

# Package ‘CVR’

October 12, 2022

**Type** Package

**Title** Canonical Variate Regression

**Version** 0.1.1

**Date** 2017-03-17

**Author** Chongliang Luo, Kun Chen.

**Maintainer** Chongliang Luo <chongliang.luo@uconn.edu>

**Description** Perform canonical variate regression (CVR) for two sets of covariates and a univariate response, with regularization and weight parameters tuned by cross validation.

**License** GPL (>= 2)

**LinkingTo** Rcpp, RcppArmadillo

**Imports** Rcpp

**Depends** R (>= 3.2.1), PMA

**RoxygenNote** 6.0.1

**NeedsCompilation** yes

**Repository** CRAN

**Date/Publication** 2017-03-22 14:13:03 UTC

## R topics documented:

CVR-package . . . . .	2
alcohol . . . . .	3
CVR . . . . .	4
cvrsolver . . . . .	6
mouse . . . . .	8
plot.CVR . . . . .	9
SimulateCVR . . . . .	10
SparseCCA . . . . .	12

<b>Index</b>	<b>14</b>
--------------	-----------

---

CVR-package

*Canonical Variate Regression*


---

## Description

Perform canonical variate regression (CVR) for two sets of covariates and a univariate response, with regularization and weight parameters tuned by cross validation.

## Details

Index of help topics:

CVR	Fit canonical variate regression with tuning parameters selected by cross validation.
CVR-package	Canonical Variate Regression
SimulateCVR	Generate simulation data.
SparseCCA	Sparse canonical correlation analysis.
alcohol	Data sets for the alcohol dependence example
cvsolver	Canonical Variate Regression.
mouse	Data sets for the mouse body weight example
plot.CVR	Plot a CVR object.

functions: cvsolver, SparseCCA, SimulateCVR, CVR, plot.CVR.

## Author(s)

Chongliang Luo, Kun Chen.

Maintainer: Chongliang Luo <chongliang.luo@uconn.edu>

## References

Chongliang Luo, Jin Liu, Dipak D. Dey and Kun Chen (2016) Canonical variate regression. Accepted by Biostatistics, doi: 10.1093/biostatistics/kxw001.

Daniela M. Witten, Robert Tibshirani and Trevor Hastie (2009) A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis. Biostatistics 10(3), 515-534.

## See Also

PMA.

alcohol

*Data sets for the alcohol dependence example***Description**

A list of 3 data frames that contains the gene expression, DNA methylation and AUD (alcohol use disorder) of 46 human subjects. The data is already screened for quality control. For the raw data see the link below. For more details see the reference.

**Usage**

```
alcohol
```

**Format**

A list of 3 data frames:

**gene** Human gene expression. A data frame of 46 rows and 300 columns.

**meth** Human DNA methylation. A data frame of 46 rows and 500 columns.

**disorder** Human AUD indicator. A data frame of 46 rows and 1 column. The first 23 subjects are AUDs and the others are matched controls.

**Source**

Alcohol dependence: <http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE49393>.

**References**

Chongliang Luo, Jin Liu, Dipak D. Dey and Kun Chen (2016) Canonical variate regression. Biostatistics, doi: 10.1093/biostatistics/kxw001.

**Examples**

```
##### Alcohol dependence example #####
data(alcohol)
gene <- scale(as.matrix(alcohol$gene))
meth <- scale(as.matrix(alcohol$meth))
disorder <- as.matrix(alcohol$disorder)
alcohol.X <- list(X1 = gene, X2 = meth)
## Not run:
foldid <- c(rep(1:5, 4), c(3,4,5), rep(1:5, 4), c(1,2,5))
## table(foldid, disorder)
## there maybe warnings due to the glm refitting with small sample size
alcohol.cvr <- CVR(disorder, alcohol.X, rankseq = 2, etaseq = 0.02,
                  family = "b", penalty = "L1", foldid = foldid )
plot(alcohol.cvr)
plot(gene %% alcohol.cvr$solution$W[[1]][, 1], meth %% alcohol.cvr$solution$W[[2]][, 1])
cor(gene %% alcohol.cvr$solution$W[[1]], meth %% alcohol.cvr$solution$W[[2]])
```

```
## End(Not run)
```

---

CVR	<i>Fit canonical variate regression with tuning parameters selected by cross validation.</i>
-----	--

---

### Description

This function fits the solution path of canonical variate regression, with tuning parameters selected by cross validation. The tuning parameters include the rank, the  $\eta$  and the  $\lambda$ .

### Usage

```
CVR(Y, Xlist, rankseq = 2, neta = 10, etaseq = NULL, nlam = 50,
     Lamseq = NULL, family = c("gaussian", "binomial", "poisson"),
     Wini = NULL, penalty = c("GL1", "L1"), nfold = 10, foldid = NULL,
     opts = list(), type.measure = NULL)
```

### Arguments

Y	A univariate response variable.
Xlist	A list of two covariate matrices as in <code>cvrsolver</code> .
rankseq	A sequence of candidate ranks. The default is a single value 2.
neta	Number of $\eta$ values. The default is 10.
etaseq	A sequence of length <code>neta</code> containing candidate $\eta$ values between 0 and 1. The default is $10^{\text{seq}(-2, \log_{10}(0.9), \text{length} = \text{neta})}$ .
nlam	Number of $\lambda$ values. The default is 50.
Lamseq	A matrix of $\lambda$ values. The column number is the number of sets in <code>Xlist</code> , and the row number is <code>nlam</code> . The default is $10^{\text{seq}(-2, 2, \text{length} = \text{nlam})}$ for each column.
family	Type of response as in <code>cvrsolver</code> . The default is "gaussian".
Wini	A list of initial loading $W$ 's. The default is from the SparseCCA solution. See SparseCCA.
penalty	Type of penalty on loading matrices $W$ 's as in <code>cvrsolver</code> . The default is "GL1".
nfold	Number of folds in cross validation. The default is 10.
foldid	Specifying training and testing sets in cross validation; random generated if not supplied. It remains the same across different rank and $\eta$ .
opts	A list of options for controlling the algorithm. The default of <code>opts\$spthresh</code> is 0.4, which means we only search sparse models with at most 40% nonzero entries in $W1$ and $W2$ . See the other options ( <code>standardization</code> , <code>maxIters</code> and <code>tol</code> ) in <code>cvrsolver</code> .
type.measure	Type of measurement used in cross validation. "mse" for Gaussian, "auc" for binomial, and "deviance" for binomial and Poisson.

**Details**

In this function, the rank,  $\eta$  and  $\lambda$  are tuned by cross validation. CVR then is refitted with all data using the selected tuning parameters. The plot function shows the tuning of  $\lambda$ , with selected rank and  $\eta$ .

**Value**

An object with S3 class "CVR" containing the following components

cverror	A matrix containing the CV errors. The number of rows is the length of etaseq and the number of columns is the length of rankseq.
etahat	Selected $\eta$ .
rankhat	Selected rank.
Lamhat	Selected $\lambda$ 's.
Alphapath	An array containing the fitted paths of the intercept term $\alpha$ .
Betapath	An array containing the fitted paths of the regression coefficient $\beta$ .
W1path, W2path	Arrays containing the fitted paths of W1 and W2.
foldid	foldid used in cross validation.
cvout	Cross validation results using selected $\eta$ and rank.
solution	A list including the solutions of $\alpha$ , $\beta$ , W1 and W2, by refitting all the data using selected tuning parameters.

**Author(s)**

Chongliang Luo, Kun Chen.

**References**

Chongliang Luo, Jin Liu, Dipak D. Dey and Kun Chen (2016) Canonical variate regression. Biostatistics, doi: 10.1093/biostatistics/kxw001.

**See Also**

[cvrsolver](#), [SparseCCA](#), [SimulateCVR](#).

**Examples**

```
##### Gaussian response #####
set.seed(42)
mydata <- SimulateCVR(family = "g", n = 100, rank = 4, p1 = 50, p2 = 70,
                    pnz = 10, beta = c(2, 1, 0, 0))
X1 <- mydata$X1;
X2 <- mydata$X2
Xlist <- list(X1 = X1, X2 = X2);
Y <- mydata$y
## fix rank = 4, tune eta and lambda
##out_cvr <- CVR(Y, Xlist, rankseq = 4, neta = 5, nlam = 25,
```

```

##          family = "g", nfold = 5)
## out_cvr$solution$W[[1]];
## out_cvr$solution$W[[2]];
### uncomment to see plots
## plot.CVR(out_cvr)
##
## Distance of subspaces
##U <- mydata$U
##Pj <- function(U) U %*% solve(t(U) %*% U, t(U))
##sum((Pj(U) - (Pj(X1 %*% out_cvr$sol$W[[1]]) + Pj(X2 %*% out_cvr$sol$W[[2]]))/2)^2)
## Precision/Recall rate
## the first 10 rows of the true W1 and W2 are set to be nonzero
##W12 <- rbind(out_cvr$sol$W[[1]], out_cvr$sol$W[[1]])
##W12norm <- apply(W12, 1, function(a)sqrt(sum(a^2)))
##prec <- sum(W12norm[c(1:10, 51:60)] != 0)/sum(W12norm != 0); prec
##rec <- sum(W12norm[c(1:10, 51:60)] != 0)/20; rec
## sequential SparseCCA, compare the Distance of subspaces and Prec/Rec
##W12s <- SparseCCA(X1, X2, 4)
## Distance larger than CVR's
##sum((Pj(U) - (Pj(X1 %*% W12s$W1) + Pj(X2 %*% W12s$W2))/2)^2)
##W12snorm <- apply(rbind(W12s$W1, W12s$W2), 1, function(a)sqrt(sum(a^2)))
## compare Prec/Rec
##sum(W12snorm[c(1:10, 51:60)] != 0)/sum(W12snorm != 0);
##sum(W12snorm[c(1:10, 51:60)] != 0)/20;

##### binary response #####
set.seed(12)
mydata <- SimulateCVR(family = "binomial", n = 300, rank = 4, p1 = 50,
                    p2 = 70, pnz = 10, beta = c(2, 1, 0, 0))
X1 <- mydata$X1; X2 <- mydata$X2
Xlist <- list(X1 = X1, X2 = X2);
Y <- mydata$y
## out_cvr <- CVR(Y, Xlist, 4, neta = 5, nlam=25, family = "b", nfold = 5)
## out_cvr$sol$W[[1]];
## out_cvr$sol$W[[2]];
## plot.CVR(out_cvr)

##### Poisson response #####
set.seed(34)
mydata <- SimulateCVR(family = "p", n = 100, rank = 4, p1 = 50,
                    p2 = 70, pnz = 10, beta = c(0.2, 0.1, 0, 0))
X1 <- mydata$X1; X2 <- mydata$X2
Xlist <- list(X1 = X1, X2 = X2);
Y <- mydata$y
## etaseq <- 10^seq(-3, log10(0.95), len = 10)
## out_cvr <- CVR(Y, Xlist, 4, neta = 5, nlam = 25, family = "p", nfold = 5)
## out_cvr$sol$W[[1]];
## out_cvr$sol$W[[2]];
## plot.CVR(out_cvr)

```

**Description**

Perform canonical variate regression with a set of fixed tuning parameters.

**Usage**

```
cvrsolver(Y, Xlist, rank, eta, Lam, family, Wini, penalty, opts)
```

**Arguments**

Y	A response matrix. The response can be continuous, binary or Poisson.
Xlist	A list of covariate matrices. Cannot contain missing values.
rank	Number of pairs of canonical variates.
eta	Weight parameter between 0 and 1.
Lam	A vector of penalty parameters $\lambda$ for regularizing the loading matrices corresponding to the covariate matrices in Xlist.
family	Type of response. "gaussian" if Y is continuous, "binomial" if Y is binary, and "poisson" if Y is Poisson.
Wini	A list of initial loading matrices W's. It must be provided. See cvr and scca for using sCCA solution as the default.
penalty	Type of penalty on W's. "GL1" for rowwise sparsity and "L1" for entrywise sparsity.
opts	A list of options for controlling the algorithm. Some of the options are: standardization: need to standardize the data? Default is TRUE. maxITers: maximum number of iterations allowed in the algorithm. The default is 300. tol: convergence criterion. Stop iteration if the relative change in the objective is less than tol.

**Details**

CVR is used for extracting canonical variates and also predicting the response for multiple sets of covariates (Xlist = list(X1, X2)) and response (Y). The covariates can be, for instance, gene expression, SNPs or DNA methylation data. The response can be, for instance, quantitative measurement or binary phenotype. The criterion minimizes the objective function

$$(\eta/2) \sum_{k < j} \|X_k W_k - X_j W_j\|_F^2 + (1 - \eta) \sum_k l_k(\alpha, \beta, Y, X_k W_k) + \sum_k \rho_k(\lambda_k, W_k),$$

s.t.  $W_k' X_k' X_k W_k = I_r$ , for  $k = 1, 2, \dots, K$ .  $l_k(\cdot)$  are general loss functions with intercept  $\alpha$  and coefficients  $\beta$ .  $\eta$  is the weight parameter and  $\lambda_k$  are the regularization parameters.  $r$  is the rank, i.e. the number of canonical pairs. By adjusting  $\eta$ , one can change the weight of the first correlation term and the second prediction term.  $\eta = 0$  is reduced rank regression and  $\eta = 1$  is sparse CCA (with orthogonal constrained W's). By choosing appropriate  $\lambda_k$  one can induce sparsity of  $W_k$ 's to select useful variables for predicting Y.  $W_k$ 's with  $B_k$ 's and  $(\alpha, \beta)$  are iterated using an ADMM algorithm. See the reference for details.

**Value**

An object containing the following components

<code>iter</code>	The number of iterations the algorithm takes.
<code>W</code>	A list of fitted loading matrices.
<code>B</code>	A list of fitted $B_k$ 's.
<code>Z</code>	A list of fitted $B_k W_k$ 's.
<code>alpha</code>	Fitted intercept term in the general loss term.
<code>beta</code>	Fitted regression coefficients in the general loss term.
<code>objvals</code>	A sequence of the objective values.

**Author(s)**

Chongliang Luo, Kun Chen.

**References**

Chongliang Luo, Jin Liu, Dipak D. Dey and Kun Chen (2016) Canonical variate regression. *Biostatistics*, doi: 10.1093/biostatistics/kxw001.

**See Also**

[SimulateCVR](#), [CVR](#).

**Examples**

```
## see SimulateCVR for simulation examples, see CVR for parameter tuning.
```

---

mouse

*Data sets for the mouse body weight example*

---

**Description**

A list of 3 data frames that contains the genotype, gene expression and body-mass index of 294 mice. The data is already screened for quality control. For the raw data see the links below. For more details see the reference.

**Usage**

```
mouse
```

**Format**

A list of 3 data frames:

**geno** Mouse genotype. A data frame of 294 rows and 163 columns.

**expr** Mouse gene expression. A data frame of 294 rows and 215 columns.

**bmi** Mouse body-mass index. A data frame of 294 rows and 1 column.



**Source**

Mouse genotype: <http://www.genetics.org/cgi/content/full/genetics.110.116087/DC1>  
 Mouse gene expression: <ftp://ftp.ncbi.nlm.nih.gov/pub/geo/DATA/SeriesMatrix/GSE2814/>  
 Mouse body-mass index: <http://labs.genetics.ucla.edu/horvath/CoexpressionNetwork/MouseWeight/>.

**References**

Chongliang Luo, Jin Liu, Dipak D. Dey and Kun Chen (2016) Canonical variate regression. Biostatistics, doi: 10.1093/biostatistics/kxw001.

**Examples**

```
##### Mouse body weight example #####
data(mouse)
expr <- scale(as.matrix(mouse$expr))
geno <- scale(as.matrix(mouse$geno))
bmi <- as.matrix(mouse$bmi)
mouse.X <- list(X1 = expr, X2 = geno)
## Not run:
mouse.cvr <- CVR(bmi, mouse.X, rankseq = 2, etaseq = 0.04, family = "g", penalty = "L1")
plot(mouse.cvr)
plot(expr %% mouse.cvr$solution$W[[1]][, 2], geno %% mouse.cvr$solution$W[[2]][, 2])
cor(expr %% mouse.cvr$solution$W[[1]], geno %% mouse.cvr$solution$W[[2]])

## End(Not run)
```

---

plot.CVR

*Plot a CVR object.*


---

**Description**

Plot the tuning of CVR

**Usage**

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

**Arguments**

x                    A CVR object.  
 ...                  Other graphical parameters used in plot.

**Details**

The first plot is mean cv error vs  $\log(\lambda)$ . The type of mean cv error is decided by `type.measure` (see parameters of CVR). The selected  $\lambda$  is marked by a vertical line in the plot. The second plot is sparsity vs  $\log(\lambda)$ . Sparsity is the proportion of non-zero elements in fitted W1 and W2. The threshold is marked by a horizontal line. Press ENTER to see the second plot, which shows the tuning of  $\eta$ .

---

 SimulateCVR

*Generate simulation data.*


---

**Description**

Generate two sets of covariates and an univariate response driven by several latent factors.

**Usage**

```
SimulateCVR(family = c("gaussian", "binomial", "poisson"), n = 100,
            rank = 4, p1 = 50, p2 = 70, pnz = 10, sigmax = 0.2,
            sigmay = 0.5, beta = c(2, 1, 0, 0), standardization = TRUE)
```

**Arguments**

<code>family</code>	Type of response. "gaussian" for continuous response, "binomial" for binary response, and "poisson" for Poisson response. The default is "gaussian".
<code>n</code>	Number of rows. The default is 100.
<code>rank</code>	Number of latent factors generating the covariates. The default is 4.
<code>p1</code>	Number of variables in X1. The default is 50.
<code>p2</code>	Number of variables in X2. The default is 70.
<code>pnz</code>	Number of variables in X1 and X2 related to the signal. The default is 10.
<code>sigmax</code>	Standard deviation of normal noise in X1 and X2. The default is 0.2.
<code>sigmay</code>	Standard deviation of normal noise in Y. Only used when the response is Gaussian. The default is 0.5.
<code>beta</code>	Numeric vector, the coefficients used to generate response from the latent factors. The default is <code>c(2, 1, 0, 0)</code> .
<code>standardization</code>	Logical. If TRUE, standardize X1 and X2 before output. The default is TRUE.

**Details**

The latent factors in  $U$  are randomly generated normal vectors,

$$X_1 = U * V_1 + \sigma_x * E_1, X_2 = U * V_2 + \sigma_x * E_2, E_1, E_2 \text{ are } N(0,1) \text{ noise matrices.}$$

The nonzero entries of  $V_1$  and  $V_2$  are generated from `Uniform([-1,-0.5]U[0.5,1])`.

For Gaussian response,

$y = U * \beta + \sigma_y * e_y$ ,  $e_y$  is  $N(0,1)$  noise vector,

for binary response,

$y \sim rbinom(n, 1, 1/(1 + \exp(-U * \beta)))$ ,

and for Poisson response,

$y \sim rpois(n, \exp(U * \beta))$ .

See the reference for more details.

### Value

X1, X2	The two sets of covariates with dimensions $n * p1$ and $n * p2$ respectively.
y	The response vector with length $n$ .
U	The true latent factor matrix with dimension $n * rank$ .
beta	The coefficients used to generate response from U. The length is rank.
V1, V2	The true loading matrices for X1 and X2 with dimensions $p1 * rank$ and $p2 * rank$ . The first $pnz$ rows are nonzero.

### Author(s)

Chongliang Luo, Kun Chen.

### References

Chongliang Luo, Jin Liu, Dipak D. Dey and Kun Chen (2016) Canonical variate regression. *Biostatistics*, doi: 10.1093/biostatistics/kxw001.

### See Also

[CVR](#), [cvrsolver](#).

### Examples

```
set.seed(42)
mydata <- SimulateCVR(family = "g", n = 100, rank = 4, p1 = 50, p2 = 70,
                    pnz = 10, beta = c(2, 1, 0, 0))
X1 <- mydata$X1
X2 <- mydata$X2
Xlist <- list(X1 = X1, X2 = X2);
Y <- mydata$y
opts <- list(standardization = FALSE, maxIters = 300, tol = 0.005)
## use sparse CCA solution as initial values, see SparseCCA()
Wini <- SparseCCA(X1, X2, 4, 0.7, 0.7)
## perform CVR with fixed eta and lambda, see cvrsolver()
fit <- cvrsolver(Y, Xlist, rank = 4, eta = 0.5, Lam = c(1, 1),
                family = "gaussian", Wini, penalty = "GL1", opts)
## check sparsity recovery
fit$W[[1]];
fit$W[[2]];
## check orthogonality
```

```
X1W1 <- X1 %*% fit$W[[1]];
t(X1W1) %*% X1W1
```

---

 SparseCCA

*Sparse canonical correlation analysis.*


---

### Description

Get sparse CCA solutions of X1 and X2. Use CCA and CCA.permute from PMA package. See PMA package for details.

### Usage

```
SparseCCA(X1, X2, rank = 2, penaltyx1 = NULL, penaltyx2 = NULL,
           nperms = 25, ifplot = 0)
```

### Arguments

X1, X2	Numeric matrices representing the two sets of covariates. They should have the same number of rows and cannot contain missing values.
rank	The number of canonical variate pairs wanted. The default is 2.
penaltyx1, penaltyx2	Numeric vectors as the penalties to be applied to X1 and X2. The defaults are seq(0.1, 0.7, len = 20). See PMA package for details.
nperms	Number of times the data should be permuted. The default is 25. Don't use too small value. See PMA package for details.
ifplot	0 or 1. The default is 0 which means don't plot the result. See PMA package for details.

### Details

This function is generally used for tuning the penalties in sparse CCA if penaltyx1 and penaltyx2 are supplied as vectors. The CCA solution is based on PMD algorithm and the tuning is based on permutation. The fitted W1 and W2 are scaled so that the diagonals of  $W1'X1'X1W1$  and  $W2'X2'X2W2$  are all 1's.

Specifically, if a single value of mild penalty is provided for both penaltyx1 and penaltyx2, the result can be used as initial values of W's in CVR. For instance, with penaltyx1 = 0.7 and penaltyx2 = 0.7, the fitted W1 and W2 are only shrunked, but mostly not zero yet.

### Value

W1	Loading matrix corresponding to X1. $X1*W1$ gives the canonical variates from X1.
W2	Loading matrix corresponding to X2. $X2*W2$ gives the canonical variates from X2.

**Author(s)**

Chongliang Luo, Kun Chen.

**References**

Daniela M. Witten, Robert Tibshirani and Trevor Hastie (2009) A penalized matrix decomposition, with applications to sparse principal components and canonical correlation analysis. *Biostatistics* 10(3), 515-534.

Chongliang Luo, Jin Liu, Dipak D. Dey and Kun Chen (2016) Canonical variate regression. *Biostatistics*, doi: 10.1093/biostatistics/kxw001.

**See Also**

[CVR](#), [CCA](#), [CCA.permute](#).

# Index

\* **datasets**

alcohol, [3](#)

mouse, [8](#)

\* **package**

CVR-package, [2](#)

alcohol, [3](#)

CCA, [13](#)

CCA.permute, [13](#)

CVR, [4](#), [8](#), [11](#), [13](#)

CVR-package, [2](#)

cvrsolver, [5](#), [6](#), [11](#)

mouse, [8](#)

plot.CVR, [9](#)

SimulateCVR, [5](#), [8](#), [10](#)

SparseCCA, [5](#), [12](#)