Package ‘OmicsPLS’

October 12, 2022

Type Package

Title Data Integration with Two-Way Orthogonal Partial Least Squares

Version 2.0.2

Date 2021-05-19

Author Said el Bouhaddani, Zhujie Gu, Jeanine Houwing-Duistermaat, Geurt Jongbloed, Szymon Kielbasa and Hae-Won Uh

Maintainer Said el Bouhaddani <s.elbouhaddani@umcutrecht.nl>

Description Performs the O2PLS data integration method for two datasets, yielding joint and dataset-specific parts for each dataset.
   The algorithm automatically switches to a memory-efficient approach to fit O2PLS to high dimensional data.
   It provides a rigorous and a faster alternative cross-validation method to select the number of components,
   as well as functions to report proportions of explained variation and to construct plots of the results.
   See the software article by el Bouhaddani et al (2018) <doi:10.1186/s12859-018-2371-3>,

License GPL-3

Encoding UTF-8

Imports graphics, stats, dplyr, ggplot2, parallel, magrittr, tibble, softImpute

Suggests testthat, knitr, rmarkdown, gplots,

RoxygenNote 7.1.1

VignetteBuilder knitr

NeedsCompilation no

Repository CRAN

Date/Publication 2021-05-19 09:40:12 UTC
\section*{R topics documented:}

\begin{verbatim}
adjR2 ................................................................. 2
crossval_o2m ....................................................... 4
crossval_o2m_adjR2 ............................................. 5
crossval_sparsity .................................................. 7
impute_matrix ...................................................... 9
loadings ............................................................... 9
loocv ................................................................. 10
loocv_combi ......................................................... 12
mse .................................................................. 14
norm_vec ........................................................... 14
o2m ................................................................. 15
OomicsPLS .......................................................... 18
orth ................................................................. 20
orth_vec ............................................................ 21
plot.o2m ............................................................ 22
predict.o2m ........................................................ 23
print.cvo2m ....................................................... 23
print.o2m .......................................................... 24
print.pre.o2m .................................................... 24
rmsep ............................................................... 25
rmsep_combi ...................................................... 25
scores ............................................................... 26
ssq ................................................................. 27
summary.o2m ..................................................... 27
thresh_n ........................................................... 28
thresh_n_gr ...................................................... 29
vnorm ............................................................. 29
\end{verbatim}

\section*{Index}

\begin{verbatim}
adjR2 ................................................................. 30
\end{verbatim}

\subsection*{adjR2 \hspace{1cm} \textit{Gridwise adjusted R2 for O2PLS}}

\section*{Description}

For (a grid of) values for \texttt{a}, \texttt{nx} and \texttt{ny}, \texttt{loocv} calculates the R2 of the joint part. Parallel computing is supported on Windows with package \texttt{parallel}.

\section*{Usage}

\begin{verbatim}
adjR2(
  X,
  Y,
  a = 1:2,
  a2 = 1,
  b2 = 1,
)
\end{verbatim}
adjR2

```r
func = o2m,
parall = F,
c1 = NULL,
stripped = TRUE,
p_thresh = 3000,
q_thresh = p_thresh,
tol = 1e-10,
max_iterations = 100
)
```

Arguments

- **X**
  Numeric matrix. Vectors will be coerced to matrix with `as.matrix` (if this is possible)

- **Y**
  Numeric matrix. Vectors will be coerced to matrix with `as.matrix` (if this is possible)

- **a**
  Vector of integers. Contains the numbers of joint components.

- **a2**
  Vector of integers. Contains the numbers of orthogonal components in X.

- **b2**
  Vector of integers. Contains the numbers of orthogonal components in Y.

- **func**
  Function to fit the O2PLS model with. Only `o2m` and `o2m_stripped` are supported.

- **parall**
  Integer. Should a parallel cluster be set up using package `parallel` (Windows)? Best is to leave it to `FALSE`.

- **cl**
  Object of class `cluster`. If `parall` is `TRUE` and `cl` is not `NULL`, calculations are parallelized over workers in `cl`.

- **stripped**
  Logical. Use the stripped version of o2m (usually when cross-validating)?

- **p_thresh**
  Integer. If X has more than `p_thresh` columns, a power method optimization is used, see `o2m2`

- **q_thresh**
  Integer. If Y has more than `q_thresh` columns, a power method optimization is used, see `o2m2`

- **tol**
  Double. Threshold for which the NIPALS method is deemed converged. Must be positive.

- **max_iterations**
  Integer. Maximum number of iterations for the NIPALS method.

Details

The use of this function is to calculate the R2 of the joint part, while varying the number of orthogonal components. Adding more joint components will increase the R2!

A parallelized version is built in -tested on windows-, use package `parallel` and set `parall=TRUE` to activate this. There should not be already a cluster object with the name `cl`. In case of some error, don’t forget to invoke `stopCluster(cl)` to end the cluster. See Task Manager (Windows) to verify that the workers are spanned/ended.

See `looocv` for more intuition.
Value

Matrix with two rows:

<table>
<thead>
<tr>
<th>adjR2X</th>
<th>Contains the joint R2 in X</th>
</tr>
</thead>
<tbody>
<tr>
<td>adjR2Y</td>
<td>Contains the joint R2 in Y</td>
</tr>
</tbody>
</table>

---

crossval_o2m

Cross-validate procedure for O2PLS

Description

Cross-validate procedure for O2PLS

Usage

crossval_o2m(
  X,
  Y,
  a,
  ax,
  ay,
  nr_folds,
  nr_cores = 1,
  stripped = TRUE,
  p_thresh = 3000,
  q_thresh = p_thresh,
  tol = 1e-10,
  max_iterations = 100
)

Arguments

<table>
<thead>
<tr>
<th>X</th>
<th>Numeric matrix. Vectors will be coerced to matrix with \texttt{as.matrix} (if this is possible)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>Numeric matrix. Vectors will be coerced to matrix with \texttt{as.matrix} (if this is possible)</td>
</tr>
<tr>
<td>a</td>
<td>Vector of positive integers. Denotes the numbers of joint components to consider.</td>
</tr>
<tr>
<td>ax</td>
<td>Vector of non-negative integers. Denotes the numbers of X-specific components to consider.</td>
</tr>
<tr>
<td>ay</td>
<td>Vector of non-negative integers. Denotes the numbers of Y-specific components to consider.</td>
</tr>
<tr>
<td>nr_folds</td>
<td>Positive integer. Number of folds to consider. Note: \texttt{kcv=N} gives leave-one-out CV. Note that CV with less than two folds does not make sense.</td>
</tr>
</tbody>
</table>
crossval_o2m_adjR2

nr_cores Positive integer. Number of cores to use for CV. You might want to use `detectCores()`.
 Defaults to 1.

stripped Logical. Use the stripped version of o2m (usually when cross-validating)?

p_thresh Integer. If X has more than p_thresh columns, a power method optimization is used, see `o2m2`

q_thresh Integer. If Y has more than q_thresh columns, a power method optimization is used, see `o2m2`

tol Double. Threshold for which the NIPALS method is deemed converged. Must be positive.

max_iterations Integer. Maximum number of iterations for the NIPALS method.

Details

This is the standard CV approach. It minimizes the sum of the prediction errors of X and Y over a three-dimensional grid of integers. Parallelization is possible on all platforms. On Windows it uses `makePSOCKcluster`, then exports all necessary objects to the workers, and then calls `parLapply`. On OSX and Linux the more friendly `mclapply` is used, which uses forking. A print method is defined, printing the minimizers and minimum in a readable way. Also the elapsed time is tracked and reported.

Value

List of class "cvo2m" with the original and sorted Prediction errors and the number of folds used.

Examples

```r
local({
  X = scale(jitter(tcrossprod(rnorm(100),runif(10))))
  Y = scale(jitter(tcrossprod(rnorm(100),runif(10))))
  crossval_o2m(X, Y, a = 1:4, ax = 1:2, ay = 1:2,
               nr_folds = 5, nr_cores = 1)
})
```
Usage

```r
crossval_o2m_adjR2(
  X,
  Y,
  a,
  ax,
  ay,
  nr_folds,
  nr_cores = 1,
  stripped = TRUE,
  p_thresh = 3000,
  q_thresh = p_thresh,
  tol = 1e-10,
  max_iterations = 100
)
```

Arguments

- **X**: Numeric matrix. Vectors will be coerced to matrix with `as.matrix` (if this is possible)
- **Y**: Numeric matrix. Vectors will be coerced to matrix with `as.matrix` (if this is possible)
- **a**: Vector of positive integers. Denotes the numbers of joint components to consider.
- **ax**: Vector of non-negative integers. Denotes the numbers of X-specific components to consider.
- **ay**: Vector of non-negative integers. Denotes the numbers of Y-specific components to consider.
- **nr_folds**: Positive integer. Number of folds to consider. Note: `kcv=N` gives leave-one-out CV. Note that CV with less than two folds does not make sense.
- **nr_cores**: Positive integer. Number of cores to use for CV. You might want to use `detectCores()`.
  - Defaults to 1.
- **stripped**: Logical. Use the stripped version of o2m (usually when cross-validating)?
- **p_thresh**: Integer. If X has more than `p_thresh` columns, a power method optimization is used, see `o2m2`
- **q_thresh**: Integer. If Y has more than `q_thresh` columns, a power method optimization is used, see `o2m2`
- **tol**: Double. Threshold for which the NIPALS method is deemed converged. Must be positive.
- **max_iterations**: Integer. Maximum number of iterations for the NIPALS method.

Details

This is an alternative way of cross-validating. It is proposed in `citation(OmicsPLS)`. This approach is (much) faster than the standard `crossval_o2m` approach and works fine even with two
folds. For each element in \( n \) it looks for \( nx \) and \( ny \) that maximize the \( R^2 \) between \( T \) and \( U \) in the O2PLS model. This approach often yields similar integer as the standard approach. We however suggest to use the standard approach to minimize the prediction error around the found integers.

Value
data.frame with four columns: MSE, \( n \), \( nx \) and \( ny \). Each row corresponds to an element in \( a \).

Examples
local({
  X = scale(jitter(tcrossprod(rnorm(100),runif(10))))
  Y = scale(jitter(tcrossprod(rnorm(100),runif(10))))
  crossval_o2m_adjR2(X, Y, a = 1:4, ax = 1:2, ay = 1:2,
                     nr_folds = 5, nr_cores = 1)
})

crossval_sparsity
Perform cross-validation to find the optimal number of variables/groups to keep for each joint component

Description
Perform cross-validation to find the optimal number of variables/groups to keep for each joint component

Usage
crossval_sparsity(
  X,
  Y,
  n,
  nx,
  ny,
  nr_folds,
  keepx_seq = NULL,
  keepy_seq = NULL,
  groupx = NULL,
  groupy = NULL,
  tol = 1e-10,
  max_iterations = 100
)

Arguments

\( X \)  Numeric matrix. Vectors will be coerced to matrix with \texttt{as.matrix} (if this is possible)
crossval_sparsity

Y
Numeric matrix. Vectors will be coerced to matrix with as.matrix (if this is possible)

n
Integer. Number of joint PLS components. Must be positive.

nx
Integer. Number of orthogonal components in X. Negative values are interpreted as 0

ny
Integer. Number of orthogonal components in Y. Negative values are interpreted as 0

nr_folds
Integer. Number of folds of CV

keepx_seq
Numeric vector. A vector indicating how many variables/groups to keep for CV in each of the joint component of X. Sparsity of each joint component will be selected sequentially.

keepy_seq
Numeric vector. A vector indicating how many variables/groups to keep for CV in each of the joint component of Y. Sparsity of each joint component will be selected sequentially.

groupx
Vector. Used when sparse = TRUE. A vector of strings indicating group names of each X-variable. Its length must be equal to the number of variables in X. The order of group names must corresponds to the order of the variables.

groupy
Vector. Used when sparse = TRUE. A vector of strings indicating group names of each Y-variable. The length must be equal to the number of variables in Y. The order of group names must corresponds to the order of the variables.

tol
Double. Threshold for which the NIPALS method is deemed converged. Must be positive.

max_iterations
Integer. Maximum number of iterations for the NIPALS method.

Value
A list containing

x_1sd
A vector with length n, giving the optimal number of variables/groups to keep for each X-joint component. One standard error rule is applied

y_1sd
A vector with length n, giving the optimal number of variables/groups to keep for each Y-joint compoenmt. One standard error rule is applied

x
A vector with length n, giving the optimal number of variables/groups to keep for each X-joint component, without applying the one standard error rule

y
A vector with length n, giving the optimal number of variables/groups to keep for each Y-joint component, without applying the one standard error rule
**impute_matrix**  
*Impute missing values in a matrix*

**Description**
Impute missing values in a matrix

**Usage**

```r
impute_matrix(X, ...)
```

**Arguments**

- `X`: A matrix with missing values in some entries.
- `...`: Further arguments for `softimpute`.

**Details**
This function is based on the `softImpute` function in its eponymous package.

**Value**
An imputed version of matrix `X`

**Examples**

```r
X <- matrix(rnorm(20*100), 20)
Xmis <- X
Xmis[sample(length(Xmis), length(Xmis)/10)] <- NA
anyNA(X)
anyNA(impute_matrix(Xmis))
```

---

**loadings**  
*Extract the loadings from an O2PLS fit*

**Description**
This function extracts loading parameters from an O2PLS fit
Usage

loadings(x, ...)

## S3 method for class 'o2m'
loadings(
  x,
  loading_name = c("Xjoint", "Yjoint", "gr_Xjoint", "gr_Yjoint", "Xorth", "Yorth"),
  subset = 0,
  sorted = FALSE,
  ...
)

Arguments

x
Object of class o2m

... For consistency

loading_name character string. One of the following: 'Xjoint', 'Yjoint', 'gr_Xjoint', 'gr_Yjoint', 'Xorth' or 'Yorth'.

subset subset of loading vectors to be extracted.

sorted Logical. Should the rows of the loadings be sorted according to the absolute magnitude of the first column?

Value

Loading matrix

See Also

scores.o2m

Examples

loadings(o2m(scale(-2:2),scale(-2:2*4),1,0,0))

---

**loocv**  
*K fold CV for O2PLS*

Description

For (a grid of) values for a, nx and ny, loocv estimates the prediction error using k-fold CV.
Usage

loocv(
  X,
  Y,
  a = 1:2,
  a2 = 1,
  b2 = 1,
  fitted_model = NULL,
  func = o2m,
  app_err = F,
  kcv,
  stripped = TRUE,
  p_thresh = 3000,
  q_thresh = p_thresh,
  tol = 1e-10,
  max_iterations = 100
)

Arguments

X Numeric matrix. Vectors will be coerced to matrix with as.matrix (if this is possible)

Y Numeric matrix. Vectors will be coerced to matrix with as.matrix (if this is possible)

a Vector of integers. Contains the numbers of joint components.

a2 Vector of integers. Contains the numbers of orthogonal components in X.

b2 Vector of integers. Contains the numbers of orthogonal components in Y.

fitted_model List. Deprecated. O2PLS model fit with o2m. Is used to calculate the apparent error without recalculating this fit.

func Function to fit the O2PLS model with. Only o2m and o2m_stripped are supported.

app_err Logical. Deprecated. Should the apparent error also be computed?

kcv Integer. The value of k, i.e. the number of folds. Choose N for LOO-CV.

stripped Logical. Use the stripped version of o2m (usually when cross-validating)?

p_thresh Integer. If X has more than p_thresh columns, a power method optimization is used, see o2m2

q_thresh Integer. If Y has more than q_thresh columns, a power method optimization is used, see o2m2

tol Double. Threshold for which the NIPALS method is deemed converged. Must be positive.

max_iterations Integer. Maximum number of iterations for the NIPALS method.
Details

Note that this function can be easily parallelized (on Windows e.g. with the parallel package.).

The parameters a, a2 and b2 can be integers or vectors of integers. A for loop is used to loop over all combinations. The resulting output is a list, which is more easy to interpret if you use array(unlist(output_of_loocv$CVerr)) as in the example below. The array will have varying a along the first dimension and a2 and b2 along the second and third respectively. Typing example(loocv) (hopefully) clarifies this function.

Value

List with two numeric vectors:

CVerr Contains the k-fold CV estimated RMSEP
Fiterr Contains the apparent error

Usage

loocv_combi(X, Y, a = 1:2, a2 = 1, b2 = 1, fitted_model = NULL, func = o2m, app_err = F, kcv, stripped = TRUE, p_thresh = 3000, q_thresh = p_thresh, tol = 1e-10, max_iterations = 100)
Arguments

X    Numeric matrix. Vectors will be coerced to matrix with \texttt{as.matrix} (if this is possible)

Y    Numeric matrix. Vectors will be coerced to matrix with \texttt{as.matrix} (if this is possible)

a    Vector of integers. Contains the numbers of joint components.

a2   Vector of integers. Contains the numbers of orthogonal components in \( X \).

b2   Vector of integers. Contains the numbers of orthogonal components in \( Y \).

\texttt{fitted\_model} List. Deprecated. O2PLS model fit with \texttt{o2m}. Is used to calculate the apparent error without recalculating this fit.

\texttt{func} Function to fit the O2PLS model with. Only \texttt{o2m} and \texttt{o2m\_stripped} are supported.

\texttt{app\_err} Logical. Deprecated. Should the apparent error also be computed?

\texttt{kcv} Integer. The value of \( k \), i.e. the number of folds. Choose \( N \) for LOO-CV.

\texttt{stripped} Logical. Use the stripped version of \texttt{o2m} (usually when cross-validating)?

\texttt{p\_thresh} Integer. If \( X \) has more than \texttt{p\_thresh} columns, a power method optimization is used, see \texttt{o2m2}

\texttt{q\_thresh} Integer. If \( Y \) has more than \texttt{q\_thresh} columns, a power method optimization is used, see \texttt{o2m2}

\texttt{tol} Double. Threshold for which the NIPALS method is deemed converged. Must be positive.

\texttt{max\_iterations} Integer. Maximum number of iterations for the NIPALS method.

Details

Note that this function can be easily parallelized (on Windows e.g. with the \texttt{parallel} package.). If there are NAs in the \texttt{CVerr} component, this is due to an error in the fitting.

Value

List with two numeric vectors:

\texttt{CVerr} Contains the k-fold CV estimated RMSEP

\texttt{Fiterr} Contains the apparent error
### mse

**Calculate mean squared difference**

**Description**

Calculate mean squared difference

**Usage**

```r
mse(x, y = 0, na.rm = FALSE)
```

**Arguments**

- `x`: Numeric vector or matrix.
- `y`: Numeric vector or matrix. Defaults to 0.
- `na.rm`: Remove NA's?

**Details**

Is equal to `ssq(x-y)/length(c(x))`. If `x` and `y` are of unequal length, the invoked minus-operator will try to make the best out of it by recycling elements of the shorter object (usually you don’t want that). In particular if `x` is an `N x p` matrix and `y` an `N x 1` vector, `y` is subtracted from each column of `x`, and if `y=0` (default) you get the mean of `vec(x^2)`

**Value**

The mean of the squared differences elementwise.

**Examples**

```r
mse(2)
mse(1:10,2:11) == 1
mse(matrix(rnorm(500),100,5),matrix(rnorm(500),100,5))
```

### norm_vec

**Norm of a vector**

**Description**

Norm of a vector

**Usage**

```r
norm_vec(x)
```
Arguments

x Numerical vector

Value

L2 norm of a vector

---

Perform O2PLS data integration with two-way orthogonal corrections

Description

NOTE THAT THIS FUNCTION DOES NOT CENTER NOR SCALE THE MATRICES! Any normalization you will have to do yourself. It is best practice to at least center the variables though.

Usage

```r
o2m(
  X,
  Y,
  n,
  nx,
  stripped = FALSE,
  p_thresh = 3000,
  q_thresh = p_thresh,
  tol = 1e-10,
  max_iterations = 1000,
  sparse = F,
  groupx = NULL,
  groupy = NULL,
  keepx = NULL,
  keepy = NULL,
  max_iterations_sparsity = 1000
)
```

Arguments

X Numeric matrix. Vectors will be coerced to matrix with `as.matrix` (if this is possible)

Y Numeric matrix. Vectors will be coerced to matrix with `as.matrix` (if this is possible)

n Integer. Number of joint PLS components. Must be positive.

nx Integer. Number of orthogonal components in `X`. Negative values are interpreted as 0
ny  Integer. Number of orthogonal components in Y. Negative values are interpreted as 0.
stripped Logical. Use the stripped version of o2m (usually when cross-validating)?
p_thresh Integer. If X has more than p_thresh columns, a power method optimization is used, see o2m2
q_thresh Integer. If Y has more than q_thresh columns, a power method optimization is used, see o2m2
tol Double. Threshold for which the NIPALS method is deemed converged. Must be positive.
max_iterations Integer. Maximum number of iterations for the NIPALS method.
sparse Boolean. Default value is FALSE, in which case O2PLS will be fitted. Set to TRUE for GO2PLS.
groupx Vector. Used when sparse = TRUE. A vector of strings indicating group names of each X-variable. Its length must be equal to the number of variables in X. The order of group names must corresponds to the order of the variables.
groupy Vector. Used when sparse = TRUE. A vector of strings indicating group names of each Y-variable. The length must be equal to the number of variables in Y. The order of group names must corresponds to the order of the variables.
keepx Vector. Used when sparse = TRUE. A vector of length n indicating how many variables (or groups if groupx is provided) to keep in each of the joint component of X. If the input is an integer, all the components will have the same amount of variables or groups retained.
keepy Vector. Used when sparse = TRUE. A vector of length n indicating how many variables (or groups if groupx is provided) to keep in each of the joint component of Y. If the input is an integer, all the components will have the same amount of variables or groups retained.
max_iterations_sparsity Integer. Used when sparse = TRUE. Maximum number of iterations for the NIPALS method for GO2PLS.

Details

If both nx and ny are zero, o2m is equivalent to PLS2 with orthonormal loadings. This is a ‘slower’ (in terms of memory) implementation of O2PLS, and is using svd, use stripped=T for a stripped version with less output. If either ncol(X) > p_thresh or ncol(Y) > q_thresh, the NIPALS method is used which does not store the entire covariance matrix. The squared error between iterands in the NIPALS approach can be adjusted with tol. The maximum number of iterations in the NIPALS approach is tuned by max_iterations.

Value

A list containing

Tt Joint X scores
w. Joint X loadings
U Joint Y scores
C. Joint Y loadings
E Residuals in X
Ff Residuals in Y
T_Yosc Orthogonal X scores
P_Yosc. Orthogonal X loadings
W_Yosc Orthogonal X weights
U_Xosc Orthogonal Y scores
P_Xosc. Orthogonal Y loadings
C_Xosc Orthogonal Y weights
B_U Regression coefficient in \( T \sim U \)
B_T. Regression coefficient in \( U \sim T \)
H_TU Residuals in \( T \) in \( T \sim U \)
H_UT Residuals in \( U \) in \( U \sim T \)
X_hat Prediction of X with Y
Y_hat Prediction of Y with X
R2X Variation (measured with \( ssq \)) of the modeled part in X (defined by joint + orthogonal variation) as proportion of variation in X
R2Y Variation (measured with \( ssq \)) of the modeled part in Y (defined by joint + orthogonal variation) as proportion of variation in Y
R2Xcorr Variation (measured with \( ssq \)) of the joint part in X as proportion of variation in X
R2Ycorr Variation (measured with \( ssq \)) of the joint part in Y as proportion of variation in Y
R2X_Y0 Variation (measured with \( ssq \)) of the orthogonal part in X as proportion of variation in X
R2Y_X0 Variation (measured with \( ssq \)) of the orthogonal part in Y as proportion of variation in Y
R2Xhat Variation (measured with \( ssq \)) of the predicted X as proportion of variation in X
R2Yhat Variation (measured with \( ssq \)) of the predicted Y as proportion of variation in Y
W_gr Joint loadings of X at group level (only available when GO2PLS is used)
C_gr Joint loadings of Y at group level (only available when GO2PLS is used)

See Also

summary.o2m, plot.o2m, crossval_o2m_adjR2, crossval_sparsity
Examples

test_X <- scale(matrix(rnorm(100*10),100,10))
test_Y <- scale(matrix(rnorm(100*11),100,11))
# -------- Default run --------
o2m(test_X, test_Y, 3, 2, 1)
# -------- Stripped version --------
o2m(test_X, test_Y, 3, 2, 1, stripped = TRUE)
# -------- High dimensional version --------
o2m(test_X, test_Y, 3, 2, 1, p_thresh = 1)
# ------ High D and stripped version ------
o2m(test_X, test_Y, 3, 2, 1, stripped = TRUE, p_thresh = 1)
# ------ Now with more iterations ------
o2m(test_X, test_Y, 3, 2, 1, stripped = TRUE, p_thresh = 1, max_iterations = 1e6)

---

OmicsPLS

Data integration with O2PLS: Two-Way Orthogonal Partial Least Squares

Description

The OmicsPLS package is an R package for penalized integration of heterogeneous omics data. The software articles are published in (el Bouhaddani et al, 2018, doi: 10.1186/s1285901823713) and (Gu et al, 2020, doi: 10.1186/s12859021039583). OmicsPLS includes the O2PLS fit, the GO2PLS fit, cross-validation tools and some misc functions.

Model and assumptions

Note that the rows of X and Y are the subjects and columns are variables. The number of columns may be different, but the subjects should be the same in both datasets.

The O2PLS model (Trygg & Wold, 2003) decomposes two datasets X and Y into three parts.

• 1. A joint part, representing the relationship between X and Y
• 2. An orthogonal part, representing the unrelated latent variation in X and Y separately.
• 3. A noise part capturing all residual variation.

See also the corresponding paper (el Bouhaddani et al, 2018).

Fitting

The O2PLS fit is done with o2m. For data X and Y you can run o2m(X, Y, n, nx, ny) for an O2PLS fit with n joint and nx, ny orthogonal components. See the help page of o2m for more information on parameters. There are four ways to obtain an O2PLS fit, depending on the dimensionality.

• For the not-too-high dimensional case, you may use o2m with default parameters. E.g. o2m(X, Y, n, nx, ny).
• In case you only want the parameters, you may add stripped = TRUE to obtain a stripped version of o2m which avoids calculating and storing some matrices. E.g. o2m(X, Y, n, nx, ny, stripped=TRUE).
• For high dimensional cases, defined by \( n_{\text{col}}(X) > p_{\text{thresh}} \) and \( n_{\text{col}}(Y) > q_{\text{thresh}} \), a NI-PALS approach is used which avoids storing large matrices. E.g. \( \text{o2m}(X,Y,n,nx,ny,p_{\text{thresh}}=3000,q_{\text{thresh}}=3000) \). The thresholds are by default both at 3000 variables.

• If you want a stripped version in the high dimensional case, add \( \text{stripped} = \text{TRUE} \). E.g. \( \text{o2m}(X,Y,n,nx,ny,\text{stripped}=\text{TRUE},p_{\text{thresh}}=3000,q_{\text{thresh}}=3000) \).

• For GO2PLS, add \( \text{sparsity} = \text{TRUE} \) and specify how many variables or groups to retain. E.g. \( \text{o2m}(X,Y,n,nx,ny,\text{sparsity}=\text{TRUE},\text{keepx}, \text{keepy}) \).

Obtaining results

After fitting an O2PLS model, by running e.g. \( \text{fit} = \text{o2m}(X,Y,n,nx,ny) \), the results can be visualised. Use \( \text{plot(fit,...)} \) to plot the desired loadings with/without ggplot2. Use \( \text{summary(fit,...)} \) to see the relative explained variances in the joint/orthogonal parts. Also plotting the joint scores \( \text{fit$Tt} \), \( \text{fit$U} \) and orthogonal scores \( \text{fit$T_Yosc} \), \( \text{fit$U_Xosc} \) are of help.

Cross-validating

Determining the number of components \( n, nx, ny \) is an important task. For this we have two methods. See \text{citation("OmicsPLS")} for our proposed approach for determining the number of components, implemented in \text{crossval_o2m_adjR2}!

• Cross-validation (CV) is done with \text{crossval_o2m} and \text{crossval_o2m_adjR2}, both have built in parallelization which relies on the \text{parallel} package. Usage is something like \text{crossval_o2m}(X, Y, a, ax, ay, nr\_folds) where \( a, ax, ay \) are vectors of integers. See the help pages. \( nr\_folds \) is the number of folds, with \( nr\_folds = nrow(X) \) for Leave-One-Out CV.

• For \text{crossval_o2m_adjR2} the same parameters are to be specified. This way of cross-validating is (potentially much) faster than the standard approach. It is also recommended over the standard CV.

• To cross-validate the number of variables to keep, use \text{crossval_sparsity}.

S3 methods

There are S3 methods implemented for a fit obtained with \text{o2m}, i.e. \text{fit} \leftarrow \text{o2m}(X,Y,n,nx,ny)

• Use \text{plot(fit)} to plot the loadings, see above.

• Use \text{loadings(fit)} to extract a matrix with loading values

• Use \text{scores(fit)} to extract the scores

• Use \text{print} and \text{summary} to print and summarize the fit object

Imputation

When the data contains missing values, one should impute them prior to using O2PLS. There are many sophisticated approaches available, such as MICE and MissForest, and no one approach is the best for all situations. To still allow users to quickly impute missing values in their data matrix, the \text{impute\_matrix} function is implemented. It relies on the \text{softImpute} function+package and imputes based on the singular value decomposition.
Also some handy tools are available

- `orth(X)` is a function to obtain an orthogonalized version of a matrix or vector `X`.
- `ssq(X)` is a function to calculate the sum of squares (or squared Frobenius norm) of `X`. See also `vnorm` for calculating the norm of each column in `X`.
- `mse(x, y)` returns the mean squared difference between two matrices/vectors.

Citation

If you use the OmicsPLS R package in your research, please cite the corresponding software paper:


The bibtex entry can be obtained with command `citation("OmicsPLS")`. Thank you!

The original paper proposing O2PLS is


Author(s)

Said el Bouhaddani (<s.elbouhaddani@umcutrecht.nl>, Twitter: @selbouhaddani), Zhujie Gu, Szymon Kielbasa, Geurt Jongbloed, Jeanine Houwing-Duistermaat, Hae-Won Uh.

Maintainer: Said el Bouhaddani (<s.elbouhaddani@umcutrecht.nl>).

---

**Orthogonalize a matrix**

**Description**

Orthogonalize a matrix

**Usage**

```r
orth(X, X_true = NULL, type = c("QR", "SVD"))
```

**Arguments**

- `X` Numeric vector or matrix.
- `X_true` (optional) A 'true' matrix/vector. Used to correct the sign of the orthonormalized `X` if QR is used. Only the first column is corrected.
- `type` A character or numeric. Should be one of "QR" or "SVD".
**orth_vec**

**Details**

Choosing `type='QR'` uses a QR decomposition of $X$ to produce orthonormal columns. For `type='SVD'` it uses an SVD decomposition. The columns are corrected for sign.

**Value**

An orthogonalized representation of $X$

**Examples**

- `orth(c(3,4))`
- `round(crossprod(orth(matrix(rnorm(500),100,5))),4)`
- `orth(matrix(1:9,3,3),type='QR')[,1] - orth(1:3); orth(matrix(1:9,3,3),type='SVD')[,1] - orth(1:3)`

---

**orth_vec**

*Orthogonalize a sparse loading vector with regard to a matrix*

**Description**

Orthogonalize a sparse loading vector with regard to a matrix

**Usage**

`orth_vec(x, W)`

**Arguments**

- `x` sparse loading vector to be orthogonalized
- `W` sparse loading matrix of the previous loading vectors

**Value**

A sparse loading vector
Plot one or two loading vectors for class o2m

Description
This function plots one or two loading vectors, by default with ggplot2.

Usage
## S3 method for class 'o2m'
plot(
x, 
loading_name = c("Xjoint", "Yjoint", "gr_Xjoint", "gr_Yjoint", "Xorth", "Yorth"), 
i = 1, 
j = NULL, 
use_ggplot2 = TRUE, 
label = c("number", "colnames"), 
... 
)

Arguments

x An O2PLS fit, with class 'o2m'

loading_name character string. One of the following: 'Xjoint', 'Yjoint', 'gr_Xjoint', 'gr_Yjoint', 'Xorth' or 'Yorth'.

i Integer. First component to be plotted.

j NULL (default) or Integer. Second component to be plotted.

use_ggplot2 Logical. Default is TRUE. If FALSE, the usual plot device will be used.

label Character, either 'number' or 'colnames'. The first option prints numbers, the second prints the colnames

... Further arguments to geom_text, such as size, col, alpha, etc.

Value
If use_ggplot2 is TRUE a ggplot2 object. Else NULL.

See Also
summary.o2m
predict.o2m Predicts X or Y

Description
Predicts X or Y based on new data on Y or X

Usage
## S3 method for class 'o2m'
predict(object, newdata, XorY = c("X", "Y"), ...)

Arguments
object List. Should be of class o2m.
newdata New data, which one of X or Y is specified in XorY.
XorY Character specifying whether newdata is X or Y.
... For compatibility

Details
Prediction is done after correcting for orthogonal parts.

Value
Predicted Data

Examples
predict(o2m(scale(1:10), scale(1:10), 1, 0, 0), newdata = scale(1:5), XorY = "X")

print.cvo2m Cross-validate procedure for O2PLS

Description
Cross-validate procedure for O2PLS

Usage
## S3 method for class 'cvo2m'
print(x, include_matrix = FALSE, ...)

Details
Cross-validate procedure for O2PLS

Value

Examples

Arguments

x List of class "cvo2m", produced by crossval_o2m.
include_matrix Logical. Should the 3-d array with Prediction errors also be printed.
... For consistency.

print.o2m

Description

This function is the print method for an O2PLS fit

Usage

## S3 method for class 'o2m'
print(x, ...)

Arguments

x An O2PLS fit (an object of class o2m)
... For consistency

print.pre.o2m

Description

This function is the print method for an O2PLS fit

Usage

## S3 method for class 'pre.o2m'
print(x, ...)

Arguments

x An O2PLS fit (an object of class o2m)
... For consistency
**rmsep**

*Root MSE of Prediction*

**Description**

Calculates the Root MSE of prediction on test data. Only tested to work inside `loocv`.

**Usage**

```r
rmse(Xtst, Ytst, fit, combi = FALSE)
```

**Arguments**

- **Xtst**: Numeric vector or matrix.
- **Ytst**: Numeric vector or matrix.
- **fit**: `o2m` fit (on data without `Xtst` and `Ytst`).
- **combi**: Logical. Should the symmetrized MSE be used, i.e. add both MSEs. Not yet implemented, but see `rmsep_combi`.

**Details**

This function is the building block for `loocv`, as it produced the prediction error for test (left out) data.

**Value**

Mean squares difference between predicted Y and true Y

---

**rmsep_combi**

*Symmetrized root MSE of Prediction*

**Description**

Calculates the symmetrized root MSE of prediction on test data. *Expected* to work in combination with `loocv`.

**Usage**

```r
rmse_combi(Xtst, Ytst, fit)
```

**Arguments**

- **Xtst**: Numeric vector or matrix.
- **Ytst**: Numeric vector or matrix.
- **fit**: `o2m` fit (on data without `Xtst` and `Ytst`).
Details

This function is the building block for `loocv`, as it produced the prediction error for test (left out) data.

This is a symmetrized version of `rmsep`, and is used by `loocv`. The prediction error of both $\text{X}_{\text{tst}}$ and $\text{Y}_{\text{tst}}$ are calculated and summed. Whether this is a good idea depends: If $\text{X}$ and $\text{Y}$ have similar meanings (LC-MS versus MALDI) this is a good thing to do. If the two matrices do not have similar meanings, (Metabolomics versus Transcriptomics) then you may want to not sum up the two prediction errors or include weights in the sum.

Value

Mean squares difference between predicted $\text{Y}$ and true $\text{Y}$

---

**scores**

*Extract the scores from an O2PLS fit*

Description

This function extracts score matrices from an O2PLS fit

Usage

```r
scores(x, ...)
```

## S3 method for class 'o2m'

```r
scores(  
  x,  
  which_part = c("Xjoint", "Yjoint", "Xorth", "Yorth"),  
  subset = 0,  
  ...  
)
```

Arguments

- `x` Object of class `o2m`
- `...` For consistency
- `which_part` character string. One of the following: 'Xjoint', 'Yjoint', 'Xorth' or 'Yorth'.
- `subset` subset of scores vectors to be extracted.

Value

Scores matrix

See Also

`loadings.o2m`
ssq

Examples
scores(o2m(scale(-2:2),scale(-2:2*4),1,0,0))

ssq  Calculate Sum of Squares

Description
Calculate Sum of Squares

Usage
ssq(X)

Arguments
X  Numeric vector or matrix.

Details
This is the Frobenius norm of X.

Value
The sum of squared elements of X

Examples
ssq(tcrossprod(1:5))
ssq(rnorm(1e5))/1e5

summary.o2m  Summary of an O2PLS fit

Description
Until now only variational summary given by the R2’s is outputted

Usage
## S3 method for class 'o2m'
summary(object, digits = 3, ...)
thresh_n

Soft thresholding a vector with respect to a number of variables

Description

Soft thresholding a vector with respect to a number of variables

Usage

thresh_n(x, keepx)

Arguments

x                Numerical vector
keepx            How many non-zero

Value

Soft-thresholded vector

Example

summary(o2m(scale(-2:2),scale(-2:2*4),1,0,0))
thresh_n_gr

Soft thresholding a vector with respect to a number of groups

Description
Soft thresholding a vector with respect to a number of groups

Usage
thresh_n_gr(w, keep_gr, index_gr)

Arguments
w Numerical loading vector
keep_gr How many groups to retain
index_gr List of index and size. index are the index of variables belongs to the group in the original vector, size is the group size

Value
A list containing sparse loading vector and names of the selected groups

vnorm

Norm of a vector or columns of a matrix

Description
Norm of a vector or columns of a matrix

Usage
vnorm(x)

Arguments
x Numeric vector or matrix.

Value
(columnwise) Euclidian norm of x

Examples
vnorm(orth(1:5))
vnorm(matrix(1:9,3,3))^2 - colSums(matrix(1:9,3)^2)
Index

* OmicsPLS
  OmicsPLS, 18

adjR2, 2

crossval_o2m, 4, 19, 24
crossval_o2m_adjR2, 5, 17, 19
crossval_sparsity, 7, 17, 19

detectCores, 5, 6

impute_matrix, 9, 19

loadings, 9, 19
loadings.o2m, 26
loocv, 3, 10, 12, 25, 26
loocv.combi, 12

makePSOCKcluster, 5
mclapply, 5
mse, 14, 20

norm_vec, 14

o2m, 3, 11, 13, 15, 18, 23, 25, 28
o2m2, 3, 5, 6, 11, 13, 16
o2m_stripped, 3, 11, 13
OmicsPLS, 18
orth, 20, 20
orth_vec, 21

parLapply, 5
plot, 19
plot.o2m, 17, 22, 28
predict.o2m, 23
print, 19
print.cvo2m, 23
print.o2m, 24
print.pre.o2m, 24

rmsep, 25, 26

rmsep.combi, 25, 25

scores, 19, 26
scores.o2m, 10
softimpute, 9, 19
ssq, 17, 20, 27
summary, 19
summary.o2m, 17, 22, 27
svd, 16

thresh_n, 28
thresh_n_gr, 29

vnorm, 20, 29