Package ‘dbstats’

October 17, 2017

Type Package
Title Distance-Based Statistics
Version 1.0.5
Date 2017-10-17
Author Boj, Eva <evabo@ub.edu>, Caballe, Adria <adria.caballe@upc.edu>, Delicado, Pedro <pedro.delicado@upc.edu> and Fortiana, Josep <fortiana@ub.edu>.
Maintainer Josep Fortiana <fortiana@ub.edu>
Description Prediction methods where explanatory information is coded as a matrix of distances between individuals. Distances can either be directly input as a distances matrix, a squared distances matrix, an inner-products matrix or computed from observed predictors.
License GPL-2
LazyLoad no
Repository CRAN
Depends R (>= 2.10.0), cluster, pls
Suggests proxy
NeedsCompilation no
Date/Publication 2017-10-17 14:39:58 UTC

R topics documented:

  dbstats-package .................................................. 2
  as.D2 ......................................................... 4
  as.Gram ......................................................... 5
  D2toDist ......................................................... 5
  D2toG .......................................................... 6
  dbglm .......................................................... 7
  dblm ......................................................... 12
  dbplsr ....................................................... 15
  disttoD2 ...................................................... 18
  GtoD2 ......................................................... 19
  ldbglm ......................................................... 20
Description

This package contains functions for distance-based prediction methods.

These are methods for prediction where predictor information is coded as a matrix of distances between individuals.

In the currently implemented methods the response is a univariate variable as in the ordinary linear model or in the generalized linear model.

Distances can either be directly input as an distances matrix, a squared distances matrix, an inner-products matrix (see gto02) or computed from observed explanatory variables.

Notation convention: in distance-based methods we must distinguish observed explanatory variables which we denote by Z or z, from Euclidean coordinates which we denote by X or x. For explanation on the meaning of both terms see the bibliography references below.

Observed explanatory variables z are possibly a mixture of continuous and qualitative explanatory variables or more general quantities.

dbstats does not provide specific functions for computing distances, depending instead on other functions and packages, such as:

- `dist` in the `stats` package.
- `dist` in the `proxy` package. When the `proxy` package is loaded, its `dist` function supersedes the one in the `stats` package.
- `daisy` in the `cluster` package. Compared to both instances of `dist` above whose input must be numeric variables, the main feature of `daisy` is its ability to handle other variable types as well (e.g. nominal, ordinal, (a)symmetric binary) even when different types occur in the same data set.

Actually the last statement is not hundred percent true: it refers only to the default behaviour of both `dist` functions, whereas the `dist` function in the `proxy` package can evaluate distances
between observations with a user-provided function, entered as a parameter, hence it can deal with any type of data. See the examples in \texttt{pr.DB}.

Functions of \texttt{dbstats} package:

Linear and local linear models with a continuous response:

- \texttt{dblm} for distance-based linear models.
- \texttt{1dlbm} for local distance-based linear models.
- \texttt{dbplsr} for distance-based partial least squares.

Generalized linear and local generalized linear models with a numeric response:

- \texttt{dbglm} for distance-based generalized linear models.
- \texttt{1dbglm} for local distance-based generalized linear models.

Details

<table>
<thead>
<tr>
<th>Package</th>
<th>dbstats</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Package</td>
</tr>
<tr>
<td>Version</td>
<td>1.0.1</td>
</tr>
<tr>
<td>Date</td>
<td>2011-06-21</td>
</tr>
<tr>
<td>License</td>
<td>GPL-2</td>
</tr>
<tr>
<td>LazyLoad</td>
<td>yes</td>
</tr>
</tbody>
</table>

Author(s)

Boj, Eva <evaboj@ub.edu>, Caballe, Adria <adria.caballe@upc.edu>, Delicado, Pedro <pedro.delicado@upc.edu> and Fortiana, Josep <fortiana@ub.edu>

References


---

**as.D2**

**D2 objects**

**Description**

as.D2 attempts to turn its argument into a D2 class object.

is.D2 tests if its argument is a (strict) D2 class object.

**Usage**

as.D2(x)

is.D2(x)

**Arguments**

x an R object.

**Value**

An object of class D2 containing the squared distances matrix between individuals.

**Author(s)**

Boj, Eva <evaboj@ub.edu>, Caballe, Adria <adria.caballe@upc.edu>, Delicado, Pedro <pedro.delicado@upc.edu> and Fortiana, Josep <fortiana@ub.edu>

**See Also**

D2toG, disttoD2, D2toDist and GtoD2 for conversions.
as.Gram

Description
as.Gram attempts to turn its argument into a Gram class object.
is.Gram tests if its argument is a (strict) Gram class object.

Usage
as.Gram(x)
is.Gram(x)

Arguments
x an R object.

Value
A Gram class object. Weighted centered inner products matrix of the squared distances matrix.

Author(s)
Boj, Eva <evaboj@ub.edu>, Caballe, Adria <adria.caballe@upc.edu>, Delicado, Pedro <pedro.delicado@upc.edu>
and Fortiana, Josep <fortiana@ub.edu>

See Also
D2toG, disttoD2, D2toDist and GtoD2 for conversions.

D2toDist

Description
Converts D2 class object into dist class object.

Usage
D2toDist(D2)

Arguments
D2 D2 object. Squared distances matrix between individuals.
D2toG

Distance conversion: D2 to G

Description
Converts D2 class object into Gram class object.

Usage
D2toG(D2, weights)

Arguments
D2 D2 object. Squared distances matrix between individuals.
weights an optional numeric vector of weights. By default all individuals have the same weight.

Value
An object of class Gram containing the Doubly centered inner product matrix of D2.

Author(s)
Boj, Eva <evaboj@ub.edu>, Caballe, Adria <adria.caballe@upc.edu>, Delicado, Pedro <pedro.delicado@upc.edu> and Fortiana, Josep <fortiana@ub.edu>
dbglm

See Also

GtoD2
disttod2
D2toDist

Examples

X <- matrix(rnorm(100*3),nrow=100)
D2 <- as.matrix(dist(X)**2)
class(D2) <- "D2"
G <- D2toG(D2,weights=NULL)

---

dbglm

Distance-based generalized linear models

Description

dbglm is a variety of generalized linear model where explanatory information is coded as distances between individuals. These distances can either be computed from observed explanatory variables or directly input as a squared distances matrix.

Response and link function as in the glm function for ordinary generalized linear models.

Notation convention: in distance-based methods we must distinguish observed explanatory variables which we denote by Z or z, from Euclidean coordinates which we denote by X or x. For explanation on the meaning of both terms see the bibliography references below.

Usage

## S3 method for class 'formula'

```
dbglm(formula, data, family=gaussian, method="GCV", full.search=TRUE,...,
metric="euclidean", weights, maxiter=100, eps1=1e-10,
eps2=1e-10, rel.gvar=0.95, eff.rank=NULL, offset, mustart=NULL, range.eff.rank)
```

## S3 method for class 'dist'

```
dbglm(distance, y, family=gaussian, method="GCV", full.search=TRUE, weights,
maxiter=100, eps1=1e-10, eps2=1e-10, rel.gvar=0.95, eff.rank=NULL,
offset, mustart=NULL, range.eff.rank,...)
```

## S3 method for class 'D2'

```
dbglm(D2, y,..., family=gaussian, method="GCV", full.search=TRUE, weights,maxiter=100,
eps1=1e-10, eps2=1e-10, rel.gvar=0.95, eff.rank=NULL, offset,
mustart=NULL, range.eff.rank)
```

## S3 method for class 'Gram'

```
\texttt{dbglm(G, y, ..., family=gaussian, method ="GCV", full.search=TRUE, weights,maxiter=100, eps1=1e-10, eps2=1e-10, rel.gvar=0.95, eff.rank=NULL, offset, mustart=NULL, range.eff.rank)}

**Arguments**

- **formula**: an object of class \texttt{formula}. A formula of the form \(y \sim Z\). This argument is a remnant of the \texttt{glm} function, kept for compatibility.
- **data**: an optional data frame containing the variables in the model (both response and explanatory variables, either the observed ones, \(Z\), or a Euclidean configuration \(X\)).
- **y**: (required if no formula is given as the principal argument). Response (dependent variable) must be numeric, factor, matrix or data.frame.
- **distance**: a \texttt{dist} or dissimilarity class object. See functions \texttt{dist} in the package \texttt{stats} and \texttt{daisy} in the package \texttt{cluster}.
- **D2**: a \texttt{D2} class object. Squared distances matrix between individuals. See the Details section in \texttt{dblm} to learn the usage.
- **G**: a \texttt{Gram} class object. Doubly centered inner product matrix of the squared distances matrix \(D2\). See details in \texttt{dblm}.
- **family**: a description of the error distribution and link function to be used in the model. This can be a character string naming a family function, a family function or the result of a call to a family function. (See \texttt{family} for details of family functions.)
- **metric**: metric function to be used when computing distances from observed explanatory variables. One of "euclidean" (the default), "manhattan", or "gower".
- **weights**: an optional numeric vector of prior weights to be used in the fitting process. By default all individuals have the same weight.
- **method**: sets the method to be used in deciding the effective rank, which is defined as the number of linearly independent Euclidean coordinates used in prediction. There are five different methods: "AIC", "BIC", "GCV" (default), "eff.rank" and "rel.gvar". GCV take the effective rank minimizing a cross-validatory quantity. AIC and BIC take the effective rank minimizing, respectively, the Akaike or Bayesian Information Criterion (see \texttt{AIC} for more details).
- **full.search**: sets which optimization procedure will be used to minimize the modelling criterion specified in method. Needs to be specified only if method is "AIC", "BIC" or "GCV". If full.search=TRUE, effective rank is set to its global best value, after evaluating the criterion for all possible ranks. Potentially too computation-ally expensive. If full.search=FALSE, the \texttt{optimize} function is called. Then computation time is shorter, but the result may be found a local minimum.
- **maxiter**: maximum number of iterations in the iterated \texttt{dblm} algorithm. (Default = 100)
- **eps1**: stopping criterion 1, "DevStat": convergence tolerance \(\text{eps1}\), a positive (small) number; the iterations converge when \(|\text{dev} - \text{dev}_{\text{old}}|/|\text{dev}| < \text{eps1}\). Stationarity of deviance has been attained.
- **eps2**: stopping criterion 2, "mustat": convergence tolerance \(\text{eps2}\), a positive (small) number; the iterations converge when \(|\text{mu} - \text{mu}_{\text{old}}|/|\text{mu}| < \text{eps2}\). Stationarity of fitted.values \(\text{mu}\) has been attained.
relative geometric variability (a real number between 0 and 1). In each dBglm iteration, take the lowest effective rank, with a relative geometric variability higher or equal to rel.gvar. Default value (rel.gvar=0.95) uses the 95% of the total variability.

eff.rank integer between 1 and the number of observations minus one. Number of Euclidean coordinates used for model fitting in each dBlm iteration. If specified its value overrides rel.gvar. When eff.rank=NULL (default), calls to dBlm are made with method=rel.gvar.

offset this can be used to specify an a priori known component to be included in the linear predictor during fitting. This should be NULL or a numeric vector of length equal to the number of cases.
mustart starting values for the vector of means.
range.eff.rank vector of size two defining the range of values for the effective rank with which the dBlm iterations will be evaluated (must be specified when method is "AIC", "BIC" or "GCV"). The range should be restrict between c(1, n-1).

... arguments passed to or from other methods to the low level.

Details

The various possible ways for inputting the model explanatory information through distances, or their squares, etc., are the same as in dBlm.

For gamma distributions, the domain of the canonical link function is not the same as the permitted range of the mean. In particular, the linear predictor might be negative, obtaining an impossible negative mean. Should that event occur, dBglm stops with an error message. Proposed alternative is to use a non-canonical link function.

Value

A list of class dBglm containing the following components:

residuals the working residuals, that is the dBlm residuals in the last iteration of dBlm fit.
fitted.values the fitted mean values, results of final dBlm iteration.
family the family object used.
deviance measure of discrepancy or badness of fit. Proportional to twice the difference between the maximum achievable log-likelihood and that achieved by the current model.
aic.model a version of Akaike's Information Criterion. Equal to minus twice the maximized log-likelihood plus twice the number of parameters. Computed by the aic component of the family. For binomial and Poisson families the dispersion is fixed at one and the number of parameters is the number of coefficients. For gaussian, Gamma and inverse gaussian families the dispersion is estimated from the residual deviance, and the number of parameters is the number of coefficients plus one. For a gaussian family the MLE of the dispersion is used so this is a valid value of AIC, but for Gamma and inverse gaussian families it is not. For families fitted by quasi-likelihood the value is NA.
bic.model a version of the Bayesian Information Criterion. Equal to minus twice the maximized log-likelihood plus the logarithm of the number of observations by the number of parameters (see, e.g., Wood 2006).

null.deviance the deviance for the null model. The null model will include the offset, and an intercept if there is one in the model. Note that this will be incorrect if the link function depends on the data other than through the fitted mean: specify a zero offset to force a correct calculation.

iter number of Fisher scoring (dblm) iterations.
prior.weights the original weights.
weights the working weights, that are the weights in the last iteration of dblm fit.
df.residual the residual degrees of freedom.
df.null the residual degrees of freedom for the null model.
y the response vector used.
convcrit convergence criterion. One of: "DevStat" (stopping criterion 1), "muStat" (stopping criterion 2), "maxiter" (maximum allowed number of iterations has been exceeded).

H hat matrix projector of the last dblm iteration.
rel.gvar the relative geometric variability in the last dblm iteration.
eff.rank the working effective rank, that is the eff.rank in the last dblm iteration.
varmu vector of estimated variance of each observation.
dev.resids deviance residuals
call the matched call.

Objects of class "dbglm" are actually of class c("dbglm", "dblm"), inheriting the plot.dblm method from class "dblm".

Note

When the Euclidean distance is used the dbglm model reduces to the generalized linear model (glm).

Author(s)

Boj, Eva <evaboj@ub.edu>, Caballe, Adria <adria.caballe@upc.edu>, Delicado, Pedro <pedro.delicado@upc.edu> and Fortiana, Josep <fortiana@ub.edu>

References


See Also

summary.dbglm for summary.
plot.dbglm for plots.
predict.dbglm for predictions.
dblm for distance-based linear models.

Examples

```r
## CASE POISSON
z <- rnorm(100)
y <- rpois(100, exp(1+z))
glm1 <- glm(y ~ z, family = poisson(link = "log"))
D2 <- as.matrix(dist(z))^2
class(D2) <- "D2"
dbglm1 <- dbglm(D2, y, family = poisson(link = "log"), method="rel.gvar")

plot(z,y)
points(z,glm1$fitted.values,col=2)
points(z,dbglm1$fitted.values,col=3)
sum((glm1$fitted.values-y)^2)
sum((dbglm1$fitted.values-y)^2)

## CASE BINOMIAL
y <- rbinom(100, 1, plogis(z))
# needs to set a starting value for the next fit
glm2 <- glm(y ~ z, family = binomial(link = "logit"))
D2 <- as.matrix(dist(z))^2
class(D2) <- "D2"
dbglm2 <- dbglm(D2, y, family = binomial(link = "logit"), method="rel.gvar")

plot(z,y)
points(z,glm2$fitted.values,col=2)
points(z,dbglm2$fitted.values,col=3)
sum((glm2$fitted.values-y)^2)
sum((dbglm2$fitted.values-y)^2)
```
**dblm**

Distance-based linear model

**Description**

dblm is a variety of linear model where explanatory information is coded as distances between individuals. These distances can either be computed from observed explanatory variables or directly input as a squared distances matrix. The response is a continuous variable as in the ordinary linear model. Since distances can be computed from a mixture of continuous and qualitative explanatory variables or, in fact, from more general quantities, dblm is a proper extension of lm.

Notation convention: in distance-based methods we must distinguish *observed explanatory variables* which we denote by Z or z, from *Euclidean coordinates* which we denote by X or x. For explanation on the meaning of both terms see the bibliography references below.

**Usage**

```r
## S3 method for class 'formula'
dblm(formula,data,...,metric="euclidean",method="OCV",full.search=TRUE, weights,rel.gvar=0.95,eff.rank)

## S3 method for class 'dist'
dblm(distance,y,...,method="OCV",full.search=TRUE, weights,rel.gvar=0.95,eff.rank)

## S3 method for class 'D2'
dblm(D2,y,...,method="OCV",full.search=TRUE,weights,rel.gvar=0.95, eff.rank)

## S3 method for class 'Gram'
dblm(G,y,...,method="OCV",full.search=TRUE,weights,rel.gvar=0.95, eff.rank)
```

**Arguments**

- `formula` an object of class `formula`. A formula of the form `y~Z`. This argument is a remnant of the `lm` function, kept for compatibility.
- `data` an optional data frame containing the variables in the model (both response and explanatory variables, either the observed ones, Z, or a Euclidean configuration X).
- `y` (required if no formula is given as the principal argument). Response (dependent variable) must be numeric, matrix or data.frame.
- `distance` a `dist` or dissimilarity class object. See functions `dist` in the package `stats` and `daisy` in the package `cluster`.
- `D2` a D2 class object. Squared distances matrix between individuals.
G
metric
method
full.search
weights
rel.gvar
eff.rank

a Gram class object. Doubly centered inner product matrix of the squared distances matrix D2.

metric function to be used when computing distances from observed explanatory variables. One of "euclidean" (default), "manhattan", or "gower".

sets the method to be used in deciding the effective rank, which is defined as the number of linearly independent Euclidean coordinates used in prediction. There are six different methods: "AIC", "BIC", "OCV" (default), "GCV", "eff.rank" and "rel.gvar". OCV and GCV take the effective rank minimizing a cross-validatory quantity (either ocv or gcv). AIC and BIC take the effective rank minimizing, respectively, the Akaike or Bayesian Information Criterion (see AIC for more details). The optimizacion procedure to be used in the above four methods can be set with the full.search optional parameter.

When method is eff.rank, the effective rank is explicitly set by the user through the eff.rank optional parameter which, in this case, becomes mandatory.

When method is rel.gvar, the fraction of the data geometric variability for model fitting is explicitly set by the user through the rel.gvar optional parameter which, in this case, becomes mandatory.

sets which optimization procedure will be used to minimize the modelling criterion specified in method. Needs to be specified only if method is "AIC", "BIC", "OCV" or "GCV". If full.search=TRUE, effective rank is set to its global best value, after evaluating the criterion for all possible ranks. Potentially too computationally expensive. If full.search=FALSE, the optimize function is called. Then computation time is shorter, but the result may be found a local minimum.

an optional numeric vector of weights to be used in the fitting process. By default all individuals have the same weight.

relative geometric variability (real between 0 and 1). Take the lowest effective rank with a relative geometric variability higher or equal to rel.gvar. Default value (rel.gvar=0.95) uses a 95% of the total variability. Applies only rel.gvar if method="rel.gvar".

integer between 1 and the number of observations minus one. Number of Euclidean coordinates used for model fitting. Applies only if method="eff.rank".

arguments passed to or from other methods to the low level.

Details

The dblm model uses the distance matrix between individuals to find an appropriate prediction method. There are many ways to compute and calculate this matrix, besides the three included as parameters in this function. Several packages in R also study this problem. In particular dist in the package stats and daisy in the package cluster (the three metrics in dblm call the daisy function).

Another way to enter a distance matrix to the model is through an object of class "D2" (containing the squared distances matrix). An object of class "dist" or "dissimilarity" can easily be transformed into one of class "D2". See disttoD2. Reciprocally, an object of class "D2" can be transformed into one of class "dist". See D2toDist.

S3 method Gram uses the Doubly centered inner product matrix G=XX'. Its also easily to transformed into one of class "D2". See D2toG and GtoD2.
The weights array is adequate when responses for different individuals have different variances. In this case the weights array should be (proportional to) the reciprocal of the variances vector.

When using method `method = "eff.rank"` or `method = "rel.gvar"`, a compromise between possible consequences of a bad choice has to be reached. If the rank is too large, the model can be overfitted, possibly leading to an increased prediction error for new cases (even though R2 is higher). On the other hand, a small rank suggests a model inadequacy (R2 is small). The other four methods are less error prone (but still they do not guarantee good predictions).

**Value**

A list of class `dblm` containing the following components:

- `residuals`  the residuals (response minus fitted values).
- `fitted.values`  the fitted mean values.
- `df.residuals`  the residual degrees of freedom.
- `weights`  the specified weights.
- `y`  the response used to fit the model.
- `H`  the hat matrix projector.
- `call`  the matched call.
- `rel.gvar`  the relative geometric variability, used to fit the model.
- `eff.rank`  the dimensions chosen to estimate the model.
- `ocv`  the ordinary cross-validation estimate of the prediction error.
- `gcv`  the generalized cross-validation estimate of the prediction error.
- `aic`  the Akaike Value Criterium of the model (only if `method = "AIC"`).
- `bic`  the Bayesian Value Criterium of the model (only if `method = "BIC"`).

**Note**

When the Euclidean distance is used the `dblm` model reduces to the linear model (`lm`).

**Author(s)**

Boj, Eva <evaboj@ub.edu>, Caballe, Adria <adria.caballe@upc.edu>, Delicado, Pedro <pedro.delicado@upc.edu> and Fortiana, Josep <fortiana@ub.edu>

**References**


**See Also**

- `summary.dbplsr` for summary.
- `plot.dbplsr` for plots.
- `predict.dbplsr` for predictions.
- `ldblm` for distance-based local linear models.

**Examples**

```r
# easy example to illustrate usage of the dblm function
n <- 100
p <- 3
k <- 5

Z <- matrix(rnorm(n*p),nrow=n)
b <- matrix(runif(p)*k,nrow=p)
s <- 1
e <- rnorm(n)*s
y <- Z*b + e
D<-dist(Z)

dblm1 <- dblm(D,y)
lm1 <- lm(y~Z)
# the same fitted values with the lm
mean(lm1$fitted.values-dblm1$fitted.values)
```

---

**dbplsr**

*Distance-based partial least squares regression*

**Description**

dbplsr is a variety of partial least squares regression where explanatory information is coded as distances between individuals. These distances can either be computed from observed explanatory variables or directly input as a squared distances matrix.

Since distances can be computed from a mixture of continuous and qualitative explanatory variables or, in fact, from more general quantities, dbplsr is a proper extension of plsr.

Notation convention: in distance-based methods we must distinguish *observed explanatory variables* which we denote by Z or z, from *Euclidean coordinates* which we denote by X or x. For explanation on the meaning of both terms see the bibliography references below.
Usage

```r
# S3 method for class 'formula'
dbplsr(formula, data, ..., metric = "euclidean",
       method = "ncomp", weights, ncomp)
```

```r
# S3 method for class 'dist'
dbplsr(distance, y, ..., weights, ncomp = ncomp, method = "ncomp")
```

```r
# S3 method for class 'D2'
dbplsr(D2, y, ..., weights, ncomp = ncomp, method = "ncomp")
```

```r
# S3 method for class 'Gram'
dbplsr(G, y, ..., weights, ncomp = ncomp, method = "ncomp")
```

Arguments

- **formula**: an object of class `formula`. A formula of the form `y~Z`. This argument is a remnant of the `plsr` function, kept for compatibility.
- **data**: an optional data frame containing the variables in the model (both response and explanatory variables, either the observed ones, Z, or a Euclidean configuration X).
- **y**: (required if no formula is given as the principal argument). Response (dependent variable) must be numeric, matrix or data.frame.
- **distance**: a `dist` or dissimilarity class object. See functions `dist` in the package `stats` and `daisy` in the package `cluster`.
- **D2**: a `D2` class object. Squared distances matrix between individuals.
- **G**: a `Gram` class object. Weighted centered inner products matrix of the squared distances matrix `D2`. See details in `dblm`.
- **metric**: metric function to be used when computing distances from observed explanatory variables. One of "euclidean" (default), "manhattan", or "gower".
- **method**: sets the method to be used in deciding how many components needed to fit the best model for new predictions. There are five different methods, "AIC", "BIC", "OCV", "GCV" and "ncomp" (default). OCV and GCV find the number of components that minimizes the Cross-validation coefficient (ocv or gcv). AIC and BIC find the number of components that minimizes the Akaike or Bayesian Information Criterion (see AIC for more details).
- **weights**: an optional numeric vector of weights to be used in the fitting process. By default all individuals have the same weight.
- **ncomp**: the number of components to include in the model.
- **...**: arguments passed to or from other methods to the low level.

Details

Partial least squares (PLS) is a method for constructing predictive models when the factors (Z) are many and highly collinear. A PLS model will try to find the multidimensional direction in the Z
space that explains the maximum multidimensional variance direction in the Y space. `dbplsr` is particularly suited when the matrix of predictors has more variables than observations. By contrast, standard regression (`dlm`) will fail in these cases.

The various possible ways for inputting the model explanatory information through distances, or their squares, etc., are the same as in `dlm`.

The number of components to fit is specified with the argument `ncomp`.

**Value**

A list of class `dbplsr` containing the following components:

- `residuals`: a list containing the residuals (response minus fitted values) for each iteration.
- `fitted.values`: a list containing the fitted values for each iteration.
- `fk`: a list containing the scores for each iteration.
- `bk`: regression coefficients. `fitted.values = fk*bk`
- `Pk`: orthogonal projector on the one-dimensional linear space by `fk`.
- `ncomp`: number of components included in the model.
- `ncomp.opt`: optimum number of components according to the selected method.
- `weights`: the specified weights.
- `method`: the using method.
- `y`: the response used to fit the model.
- `H`: the hat matrix projector.
- `G0`: initial weighted centered inner products matrix of the squared distance matrix.
- `Gk`: weighted centered inner products matrix in last iteration.
- `gvar`: total weighted geometric variability.
- `gvec`: the diagonal entries in `G0`.
- `gvar.iter`: geometric variability for each iteration.
- `ocv`: the ordinary cross-validation estimate of the prediction error.
- `gcv`: the generalized cross-validation estimate of the prediction error.
- `aic`: the Akaike Value Criterion of the model.
- `bic`: the Bayesian Value Criterion of the model.

**Note**

When the Euclidean distance is used the `dbplsr` model reduces to the traditional partial least squares (`plsr`).

**Author(s)**

Boj, Eva <evaboj@ub.edu>, Caballe, Adria <adria.caballe@upc.edu>, Delicado, Pedro <pedro.delicado@upc.edu> and Fortiana, Josep <fortiana@ub.edu>
References


See Also

summary.dbplsr for summary.
plot.dbplsr for plots.
predict.dbplsr for predictions.

Examples

```r
# require(pls)
library(pls)
data(yarn)
## Default methods:
yarn.dbplsr <- dbplsr(density ~ NIR, data = yarn, ncomp=6, method="GCV")
```

### disttoD2

*Distance conversion: dist to D2*

Description

Converts dist or dissimilarity class object into D2 class object.

Usage

```r
disttoD2(distance)
```

Arguments

distance dist or dissimilarity class object. See functions `dist` in the package stats and `daisy` in the package cluster.
Value

An object of class D2 containing the squared distances matrix between individuals.

Author(s)

Boj, Eva <evaboj@ub.edu>, Caballe, Adria <adria.caballe@upc.edu>, Delicado, Pedro <pedro.delicado@upc.edu> and Fortiana, Josep <fortiana@ub.edu>

See Also

GtoD2
D2toG
D2toDist

Examples

```r
X <- matrix(rnorm(100*3),nrow=100)
distance <- daisy(X,"manhattan")
D2 <- disttoD2(distance)
```

---

**GtoD2**

*Distance conversion: dist to D2*

Description

Converts Gram class object into D2 class object

Usage

```r
GtoD2(G)
```

Arguments

- `G` Gram class object. Weighted centered inner products matrix of the squared distances matrix.

Value

An object of class D2 containing the squared distances matrix between individuals.

Author(s)

Boj, Eva <evaboj@ub.edu>, Caballe, Adria <adria.caballe@upc.edu>, Delicado, Pedro <pedro.delicado@upc.edu> and Fortiana, Josep <fortiana@ub.edu>
See Also

D2toG
disttoD2
D2toDist

Examples

```r
X <- matrix(rnorm(100*3),nrow=100)
D2 <- as.matrix(dist(X)^2)
class(D2) <- "D2"
G <- D2toG(D2,weights=NULL)
class(G) <- "Gram"
D22 <- GtoD2(G)
```

**ldbglm**

*Local distance-based generalized linear model*

Description

ldbglm is a localized version of a distance-based generalized linear model. As in the global model dbglm, explanatory information is coded as distances between individuals.

Neighborhood definition for localizing is done by the (semi)metric dist1 whereas a second (semi)metric dist2 (which may coincide with dist1) is used for distance-based prediction. Both dist1 and dist2 can either be computed from observed explanatory variables or directly input as a squared distances matrix or as a Gram matrix. Response and link function are as in the dbglm function for ordinary generalized linear models. The model allows for a mixture of continuous and qualitative explanatory variables or, in fact, from more general quantities such as functional data.

Notation convention: in distance-based methods we must distinguish observed explanatory variables which we denote by Z or z, from Euclidean coordinates which we denote by X or x. For explanation on the meaning of both terms see the bibliography references below.

Usage

```r
## S3 method for class 'formula'
ldbglm(formula,data,...,family=gaussian(),kind.of.kernel=1,
   metric1="euclidean",metric2=metric1,method.h="GCV",weights,
   user.h=NULL,h.range=NULL,oh=10,k.knn=3,
   rel.gvar=0.95,eff.rank=NULL,maxiter=100,eps1=1e-10,
   eps2=1e-10)

## S3 method for class 'dist'
ldbglm(dist1,dist2=dist1,y,family=gaussian(),kind.of.kernel=1,
   method.h="GCV",weights,user.h=quantile(dist1,.25),
```
Arguments

formula an object of class \texttt{formula}. A formula of the form \texttt{y~z}. This argument is a remnant of the \texttt{loess} function, kept for compatibility.

data an optional data frame containing the variables in the model (both response and explanatory variables, either the observed ones, \texttt{Z}, or a Euclidean configuration \texttt{X}).

y (required if no formula is given as the principal argument). Response (dependent variable) must be numeric, matrix or data.frame.

dist1 a \texttt{dist} or dissimilarity class object. Distances between observations, used for neighborhood localizing definition. Weights for observations are computed as a decreasing function of their \texttt{dist1} distances to the neighborhood center, e.g. a new observation whose response has to be predicted. These weights are then entered to a \texttt{dbglm}, where distances are evaluated with \texttt{dist2}.

dist2 a \texttt{dist} or dissimilarity class object. Distances between observations, used for fitting \texttt{dbglm}. Default \texttt{dist2=dist1}.

D2.1 a \texttt{D2} class object. Squared distances matrix between individuals. One of the alternative ways of entering distance information to a function. See the Details section in \texttt{dblm}. See above \texttt{dist1} for explanation of its role in this function.

D2.2 a \texttt{D2} class object. Squared distances between observations. One of the alternative ways of entering distance information to a function. See the Details section in \texttt{dblm}. See above \texttt{dist2} for explanation of its role in this function. Default \texttt{D2.2=D2.1}.

G1 a \texttt{Gram} class object. Doubly centered inner product matrix associated with the squared distances matrix \texttt{D2.1}.

G2 a \texttt{Gram} class object. Doubly centered inner product matrix associated with the squared distances matrix \texttt{D2.2}. Default \texttt{G2=G1}

family a description of the error distribution and link function to be used in the model. This can be a character string naming a family function, a family function or the result of a call to a family function. (See \texttt{family} for details of family functions.)
kind.of.kernel: integer number between 1 and 6 which determines the user's choice of smoothing kernel. (1) Epanechnikov (Default), (2) Biweight, (3) Triweight, (4) Normal, (5) Triangular, (6) Uniform.

metric1: metric function to be used when computing dist1 from observed explanatory variables. One of "euclidean" (default), "manhattan", or "gower".

metric2: metric function to be used when computing dist2 from observed explanatory variables. One of "euclidean" (default), "manhattan", or "gower".

method.h: sets the method to be used in deciding the optimal bandwidth h. There are four different methods, AIC, BIC, GCV (default) and user.h. GCV take the optimal bandwidth minimizing a cross-validatory quantity. AIC and BIC take the optimal bandwidth minimizing, respectively, the Akaike or Bayesian Information Criterion (see AIC for more details). When method.h is user.h, the bandwidth is explicitly set by the user through the user.h optional parameter which, in this case, becomes mandatory.

weights: an optional numeric vector of weights to be used in the fitting process. By default all individuals have the same weight.

user.h: global bandwidth user.h, set by the user, controlling the size of the local neighborhood of Z. Smoothing parameter (Default: 1st quartile of all the distances d(i,j) in dist1). Applies only if method.h="user.h".

h.range: a vector of length 2 giving the range for automatic bandwidth choice. (Default: quantiles 0.05 and 0.5 of d(i,j) in dist1).

noh: number of bandwidth h values within h.range for automatic bandwidth choice (if method.h!="user.h").

k.knn: minimum number of observations with positive weight in neighborhood localizing. To avoid runtime errors due to a too small bandwidth originating neighborhoods with only one observation. By default k.nn=3.

rel.gvar: relative geometric variability (a real number between 0 and 1). In each dblm iteration, take the lowest effective rank, with a relative geometric variability higher or equal to rel.gvar. Default value (rel.gvar=0.95) uses the 95% of the total variability.

eff.rank: integer between 1 and the number of observations minus one. Number of Euclidean coordinates used for model fitting in each dblm iteration. If specified its value overrides rel.gvar. When eff.rank=NULL (default), calls to dblm are made with method=rel.gvar.

maxiter: maximum number of iterations in the iterated dblm algorithm. (Default = 100)

eps1: stopping criterion 1, "DevStat": convergence tolerance eps1, a positive (small) number; the iterations converge when |dev - dev_{old}|/(|dev|) < eps1. Stationarity of deviance has been attained.

eps2: stopping criterion 2, "mustat": convergence tolerance eps2, a positive (small) number; the iterations converge when |mu - mu_{old}|/(|mu|) < eps2. Stationarity of fitted.values mu has been attained.

... arguments passed to or from other methods to the low level.
Details

The various possible ways for inputting the model explanatory information through distances, or their squares, etc., are the same as in `dblm`.

The set of bandwidth $h$ values checked in automatic bandwidth choice is defined by $h$.range and $noh$, together with $k$.knn. For each $h$ in it a local generalized linear model is fitted and the optimal $h$ is decided according to the statistic specified in method $h$.

kind.of.kernel designates which kernel function is to be used in determining individual weights from dist1 values. See density for more information.

For gamma distributions, the domain of the canonical link function is not the same as the permitted range of the mean. In particular, the linear predictor might be negative, obtaining an impossible negative mean. Should that event occur, `dbglm` stops with an error message. Proposed alternative is to use a non-canonical link function.

Value

A list of class `1dbglm` containing the following components:

- residuals: the residuals (response minus fitted values).
- fitted.values: the fitted mean values.
- h.opt: the optimal bandwidth $h$ used in the fitting process (if method $h!=$user $h$).
- family: the family object used.
- y: the response variable used.
- S: the Smoother hat projector.
- weights: the specified weights.
- call: the matched call.
- dist1: the distance matrix (object of class "D2" or "dist") used to calculate the weights of the observations.
- dist2: the distance matrix (object of class "D2" or "dist") used to fit the `dbglm`.

Objects of class "1dbglm" are actually of class c("1dbglm", "1dblm"), inheriting the `plot.1dblm` and `summary.1dblm` method from class "1dblm".

Note

Model fitting is repeated $n$ times ($n=\text{number of observations}$) for each bandwidth ($noh \times n$ times). For a $noh$ too large or a sample with many observations, the time of this function can be very high.

Author(s)

Boj, Eva <evaboju@ub.edu>, Caballe, Adria <adria.caballe@upc.edu>, Delicado, Pedro <pedro.delicado@upc.edu> and Fortiana, Josep <fortiana@ub.edu>
References


See Also

dbglm for distance-based generalized linear models.
ldblm for local distance-based linear models.
summary.ldbglm for summary.
plot.ldbglm for plots.
predict.ldbglm for predictions.

Examples

```r
# example of ldbglm usage
z <- rnorm(100)
y <- rbinom(100, 1, plogis(z))
D2 <- as.matrix(dist(z))^2
class(D2) <- "D2"

# Distance-based generalized linear model
$dbglm2 <- dbglm(D2, y, family=binomial(link = "logit"), method="rel.gvar")

# Local Distance-based generalized linear model
ldbglm2 <- ldbglm(D2, y, family=binomial(link = "logit"), noh=3)

# check the difference of both
sum((y-ldbglm2$fit)^2)
sum((y-$dbglm2$fit)^2)
plot(z, y)
points(z, ldbglm2$fit, col=3)
points(z, $dbglm2$fit, col=2)
```
ldblm

Local distance-based linear model

Description

ldblm is a localized version of a distance-based linear model. As in the global model dblm, explanatory information is coded as distances between individuals.

Neighborhood definition for localizing is done by the (semi)metric dist1 whereas a second (semi)metric dist2 (which may coincide with dist1) is used for distance-based prediction. Both dist1 and dist2 can either be computed from observed explanatory variables or directly input as a squared distances matrix or as a Gram matrix. The response is a continuous variable as in the ordinary linear model. The model allows for a mixture of continuous and qualitative explanatory variables or, in fact, from more general quantities such as functional data.

Notation convention: in distance-based methods we must distinguish observed explanatory variables which we denote by Z or z, from Euclidean coordinates which we denote by X or x. For explanation on the meaning of both terms see the bibliography references below.

Usage

```r
## S3 method for class 'formula'
ldblm(formula, data,..., kind.of.kernel=1,
      metric1="euclidean", metric2=metric1, method.h="GCV", weights,
      user.h=NULL, h.range=NULL, noh=10, k.knn=3, rel.gvar=0.95, eff.rank=NULL)

## S3 method for class 'dist'
ldblm(dist1, dist2=dist1, y, kind.of.kernel=1,
      method.h="GCV", weights, user.h=quantile(dist1,.25),
      h.range=quantile(as.matrix(dist1),c(.05,.5)), noh=10,
      k.knn=3, rel.gvar=0.95, eff.rank=NULL,...)

## S3 method for class 'D2'
ldblm(D2.1, D2.2=D2.1, y, kind.of.kernel=1, method.h="GCV",
      weights, user.h=quantile(D2.1,.25)^.5,
      h.range=quantile(as.matrix(D2.1),c(.05,.5))^ .5, noh=10, k.knn=3,
      rel.gvar=0.95, eff.rank=NULL,...)

## S3 method for class 'Gram'
ldblm(G1, G2=G1, y, kind.of.kernel=1, method.h="GCV",
      weights, user.h=NULL, h.range=NULL, noh=10, k.knn=3, rel.gvar=0.95,
      eff.rank=NULL,...)
```

Arguments

- `formula` an object of class `formula`. A formula of the form `y~Z`. This argument is a remnant of the `loess` function, kept for compatibility.
data an optional data frame containing the variables in the model (both response and explanatory variables, either the observed ones, Z, or a Euclidean configuration X).

y (required if no formula is given as the principal argument). Response (dependent variable) must be numeric, matrix or data.frame.

dist1 a dist or dissimilarity class object. Distances between observations, used for neighborhood localizing definition. Weights for observations are computed as a decreasing function of their dist1 distances to the neighborhood center, e.g. a new observation whose response has to be predicted. These weights are then entered to a dblm, where distances are evaluated with dist1.

dist2 a dist or dissimilarity class object. Distances between observations, used for fitting dblm. Default dist2=dist1.

D2.1 a D2 class object. Squared distances matrix between individuals. One of the alternative ways of entering distance information to a function. See the Details section in dblm. See above dist1 for explanation of its role in this function.

D2.2 a D2 class object. Squared distances between observations. One of the alternative ways of entering distance information to a function. See the Details section in dblm. See above dist2 for explanation of its role in this function. Default D2.2=D2.1.

G1 a Gram class object. Doubly centered inner product matrix associated with the squared distances matrix D2.1.

G2 a Gram class object. Doubly centered inner product matrix associated with the squared distances matrix D2.2. Default G2=G1

kind.of.kernel integer number between 1 and 6 which determines the user’s choice of smoothing kernel. (1) Epanechnikov (Default), (2) Biweight, (3) Triweight, (4) Normal, (5) Triangular, (6) Uniform.

metric1 metric function to be used when computing dist1 from observed explanatory variables. One of "euclidean" (default), "manhattan", or "gower".

metric2 metric function to be used when computing dist2 from observed explanatory variables. One of "euclidean" (default), "manhattan", or "gower".

method.h sets the method to be used in deciding the optimal bandwidth h. There are five different methods, AIC, BIC, OCV, GCV (default) and user.h. OCV and GCV take the optimal bandwidth minimizing a cross-validatory quantity (either ocv or gcv). AIC and BIC take the optimal bandwidth minimizing, respectively, the Akaike or Bayesian Information Criterion (see AIC for more details). When method.h is user.h, the bandwidth is explicitly set by the user through the user.h optional parameter which, in this case, becomes mandatory.

weights an optional numeric vector of weights to be used in the fitting process. By default all individuals have the same weight.

user.h global bandwidth user.h, set by the user, controlling the size of the local neighborhood of Z. Smoothing parameter (Default: 1st quartile of all the distances d(i,j) in dist1). Applies only if method.h="user.h".

h.range a vector of length 2 giving the range for automatic bandwidth choice. (Default: quantiles 0.05 and 0.5 of d(i,j) in dist1).
ldblm

noh number of bandwidth h values within h.range for automatic bandwidth choice (if method.h!="user.h").

k.knn minimum number of observations with positive weight in neighborhood localizing. To avoid runtime errors due to a too small bandwidth originating neighborhoods with only one observation. By default k.nn=3.

rel.gvar relative geometric variability (a real number between 0 and 1). In each dlbm iteration, take the lowest effective rank, with a relative geometric variability higher or equal to rel.gvar. Default value (rel.gvar=0.95) uses the 95% of the total variability.

eff.rank integer between 1 and the number of observations minus one. Number of Euclidean coordinates used for model fitting in each dlbm iteration. If specified its value overrides rel.gvar. When eff.rank=NULL (default), calls to dlbm are made with method=rel.gvar.

... arguments passed to or from other methods to the low level.

Details

There are two semi-metrics involved in local linear distance-based estimation: dist1 and dist2. Both semi-metrics can coincide. For instance, when dist1=||x_i-x_j|| and dist2=||(x_i,x_i^2,x_i^3)-(x_j,x_j^2,x_j^3)|| the estimator for new observations coincides with fitting a local cubic polynomial regression.

The set of bandwidth h values checked in automatic bandwidth choice is defined by h.range and noh, together with k.knn. For each h in it a local linear model is fitted and the optimal h is decided according to the statistic specified in method.h.

kind.of.kernel designates which kernel function is to be used in determining individual weights from dist1 values. See density for more information.

Value

A list of class dlbm containing the following components:

residuals the residuals (response minus fitted values).
fitted.values the fitted mean values.
h.opt the optimal bandwidth h used in the fitting process (if method.h!="user.h").
S the Smoother hat projector.
weights the specified weights.
y the response variable used.
call the matched call.
dist1 the distance matrix (object of class "D2" or "dist") used to calculate the weights of the observations.
dist2 the distance matrix (object of class "D2" or "dist") used to fit the dlbm.

Note

Model fitting is repeated n times (n= number of observations) for each bandwidth (noh*n times). For a noh too large or a sample with many observations, the time of this function can be very high.
ldblm

Author(s)
Boj, Eva <evaboj@ub.edu>, Caballe, Adria <adria.caballe@upc.edu>, Delicado, Pedro <pedro.delicado@upc.edu> and Fortiana, Josep <fortiana@ub.edu>

References

See Also
dblm for distance-based linear models.
ldbg1m for local distance-based generalized linear models.
summary.ldblm for summary.
plot.ldblml for plots.
predict.ldblml for predictions.

Examples

# example to use of the ldblm function
n <- 100
p <- 1
k <- 5

Z <- matrix(rnorm(n*p),nrow=n)
b1 <- matrix(runif(p)*k,nrow=p)
b2 <- matrix(runif(p)*k,nrow=p)
b3 <- matrix(runif(p)*k,nrow=p)
s <- 1
e <- rnorm(n)*s

y <- Z*b1 + Z^2*b2 + Z^3*b3 + e
D2 <- as.matrix(dist(Z)^2)
class(D2) <- "D2"

ldblm1 <- ldblm(y~Z,kind.of.kernel=1,method="GCV",n=3,k.knn=3)
Description

Six plots (selected by which) are available: a plot of residual vs fitted values, the Q-Q plot of normality, a Scale-Location plot of sqrt(|residuals|) against fitted values. A plot of Cook's distances versus row labels, a plot of residuals against leverages, and the optimal effective rank of "Ov", "GCV", "AIC" or "BIC" method (only if one of these four methods have been chosen in function dblm). By default, only the first three and 5 are provided.

Usage

## S3 method for class 'dblm'
plot(x, which=c(1:3, 5), id.n=3, main="",
     cook.levels = c(0.5, 1), cex.id = 0.75,
     type.pred=c("link", "response"), ...)

Arguments

- **x** an object of class dblm or dbglm.
- **which** if a subset of the plots is required, specify a subset of the numbers 1:6.
- **id.n** number of points to be labelled in each plot, starting with the most extreme.
- **main** an overall title for the plot. Only if one of the six plots is selected.
- **cook.levels** levels of Cook’s distance at which to draw contours.
- **cex.id** magnification of point labels.
- **type.pred** the type of prediction (required only for a dbglm class object). Like predict.dbglm, the default "link" is on the scale of the linear predictors; the alternative "response" is on the scale of the response variable.
- **...** other parameters to be passed through to plotting functions.

Details

The five first plots are very useful to the residual analysis and are the same that plot.lm. A plot of residuals against fitted values sees if the variance is constant. The qq-plot checks if the residuals are normal (see qqnorm). The plot between "Scale-Location" and the fitted values takes the square root of the absolute residuals in order to diminish skewness. The Cook’s distance against the row labels, measures the effect of deleting a given observation (estimate of the influence of a data point). Points with a large Cook’s distance are considered to merit closer examination in the analysis. Finally, the Residual-Leverage plot also shows the most influence points (labelled by Cook’s distance). See cooks.distance.
The last plot, allows to view the "OCV" (just for `dblm`), "GCV", "AIC" or "BIC" criterion according to the used rank in the `dblm` or `dbglm` functions, and chosen the minimum. Applies only if the parameter `full.search` its `TRUE`.

**Author(s)**

Boj, Eva <evaboj@ub.edu>, Caballe, Adria <adria.caballe@upc.edu>, Delicado, Pedro <pedro.delicado@upc.edu> and Fortiana, Josep <fortiana@ub.edu>

**References**


**See Also**

`dblm` for distance-based linear models.

`dbglm` for distance-based generalized linear models.

**Examples**

```r
n <- 64
p <- 4
k <- 3

Z <- matrix(rnorm(n*p),nrow=n)
b <- matrix(runif(p)*k,nrow=p)
s <- 1
e <- rnorm(n)*s
y <- Z*b + e
dblm1 <- dblm(y~Z,metric="gower",method="GCV", full.search=FALSE)
plot(dblm1)
plot(dblm1,which=4)
```
plot.dbplsr  

Plots for a dbplsr object

Description

Four plots (selected by which) are available: plot of scores, response vs scores, R2 contribution in each component and the value of "OCV", "GCV", "AIC" or "BIC" vs the number of component chosen.

Usage

```r
## S3 method for class 'dbplsr'
plot(x, which=c(1L:4L), main="", scores.comps=1:2,
    component=1, method=c("OCV","GCV","AIC","BIC"), ...)
```

Arguments

- `x`: an object of class dbplsr.
- `which`: if a subset of the plots is required, specify a subset of the numbers 1:4.
- `main`: an overall title for the plot. Only if one of the four plots is selected.
- `scores.comps`: array containing the component scores crossed in the first plot (default the first two).
- `component`: numeric value. Component vs response in the second plot (Default the first component).
- `method`: chosen method "OCV", "GCV", "AIC" or "BIC" in the last plot.
- `...`: other parameters to be passed through to plotting functions.

Author(s)

Boj, Eva <evaboj@ub.edu>, Caballe, Adria <adria.caballe@upc.edu>, Delicado, Pedro <pedro.delicado@upc.edu> and Fortiana, Josep <fortiana@ub.edu>

References


See Also

dbplsr for distance-based partial least squares.

Examples

```r
#require(pls)
library(pls)
data(yarn)
## Default methods:
yarn.dbplsr <- dbplsr(density ~ NIR, data = yarn, ncomp=6, method="GCV")
plot(yarn.dbplsr,scores.comps=1:3)
```

---

**plot.ldblm**

*Plots for objects of classes ldblm or ldbglm*

**Description**

Three plots (selected by `which`) are available: a plot of fitted values vs response, a plot of residuals vs fitted and the optimal bandwidth \( h \) of "OCV", "GCV", "AIC" or "BIC" criterion (only if one of these four methods have been chosen in the ldblm function). By default, only the first and the second are provided.

**Usage**

```r
## S3 method for class 'ldblm'
plot(x,which=c(1,2),id.n=3,main="",...)
```

**Arguments**

- `x` an object of class `ldblm` or `ldbglm`.
- `which` if a subset of the plots is required, specify a subset of the numbers 1:3.
- `id.n` number of points to be labelled in each plot, starting with the most extreme.
- `main` an overall title for the plot. Only if one of the three plots is selected.
- `...` other parameters to be passed through to plotting functions.

**Author(s)**

Boj, Eva <evaboj@ub.edu>, Caballe, Adria <adria.caballe@upc.edu>, Delicado, Pedro <pedro.delicado@upc.edu> and Fortiana, Josep <fortiana@ub.edu>
References


See Also

*ldblm* for local distance-based linear models.

*ldbgglm* for local distance-based generalized linear models.

Examples

```r
# example to use of the ldblml function
n <- 100
p <- 1
k <- 5

Z <- matrix(rnorm(n*p),nrow=n)
b1 <- matrix(runif(p)*k,nrow=p)
b2 <- matrix(runif(p)*k,nrow=p)
b3 <- matrix(runif(p)*k,nrow=p)
s <- 1
e <- rnorm(n)*s

y <- Z*b1 + Z^2*b2 + Z^3*b3 + e
D2 <- as.matrix(dist(Z))^2
class(D2) <- "D2"

ldblm1 <- ldblml(D2,y=y,kind.of.kernel=1,method.h="AIC",noh=5,h.knn=NULL)
plot(ldblm1)
plot(ldblm1,which=3)
```
predict.dbglm

Predicted values for a dbglm object

Description

predict.dbglm returns the predicted values, obtained by testing the generalized distance regression function in the new data (newdata).

Usage

```r
## S3 method for class 'dbglm'
predict(object, newdata, type.pred = c("link", "response"),
        type.var = "Z", ...)
```

Arguments

- `object`: an object of class `dbglm`. Result of `dbglm`.
- `newdata`: data.frame or matrix which contains the values of Z (if `type.var = "Z"`). The squared distances between k new individuals and the original n individuals (only if `type.var = "D2"`). Finally, the G inner products matrix (if `type.var = "G"`).
- `type.pred`: the type of prediction (required). The default "link" is on the scale of the linear predictors; the alternative "response" is on the scale of the response variable.
- `type.var`: set de type of newdata. Can be "Z" if newdata contains the values of the explanatory variables, "D2" if contains the squared distances matrix or "G" if contains the inner products matrix.
- `...`: arguments passed to or from other methods to the low level.

Details

The predicted values may be the expected mean values of response for the new data (`type.pred = "response"`), or the linear predictors evaluated in the estimated `dblm` of the last iteration.

In classical linear models the mean and the linear predictor are the same (makes use of the identity link). However, other distributions such as Poisson or binomial, the link could change. It's easy to get the predicted mean values, as these are calculated by the inverse link of linear predictors. See `family` to view how to use `linkfun` and `linkinv`.

Value

predict.dbglm produces a vector of predictions for the k new individuals.

Note

Look at which way (or `type.var`) was made the dbglm call. The parameter `type.var` must be consistent with the data type that is introduced to `dbglm`. 
Author(s)

Boj, Eva <evaboj@ub.edu>, Caballe, Adria <adria.caballe@upc.edu>, Delicado, Pedro <pedro.delicado@upc.edu> and Fortiana, Josep <fortiana@ub.edu>

References


See Also

dbgelm for distance-based generalized linear models.

Examples

```r
z <- rnorm(100)
y <- rpois(100, exp(1+z))
glm1 <- glm(y ~ z, family=quasi("identity"))
dbgelm1 <- dbglm(y~z,family=quasi("identity"), method="rel.gvar")

newdata<-

pr1 <- predict(dbglm1,newdata,type.pred="response",type.var="Z")
print(pr1)
plot(z,y)
points(z,dbglm1$fit,t,col=2)
points(0,pr1,col=2)
abline(v=0,col=2)
abline(h=pr1,col=2)
```
predict.dbdm

Predicted values for a dblm object

Description

predict.dblm returns the predicted values, obtained by evaluating the distance regression function in the new data (newdata). newdata can be the values of the explanatory variables of these new cases, the squared distances between these new individuals and the originals ones, or rows of new doubly weighted and centered inner products matrix G.

Usage

```r
## S3 method for class 'dblm'
predict(object, newdata, type.var = "Z", ...)
```

Arguments

- `object`: an object of class dblm. Result of `dblm`
- `newdata`: data.frame or matrix which contains the values of Z (if `type.var = "Z"`). The squared distances between k new individuals and the original n individuals (only if `type.var = "D2"`). Finally, the G inner products matrix (if `type.var = "G"`).
- `type.var`: set de type.var of newdata. Can be "Z" if newdata contains the values of the explanatory variables, "D2" if contains the squared distances matrix or "G" if contains the inner products matrix.
- `...`: arguments passed to or from other methods to the low level.

Value

predict.dblm produces a vector of predictions for the k new individuals.

Note

Look at which way (or `type.var`) was made the dblm call. The parameter `type.var` must be consistent with the data type that is introduced to dblm.

Author(s)

Boj, Eva <evaboj@ub.edu>, Caballe, Adria <adria.caballe@upc.edu>, Delicado, Pedro <pedro.delicado@upc.edu> and Fortiana, Josep <fortiana@ub.edu>

References


**See Also**

dblm for distance-based linear models.

**Examples**

```r
# prediction of new observations newdata
n <- 100
p <- 3
k <- 5

Z <- matrix(rnorm(n*p),nrow=n)
b <- matrix(runif(p)*k,nrow=p)
s <- 1
e <- rnorm(n)*s
y <- Z%*%b + e

D <- dist(Z)
D2 <- disttoD2(D)
D2_train <- D2[1:90,1:90]
class(D2_train)<-"D2"

dblm1 <- dblm(D2_train,y[1:90])

newdata <- D2[91:100,1:90]
predict(dblm1,newdata,type.var="D2")
```

**predict.dbplsr**

*Predicted values for a dbpls object*

**Description**

`predict.dbplsr` returns the predicted values, obtained by evaluating the Distance-based partial least squares function in the new data (`newdata`). `newdata` can be the values of the explanatory variables of these new cases, the squared distances between these new individuals and the originals ones, or rows of new doubly weighted and centered inner products matrix G.
Usage

```r
## S3 method for class 'dbplsr'
predict(object, newdata, type.var = "Z", ...)
```

Arguments

- **object**: an object of class `dbplsr`. Result of `dbplsr`.
- **newdata**: data.frame or matrix which contains the values of Z (if `type.var = "Z"`). The squared distances between k new individuals and the original n individuals (only if `type.var = "D2"`). Finally, the G inner products matrix (if `type.var = "G"`).
- **type.var**: set de type of newdata. Can be "Z" if newdata contains the values of the explanatory variables, "D2" if contains the squared distances matrix or "G" if contains the inner products matrix.
- ... arguments passed to or from other methods to the low level.

Value

`predict.dbplsr` produces a vector of predictions for the k new individuals.

Note

Look at which way (or `type.var`) was made the `dbplsr` call. The parameter `type.var` must be consistent with the data type that is introduced to `dbplsr`.

Author(s)

Boj, Eva <evaboj@ub.edu>, Caballe, Adria <adria.caballe@upc.edu>, Delicado, Pedro <pedro.delicado@upc.edu> and Fortiana, Josep <fortiana@ub.edu>

References


See Also
  dbplsr for distance-based partial least squares.

Examples

```r
#require(pls)
# prediction of new observations newdata
library(pls)
data(yarn)
## Default method:
yarn.dbplsr <- dbplsr(density[1:27] ~ NIR[1:27,], data = yarn, ncomp=6, method="GCV")
pr_yarn_28 <- predict(yarn.dbplsr,newdata=t(as.matrix(yarn$NIR[28,])))
print(pr_yarn_28)
print(yarn$density[28])
```

---

**predict.ldbglm**  
*Predicted values for a ldbglm object*

**Description**

`predict.ldbglm` returns the predicted values, obtained by evaluating the local distance-based generalized linear model in the new data (newdata2), using newdata1 to estimate the "kernel weights".

**Usage**

```r
## S3 method for class 'ldbglm'
predict(object,newdata1,newdata2=newdata1,
new.k.knn=3,type.pred=c("link","response"),
type.var="Z",...)
```

**Arguments**

- **object**: an object of class ldbglm. Result of `ldbglm`.
- **newdata1**: data.frame or matrix which contains the values of Z (if type.var="Z"). The squared distances between k new individuals and the original n individuals (only if type.var="D2"). Finally, the G inner products matrix (if type.var="G"). newdata1 is used to compute kernels and local weights.
- **newdata2**: the same logic as newdata1. newdata2 is used to compute the distance-based generalized regressions with `dbglm`. If newdata2=NULL, newdata2 <- newdata1.
- **new.k.knn**: setting a minimum bandwidth in order to check that a candidate bandwidth h doesn’t contains DB linear models with only one observation. If new.h.knn=NULL, takes the distance that includes the 3 nearest neighbors for each new individual row.
**predict.ldbglm**

`type.pred`  the type of prediction (required). The default `link` is on the scale of the linear predictors; the alternative "response" is on the scale of the response variable.

`type.var`  set the type of the newdata parameter. Can be "Z" if newdata contains the values of the explanatory variables, "D2" if contains the squared distances matrix or "G" if contains the inner products matrix.

... arguments passed to or from other methods to the low level.

**Value**

A list of class `predict.ldbglm` containing the following components:

- `fit` predicted values for the k new individuals.
- `newS` matrix (with dimension (k,n)) of weights used to compute the predictions.

**Note**

Look at which way (or `type.var`) was made the `ldbglm` call. The parameter `type.var` must be consistent with the data type that is introduced to `ldbglm`.

**Author(s)**

Boj, Eva <evaboj@ub.edu>, Caballe, Adria <adria.caballe@upc.edu>, Delicado, Pedro <pedro.delicado@upc.edu> and Fortiana, Josep <fortiana@ub.edu>

**References**


**See Also**

`ldbglm` for local distance-based generalized linear models.
Examples

```r
# example to use of the predict.ldblm function
z <- rnorm(100)
y <- rpois(100, exp(1+z))
glm5 <- glm(y ~ z, family=quasi("identity"))
ldbglm5 <- ldbglm(dist(z), y=y, family=quasi("identity"), noh=3)
plot(z, y)
points(z, glm5$fitt, col=2)
points(z, ldbglm5$fitt, col=3)

pr_ldbglm5 <- predict(ldbglm5, as.matrix(dist(z)^2), type.pred="response", type.var="D2")
max(pr_ldbglm5$fit - ldbglm5$fitt)
```

---

**predict.ldblm**

Predicted values for a ldblm object

**Description**

`predict.ldblm` returns the predicted values, obtained by evaluating the local distance-based linear model in the new data (`newdata2`), using `newdata1` to estimate the "kernel weights".

**Usage**

```r
## S3 method for class 'ldblm'
predict(object, newdata1, newdata2 = newdata1, 
         new.k.knn = NULL, type.var = "Z", ...)  
```

**Arguments**

- `object` an object of class `ldblm`. Result of `ldblm`.
- `newdata1` data.frame or matrix which contains the values of Z (if `type.var="Z"`). The squared distances between k new individuals and the original n individuals (only if `type.var="D2"`). Finally, the G inner products matrix (if `type.var="G"`). `newdata1` is used to compute kernels and local weights.
- `newdata2` the same logic as `newdata1`. `newdata2` is used to compute the Distance-based Regressions with (`dblm`). If `newdata2=NULL`, `newdata2 <- newdata1`.
- `new.k.knn` setting a minimum bandwidth in order to check that a candidate bandwidth h doesn’t contains DB linear models with only one observation. If `new.h.knn=NULL`, takes the distance that includes the 3 nearest neighbors for each new individual row.
- `type.var` set de type of the newdata parameter. Can be "Z" if newdata contains the values of the explanatory variables, "D2" if contains the squared distances matrix or "G" if contains the inner products matrix.
- `...` arguments passed to or from other methods to the low level.
predict.ldblm

Value

A list of class predict.ldblm containing the following components:

- **fit**: predicted values for the k new individuals.
- **newS**: matrix (with dimension (k,n)) of weights used to compute the predictions.

Note

Look at which way (or type.var) was made the ldblm call. The parameter type.var must be consistent with the data type that is introduced to ldblm.

Author(s)

Boj, Eva <evabo@ub.edu>, Caballe, Adria <adria.caballe@upc.edu>, Delicado, Pedro <pedro.delicado@upc.edu> and Fortiana, Josep <fortiana@ub.edu>

References


See Also

ldblm for local distance-based linear models.

Examples

```r
# example to use of the predict.ldblm function

n <- 100
p <- 1
k <- 5

Z <- matrix(rnorm(n*p),nrow=n)
b1 <- matrix(runif(p)*k,nrow=p)
b2 <- matrix(runif(p)*k,nrow=p)
b3 <- matrix(runif(p)*k,nrow=p)
s <- 1
```
summary.dbglm

Summarizing distance-based generalized linear model fits

Description

summary method for class "dbglm"

Usage

## S3 method for class 'dbglm'
summary(object, dispersion, ...)

Arguments

- **object**: an object of class dbglm. Result of dbglm.
- **dispersion**: the dispersion parameter for the family used. Either a single numerical value or NULL (the default).
- **...**: arguments passed to or from other methods to the low level.

Value

A list of class summary.dbglm containing the following components:

- **call**: the matched call.
- **family**: the family object used.
- **deviance**: measure of discrepancy or goodness of fit. Proportional to twice the difference between the maximum log likelihood achievable and that achieved by the model under investigation.
- **aic**: Akaike's An Information Criterion.

```
e <- rnorm(n)*s
y <- Z*b1 + Z^2*b2 + Z^3*b3 + e
d <- as.matrix(dist(Z))
D2 <- D^2
newdata1 <- 0
ldblm1 <- ldlm(y~Z,kind.of.kernel=1,method="GCV",noh=3,k.knn=3)
pr1 <- predict(ldblm1,newdata1)
print(pr1)
plot(Z,y)
points(0,pr1$fit,col=2)
abline(v=0,col=2)
abline(h=pr1$fit,col=2)
```
df.residual: the residual degrees of freedom.
df.null: the residual degrees of freedom for the null model.
null.deviance: the deviance for the null model.
iter: number of Fisher Scoring (dblm) iterations.
deviance.resid: the deviance residuals for each observation: sign(y-mu)*sqrt(di).
pears.resid: the raw residual scaled by the estimated standard deviation of y.
dispersion: the dispersion is taken as 1 for the binomial and Poisson families, and otherwise estimated by the residual Chi-squared statistic (calculated from cases with non-zero weights) divided by the residual degrees of freedom.
gvar: weighted geometric variability of the squared distance matrix.
gvec: diagonal entries in weighted inner products matrix G.
convcrit: convergence criterion. One of: "DevStat" (stopping criterion 1), "muStat" (stopping criterion 2), "maxiter" (maximum allowed number of iterations has been exceeded).

Author(s)
Boj, Eva <evabo@ub.edu>, Caballe, Adria <adria.caballe@upc.edu>, Delicado, Pedro <pedro.delicado@upc.edu> and Fortiana, Josep <fortiana@ub.edu>

References


See Also
`dbglm` for distance-based generalized linear models.
**summary.dbml**  
*Summarizing distance-based linear model fits*

**Description**  
summary method for class "dblm"

**Usage**  
```r  
## S3 method for class 'dblm'
summary(object,...)
```

**Arguments**  
- `object` an object of class dblm. Result of dblm.
- `...` arguments passed to or from other methods to the low level.

**Value**  
A list of class summary.dbml containing the following components:

- `residuals` the residuals (response minus fitted values).
- `sigma` the residual standard error.
- `r.squared` the coefficient of determination R2.
- `adj.r.squared` adjusted R-squared.
- `rdf` the residual degrees of freedom.
- `call` the matched call.
- `gvar` weighted geometric variability of the squared distance matrix.
- `gvec` diagonal entries in weighted inner products matrix G.
- `method` method used to decide the effective rank.
- `eff.rank` integer between 1 and the number of observations minus one. Number of Euclidean coordinates used for model fitting. Applies only if method="eff.rank".
- `rel.gvar` relative geometric variability (real between 0 and 1). Take the lowest effective rank with a relative geometric variability higher or equal to rel.gvar. Default value (rel.gvar=0.95) uses a 95% of the total variability. Applies only rel.gvar if method="rel.gvar".
- `crit.value` value of criterion defined in method.

**Author(s)**  
Boj, Eva <evaboju@ub.edu>, Caballe, Adria <adria.caballe@upc.edu>, Delicado, Pedro <pedro.delicado@upc.edu> and Fortiana, Josep <fortiana@ub.edu>
References


See Also
dblm for distance-based linear models.

summary.dbplsr

Summary of distance-based partial least squares fits

Description

summary method for class "dbplsr"

Usage

## S3 method for class 'dbplsr'
summary(object,...)

Arguments

object an object of class dbplsr. Result of dbplsr.

... arguments passed to or from other methods to the low level.

Value

A list of class summary.dbplsr containing the following components:

- ncomp the number of components of the model.
- r.squared the coefficient of determination R2.
- adj.r.squared adjusted R-squared.
- call the matched call.
- residuals a list containing the residuals for each iteration (response minus fitted values).
- sigma the residual standard error.
gvar total weighted geometric variability.
gvec the diagonal entries in G0.
gvar.iter geometric variability for each iteration.
method the using method to set ncomp.
crit.value value of criterion defined in method.
ncomp.opt optimum number of components according to the selected method.

Author(s)
Boj, Eva <evaboj@ub.edu>, Caballe, Adria <adria.caballe@upc.edu>, Delicado, Pedro <pedro.delicado@upc.edu> and Fortiana, Josep <fortiana@ub.edu>

References


See Also
dbplsr for distance-based partial least squares.

Examples
```r
# require(pls)
library(pls)
data(yarn)
## Default methods:
yarn.dbplsr <- dbplsr(density ~ NIR, data = yarn, ncomp=6, method="GCV")
summary(yarn.dbplsr)
```
**summary ldbglm**

*Summary local distance-based generalized linear model fits*

**Description**

Summary method for class "ldbglm".

**Usage**

```r
## S3 method for class 'ldbglm'
summary(object, dispersion = NULL, ...)
```

**Arguments**

- `object` an object of class ldbglm. Result of ldbglm.
- `dispersion` the dispersion parameter for the family used. Either a single numerical value or NULL (the default)
- `...` arguments passed to or from other methods to the low level.

**Value**

A list of class summary ldbglm containing the following components:

- `nobs` number of observations.
- `trace.hat` Trace of smoother matrix.
- `call` the matched call.
- `family` the family object used.
- `deviance` measure of discrepancy or goodness of fit. Proportional to twice the difference between the maximum log likelihood achievable and that achieved by the model under investigation.
- `df.residual` the residual degrees of freedom.
- `null.deviance` the deviance for the null model.
- `df.null` the residual degrees of freedom for the null model.
- `iter` number of Fisher Scoring (dblml) iterations.
- `deviance.resid` the deviance residuals for each observation: sign(y-mu)*sqrt(di).
- `pears.resid` the raw residual scaled by the estimated standard deviation of y.
- `dispersion` the dispersion is taken as 1 for the binomial and Poisson families, and otherwise estimated by the residual Chi-squared statistic (calculated from cases with non-zero weights) divided by the residual degrees of freedom.
- `kind.kernel` smoothing kernel function.
- `method.h` method used to decide the optimal bandwidth.
- `h.opt` the optimal bandwidth h used in the fitting process (if method.h!=user.h).
- `crit.value` value of criterion defined in method.h.
**Author(s)**

Boj, Eva <evaboj@ub.edu>, Caballe, Adria <adria.caballe@upc.edu>, Delicado, Pedro <pedro.delicado@upc.edu> and Fortiana, Josep <fortiana@ub.edu>

**References**


**See Also**

`ldbg1m` for local distance-based generalized linear models.

---

**summary.ldblm**  
*Summarizing local distance-based linear model fits*

**Description**

summary method for class "ldblm".

**Usage**

```r
## S3 method for class 'ldblm'
summary(object,...)
```

**Arguments**

- `object` an object of class ldbl1. Result of ldbl1.
- `...` arguments passed to or from other methods to the low level.
Value

A list of class `summary.ldblm` containing the following components:

- `nobs` number of observations.
- `r.squared` the coefficient of determination R2.
- `trace.hat` Trace of smoother matrix.
- `call` the matched call.
- `residuals` the residuals (the response minus fitted values).
- `family` the `family` object used.
- `kind.kernel` smoothing kernel function.
- `method.h` method used to decide the optimal bandwidth.
- `h.opt` the optimal bandwidth h used in the fitting process (if `method.h!=user.h`).
- `crit.value` value of criterion defined in `method.h`.

Author(s)

Boj, Eva <evaboj@ub.edu>, Caballe, Adria <adria.caballe@upc.edu>, Delicado, Pedro <pedro.delicado@upc.edu> and Fortiana, Josep <fortiana@ub.edu>

References


See Also

`ldblm` for local distance-based linear models.
Index

AIC, 8, 13, 16, 22, 26
as.D2, 4
as.Gram, 5
cooks.distance, 29
d2toDist, 4, 5, 7, 13, 19, 20
d2toG, 4–6, 6, 13, 19, 20
daisy, 2, 8, 12, 13, 16, 18
dbgglm, 3, 7, 20, 21, 23, 24, 29, 30, 34, 35, 39, 43, 44
dblm, 3, 8, 9, 11, 12, 16, 17, 21, 23, 26–30, 36, 37, 41, 45, 46
dbplsr, 3, 15, 32, 38, 39, 46, 47
dbstats (dbstats-package), 2
dbstats-package, 2
density, 23, 27
dist, 2, 6, 8, 12, 13, 16, 18
disttoD2, 4–7, 13, 18, 20
family, 8, 9, 11, 12, 16, 17, 21, 23, 26–30, 36, 37, 41, 45, 46
formula, 8, 12, 16, 21, 25
glm, 8
gtoD2, 2, 4–7, 13, 19, 19
is.D2 (as.D2), 4
is.Gram (as.Gram), 5
ldbglm, 3, 20, 28, 32, 33, 39, 40, 48, 49
ldblm, 3, 15, 24, 25, 32, 33, 41, 42, 49, 50
lm, 12
loess, 21, 25
optimize, 8, 13
plot.dbglm, 11
plot.dbglm (plot.dbglm), 29
plot.dblm, 10, 15, 29
plot.dbplsr, 18, 31
plot.ldbglm, 24
plot.ldbglm (plot.ldbglm), 32
plot.ldbglm, 23, 28, 32
plot.lm, 29
pls, 16
pr_DB, 3
predict.dbglm, 11, 29, 34
predict.dbglm, 15, 36
predict.dbplsr, 18, 37
predict.ldbglm, 24, 39
predict.ldbglm, 28, 41
print.dbglm (dbglm), 7
print.dblm (dblm), 12
print.dbplsr (dbplsr), 15
print.ldbglm (ldbglm), 20
print.ldbglm (ldbglm), 25
print.predict.ldbglm (predict.ldbglm), 39
print.predict.ldbglm (predict.ldbglm), 41
print.summary.dbglm (summary.dbglm), 43
print.summary.dblm (summary.dbglm), 45
print.summary.dbplsr (summary.dbplsr), 46
print.summary.ldbglm (summary.ldbglm), 48
print.summary.ldbglm (summary.ldbglm), 49
qqnorm, 29
summary.dbglm, 11, 43
summary.dblm, 15, 45
summary.dbplsr, 18, 46
summary.ldbglm, 24, 48
summary.ldbglm, 23, 28, 49

51