Package 'VIGoR'

June 5, 2023

Type Package

Title Variational Bayesian Inference for Genome-Wide Regression

Version 1.1.2

Date 2023-06-05

Description Conducts linear regression using variational Bayesian inference, particularly optimized for genome-wide association mapping and whole-genome prediction which use a number of DNA markers as the explanatory variables. Provides seven regression models which select the important variables (i.e., the variables related to response variables) among the given explanatory variables in different ways (i.e., model structures).

License MIT + file LICENSE

NeedsCompilation yes

Author Akio Onogi [aut, cre, cph],
R Core Team [ctb] (Provide Rdynload.h and Boolean.h),
Hiroyoshi Iwata [cph],
Takuji Nishimura [ctb] (Developer of Mersenne twister in header1.h),
Makoto Matsumoto [ctb] (Developer of Mersenne twister in header1.h),
STRUCTURE software contributors [ctb] (Provide snorm and RNormal functions in header2.h),
Alan Miller [ctb] (Program mylgamma function in header2.h),
Peter Beerli [ctb] (Translate mylgamma function in header2.h)

Maintainer Akio Onogi <onogiakio@gmail.com>

Repository CRAN

Date/Publication 2023-06-05 05:30:02 UTC

R topics documented:

hyperpara *Calculation of hyperparameter values*

Description

This function determines the hyperparameter values of regression methods, based on two assumptions.

Usage

```
hyperpara(X, Mvar, Model = c("BL", "EBL", "BayesA", "BayesB", "BayesC", "BRR"),
        Kappa = 0.01, Xtype = c("Geno", "Var"), f = 0, BL.Phi = 1, EBL.Phi = 0.1,
          EBL.Omega = 0.1, EBL.Psi = 1, Nu = 5, Verbose=FALSE)
```

Arguments

X	An (N x P) matrix, where N and P denote the number of samples and variables, respectively. When X is SNP genotypes as specified by Xtype = "Geno", SNP genotypes should be coded with values between 0 and 2. SNP genotypes are used to calculate sum (2*Q*(1-Q)*(1+f)) where Q is a vector of allele frequencies and f is the inbreeding coefficient. Missing values in X are not allowed. When X is "Var"(variables other than SNPs), variances of variables are used instead of sum (2*Q*(1-Q)*(1+f)).
Mvar	A scalar or vector denoting the assumed proportion of variance of Y that can be explained by X. Mvar is < 1.0 in BL and EBL, and <= 1.0 in the other methods.
Model	One of the six regression methods (BL, EBL, BayesA, BayesB, BayesC, BRR).
Kappa	A scalar or vector containing the assumed proportion of variables with NON-ZERO EFFECTS. Used when BL, EBL, BayesB, and BayesC. Kappa is set to 1 when BayesA and BRR are used.
Xtype	Allowed Xtypes are "Geno" and "Var". Enter "Geno" when X contains the SNP genotypes and "Var" when X contains variables other than SNP genotypes.
f	A scalar representing the inbreeding coefficient. Enter 1 for inbred species.
BL.Phi	A scalar or vector containing Phi values of BL.
EBL.Phi	A scalar or vector containing Phi values of EBL.
EBL.Omega	A scalar or vector containing Omega values of EBL.
EBL.Psi	A scalar or vector containing Psi values of EBL.
Nu	A scalar or vector containing Nu values of BayesA, BayesB, BayesC, and BRR.
Verbose	Specifies whether to print information (TRUE) or not (FALSE).

Details

To run vigor, users must specify the following hyperparameter values.

- BL: Phi, Omega
- EBL: Phi, Omega, Psi, Theta
- BayesA: Nu, S2
- BayesB: Nu, S2, Kappa
- BayesC: Nu, S2, Kappa
- BRR: Nu, S2

This function calculates the Omega of BL; Theta of EBL; S2 of BayesA, BayesB, BayesC, and BRR. Mvar, Kappa, and the other hyperparameters required by each method are specified by the user. The definitions of Mvar and Kappa are intuitively understandable and relatively easy to specify (see Arguments). For the other hyperparameters, the default values are recommended. When the arguments of hyperpara are vectors, all value-combinations are returned as a matrix. The hyperparameters are explained in the details of vigor and in the pdf manual of VIGoR (Onogi 2021).

Value

This function returns a vector when yielding a single hyperparameter set, and a matrix when yielding multiple hyperparameter sets. The rows and columns of the matrix correspond to the sets (value combinations) and the hyperparameters, respectively. See examples below.

References

Onogi A., Variational Bayesian inference for genome-wide regression: joint estimation of multiple learners, in prep.
Onogi A. & Iwata H., 2016 VIGoR: Variational Bayesian Inference for Genome-Wide Regression. Journal of Open Research Software, 4: e11
Onogi A., 2021, Documents for VIGoR ver. 1.1.0, https://github.com/Onogi/VIGoR

See Also

vigor

Examples

```
#data
data(sampledata)
dim(X) #500 samples and 1000 variables
unique(X[1:(100*1000)]) #coded as 0, 1, 2

#A single Mvar (0.5) and Kappa (0.01) value is assumed for BL.
#A vector is returned.
hyperpara(X, 0.5, "BL", 0.01, Verbose = TRUE)
```

```
#Phi is set to 1 as default. To change Phi, use BL.Phi.
hyperpara(X, 0.5, "BL", 0.01, BL.Phi = 5)

#Calculate multiple hyperparameter value sets of BayesC assuming that Kappa is 0.1 and 0.01.
#A matrix is returned.
hyperpara(X, 0.5, "BayesC", c(0.1, 0.01))

#The output vector can be used as the argument of vigor
ETA <- list(list(model = "BayesB",X = X,
                 H = hyperpara(X, 0.5, "BayesC", c(0.1, 0.01))))
Result <- vigor(Y, ETA, Function = "tuning")
Result$Metrics

#Calculate hyperparameter values of EBL
hyperpara(X, c(0.2, 0.5), "EBL", c(0.1, 0.01), EBL.Omega = c(0.5, 1))
#Total 2 (Mvar) x 2 (Kappa) x 2 (EBL.Omega) = 8 sets are returned.
```

`predict_vigor` *Predict Y of new data using a training result with vigor*

Description

This function predicts Y of new data using a training result with vigor.

Usage

```
predict_vigor(training_result, newX)
```

Arguments

`training_result`	Result of vigor
`newX`	A list of X. newX contains X for each learner. The length and order of learners should be same as those of ETA used for training. Each element of newX is a matrix. When BLUP is used, X is an n1 x n2 relationship matrix where n1 and n2 are the numbers of samples in test and training data, respectively. When the other methods are used, X is an n1 x p matrix where p is the number of explanatory variables. p should be the same as the training data. When the intercept is added automatically, newX needs not to include the intercept.

Details

This function predict Y of new data (newX). When multiple learners are included in the model, predicted values are calculated for each element of newX, and the summation of all predicted values is returned (predicted values of each learner are not returned). When contributions of each learner (method) are of interest, add NULL to the elements of newX to be ignored (see examples).

Value

A vector of predicted values is returned.

See Also

vigor

Examples

```
#data
data(sampledata)
dim(X) #Matrix of SNP genotypes (explanatory variables)
dim(Z) #Matrix of a fixed effect (explanatory variables)
length(Y) #Vector of response variables

#Train EBL using the first 80 percent of data
#Then predict the remaining 20 percent data
ETA <- list(list(model = "EBL",X = X[1:400, ]))
Train <- vigor(Y[1:400], ETA)
newX <- list(X[401:500, ])
Predict <- predict_vigor(Train, newX)
plot(Y[401:500], Predict)
#When the intercept is automatically added when training,
#the intercept is again automatically added to predicted values

#Use multiple regression methods
#Fit additive and dominance effects using BayesC with different shrinkage levels
#Also fixed effects are added
X.d <- X
X.d[X == 2] <- 0 #heterozygotes are 1 and homozygotes are 0
Z.matrix <- model.matrix(~ Z)
ETA <- list(list(model = "FIXED", X = Z.matrix[1:400, ]),
            list(model = "BayesC", X = X[1:400, ], H = c(5, 0.1, 0.01)),
            list(model = "BayesC", X = X.d[1:400, ], H = c(5, 0.1, 0.001)))
Train <- vigor(Y[1:400], ETA)
newX <- list(Z.matrix[401:500, ], X[401:500, ], X.d[401:500, ])
Predict <- predict_vigor(Train, newX)
plot(Y[401:500], Predict)

#When fixed effects are specified using formula,
Data <- data.frame(Z = factor(Z))
ETA <- list(list(~ Z, model = "FIXED", data = Data[1:400, ,drop=FALSE]),
            list(model = "BayesB", X = X[1:400, ], H = c(5, 0.1, 0.01)),
            list(model = "BayesB", X = X.d[1:400, ], H = c(5, 0.1, 0.001)))
Train <- vigor(Y[1:400], ETA)
newX <- list(~ Z, data = Data[401:500, , drop=FALSE], X[401:500, ], X.d[401:500, ])
Predict <- predict_vigor(Train, newX)
plot(Y[401:500], Predict)
#NOTE: please confirm that levels of fixed effects are consistent
#between the training and testing data
```

```
#Contributions of each learner can be assessed by filling newX with NULL
##Contribution of additive effect
newX <- list(NULL, X[401:500, ], NULL)
Predict <- predict_vigor(Train, newX)
plot(Y[401:500], Predict)

##Contribution of dominance effect
newX <- list(NULL, NULL, X.d[401:500, ])
Predict <- predict_vigor(Train, newX)
plot(Y[401:500], Predict)
```

---

vigor *Variational Bayesian inference for genome-wide regression*

Description

This function performs Bayesian genome-wide regression using variational Bayesian algorithms. The available regression methods are Bayesian lasso (BL), extended Bayesian lasso (EBL), BayesA, BayesB, BayesC, Bayesian ridge regression (BRR), BLUP, and fixed effects (FIXED) (fixed effects mean regression using noninformative priors). This function allows multiple regression methods (learners) in a single model. For example, additive effects and interaction effects can be incorporated in a single model using BL and BayesB which provide different shrinkage levels.

Usage

```
vigor(Y, ETA, Function = c("fitting", "tuning", "cv"), Nfold = 5, CVFoldTuning = 5,
      Partition = NULL, Thresholdvalue = 1e-5,
    Maxiteration = 1000, RandomIni = TRUE, Metrics = c("rmse", "cor"), Verbose = TRUE)
```

Arguments

Y	An N-length vector of response variables, where N is the number of samples. Missing data (coded as NA) are allowed.
ETA	A nested list to specify regression methods, explanatory variables, and hyperparameters. The length of ETA is the number of methods (learners) incorporated in a single model. See details below.
Function	One of the strings "fitting", "tuning", and "cv". See details below.
Nfold	An integer value. When n > 1, n-fold cross-validation (CV) is performed on randomly partitioned individuals. When the integer is -1, leave-one-out CV is conducted. Used when Function = "cv" and Partition == NULL.
CVFoldTuning	An integer specifying the fold number of the CV in hyperparameter tuning. Used when Function = "cv" or "tuning" and multiple hyperparameter sets are given.
Partition	A matrix defining the partitions of CV. See details and examples below. Used when Function = "cv".
Thresholdvalue	Specifies the convergence threshold. Smaller values indicate stricter thresholds.

`Maxiteration`	Maximum number of iterations.
`RandomIni`	If TRUE, the initial values of the SNP effects are randomly determined. Otherwise, they are set to 0.
`Metrics`	One of the strings "rmse" and "cor" to specify the metrics used in CV. rmse and cor use RMSE and Pearson correlation, respectively.
`Verbose`	If TRUE, print the run information to the console.

Details

Regression methods

Vigor supports the following regression methods;

- BL (Bayesian lasso)
- EBL (extended Bayesian lasso)
- BayesA
- BayesB
- BayesC
- BRR (Bayesian ridge regression)
- BLUP
- FIXED (fixed effects)

These methods can be included in a single model simultaneously with different explanatory variables. For the details of these methods and the theoretical backgrounds of vigor, see the pdf document (Onogi 2021).

ETA

Each element (list) of ETA consists of the following objects.

- model : One of strings representing regression methods, "BL", "EBL", "BayesA", "BayesB", "BayesC", "BRR", "BLUP", or "FIXED"
- X : An explanatory variables (e.g., SNP genotypes) of (N x P) matrix, where N and P denote the number of samples and variables, respectively
- K : An N x N kernel matrix (e.g., genomic relationship matrix) used when BLUP is specified as model. When BLUP is specified but K is lacked, the linear kernel is created from X as scale(X)*t(scale(X))/ncol(X).
- H : A vector or matrix including hyperparameters
- data : A data frame containing fixed effects. Used to model FIXED using formula.

Specification of model is essential for all methods. For regression methods except for BLUP and FIXED, X is essential. For BLUP, either X or K is essential. For FIXED, either X or formula with data is essential.

Hyperparameters

The regression methods require hyperparameters as H in ETA. H can be a vector or matrix. Below is the order of hyperparameters in H (order of columns in the case of matrix). Default values are shown in parenthesis.

- BL : Phi(1), Omega(1)
- EBL : Phi(0.1), Omega(0.1), Psi(1), Theta(0.1)
- BayesA : Nu(5), S2(0.01)
- BayesB : Nu(5), S2(0.1), Kappa(0.01)
- BayesC : Nu(5), S2(0.1), Kappa(0.01)
- BRR : Nu(5), S2(0.01)
- BLUP : Nu(5), S2(0.3)

Note that Kappa is the proportion of explanatory variables with NON-ZERO EFFECTS. Also note that Y is standardized automatically. To specify multiple hyperparameter sets, give an S x Nh matrix where S is the number of sets and Nh is the number of hyperparameters of the method to ETA. See the pdf document (Onogi 2021) for the details of hyperparameters.

Functions
The functions of vigor are "fitting", "tuning", and "cv".

- "fitting" : Fits the specified regression model to the data. When H (hyperparameters) includes multiple hyperparameter sets (i.e., H is a matrix), only the first set is used.
- "tuning" : Selects the best hyperparameter set using CV for tuning. This set is then used for model fitting. The CV is performed on randomly partitioned data. The number of folds is determined by CVFoldTuning. When multiple hyperparameter sets are given to multiple methods, all combinations of hyperparameters are attempted.
- "cv" : Conducts CV and returns the predicted values. When multiple hyperparameter sets are given, tuning is performed at each fold of the CV. As in Tuning, when multiple hyperparameter sets are given to multiple methods, all combinations of hyperparameters are attempted.

Partition matrix
The following is a possible Partition of 20 individuals evaluated in a five-fold CV:

14 11 3 2 7
5 4 20 10 9
6 8 16 15 12
18 13 17 1 19

Sample (row numbers in Y/X/K) 14, 5, 6, and 18 are removed from the training set at the first fold of the five-fold CV. Samples 11, 4, 8, and 13 are removed at the next fold. This process is repeated up to the fold number of the CV. If the number of samples N is 19, the gap is filled with -9. For example,

8 6 3 14 18
12 4 1 15 5
17 9 13 11 10
19 16 7 2 -9

An example of random sampling validation in which samples can be sampled more than once is shown below.

18 3 11 16 13
17 8 13 13 18
7 15 14 19 7
1 13 12 7 2

Samples 18, 13, and 7 are repeatedly used as testing samples.

Random partitioning outputs a Partition matrix, which can be input as the Partition matrix in subsequent analysis.

Intercept
If No FIXED is given in ETA, vigor automatically adds the intercept to the regression model as a fixed effect (FIXED). If FIXED is given by the user, vigor regards the first column as the intercept

Standardization
Vigor standardizes Y (response variables). Although most returned values are scaled back to the original scale, the lower bound of the marginal log likelihood is returned as the standardized scale.

Value

When Function = "fitting" or "tuning", a list containing the following elements is returned.

`$LB`	Lower bound of the marginal log likelihood of Y.
`$ResidualVar`	Residual variances (1/Tau02) at each iteration (from start to end).
`$H`	Used hyperparameters.
`$Fittedvalue`	Fitted values.
`$Metrics`	Metrics for hyperparameter tuning. Returned when Function = "tuning"
`$ETA`	A list containing results for each method. • Beta: Posterior means of regression coefficients of X • Sd.beta: Posterior standard deviations of Beta (uncertainty of Beta) • Sigma2: Posterior means of variance of Beta or U • Rho: Posterior means of model-inclusion probabilities • U: Posterior means of random effects of BLUP • Sd.u: Posterior standard deviations of U (uncertainty of U) • iK: Inverse of K Beta and Sd.beta are returned for BL, EBL, BayesA, BayesB, BayesC, and FIXED, and U, Sd.u, and iK are returned for BLUP. Rho is returned for BayesB and BayesC. Sigma2 is returned for methods except for BL and EBL.
`$AddIntercept`	True when the intercept was added automatically.

When Function = "cv", a list containing the following elements is returned.

`$Prediction`	A vector of predicted values
`$Metrics`	Metrics of hyperparameter tuning. Chosen sets and corresponding metrics at each fold are returned.

`$Partition`	A matrix representing the partition used in random partitioning. This matrix can be used as the argument Partition in subsequent analyses.
`$AddIntercept`	True when the intercept was added automatically.

Author(s)

Akio Onogi

References

Onogi A., Variational Bayesian inference for genome-wide regression: joint estimation of multiple learners, in prep.
Onogi A. & Iwata H., 2016 VIGoR: Variational Bayesian Inference for Genome-Wide Regression. Journal of Open Research Software, 4: e11
Onogi A., 2021, Documents for VIGoR ver. 1.1.0, https://github.com/Onogi/VIGoR

Examples

```
#DATA###########################################################################
data(sampledata)
dim(X) #Matrix of SNP genotypes (explanatory variables)
dim(Z) #Matrix of a fixed effect (explanatory variables)
length(Y) #Vector of response variables


#Fitting#########################################################################
#Example 1: Fit SNP genotypes with BayesC
ETA <- list(list(model = "BayesC", X = X))
Result <- vigor(Y, ETA)
##see estimated SNP effects
plot(abs(Result$ETA[[1]]$Beta), pch = 20)
##10 SNPs at 1, 101, ..., 901 have non-zero effects
abline(v = seq(1, 1000, 100), col = 2, lty = 2)
##see inclusion probability
plot(Result$ETA[[1]]$Rho, pch = 20)
abline(v = seq(1, 1000, 100), col = 2, lty = 2)
##Intercept is added automatically as the last learner
Result$ETA[[2]]$Beta


#Example 2: Fit fixed effects and SNP genotypes
##There are two approaches to fit fixed effects
##(1) Create model matrix
Z #Z consists of three categories (A, B, and C)
Z.matrix <- model.matrix(~ Z)
head(Z.matrix) #The first column is the intercept
##Fit with EBL
ETA <- list(list(model = "FIXED", X = Z.matrix),
            list(model = "EBL", X = X))
Result <- vigor(Y, ETA)
```

```
##Estimated fixed effects (intercept, B, and C)
Result$ETA[[1]]$Beta
##Estimated SNP effects
plot(abs(Result$ETA[[2]]$Beta), pch = 20)
abline(v = seq(1, 1000, 100), col = 2, lty = 2)
##NOTE: when FIXED is added by user, the intercept is not automatically added.
##Thus, variables in FIXED should contain the intercept.

##(2) Use formula
Data <- data.frame(Z = factor(Z))
ETA <- list(list(~ Z, model = "FIXED", data = Data),
            list(model = "BayesA", X = X))
Result <- vigor(Y, ETA)
##Estimated fixed effects (intercept, B, and C)
Result$ETA[[1]]$Beta
plot(abs(Result$ETA[[2]]$Beta), pch = 20)
abline(v=seq(1,1000,100),col=2,lty=2)
##NOTE: formula automatically adds the intercept


#Example 3: Multiple regression methods in a single model
##Some SNPs in X have dominance (non-additive) effects
##Fit SNP genotypes coded as additive and dominance with different shrinkage levels
X.d <- X
X.d[X == 2] <- 0 #heterozygotes are 1 and homozygotes are 0
ETA <- list(list(~ Z, model = "FIXED", data = Data),
            list(model = "BayesC", X = X, H = c(5, 0.1, 0.01)),
            list(model = "BayesC", X = X.d, H = c(5, 0.1, 0.001)))
Result <- vigor(Y, ETA)
##Inclusion probability for additive effects
plot(Result$ETA[[2]]$Rho, pch = 20)
abline(v = seq(1, 1000, 100), col = 2, lty = 2)
##Inclusion probability for dominance effects
plot(Result$ETA[[3]]$Rho, pch = 20)
##SNPs at 1, 201, ..., 801 have non-zero effects
abline(v = seq(1, 1000, 200), col = 2, lty = 2)

##Fit additive and dominance effects with different learners
ETA <- list(list(~ Z, model = "FIXED", data = Data),
            list(model = "BL", X = X, H = c(1, 0.01)),
            list(model = "BayesC", X = X.d, H = c(5, 0.1, 0.001)))
Result <- vigor(Y, ETA)
plot(abs(Result$ETA[[2]]$Beta), pch = 20)
abline(v = seq(1, 1000, 100), col = 2, lty = 2)
plot(Result$ETA[[3]]$Rho, pch = 20)
abline(v = seq(1, 1000, 200), col = 2, lty = 2)


#Tuning hyperparameters#########################################################
#Example 4: Model fitting after hyperparameter tuning with cross-validation
##Candidate hyperparameter values are determined with hyperpara
##Use BayesB
ETA <- list(list(~ Z, model = "FIXED", data = Data),
```

```
            list(model = "BayesB", X = X,
                 H = hyperpara(X, 0.5, "BayesB", c(0.1,0.01))))
Result <- vigor(Y, ETA, Function = "tuning")
##See the tuned result
Result$Metrics
##The model was fitted to the full data with the best set
Result$H
plot(Result$ETA[[2]]$Rho, pch = 20)
abline(v = seq(1, 1000, 100), col = 2, lty = 2)
##See how much the model was fitted
plot(Y, Result$Fittedvalue); abline(0, 1)

##When multiple learners used, all combinations of hyperparameter sets are compared
ETA <- list(list(~ Z, model = "FIXED", data = Data),
            list(model = "BayesB", X = X,
                 H = hyperpara(X, 0.5, "BayesB", c(0.1,0.01))),
            list(model = "BayesC", X = X.d,
                 H = hyperpara(X, 0.5, "BayesC", c(0.1,0.01))))
Result <- vigor(Y, ETA, Function = "tuning")
##BayesB and BayesC have two candidate sets, respectively.
##Thus, total 2 x 2 = 4 combinations are compared.
Result$Metrics
##The model was fitted to the full data with the best combination.
Result$H
plot(Result$ETA[[2]]$Rho, pch = 20)
abline(v = seq(1, 1000, 100), col = 2, lty = 2)
plot(Result$ETA[[3]]$Rho, pch = 20)
abline(v = seq(1, 1000, 200), col = 2, lty = 2)


#Cross-validation#####################################################################
#Example 5: Cross-validation with random splitting
ETA <- list(list(~ Z, model = "FIXED", data = Data),
            list(model = "BayesC", X = X,
                 H = hyperpara(X, 0.5, "BayesC", c(0.1,0.01))))
Result <- vigor(Y, ETA, Function="cv")
##Because two hyperparameter sets are provided,
##nested CV is conducted at each fold to tune hyperparameters
##See which set was selected at each fold
Result$Metrics
##See predicted values
plot(Y, Result$Prediction)
cor(Y, Result$Prediction)

#Example 6: Cross-validation with the specified splitting
##Perform CV using the same partition as Example 5. Use EBL
ETA <- list(list(~ Z, model = "FIXED", data = Data),
            list(model = "EBL", X = X))
Result2 <- vigor(Y, ETA, Function="cv", Partition = Result$Partition)
plot(Y, Result2$Prediction)
cor(Y, Result2$Prediction)
```

X *An example of SNP genotypes (explanatory variables)*

Description

An example of SNP genotypes consisting of 1000 SNPs and 500 samples. The genotypes are coded as 0 (AA), 1 (AB), and 2 (BB). The genotypes were randomly generated as described in the details. The first, 101th, ..., and 901th SNP have additive effects. The first, 201th, ..., and 801th SNPs have also dominance effects.

Details

X was generated by

```
N <- 500
P <- 1000
X <- matrix(sample(c(0, 1, 2), N * P, replace = T,
prob = c(0.49, 0.42, 0.09)), nc = P)
```

See Also

Y, Z

Examples

```
data(sampledata)
dim(X) #500 samples and 1000 SNPs
unique(X[1:(500*1000)]) #0,1,2
```

Y *An example of response variables.*

Description

A vector consisting of 500 samples.

Details

Y mimicked phenotypic values in quantitative genetics. Y was created from X (SNP genotypes) and Z (fixed effect). 10 SNPs in X have additive effects, and 5 out of 10 SNPs have dominance effects further. Y also include environmental noises. The variances of additive, dominance, and noise are approximately 1 : 1 : 1.

See Also

X, Z

Examples

```
data(sampledata)
length(Y) #500 samples
any(is.na(Y)) #FALSE. Y has no missing (but it's allowed).
```

Z *An example of fixed effects (explanatory variables)*

Description

An example of fixed effects for 500 samples. The effect consists of three levels, "A", "B", and "C.

Details

Compared with the mean of "A", the mean of "B" is 3 lower and the mean of "C" is 3 higher.

See Also

Y, X

Examples

```
data(sampledata)
dim(Z) #500 samples and 1 effect
unique(Z) #"A", "B", and "C"
```

Index