Package ‘LUCIDus’

April 6, 2022

Type Package

Title Latent Unknown Clustering with Integrated Data

Version 2.1.5-1

Author Yinqi Zhao, David V. Conti, Cheng Peng, Zhao Yang

Maintainer Yinqi Zhao <yinqiz@usc.edu>

Description
An implementation of LUCID model (Peng (2019) <doi:10.1093/bioinformatics/btz667>). LUCID conducts integrated clustering using exposures, omics data (and outcome of interest). An EM algorithm is implemented to estimate MLE of LUCID model. LUCID features integrated variable selection, incorporation of missing omics data, bootstrap inference and visualization via Sankey diagram.

Depends R (>= 3.6.0)

License GPL-3

Encoding UTF-8

RoxygenNote 7.1.2

LazyData true

URL https://github.com/USCbiostats/LUCIDus

Suggests knitr, testthat (>= 3.0.0), rmarkdown

VignetteBuilder knitr

Imports mclust, nnet, networkD3, boot, glasso, glmnet, jsonlite, progress

Config/testthat/edition 3

NeedsCompilation no

Repository CRAN

Date/Publication 2022-04-06 10:32:30 UTC

R topics documented:

`boot.lucid` *Inference of LUCID model based on bootstrap resampling*

Description

Generate `R` bootstrap replicates of LUCID parameters and derive confidence interval (CI) base on bootstrap. Bootstrap replicates are generated based on nonparameteric resampling, implemented by `ordinary` method of codeboot::boot function.

Usage

```
boot.lucid(G, Z, Y, CoG = NULL, CoY = NULL, model, conf = 0.95, R = 100)
```

Arguments

`G` Exposures, a numeric vector, matrix, or data frame. Categorical variable should be transformed into dummy variables. If a matrix or data frame, rows represent observations and columns correspond to variables.

`Z` Omics data, a numeric matrix or data frame. Rows correspond to observations and columns correspond to variables.

`Y` Outcome, a numeric vector. Categorical variable is not allowed. Binary outcome should be coded as 0 and 1.

`CoG` Optional, covariates to be adjusted for estimating the latent cluster. A numeric vector, matrix or data frame. Categorical variable should be transformed into dummy variables.

`CoY` Optional, covariates to be adjusted for estimating the association between latent cluster and the outcome. A numeric vector, matrix or data frame. Categorical variable should be transformed into dummy variables.

`model` A LUCID model fitted by `est.lucid`.

`conf` A numeric scalar between 0 and 1 to specify confidence level(s) of the required interval(s).

`R` An integer to specify number of bootstrap replicates for LUCID model. If feasible, it is recommended to set R > 1000. However, the convergence speed of LUCID varies greatly depending on data. If it takes very long time to run 1000 replicates, it is recommend to set smaller values for R, such as 200.

Value

A list, containing the following components:

`beta` effect estimate for each exposure

`mu` cluster-specific mean for each omics feature

`gamma` effect estiamte for the association btween latent cluster and outcome

`bootstrap` The `boot` object returned by `boot:boot`

Examples

```
## Not run:
# use simulated data
G <- sim_data$G
Z <- sim_data$Z
Y_normal <- sim_data$Y_normal

# fit lucid model
fit1 <- est.lucid(G = G, Z = Z, Y = Y_normal, family = "normal", K = 2,
seed = 1008)

# conduct bootstrap resampling
boot1 <- boot.lucid(G = G, Z = Z, Y = Y_normal, model = fit1, R = 100)

# check distribution for bootstrap replicates of the variable of interest
plot(boot1$bootstrap, 1)

# use 90% CI
boot2 <- boot.lucid(G = G, Z = Z, Y = Y_normal, model = fit1, R = 100, conf = 0.9)

## End(Not run)
```

`est.lucid` *Fit LUCID model to conduct integrated clustering*

Description

The Latent Unknown Clustering with Integrated Data (LUCID) performs integrative clustering using multi-view data. LUCID model is estimated via EM algorithm for model-based clustering. It also features variable selection, integrated imputation, bootstrap inference and visualization via Sankey diagram.

Usage

```
est.lucid(
  G,
  Z,
  Y,
  CoG = NULL,
  CoY = NULL,
  K = 2,
  family = c("normal", "binary"),
  useY = TRUE,
  tol = 0.001,
  max_itr = 1000,
  max_tot.itr = 10000,
  Rho_G = 0,
  Rho_Z_Mu = 0,
  Rho_Z_Cov = 0,
  modelName = "VVV",
  seed = 123,
  init_impute = c("mclust", "lod"),
  init_par = c("mclust", "random"),
  verbose = FALSE
)
```

Arguments

`G` Exposures, a numeric vector, matrix, or data frame. Categorical variable should be transformed into dummy variables. If a matrix or data frame, rows represent observations and columns correspond to variables.

`Z` Omics data, a numeric matrix or data frame. Rows correspond to observations and columns correspond to variables.

`Y` Outcome, a numeric vector. Categorical variable is not allowed. Binary outcome should be coded as 0 and 1.

`CoG` Optional, covariates to be adjusted for estimating the latent cluster. A numeric vector, matrix or data frame. Categorical variable should be transformed into dummy variables.

`CoY` Optional, covariates to be adjusted for estimating the association between latent cluster and the outcome. A numeric vector, matrix or data frame. Categorical variable should be transformed into dummy variables.

`K` Number of latent clusters. An integer greater or equal to 2. User can use `lucid` to determine the optimal number of latent clusters.

`family` Distribution of outcome. For continuous outcome, use "normal"; for binary outcome, use "binary". Default is "normal".

`useY` Flag to include information of outcome when estimating the latent cluster. Default is TRUE.

`tol` Tolerance for convergence of EM algorithm. Default is 1e-3.

`max_itr` Max number of iterations for EM algorithm.

`max_tot.itr` Max number of total iterations for `est.lucid` function. `est.lucid` may conduct EM algorithm for multiple times if the algorithm fails to converge.

`Rho_G` A scalar. Penalty to conduct LASSO regularization and obtain a sparse estimation for effect of exposures. If user wants to tune the penalty, use the wrapper function `lucid`

`Rho_Z_Mu` A scalar. Penalty to conduct LASSO regularization and obtain a sparse estimation of cluster-specific mean for omics data. If user wants to tune the penalty, use the wrapper function `lucid`

`Rho_Z_Cov` A scalar. Penalty to conduct graphic LASSO regularization and obtain a sparse estimation of cluster-specific variance-covariance matrices for omics data. If user wants to tune the penalty, use the wrapper function `lucid`

`modelName` The variance-covariance structure for omics data. See `mclust::mclustModelNames` for details.

`seed` An integer to initialize the EM algorithm or imputing missing values. Default is 123.

`init_impute` Method to initialize the imputation of missing values in LUCID. "mclust" will use `mclust:imputeData` to implement EM Algorithm for Unrestricted General Location Model to impute the missing values in omics data; `lod` will initialize the imputation via relacing missing values by LOD / sqrt(2). LOD is determined by the minimum of each variable in omics data.

`init_par` Method to initialize the EM algorithm. "mclust" will use mclust model to initialize parameters; "random" initialize parameters from uniform distribution.

`verbose` A flag indicates whether detailed information for each iteration of EM algorithm is printed in console. Default is FALSE.

Value

A list which contains the several features of LUCID, including:

`pars` Estimates of parameters of LUCID, including beta (effect of exposure), mu (cluster-specific mean for omics data), sigma (cluster-specific variance-covariance matrix for omics data) and gamma (effect estimate of association between latent cluster and outcome)

`K` Number of latent cluster

`modelName` Geometric model to estiamte variance-covariance matrix for omics data

`likelihood` The log likelihood of the LUCID model

`post.p` Posterior inclusion probability (PIP) for assigning observation i to latent cluster j

`Z` If missing values are observed, this is the complet dataset for omics data with missing values imputed by LUCID

References

Cheng Peng, Jun Wang, Isaac Asante, Stan Louie, Ran Jin, Lida Chatzi, Graham Casey, Duncan C Thomas, David V Conti, A Latent Unknown Clustering Integrating Multi-Omics Data (LUCID) with Phenotypic Traits, Bioinformatics, btz667, https://doi.org/10.1093/bioinformatics/btz667.

Examples

```
## Not run:
# use simulated data
G <- sim_data$G
Z <- sim_data$Z
Y_normal <- sim_data$Y_normal
Y_binary <- sim_data$Y_binary
cov <- sim_data$Covariate

# fit LUCID model with continuous outcome
fit1 <- est.lucid(G = G, Z = Z, Y = Y_normal, family = "normal", K = 2,
seed = 1008)

# fit LUCID model with block-wise missing pattern in omics data
Z_miss_1 <- Z
Z_miss_1[sample(1:nrow(Z), 0.3 * nrow(Z)), ] <- NA
fit2 <- est.lucid(G = G, Z = Z_miss_1, Y = Y_normal, family = "normal", K = 2)

# fit LUCID model with sporadic missing pattern in omics data
Z_miss_2 <- Z
index <- arrayInd(sample(length(Z_miss_2), 0.3 * length(Z_miss_2)), dim(Z_miss_2))
Z_miss_2[index] <- NA
# initialize imputation by imputing
fit3 <- est.lucid(G = G, Z = Z_miss_2, Y = Y_normal, family = "normal",
K = 2, seed = 1008, init_impute = "lod")
LOD
# initialize imputation by mclust
fit4 <- est.lucid(G = G, Z = Z_miss_2, Y = Y, family = "normal", K = 2,
seed = 123, init_impute = "mclust")

# fit LUCID model with binary outcome
fit5 <- est.lucid(G = G, Z = Z, Y = Y_binary, family = "binary", K = 2,
seed = 1008)

# fit LUCID model with covariates
fit6 <- est.lucid(G = G, Z = Z, Y = Y_binary, CoY = cov, family = "binary",
K = 2, seed = 1008)

# use LUCID model to conduct integrated variable selection
# select exposure
fit6 <- est.lucid(G = G, Z = Z, Y = Y_normal, CoY = NULL, family = "normal",
K = 2, seed = 1008, Rho_G = 0.1)
# select omics data
fit7 <- est.lucid(G = G, Z = Z, Y = Y_normal, CoY = NULL, family = "normal",
K = 2, seed = 1008, Rho_Z_Mu = 90, Rho_Z_Cov = 0.1, init_par = "random")


## End(Not run)
```

fill_data *Impute missing data by optimizing the likelihood function*

Description

Impute missing data by optimizing the likelihood function

Usage

```
fill_data(obs, mu, sigma, p, index)
```

Arguments

`obs` a vector of length M

`mu` a matrix of size M x K

`sigma` a matrix of size M x M x K

`p` a vector of length K

`index` a vector of length M, indicating whether a value is missing or not in the raw data

Value

an observation with updated imputed value

`gen_ci` *generate bootstrp ci (normal, basic and percentile)*

Description

generate bootstrp ci (normal, basic and percentile)

Usage

```
gen_ci(x, conf = 0.95)
```

Arguments

`x` an object return by boot function

`conf` A numeric scalar between 0 and 1 to specify confidence level(s) of the required interval(s).

Value

a matrix, the first column is t0 statistic from original model

helix_data *HELIX data*

Description

The Human Early-Life Exposome (HELIX) project is multi-center research project that aims to characterize early-life environmental exposures and associate these with omics biomarkers and child health outcomes (Vrijheid, 2014. doi: 10.1289/ehp.1307204). We used a subset of HELIX data from Exposome Data Challenge 2021 (hold by ISGlobal) as an example to illustrate LUCID model.

Usage

```
helix_data
```

Format

A list with 4 matrices corresponding to exposures (G), omics data (Z), outcome (Y) and covariates (CoY)

exposure 8 variables, representing children/maternal exposure to environmental pollutants

omics 10 proteins

outcome A continuous outcome for BMI-z score based on WHO standard, A binary outcome for body mass index categories at 6-11 years old based on WHO reference (0: Thinness or Normal; 1: Overweight or Obese)

covariate 3 covariates including mother's bmi, child sex, maternal age

lucid *A wrapper function to perform model selection for LUCID*

Description

Given a grid of K and L1 penalties (incluing Rho_G, Rho_Z_mu and Rho_Z_Cov), fit LUCID model over all combinations of K and L1 penalties to determine the optimal penalty.

Usage

```
lucid(
  G,
  Z,
  Y,
  CoG = NULL,
  CoY = NULL,
  family = "normal",
  useY = TRUE,
  K = 2:5,
  Rho_G = 0,
  Rho_Z_Mu = 0,
  Rho_Z_Cov = 0,
  ...
)
```

Arguments

`G` Exposures, a numeric vector, matrix, or data frame. Categorical variable should be transformed into dummy variables. If a matrix or data frame, rows represent observations and columns correspond to variables.

`Z` Omics data, a numeric matrix or data frame. Rows correspond to observations and columns correspond to variables.

`Y` Outcome, a numeric vector. Categorical variable is not allowed. Binary outcome should be coded as 0 and 1.

`CoG` Optional, covariates to be adjusted for estimating the latent cluster. A numeric vector, matrix or data frame. Categorical variable should be transformed into dummy variables.

`CoY` Optional, covariates to be adjusted for estimating the association between latent cluster and the outcome. A numeric vector, matrix or data frame. Categorical variable should be transformed into dummy variables.

`family` Distribution of outcome. For continuous outcome, use "normal"; for binary outcome, use "binary". Default is "normal".

`useY` Flag to include information of outcome when estimating the latent cluster. Default is TRUE.

`K` Number of latent clusters. An integer greater or equal to 2.

`Rho_G` A scalar or a vector. Penalty to conduct LASSO regularization and obtain a sparse estimation for effect of exposures. If a vector, `lucid` will fit lucid model over the grid of penalties.

`Rho_Z_Mu` A scalar or a vector. Penalty to conduct LASSO regularization and obtain a sparse estimation of cluster-specific mean for omics data. If a vector, `lucid` will fit lucid model over the grid of penalties.

`Rho_Z_Cov` Penalty to conduct graphic LASSO regularization and obtain a sparse estimation of cluster-specific variance-covariance matrices for omics data. If a vector, `lucid` will fit lucid model over the grid of penalties.

`...` Other parameters passed to `est.lucid`

Value

A list:

`best_model` the best model over different combination of tuning parameters

`tune_list` a data frame contains combination of tuning parameters and c orresponding BIC

`res_model` a list of LUCID models corresponding to each combination of tuning parameters

Examples

```
## Not run:
# use simulated data
G <- sim_data$G
Z <- sim_data$Z
Y_normal <- sim_data$Y_normal

# find the optimal model over the grid of K
tune_K <- lucid(G = G, Z = Z, Y = Y_normal, useY = FALSE, tol = 1e-3,
seed = 1, K = 2:5)

# tune penalties
tune_Rho_G <- lucid(G = G, Z = Z, Y = Y_normal, useY = FALSE, tol = 1e-3,
seed = 1, K = 2, Rho_G = c(0.1, 0.2, 0.3, 0.4))
tune_Rho_Z_mu <- lucid(G = G, Z = Z, Y = Y_normal, useY = FALSE, tol = 1e-3,
seed = 1, K = 2, Rho_Z_mu = c(10, 20, 30, 40))
tune_Rho_Z_Cov <- lucid(G = G, Z = Z, Y = Y_normal, useY = FALSE, tol = 1e-3,
seed = 1, K = 2, Rho_Z_Cov = c(0.1, 0.2, 0.3))


## End(Not run)
```

---

plot_lucid *Visualize LUCID model through a Sankey diagram*

Description

In the Sankey diagram, each node either represents a variable (exposure, omics or outcome) or a latent cluster. Each line represents an association. The color of the node represents variable type, either exposure, omics or outcome. The width of the line represents the effect size of a certain association; the color of the line represents the direction of a certain association.

Usage

```
plot_lucid(
  x,
  G_color = "dimgray",
  X_color = "#eb8c30",
  Z_color = "#2fa4da",
  Y_color = "#afa58e",
  pos_link_color = "#67928b",
  neg_link_color = "#d1e5eb",
  fontsize = 7
)
```

Arguments

x A LUCID model fitted by `est.lucid`

`G_color`	Color of node for exposure
`X_color`	Color of node for latent cluster
`Z_color`	Color of node for omics data
`Y_color`	Color of node for outcome
`pos_link_color`	Color of link corresponds to positive association
`neg_link_color`	Color of link corresponds to negative association
`fontsize`	Font size for annotation

Value

A DAG graph created by `sankeyNetwork`

Examples

```
## Not run:
# prepare data
G <- sim_data$G
Z <- sim_data$Z
Y_normal <- sim_data$Y_normal
Y_binary <- sim_data$Y_binary
cov <- sim_data$Covariate

# plot lucid model
fit1 <- est.lucid(G = G, Z = Z, Y = Y_normal, CoY = NULL, family = ”normal”,
K = 2, seed = 1008)
plot_lucid(fit1)

# change node color
plot_lucid(fit1, G_color = ”yellow”)
plot_lucid(fit1, Z_color = ”red”)

# change link color
plot_lucid(fit1, pos_link_color = ”red”, neg_link_color = ”green”)

## End(Not run)
```

`predict_lucid` *Predict cluster assignment and outcome based on LUCID model*

Description

Predict cluster assignment and outcome based on LUCID model

Usage

```
predict_lucid(model, G, Z, Y = NULL, CoG = NULL, CoY = NULL, response = TRUE)
```

Arguments

`model`	A model fitted and returned by `est.lucid`
`G`	Exposures, a numeric vector, matrix, or data frame. Categorical variable should be transformed into dummy variables. If a matrix or data frame, rows represent observations and columns correspond to variables.
`Z`	Omics data, a numeric matrix or data frame. Rows correspond to observations and columns correspond to variables.
`Y`	Outcome, a numeric vector. Categorical variable is not allowed. Binary outcome should be coded as 0 and 1.
`CoG`	Optional, covariates to be adjusted for estimating the latent cluster. A numeric vector, matrix or data frame. Categorical variable should be transformed into dummy variables.
`CoY`	Optional, covariates to be adjusted for estimating the association between latent cluster and the outcome. A numeric vector, matrix or data frame. Categorical variable should be transformed into dummy variables.
`response`	If TRUE, when predicting binary outcome, the response will be returned. If FALSE, the linear predictor is returned.

Value

A list contains predicted latent cluster and outcome for each observation

Examples

```
## Not run:
# prepare data
G <- sim_data$G
Z <- sim_data$Z
Y_normal <- sim_data$Y_normal

# fit lucid model
fit1 <- est.lucid(G = G, Z = Z, Y = Y_normal, K = 2, family = "normal")

# prediction on training set
pred1 <- predict_lucid(model = fit1, G = G, Z = Z, Y = Y_normal)
pred2 <- predict_lucid(model = fit1, G = G, Z = Z)

## End(Not run)
```

`print.lucid` *Print the output of* `est.lucid`

Description

Print the output of `est.lucid`

Usage

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

Arguments

`x`	An object of LUCID model, returned by `est.lucid`
`...`	Other arguments to be passed to `print`

`print.sumlucid` *Print the output of LUCID in a nicer table*

Description

Print the output of LUCID in a nicer table

Usage

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

Arguments

`x`	An object returned by `summary.lucid`
`...`	Other parameters to be passed to `print`

`sim_data` *A simulated dataset for LUCID*

Description

This is an example dataset to illustrate LUCID model. It is simulated by assuming there are 2 latent clusters in the data. We assume the exposures are associated with latent cluster which ultimately affects the PFAS concentration and liver injury in children. The latent clusters are also characterized by differential levels of metabolites.

Usage

```
sim_data
```

Format

A list with 5 matrices corresponding to exposures (G), omics data (Z), a continuous outcome, a binary outcome and 2 covariates (can be used either as CoX or CoY). Each matrice contains 2000 observations.

G 10 exposures

Z 10 metabolites

Y_normal Outcome, PFAS concentration in children

Y_bninary Bianry outcome, liver injury status

Covariates 2 continous covariates, can be treated as either CoX or CoY

X Latent clusters

summary_lucid *Summarize results of LUCID model*

Description

Summarize results of LUCID model

Usage

```
summary_lucid(object, boot.se = NULL)
```

Arguments

`object`	A LUCID model fitted by `est.lucid`
`boot.se`	An object returned by `boot.lucid`, which contains the bootstrap confidence intervals

Examples

```
## Not run:
# use simulated data
G <- sim_data$G
Z <- sim_data$Z
Y_normal <- sim_data$Y_normal

# fit lucid model
fit1 <- est.lucid(G = G, Z = Z, Y = Y_normal, family = "normal", K = 2,
seed = 1008)

# conduct bootstrap resampling
boot1 <- boot.lucid(G = G, Z = Z, Y = Y_normal, model = fit1, R = 100)

# summarize lucid model
summary_lucid(fit1)
```

```
# summarize lucid model with bootstrap CIs
summary_lucid(fit1, boot.se = boot1)

## End(Not run)
```

Index