior distributions.

See Also

`select.explore`

Examples

```
# note: iter = 250 for demonstrative purposes

Y <- ptsd[,1:5]

fit <- explore(Y, iter = 250,
               progress = FALSE)

summ <- summary(fit)

summ
```

`summary.ggm_compare_estimate` *Summary method for* `ggm_compare_estimate` *objects*

Description

Summarize the posterior distribution of each partial correlation difference with the posterior mean and standard deviation.

Usage

```
## S3 method for class 'ggm_compare_estimate'
summary(object, col_names = TRUE, cred = 0.95, ...)
```

Arguments

`object`	An object of class `ggm_compare_estimate`.
`col_names`	Logical. Should the summary include the column names (default is `TRUE`)? Setting to `FALSE` includes the column numbers (e.g., `1--2`).
`cred`	Numeric. The credible interval width for summarizing the posterior distributions (defaults to 0.95; must be between 0 and 1).
`...`	Currently ignored.

Value

A list containing the summarized posterior distributions.

See Also

ggm_compare_estimate

Examples

```
# note: iter = 250 for demonstrative purposes
# data
Y <- bfi

# males and females
Ymale <- subset(Y, gender == 1,
                select = -c(gender,
                            education))[,1:5]

Yfemale <- subset(Y, gender == 2,
                  select = -c(gender,
                              education))[,1:5]

# fit model
fit <- ggm_compare_estimate(Ymale,  Yfemale,
                            type = "ordinal",
                            iter = 250,
                            prior_sd = 0.25,
                            progress = FALSE)

summary(fit)
```

---

summary.ggm_compare_explore

Summary Method for `ggm_compare_explore` *Objects*

Description

Summarize the posterior hypothesis probabilities

Usage

```
## S3 method for class 'ggm_compare_explore'
summary(object, col_names = TRUE, ...)
```

Arguments

`object`	An object of class `ggm_compare_explore`.
`col_names`	Logical. Should the summary include the column names (default is `TRUE`)? Setting to `FALSE` includes the column numbers (e.g., `1--2`).
`...`	Currently ignored.

Value

An object of class `summary.ggm_compare_explore`

See Also

`ggm_compare_explore`

Examples

```
# note: iter = 250 for demonstrative purposes

# data
Y <- bfi

# males and females
Ymale <- subset(Y, gender == 1,
                   select = -c(gender,
                               education))[,1:10]

Yfemale <- subset(Y, gender == 2,
                     select = -c(gender,
                                 education))[,1:10]

##########################
### example 1: ordinal ###
##########################

# fit model
fit <- ggm_compare_explore(Ymale,  Yfemale,
                           type = "ordinal",
                           iter = 250,
                           progress = FALSE)
# summary
summ <- summary(fit)

summ
```

`summary.predictability`

Summary Method for `predictability` *Objects*

Description

Summary Method for `predictability` Objects

Usage

```
## S3 method for class 'predictability'
summary(object, cred = 0.95, ...)
```

Arguments

`object`	An object of class `predictability`.
`cred`	Numeric. The credible interval width for summarizing the posterior distributions (defaults to 0.95; must be between 0 and 1).
`...`	Currently ignored

Examples

```

Y <- ptsd[,1:5]

fit <- explore(Y, iter = 250,
               progress = FALSE)

r2 <- predictability(fit, iter = 250,
                     progress = FALSE)

summary(r2)
```

`summary.select.explore`

Summary Method for `select.explore` *Objects*

Description

Summary Method for `select.explore` Objects

Usage

```
## S3 method for class 'select.explore'
summary(object, col_names = TRUE, ...)
```

Arguments

`object`	object of class `select.explore`.
`col_names`	Logical.
`...`	Currently ignored.

Value

a data frame including the posterior mean, standard deviation, and posterior hypothesis probabilities for each relation.

Examples

```
#  data
Y <- bfi[,1:10]

# fit model
fit <- explore(Y, iter = 250,
               progress = FALSE)

# edge set
E <- select(fit,
            alternative = "exhaustive")

summary(E)
```

`summary.var_estimate` *Summary Method for* `var_estimate` *Objects*

Description

Summarize the posterior distribution of each partial correlation and regression coefficient with the posterior mean, standard deviation, and credible intervals.

Usage

```
## S3 method for class 'var_estimate'
summary(object, cred = 0.95, ...)
```

Arguments

`object`	An object of class `var_estimate`
`cred`	Numeric. The credible interval width for summarizing the posterior distributions (defaults to 0.95; must be between 0 and 1).
`...`	Currently ignored.

Value

A dataframe containing the summarized posterior distributions, including both the partial correlations and the regression coefficients.

- `pcor_results` A data frame including the summarized partial correlations
- `beta_results` A list containing the summarized regression coefficients (one data frame for each outcome)

See Also

var_estimate

Examples

```
# data
Y <- subset(ifit, id == 1)[,-1]

# fit model with alias (var_estimate also works)
fit <- var_estimate(Y, progress = FALSE)

# summary ('pcor')
print(
summary(fit, cred = 0.95),
param = "pcor",
)


# summary ('beta')
print(
summary(fit, cred = 0.95),
param = "beta",
)
```

`tas` *Data: Toronto Alexithymia Scale (TAS)*

Description

A dataset containing items from the Toronto Alexithymia Scale (TAS). There are 20 variables and 1925 observations

Usage

```
data("tas")
```

Format

A data frame with 20 variables and 1925 observations (5 point Likert scale)

Details

- 1 I am often confused about what emotion I am feeling
- 2 It is difficult for me to find the right words for my feelings
- 3 I have physical sensations that even doctors don’t understand

- `4` I am able to describe my feelings easily
- `5` I prefer to analyze problems rather than just describe them
- `6` When I am upset, I don’t know if I am sad, frightened, or angry
- `7` I am often puzzled by sensations in my body
- `8` I prefer just to let things happen rather than to understand why they turned out that way
- `9` I have feelings that I can’t quite identify
- `10` Being in touch with emotions is essential
- `11` I find it hard to describe how I feel about people
- `12` People tell me to describe my feelings more
- `13` I don’t know what’s going on inside me
- `14` I often don’t know why I am angry
- `15` I prefer talking to people about their daily activities rather than their feelings
- `16` I prefer to watch “light” entertainment shows rather than psychological dramas
- `17` It is difficult for me to reveal my innermost feelings, even to close friends
- `18` I can feel close to someone, even in moments of silence
- `19` I find examination of my feelings useful in solving personal problems
- `20` Looking for hidden meanings in movies or plays distracts from their enjoyment
- `gender` "M" (male) or "F" (female)

Note

There are three domains

Difficulty identifying feelings: items 1, 3, 6, 7, 9, 13, 14

Difficulty describing feelings: items 2, 4, 11, 12, 17

Externally oriented thinking: items 10, 15, 16, 18, 19

References

Briganti, G., & Linkowski, P. (2019). Network approach to items and domains from the Toronto Alexithymia Scale. Psychological reports.

Examples

```
data("tas")
```

`var_estimate` *VAR: Estimation*

Description

Estimate VAR(1) models by efficiently sampling from the posterior distribution. This provides two graphical structures: (1) a network of undirected relations (the GGM, controlling for the lagged predictors) and (2) a network of directed relations (the lagged coefficients). Note that in the graphical modeling literature, this model is also known as a time series chain graphical model (Abegaz and Wit 2013).

Usage

```
var_estimate(
  Y,
  rho_sd = 0.5,
  beta_sd = 1,
  iter = 5000,
  progress = TRUE,
  seed = 1,
  ...
)
```

Arguments

`Y`	Matrix (or data frame) of dimensions *n* (observations) by *p* (variables).
`rho_sd`	Numeric. Scale of the prior distribution for the partial correlations, approximately the standard deviation of a beta distribution (defaults to 0.50).
`beta_sd`	Numeric. Standard deviation of the prior distribution for the regression coefficients (defaults to 1). The prior is by default centered at zero and follows a normal distribution (Equation 9, Sinay and Hsu 2014)
`iter`	Number of iterations (posterior samples; defaults to 5000).
`progress`	Logical. Should a progress bar be included (defaults to `TRUE`) ?
`seed`	An integer for the random seed (defaults to 1).
`...`	Currently ignored.

Details

Each time series in `Y` is standardized (mean = 0; standard deviation = 1).

Value

An object of class `var_estimate` containing a lot of information that is used for printing and plotting the results. For users of **BGGM**, the following are the useful objects:

- `beta_mu` A matrix including the regression coefficients (posterior mean).

- `pcor_mu` Partial correlation matrix (posterior mean).
- `fit` A list including the posterior samples.

Note

Regularization:

A Bayesian ridge regression can be fitted by decreasing `beta_sd` (e.g., `beta_sd = 0.25`). This could be advantageous for forecasting (out-of-sample prediction) in particular.

References

Abegaz F, Wit E (2013). "Sparse time series chain graphical models for reconstructing genetic networks." *Biostatistics*, **14**(3), 586–599. doi:10.1093/biostatistics/kxt005.

Sinay MS, Hsu JS (2014). "Bayesian inference of a multivariate regression model." *Journal of Probability and Statistics*, **2014**.

Examples

```
# data
Y <- subset(ifit, id == 1)[,-1]

# use alias (var_estimate also works)
fit <- var_estimate(Y, progress = FALSE)

fit
```

`weighted_adj_mat` *Extract the Weighted Adjacency Matrix*

Description

Extract the weighted adjacency matrix (posterior mean) from `estimate`, `explore`, `ggm_compare_estimate`, and `ggm_compare_explore` objects.

Usage

```
weighted_adj_mat(object, ...)
```

Arguments

`object`	A model estimated with **BGGM**. All classes are supported, assuming there is matrix to be extracted.
`...`	Currently ignored.

Value

The weighted adjacency matrix (partial correlation matrix with zeros).

Examples

```
# note: iter = 250 for demonstrative purposes
Y <- bfi[,1:5]

# estimate
fit <- estimate(Y, iter = 250,
                progress = FALSE)

# select graph
E <- select(fit)

# extract weighted adj matrix
weighted_adj_mat(E)
```

`women_math` *Data: Women and Mathematics*

Description

A data frame containing 1190 observations (n = 1190) and 6 variables (p = 6) measured on the binary scale.

Usage

`data("women_math")`

Format

A data frame containing 1190 observations (n = 1190) and 6 variables (p = 6) measured on the binary scale (Fowlkes et al. 1988). These data have been analyzed in Tarantola (2004) and in (Madigan and Raftery 1994). The variable descriptions were copied from (section 5.2) (section 5.2, Talhouk et al. 2012)

Details

- `1` Lecture attendance (attend/did not attend)
- `2` Gender (male/female)
- `3` School type (urban/suburban)
- `4` “I will be needing Mathematics in my future work” (agree/disagree)
- `5` Subject preference (math/science vs. liberal arts)
- `6` Future plans (college/job)

References

Fowlkes EB, Freeny AE, Landwehr JM (1988). "Evaluating logistic models for large contingency tables." *Journal of the American Statistical Association*, **83**(403), 611–622. doi:10.1080/01621459.1988.10478640.

Madigan D, Raftery AE (1994). "Model selection and accounting for model uncertainty in graphical models using Occam's window." *Journal of the American Statistical Association*, **89**(428), 1535–1546.

Talhouk A, Doucet A, Murphy K (2012). "Efficient Bayesian inference for multivariate probit models with sparse inverse correlation matrices." *Journal of Computational and Graphical Statistics*, **21**(3), 739–757. doi:10.1080/10618600.2012.679239.

Tarantola C (2004). "MCMC model determination for discrete graphical models." *Statistical Modelling*, **4**(1), 39–61. doi:10.1191/1471082x04st063oa.

Examples

```
data("women_math")
```

`zero_order_cors` *Zero-Order Correlations*

Description

Estimate zero-order correlations for any type of data. Note zero-order refers to the fact that no variables are controlled for (i.e., bivariate correlations). To our knowledge, this is the only Bayesian implementation in `R` that can estiamte Pearson's, tetrachoric (binary), polychoric (ordinal with more than two cateogries), and rank based correlation coefficients.

Usage

```
zero_order_cors(
  Y,
  type = "continuous",
  iter = 5000,
  mixed_type = NULL,
  progress = TRUE
)
```

Arguments

`Y`	Matrix (or data frame) of dimensions *n* (observations) by *p* (variables).
`type`	Character string. Which type of data for `Y` ? The options include `continuous`, `binary`, `ordinal`, or `mixed`. See the note for further details.
`iter`	Number of iterations (posterior samples; defaults to 5000).

`mixed_type`	Numeric vector. An indicator of length p for which varibles should be treated as ranks. (1 for rank and 0 to assume normality). The default is currently to treat all integer variables as ranks when `type = "mixed"` and `NULL` otherwise. See note for further details.
`progress`	Logical. Should a progress bar be included (defaults to `TRUE`) ?

Details

Mixed Type:

The term "mixed" is somewhat of a misnomer, because the method can be used for data including *only* continuous or *only* discrete variables. This is based on the ranked likelihood which requires sampling the ranks for each variable (i.e., the data is not merely transformed to ranks). This is computationally expensive when there are many levels. For example, with continuous data, there are as many ranks as data points!

The option `mixed_type` allows the user to determine which variable should be treated as ranks and the "emprical" distribution is used otherwise (Hoff 2007). This is accomplished by specifying an indicator vector of length p. A one indicates to use the ranks, whereas a zero indicates to "ignore" that variable. By default all integer variables are treated as ranks.

Dealing with Errors:

An error is most likely to arise when `type = "ordinal"`. The are two common errors (although still rare):

- The first is due to sampling the thresholds, especially when the data is heavily skewed. This can result in an ill-defined matrix. If this occurs, we recommend to first try decreasing `prior_sd` (i.e., a more informative prior). If that does not work, then change the data type to `type = mixed` which then estimates a copula GGM (this method can be used for data containing **only** ordinal variable). This should work without a problem.
- The second is due to how the ordinal data are categorized. For example, if the error states that the index is out of bounds, this indicates that the first category is a zero. This is not allowed, as the first category must be one. This is addressed by adding one (e.g., `Y + 1`) to the data matrix.

Value

- `R` An array including the correlation matrices (of dimensions p by p by *iter*)
- `R_mean` Posterior mean of the correlations (of dimensions p by p)

Examples

```
# note: iter = 250 for demonstrative purposes

Y <- ptsd[,1:3]

#################################
####### example 1: Pearson's ####
#################################

fit <- zero_order_cors(Y, type = "continuous",
```

```
                       iter = 250,
                       progress = FALSE)


#################################
###### example 2: polychoric ####
#################################

fit <- zero_order_cors(Y+1, type = "ordinal",
                       iter = 250,
                       progress = FALSE)


###########################
##### example 3: rank #####
###########################

fit <- zero_order_cors(Y+1, type = "mixed",
                       iter = 250,
                       progress = FALSE)

############################
## example 4: tetrachoric ##
############################

# binary data
Y <- women_math[,1:3]

fit <- zero_order_cors(Y, type = "binary",
                       iter = 250,
                       progress = FALSE)
```

Index