ExposomeX handbook (v1.0.0)

ExposomeX: An integrated platform to expedite the discovery of "Exposure-Biology-Disease" nexus

Bin Wang, Mingiang Fang, and ExposomeX team

2023-1-1

Author list

- Bin Wang, School of Public Health, Peking University
- Mingliang Fang, Department of Environmental Science and Engineering, Fudan University
- Changxin Lan, School of Public Health, Peking University
- Guohuan Zhang, School of Public Health, Peking University
- Mengyuan Ren, School of Public Health, Peking University
- Ning Gao, School of Public Health, Peking University
- Weinan Lin, School of Public Health, Peking University
- Yanqiu Feng, School of Public Health, Peking University
- Yuting Wang, School of Public Health, Peking University

Contents

Quick start

```
#devtools::install_github("ExposomeX/exposomex", force = TRUE)
library(exposomex)

#Convert chemical name to CAS.RN
res <- InitDb()

res1 = LoadDb(PID = res$PID,
              UseExample = "example#1")

res2 = ExpoConv(PID = res$PID,
                From = "chemical",
                To = "cas.rn",
                Keys = "default")
res2
```

```
## # A tibble: 18 x 3
##    chemical cas.rn       EXC
##    <chr>    <chr>        <chr>
##  1 zinc     1162648-93-2 EX:C00212
##  2 zinc     24359-56-6   EX:C00212
##  3 zinc     25016-79-9   EX:C00212
##  4 zinc     23713-49-7   EX:C00212
##  5 zinc     7440-66-6    EX:C00212
##  6 zinc     7646-85-7    EX:C00212
##  7 zinc     9025-42-7    EX:C00212
##  8 water    1310-73-2    EX:C46309
##  9 water    39388-36-8   EX:C46309
## 10 water    13670-17-2   EX:C46309
## 11 water    14314-42-2   EX:C46309
## 12 water    558440-22-5  EX:C46309
## 13 water    7732-18-5    EX:C46309
## 14 water    67747-09-5   EX:C46309
## 15 water    7789-20-0    EX:C46309
## 16 water    14280-30-9   EX:C46309
## 17 water    3352-57-6    EX:C46309
## 18 water    60426-60-0   EX:C46309
```

```
FuncExit(PID = res$PID)
```

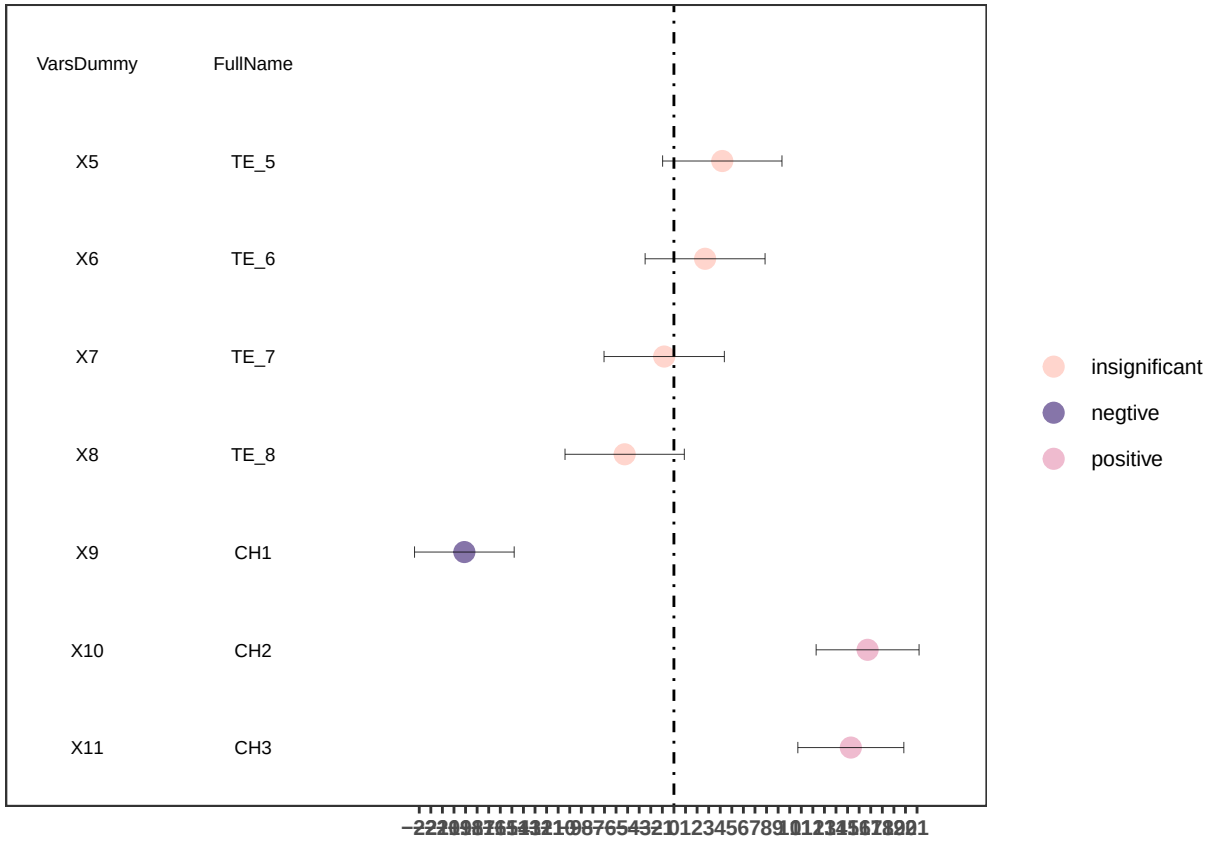
```
## [1] "Success to exit. Thanks for using ExposomeX platform!"
```

```
#Calculate the association
res <- InitCros()

res1 = LoadCros(PID = res$PID,
                UseExample = "example#1")

res2 = TransScale(PID = res$PID,
                  Group = "T",
                  Vars = "all.x",
                  Method = "normal")

res3 = CrosAsso(PID = res$PID,
```

```
                EpiDesign = "cross.sectional",
                VarsY = "Y1",
                VarsX = "X5,X6,X7,X8,X9,X10,X11",
                VarsN = "single.factor",
                VarsSel = FALSE,
                VarsSelThr = 0.1,
                IncCova = TRUE,
                Family = "gaussian")
res3$Y1_single.factor_cross.sectional_gaussian
```

```
## # A tibble: 7 x 13
##   SerialNo Vars.dummy beta.va~1    ci_l    ci_h  p.value std.e~2 formula    Beta
##   <chr>    <chr>          <dbl>   <dbl>   <dbl>    <dbl>   <dbl> <chr>     <dbl>
## 1 X5       X5              4.18   -0.978   9.33  1.14e- 1    2.63 Y1~X5     4.18
## 2 X6       X6              2.69   -2.49    7.87  3.10e- 1    2.64 Y1~X6     2.69
## 3 X7       X7             -0.841  -6.04    4.36  7.52e- 1    2.65 Y1~X7    -0.841
## 4 X8       X8             -4.26   -9.41    0.897 1.08e- 1    2.63 Y1~X8    -4.26
## 5 X9       X9            -18.1   -22.4   -13.8   8.22e-14    2.20 Y1~X9   -18.1
## 6 X10      X10            16.7    12.3    21.2   1.08e-11    2.27 Y1~X10   16.7
## 7 X11      X11            15.3    10.7    19.9   9.33e-10    2.34 Y1~X11   15.3
## # ... with 4 more variables: Beta_CI_L <dbl>, Beta_CI_H <dbl>, FullName <chr>,
## #   GroupName <chr>, and abbreviated variable names 1: beta.value, 2: std.error
```

```
res4 = VizCrosAsso(PID = res$PID,
                   VarsY = "Y1",
                   VarsN ="single.factor",
                   Layout = "forest",
                   Brightness = "dark",
                   Palette = "default1")
res4$Y1_single.factor_forest_dark_default1
```



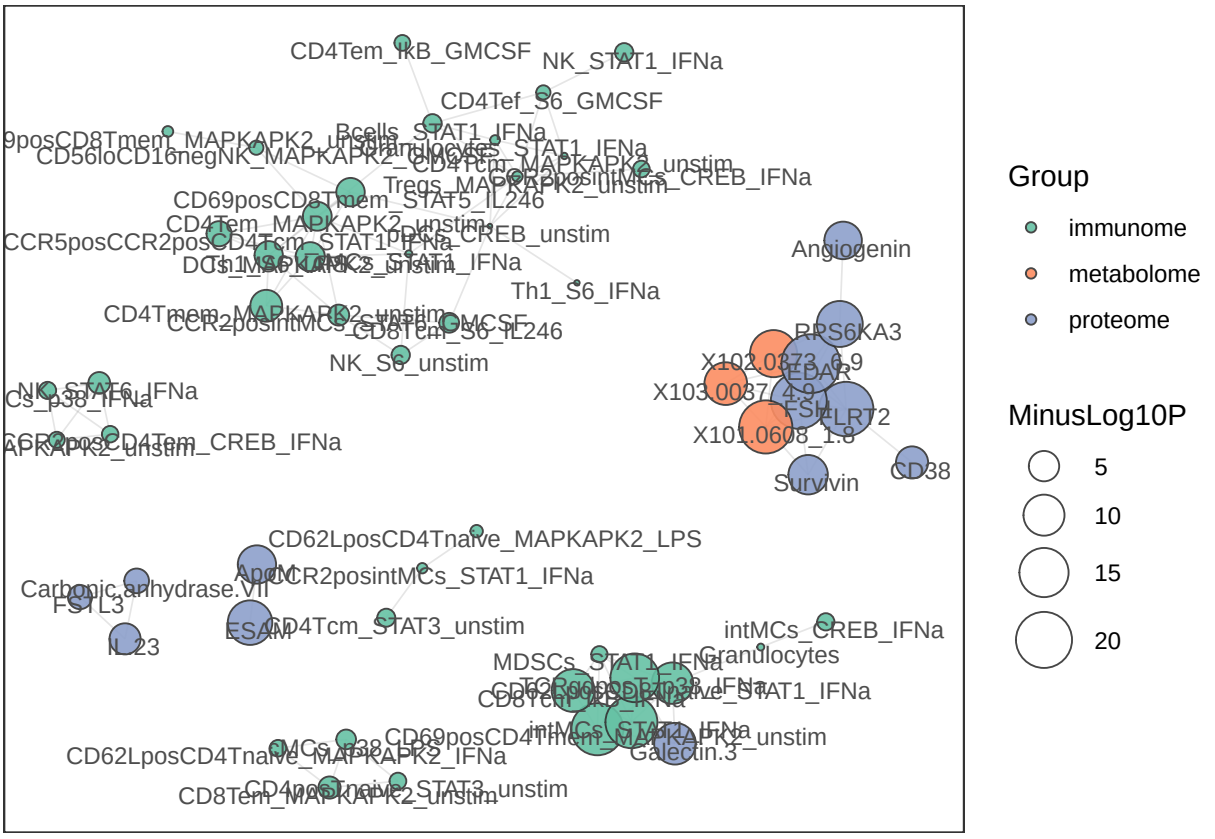

```
FuncExit(PID = res$PID)
```

```
## [1] "Success to exit. Thanks for using ExposomeX platform!"
```

```
#Build molti-omic prediction model
res <- InitMO()

res1 <- LoadMO(PID = res$PID,
               UseExample = "example#1")

res2 <- MulOmicsCros(PID = res$PID,
                     OmicGroups = "immunome,metabolome,proteome",
                     VarsY = "Y1",
                     VarsC = "all.c",
                     TuneMethod = "default",
                     TuneNum = 5,
                     RsmpMethod = "cv",
                     VarsImpThr = 0.85,
                     SG_Lrns ="enet,rf")
res2$SGModel_summary
```

```
##           Step1 Step2 resampling.method rsq_train  rsq_test
## 1 Step1 by enet  enet              cv-5 0.9707111 0.9673747
## 2   Step1 by rf  enet              cv-5 0.9817142 0.9793555
## 3 Step1 by enet    rf              cv-5 0.9917177 0.9667765
## 4   Step1 by rf    rf              cv-5 0.9958974 0.9814641
```

```
res3 <- VizMulOmicCros(PID = res$PID,
                       VarsY = "Y1",
                       NodeNum =100,
                       EdgeThr= 0.45,
                       Layout = "force-directed",
                       Brightness = "light",
                       Palette = 'default1')
res3$Networkplot$EN
```




```
#Find the biological link in protein-protein interaction mode
res = InitBioLink()

res1 = LoadBioLink(PID = res$PID,
                   UseExample = "example#1")

res2 = ConvToExpoID(PID = res$PID)
res2
```

```
## # A tibble: 221 x 8
##    SerialNo FullName GroupName DiseaseID   ExposureID MetabolomeID Prote~1 EX
##    <chr>    <chr>    <chr>     <chr>       <chr>      <chr>        <chr>   <chr>
##  1 Y1       CSNU     disease   OMIM:220100 <NA>       <NA>         <NA>    EX:D~
##  2 Y2       SCZD     disease   OMIM:181500 <NA>       <NA>         <NA>    EX:D~
##  3 Y3       TGCT     disease   OMIM:273300 <NA>       <NA>         <NA>    EX:D~
##  4 Y4       RASJ     disease   OMIM:604302 <NA>       <NA>         <NA>    EX:D~
##  5 Y5       HH3      disease   OMIM:244200 <NA>       <NA>         <NA>    EX:D~
```

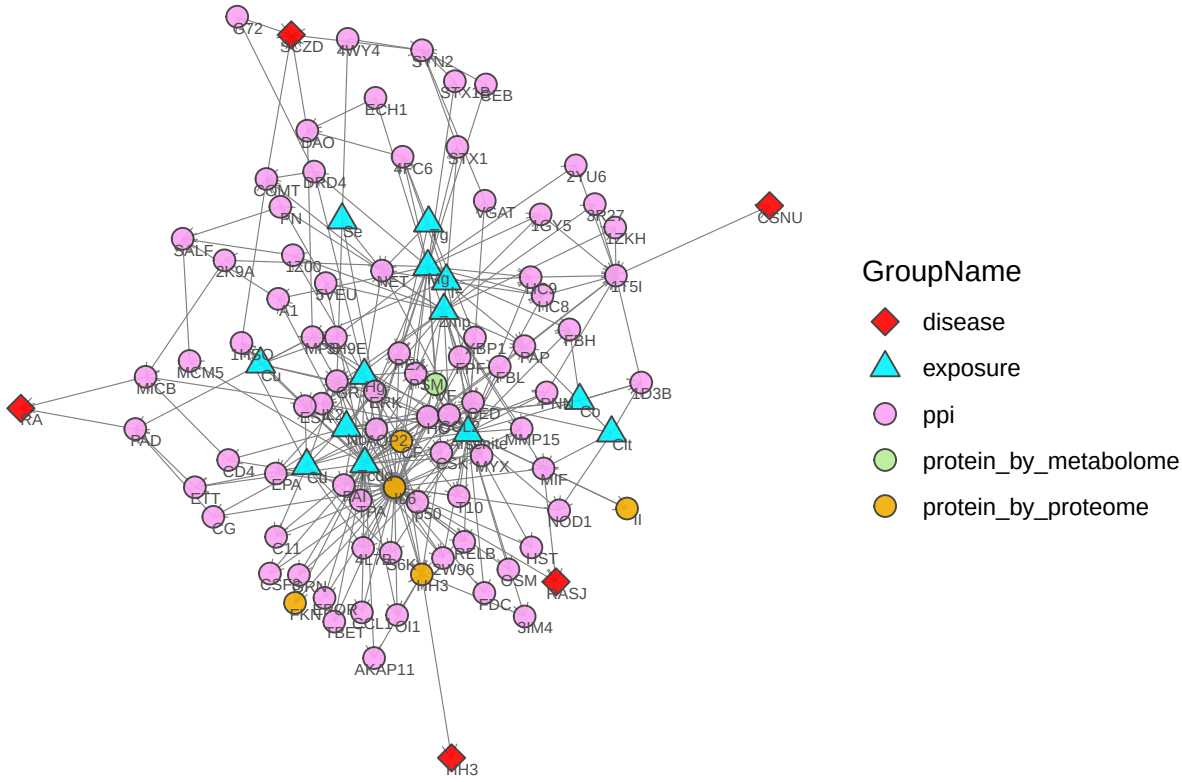
```
##  6 Y6       DECRD    disease   OMIM:616034 <NA>       <NA>         <NA>    EX:D~
##  7 Y7       PMDS1    disease   OMIM:261550 <NA>       <NA>         <NA>    EX:D~
##  8 Y8       CRMCC2   disease   OMIM:617341 <NA>       <NA>         <NA>    EX:D~
##  9 Y9       FSHD1    disease   OMIM:158900 <NA>       <NA>         <NA>    EX:D~
## 10 Y10      RA       disease   OMIM:180300 <NA>       <NA>         <NA>    EX:D~
## # ... with 211 more rows, and abbreviated variable name 1: ProteomeID
```

```
res3 = BioLink(PID = res$PID,
               Mode = "PPI",
               ChemCas = "default",
               ChemInchikey = "default",
               DiseaseID = "default",
               MetabolomeID = "default",
               MetBiospec = "blood",
               ProteomeID = "default")
res3
```

```
## $Edges
## # A tibble: 310 x 9
##    source    target     intera~1 sourc~2 targe~3 datab~4 edge_~5 sourc~6 targe~7
##    <chr>     <chr>      <chr>    <chr>   <chr>   <chr>     <dbl> <chr>   <chr>
##  1 EX:E00321 EX:P000667 Active   chemic~ protein toxcast       1 EX:E00~ EX:P00~
##  2 EX:E00321 EX:P000608 Active   chemic~ protein toxcast       1 EX:E00~ EX:P00~
##  3 EX:E00321 EX:P000493 Active   chemic~ protein toxcast       1 EX:E00~ EX:P00~
##  4 EX:E00321 EX:P000687 Active   chemic~ protein toxcast       1 EX:E00~ EX:P00~
##  5 EX:E03544 EX:P000608 Active   chemic~ protein toxcast       1 EX:E03~ EX:P00~
##  6 EX:E00321 EX:P000420 Active   chemic~ protein toxcast       1 EX:E00~ EX:P00~
##  7 EX:E47866 EX:P000466 Active   chemic~ protein toxcast       1 EX:E47~ EX:P00~
##  8 EX:E04534 EX:P000298 Active   chemic~ protein toxcast       1 EX:E04~ EX:P00~
##  9 EX:E04534 EX:P166812 Active   chemic~ protein toxcast       1 EX:E04~ EX:P16~
## 10 EX:E00321 EX:P000153 Active   chemic~ protein toxcast       1 EX:E00~ EX:P00~
## # ... with 300 more rows, and abbreviated variable names 1: interaction,
## #   2: source.class, 3: target.class, 4: database, 5: edge_type,
## #   6: source.label, 7: target.label
##
## $Nodes
## # A tibble: 126 x 3
##    id         label group
##    <chr>      <chr> <chr>
##  1 EX:E00321  Hg    exposure
##  2 EX:P000667 CCL2  ppi
##  3 EX:P000608 PAI   ppi
##  4 EX:P000493 TPA   ppi
##  5 EX:P000687 T10   ppi
##  6 EX:E03544  Tcdd  exposure
##  7 EX:P000420 EPA   ppi
##  8 EX:E47866  Zmp   exposure
##  9 EX:P000466 NET   ppi
## 10 EX:E04534  Tg    exposure
## # ... with 116 more rows
```

```
res4 = VizBioLink(PID = res$PID,
                  Mode = 'PPI',
                  Layout = "force-directed",
                  Brightness = "dark",
```

```
                  Palette = "default1")
res4
```



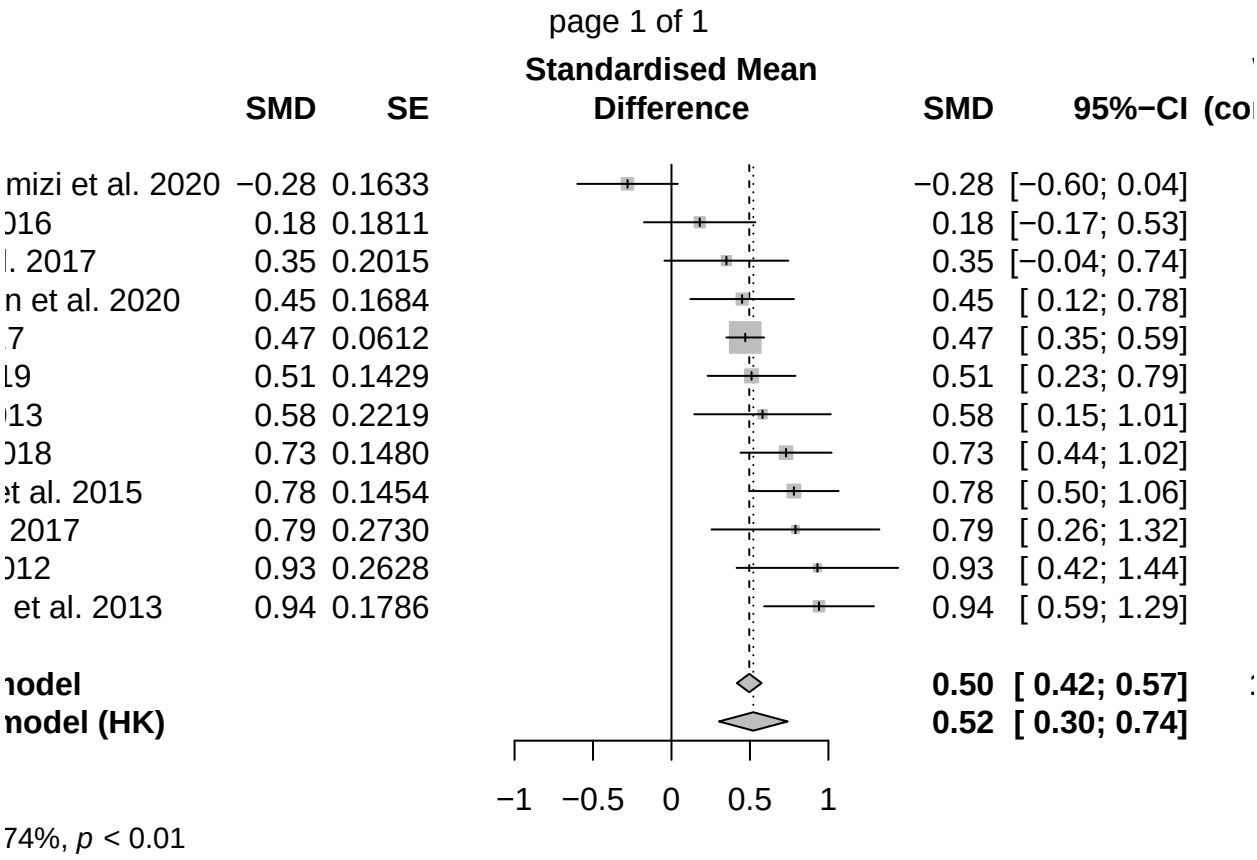

```
#Pool the effect value by meta-analysis
res = InitMeta()

res1 = LoadMeta(PID = res$PID,
                UseExample = "example#1")

res2 = MetaEffect(PID = res$PID)

res2$MetaEffect_References
```

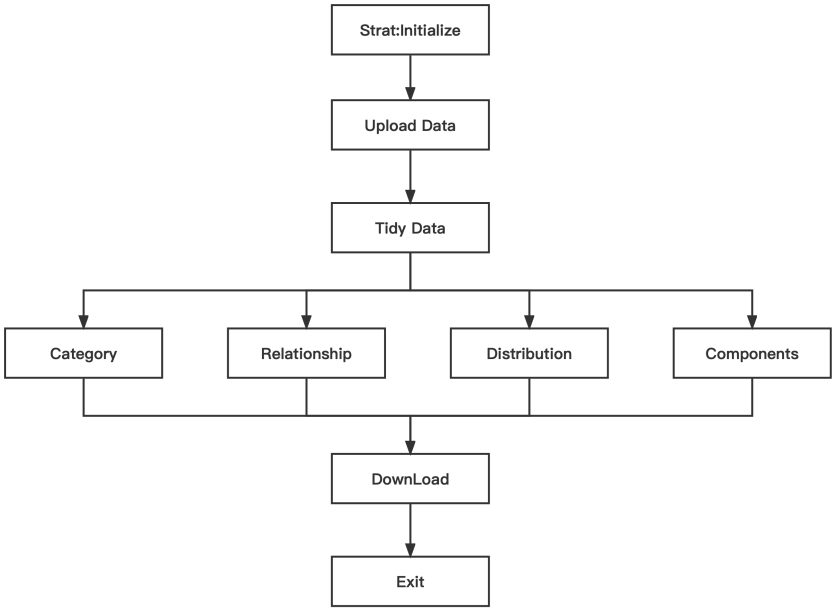
```
## $`EX_C00033&EX_D13329[references]`
## # A tibble: 12 x 29
##    PMID  Study~1 Expos~2 Recep~3 Study~4 Mediu~5 Expos~6 Corre~7 Outco~8 Autho~9
##    <chr> <chr>   <chr>   <chr>   <chr>   <chr>   <chr>   <chr>   <chr>   <chr>
##  1 2575~ <NA>    copper~ <NA>    <NA>    <NA>    <NA>    <NA>    polycy~ <NA>
##  2 2750~ <NA>    copper~ <NA>    <NA>    <NA>    <NA>    <NA>    polycy~ <NA>
##  3 2202~ <NA>    copper~ <NA>    <NA>    <NA>    <NA>    <NA>    polycy~ <NA>
##  4 2348~ <NA>    copper~ <NA>    <NA>    <NA>    <NA>    <NA>    polycy~ <NA>
##  5 2332~ <NA>    copper~ <NA>    <NA>    <NA>    <NA>    <NA>    polycy~ <NA>
##  6 2831~ <NA>    copper~ <NA>    <NA>    <NA>    <NA>    <NA>    polycy~ <NA>
##  7 3094~ <NA>    copper~ <NA>    <NA>    <NA>    <NA>    <NA>    polycy~ <NA>
##  8 3064~ <NA>    copper~ <NA>    <NA>    <NA>    <NA>    <NA>    polycy~ <NA>
##  9 3250~ <NA>    copper~ <NA>    <NA>    <NA>    <NA>    <NA>    polycy~ <NA>
```

```
## 10 <NA>  <NA>    copper~ <NA>    <NA>    <NA>    <NA>    <NA>    polycy~ <NA>
## 11 <NA>  <NA>    copper~ <NA>    <NA>    <NA>    <NA>    <NA>    polycy~ <NA>
## 12 2795~ <NA>    copper~ <NA>    <NA>    <NA>    <NA>    <NA>    polycy~ <NA>
## # ... with 19 more variables: `StudyDesign  Study^Design` <chr>,
## #   `OutcomeRelationship  (risky/protective/unrelated)` <chr>,
## #   `OR/RR/SMD/Beta/HR ( | delimited list)  OR/RR/SMD/Beta/HR^Value^95%Confidence Interval^Subgroup`
## #   `Risk Unit` <chr>, IntakeMedian <chr>, IntakeArithmeticAverage <chr>,
## #   IntakeGeometricAverage <chr>, IntakeUnit <chr>,
## #   `Recorder ( | delimited list)` <chr>, Database <chr>, Title <chr>,
## #   Date <chr>, Author <chr>, DOI <chr>, Affiliations <chr>, ...
```

```
res2$MetaEffect_Plot
```

```
## $`EX_C00033&EX_D13329[SMD](ug per mL)`
```





1 Introduction of "ExposomeX" platform

1.1 Overview and perspective

Exposome has become the hotspot of next-generation health studies. To date, there is no available effective platform to standardize the analysis of exposomic data. In ExposomeX, we aim to propose one new framework of exposomic analysis and build up one novel integrated platform to expediate the discovery of "Exposure-Biology-Disease" nexus. In ExposomeX, we have built up one integrated platform for exposomic analysis. In total, we have developed SIX major modules, which include exposomic mass spectrometry data processing (E-MS), statistical analyses of "chemicals-endogenous omics-diseases" interaction (E-STAT), exposome database search (E-DB), meta-analysis of chemical-diseases (E-META), biological explanation of chemical-diseases (E-BIO) and visualization of multiple dimension data (E-VIZ). Here, we provide R package "exposomex" to conduct the data analysis. Users can also install part of the packages, i.e., the 14 R packages including extidy (tidy data), exstat (statistical desicription), exviz (data visualization), exdb (data base), exmo (multi-omic data), excros (cross-section data), exmedt (mediation effect), expanel (panel data), exsurv (survival analysis), exmix (mixture effect), exnta (non-targeted analysis), exbiolink (biological link), exstatlink (statistical link), and exmeta (meta-analysis). User can also use these packages by web-interaction, see: http://www.exposomex.cn. In sum, we have proposed a novel framework for standardized exposomic analysis, which can be accessed using both R and online interactive platform. All the modules will keep updating. Please the user tutorials at: http://www.exposomex.cn/#/toturial.

This platform is founded by Bin Wang (Peking University, binwang@pku.edu.cn) and Mingliang Fang (Fudan University, mlfang@fudan.edu.cn). Great thanks should be given to the core development members for their significant contributions to the individual 14 R packages including Bin Wang (extidy, exdb, excros, exmix, expanel, and exstatlink), Mingliang Fang (exnta and exbiolink), Yanqiu Feng (exstat), Ning Gao (exviz), Guohuan Zhang and Yuting Wang (exmo), Mengyuan Ren (exmedt), Changxin Lan (exsurv), and Weinan Lin (exmeta). Special credits to Changxin Lan for making the codes into R packages of all modules, Ning Gao for providing help to most of the visualization funcitons, and Weinan Lin for tidying the IDs of chemicals, proteins, and diseases. The other contributors are acknowledged at http://www.exposomex.cn/#/about. Welcome to contact ExposomeX development team by E-mail: exposomex@gmail.com

1.2 Methodology

1.3 Basics

Initialize package

Make sure that the required packages is already installed.

```
# The following two packages should be installed in advance
# devtools::install_github("ExposomeX/exposomex", force = TRUE)
# install.packages("pacman")
pacman::p_load(
  "exposomex",
  "tidyverse"
  )
```

At first, you need to initialize the calculation environment using a series of initialization functions, e.g., InitCros, InitMo, InitTidy, InitViz, InitBiolink, etc. Here, we use the package "excros" for cross-sectional data analysis for example. The detailed information about the functions and returned value will be introduced in the following chapters.

```
res = InitCros()
res
```

```
## <eSet>
##   Public:
##     AddCommand: function (x)
##     AddLog: function (x)
##     clone: function (deep = FALSE)
##     EpiDesign: NULL
##     ExcecutionLog: Complete initializing the ExpoStat module.2022.12.14 14. ...
##     Expo: list
##     FileDirIn: NULL
##     FileDirOut: /home/ubuntu/@changxin/R_Exposome_1.0/output_144401GTCNXR
##     PID: 144401GTCNXR
##     RCommandLog: eSet <- InitCros(PID = Any ID your like, FileDirOut = An ...
##     VarsDel: NULL
```

Here, we can see that the returned value "res" is an R6 object. It contains an unique program ID of res$PID (e.g., "100737GJMWJA"), which is random generated by the system. Users need use it in the following step for further data process.

Upload data

```
res1 <- LoadCros(PID = res$PID,
                 UseExample = "example#1")

res1$Expo$Voca %>%
  dplyr::slice(1:20) %>%
  knitr::kable(format = "latex",
               align = "l") %>%
  kableExtra::kable_styling(full_width = F,
                            latex_options = "striped",
                            position = "left",
                            font_size = 10)
```

SerialNo	SerialNo_Raw	FullName	GroupName	Lod
Y1	Y1	Y_cont	Outcome	NA
Y2	Y2	Y_disc	Outcome	NA
C1	C1	Cov_1	Demography	NA
C2	C2	Cov_2	Demography	NA
C3	C3	Cov_3	Demography	NA
C4	C4	Cov_4	Demography	NA
C5	C5	Cov_5	Demography	NA
C6	C6	Cov_6	Demography	NA
X1	X1	TE_1	Chemical	0.5
X2	X2	TE_2	Chemical	0.5
X3	X3	TE_3	Chemical	0.5
X4	X4	TE_4	Chemical	0.5
X5	X5	TE_5	Chemical	0.5
X6	X6	TE_6	Chemical	0.5
X7	X7	TE_7	Chemical	0.5
X8	X8	TE_8	Chemical	0.5
X9	X9	CH1	Chemical	5.0
X10	X10	CH2	Chemical	5.0
X11	X11	CH3	Chemical	5.0
X12	X12	CH4	Chemical	5.0

```
res1$Expo$Data %>%
  dplyr::select(SampleID:C2) %>%
  dplyr::slice(1:20) %>%
  knitr::kable(format = "latex",
               align = "l") %>%
  kableExtra::kable_styling(full_width = F,
                            latex_options = "striped",
                            position = "left",
                            font_size = 10)
```

SampleID	SubjectID	Group	Y1	Y2	C1	C2
Tr1	S1	train	-101	1	26.86773	25.35056
Tr2	S2	train	-51	0	30.91822	23.94432
Tr3	S3	train	-37	0	25.82186	23.04579
Tr4	S4	train	-61	1	37.97640	21.21191
Tr5	S5	train	-28	0	31.64754	19.53762
Tr6	S6	train	-8	0	25.89766	20.77442
Tr7	S7	train	-63	1	32.43715	27.00009
Tr8	S8	train	-35	0	33.69162	22.13620
Tr9	S9	train	-14	0	32.87891	19.84672
Tr10	S10	train	-99	1	28.47306	29.60787
Tr11	S11	train	-60	0	37.55891	25.27530
Tr12	S12	train	-32	0	31.94922	23.28406
Tr13	S13	train	-73	0	26.89380	27.17545
Tr14	S14	train	-18	0	18.92650	26.65927
Tr15	S15	train	-48	0	35.62465	22.14227
Tr16	S16	train	-20	0	29.77533	30.61831
Tr17	S17	train	-9	0	29.91905	23.23492
Tr18	S18	train	-98	1	34.71918	19.72652
Tr19	S19	train	-70	0	34.10611	23.56680
Tr20	S20	train	-36	0	32.96951	24.62261

Tidy data

```
res2 <- DelMiss(res$PID)

res2$Expo$Voca %>%
  dplyr::slice(1:20) %>%
  knitr::kable(format = "latex",
               align = "l") %>%
  kableExtra::kable_styling(full_width = F,
                            latex_options = "striped",
                            position = "left",
                            font_size = 10)
```

SerialNo	SerialNo_Raw	FullName	GroupName	Lod
Y1	Y1	Y_cont	Outcome	NA
Y2	Y2	Y_disc	Outcome	NA
C1	C1	Cov_1	Demography	NA
C2	C2	Cov_2	Demography	NA
C3	C3	Cov_3	Demography	NA
C4	C4	Cov_4	Demography	NA
C5	C5	Cov_5	Demography	NA
C6	C6	Cov_6	Demography	NA
X1	X1	TE_1	Chemical	0.5
X2	X2	TE_2	Chemical	0.5
X3	X3	TE_3	Chemical	0.5
X4	X4	TE_4	Chemical	0.5
X5	X5	TE_5	Chemical	0.5
X6	X6	TE_6	Chemical	0.5
X7	X7	TE_7	Chemical	0.5
X8	X8	TE_8	Chemical	0.5
X9	X9	CH1	Chemical	5.0
X10	X10	CH2	Chemical	5.0
X11	X11	CH3	Chemical	5.0
X12	X12	CH4	Chemical	5.0

```
res2$Expo$Data %>%
  dplyr::select(SampleID:C2) %>%
  dplyr::slice(1:20) %>%
  knitr::kable(format = "latex",
               align = "l") %>%
  kableExtra::kable_styling(full_width = F,
                            latex_options = "striped",
                            position = "left",
                            font_size = 10)
```

SampleID	SubjectID	Group	Y1	Y2	C1	C2
Tr1	S1	train	-101	1	26.86773	25.35056
Tr2	S2	train	-51	0	30.91822	23.94432
Tr3	S3	train	-37	0	25.82186	23.04579
Tr4	S4	train	-61	1	37.97640	21.21191
Tr5	S5	train	-28	0	31.64754	19.53762
Tr6	S6	train	-8	0	25.89766	20.77442
Tr7	S7	train	-63	1	32.43715	27.00009
Tr8	S8	train	-35	0	33.69162	22.13620
Tr9	S9	train	-14	0	32.87891	19.84672
Tr10	S10	train	-99	1	28.47306	29.60787
Tr11	S11	train	-60	0	37.55891	25.27530
Tr12	S12	train	-32	0	31.94922	23.28406
Tr13	S13	train	-73	0	26.89380	27.17545
Tr14	S14	train	-18	0	18.92650	26.65927
Tr15	S15	train	-48	0	35.62465	22.14227
Tr16	S16	train	-20	0	29.77533	30.61831
Tr17	S17	train	-9	0	29.91905	23.23492
Tr18	S18	train	-98	1	34.71918	19.72652
Tr19	S19	train	-70	0	34.10611	23.56680
Tr20	S20	train	-36	0	32.96951	24.62261

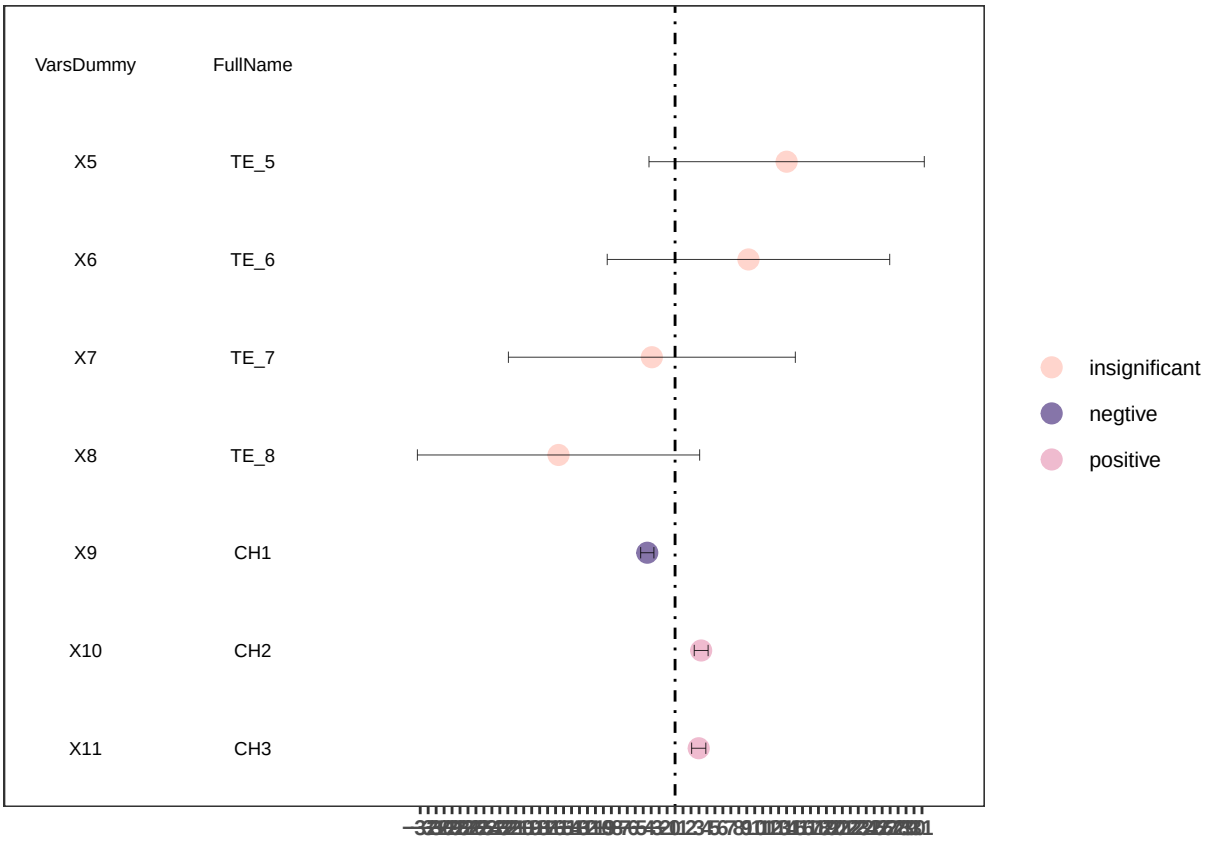
Modeling

```
res3 = CrosAsso(PID = res$PID,
                EpiDesign = "cohort",
                VarsY = "Y1",
                VarsX = "X5,X6,X7,X8,X9,X10,X11",
                VarsN = "single.factor" ,
                VarsSel = F,
                VarsSelThr = 0.1,
                IncCova = T,
                Family = "gaussian",
                RepMsr = F,
                Corstr = "ar1")

res3$Y1_single.factor_cohort_gaussian %>%
  dplyr::select(SerialNo:ci_h) %>%
  dplyr::slice(1:20) %>%
  knitr::kable(format = "latex",
               align = "l") %>%
  kableExtra::kable_styling(full_width = F,
                            latex_options = "striped",
                            position = "left",
                            font_size = 10)
```

SerialNo	Vars.dummy	beta.value	ci_l	ci_h
X5	X5	14.005401	-3.277425	31.288227
X6	X6	9.214098	-8.517261	26.945457
X7	X7	-2.915070	-20.927230	15.097090
X8	X8	-14.634061	-32.354215	3.086094
X9	X9	-3.489654	-4.319254	-2.660055
X10	X10	3.275350	2.404960	4.145739
X11	X11	2.968116	2.078754	3.857479

Visualize model

```
res4 = VizCrosAsso(PID = res$PID,
                   VarsY = "Y1",
                   VarsN="single.factor",
                   Layout = "forest",
                   Brightness = "dark",
                   Palette = "default1")
res4$Y1_single.factor_forest_dark_default1
```




```
FuncExit(PID = res$PID)
```

```
## [1] "Success to exit. Thanks for using ExposomeX platform!"
```

2 ExpoDb module

2.1 Application domain

ExpoDB module is designed as a convenient tool to explore the data, as well as facilitating to find the biological relationship between exposure and diseases from the perspective of bioinformatics. This module adopts the most frequently-used and authoritative databases, e.g., T3DB, CTD, ToxCast, StringDB, STITCH, KEGG, and GO.

2.2 Theory

2.3 Work pipeline

Users can easily get the modeling results by following the detailed instructions in each step. It mainly has five functions, including searching the related information (Dictionary), converting the IDs between different databases (Convert), exploring the nexus between exposure, protein, phenotype, and disease (Nexus), annotating the non-target features from high-resolution mass spectrometry (Annotation), and learning the nomenclature of "ExposomeX" platform (Abbreviation).

```
# The following two packages should be installed in advance
# devtools::install_github("ExposomeX/exdb", force = TRUE)
# devtools::install_github("ExposomeX/extidy", force = TRUE)

#library(exdb)
# library(extidy)

# devtools::install_github("ExposomeX/exposomex", force = TRUE)
library(exposomex)
```

```
res = InitDb()

res1 = LoadDb(PID=res$PID,
              UseExample="example#1")

res2 = ExpoAbbr(PID=res$PID,
                OutPath = 'default',
                Keys = "default")
res2
```

```
## # A tibble: 12 x 3
##    Keywords     FullName        Group
##    <chr>        <chr>           <chr>
##  1 CrossSection Cross sectional EpiDesign
##  2 Cros         Cross sectional EpiDesign
##  3 CaseControl  Case control    EpiDesign
##  4 CaseCtr      Case control    EpiDesign
##  5 Cohort       Cohort          EpiDesign
##  6 Panel        Panel           EpiDesign
##  7 Longitudinal Longitudinal    EpiDesign
##  8 Longi        Longitudinal    EpiDesign
##  9 CaseFollow   Case followup   EpiDesign
## 10 TimeSeries   Time series     EpiDesign
## 11 TimeSeri     Time series     EpiDesign
## 12 GO           Gene ontology   <NA>
```

```
res3 = ExpoDict(PID=res$PID,
                OutPath = 'default',
                Class = "GO",
                Keys = "default")
res3
```

```
## # A tibble: 7 x 6
##   go_id      Term                              Ontology Definit~1 Synonym Secon~2
##   <chr>      <chr>                             <chr>    <chr>     <chr>   <lgl>
## 1 GO:0000001 mitochondrion inheritance         BP       The dist~ mitoch~ NA
## 2 GO:0004857 enzyme inhibitor activity         MF       Binds to~ metall~ NA
## 3 GO:0004857 enzyme inhibitor activity         MF       Binds to~ GO:004~ NA
## 4 GO:0004866 endopeptidase inhibitor activity MF       Stops, p~ alpha-~ NA
## 5 GO:0004866 endopeptidase inhibitor activity MF       Stops, p~ endopr~ NA
## 6 GO:0004866 endopeptidase inhibitor activity MF       Stops, p~ protei~ NA
## 7 GO:0030414 peptidase inhibitor activity      MF       Stops, p~ protea~ NA
## # ... with abbreviated variable names 1: Definition, 2: Secondary
```

```
res4 = ExpoConv(PID=res$PID,
                OutPath = 'default',
                From = "chemical",
                To = "cas.rn",
                Keys = "default")
res4
```

```
## # A tibble: 18 x 3
##    chemical cas.rn       EXC
##    <chr>    <chr>        <chr>
##  1 zinc     1162648-93-2 EX:C00212
##  2 zinc     24359-56-6   EX:C00212
##  3 zinc     25016-79-9   EX:C00212
##  4 zinc     23713-49-7   EX:C00212
##  5 zinc     7440-66-6    EX:C00212
##  6 zinc     7646-85-7    EX:C00212
##  7 zinc     9025-42-7    EX:C00212
##  8 water    1310-73-2    EX:C46309
##  9 water    39388-36-8   EX:C46309
## 10 water    13670-17-2   EX:C46309
## 11 water    14314-42-2   EX:C46309
## 12 water    558440-22-5  EX:C46309
## 13 water    7732-18-5    EX:C46309
## 14 water    67747-09-5   EX:C46309
## 15 water    7789-20-0    EX:C46309
## 16 water    14280-30-9   EX:C46309
## 17 water    3352-57-6    EX:C46309
## 18 water    60426-60-0   EX:C46309
```

```
res5 = ExpoNexus(PID=res$PID,
                 OutPath = 'default',
                 ClassA = "chemical",
                 ClassB = "protein",
                 KeysA = "default",
                 KeysB = "default")
res5
```

```
## # A tibble: 556 x 8
```

```
##    EXC       inchikey                cas.rn EXP   ENSP  Unipr~1 datab~2 remarks
##    <chr>     <chr>                   <chr>  <chr> <chr> <chr>   <chr>   <chr>
##  1 EX:C07343 OWTFKEBRIAXSMO-UHFFFAOY~ 11118~ EX:P~ ENSP~ <NA>    ctd     affect~
##  2 EX:C07343 OWTFKEBRIAXSMO-UHFFFAOY~ 11118~ EX:P~ ENSP~ Q15057  ctd     affect~
##  3 EX:C07343 OWTFKEBRIAXSMO-UHFFFAOY~ 11118~ EX:P~ ENSP~ F8WAU0  ctd     affect~
##  4 EX:C07343 OWTFKEBRIAXSMO-UHFFFAOY~ 11118~ EX:P~ ENSP~ C9J8L1  ctd     affect~
##  5 EX:C07343 OWTFKEBRIAXSMO-UHFFFAOY~ 11118~ EX:P~ <NA>  <NA>    ctd     affect~
##  6 EX:C07343 OWTFKEBRIAXSMO-UHFFFAOY~ 11118~ EX:P~ <NA>  Q15057  ctd     affect~
##  7 EX:C07343 OWTFKEBRIAXSMO-UHFFFAOY~ 11118~ EX:P~ <NA>  F8WAU0  ctd     affect~
##  8 EX:C07343 OWTFKEBRIAXSMO-UHFFFAOY~ 11118~ EX:P~ <NA>  C9J8L1  ctd     affect~
##  9 EX:C07343 OWTFKEBRIAXSMO-UHFFFAOY~ 15502~ EX:P~ ENSP~ <NA>    ctd     affect~
## 10 EX:C07343 OWTFKEBRIAXSMO-UHFFFAOY~ 15502~ EX:P~ ENSP~ Q15057  ctd     affect~
## # ... with 546 more rows, and abbreviated variable names 1: Uniprot_KB,
## #   2: database
```

```
res6 = ExpoAnno(PID=res$PID,
                OutPath = 'default',
                MassToCharge = "default",
                AdductPos = "all",
                AdductNeg = "all",
                Accuracy = 5)
res6
```

```
## # A tibble: 41 x 9
##    IonMass Bias_PPM IonMode  Adduct Name            MassC~1 Monoi~2 SMILES Group
##      <dbl>    <dbl> <chr>    <chr>  <chr>             <dbl>   <dbl> <chr>  <chr>
##  1     200    0.364 positive M+2Na  1,4-Bis(4-chlo~    354.    354. ClC1=~ pare~
##  2     200    0.127 positive M+2Na  (S)-1-(4-Bromo~    354.    354. CN(C)~ prec~
##  3     200    0.127 positive M+2Na  (S)-1-(4-Bromo~    354.    354. CN(C)~ prec~
##  4     200    0.127 positive M+2Na  (S)-1-(4-Bromo~    354.    354. CN(C)~ prec~
##  5     200    0.127 positive M+2Na  (S)-1-(4-Bromo~    354.    354. CN(C)~ prec~
##  6     200    0.127 positive M+2Na  (S)-1-(4-Bromo~    354.    354. CN(C)~ prec~
##  7     200    0.127 positive M+2Na  (S)-1-(4-Bromo~    354.    354. CN(C)~ prec~
##  8     200    3.99  positive M+2Na  2-Chloro-.beta~    354.    354. c1ccc~ prec~
##  9     200    3.99  positive M+2Na  2-Chloro-.beta~    354.    354. c1ccc~ prec~
## 10     200    3.99  positive M+2Na  2-Chloro-.beta~    354.    354. c1ccc~ prec~
## # ... with 31 more rows, and abbreviated variable names 1: MassConv,
## #   2: Monoisotopic_Mass
```

```
FuncExit(PID = res$PID)
```

```
## [1] "Success to exit. Thanks for using ExposomeX platform!"
```

3 ExpoTidy module

3.1 Application domain

3.2 Theory

3.3 Work pipeline

```
# The following two packages should be installed in advance

# devtools::install_github("ExposomeX/extidy", force = TRUE)

# library(extidy)
library(tidyverse)

# devtools::install_github("ExposomeX/exposomex", force = TRUE)
library(exposomex)
```

```
res = InitTidy()
res1 = LoadTidy(PID=res$PID,
                UseExample="example#1")
res1$Expo$Data
```

```
## # A tibble: 241 x 239
##    SampleID Subjec~1 Group    Y1    Y2    C1    C2    C3    C4    C5    C6    X1
##    <chr>    <chr>    <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
##  1 Tr1      S1       train     1  -101  26.9  25.4     3     0     1     3 NA
##  2 Tr2      S2       train     0   -51  30.9  23.9     1     1     2     2  1.23
##  3 Tr3      S3       train     0   -37  25.8  23.0     2     3     1     2  1.18
##  4 Tr4      S4       train     1   -61  38.0  21.2     1     2     2     3  1.90
##  5 Tr5      S5       train     0   -28  31.6  19.5     1     0     2     2 NA
##  6 Tr6      S6       train     0    -8  25.9  20.8     3     3     1     2  1.99
##  7 Tr7      S7       train     1   -63  32.4  27.0     2     3     2     1  1.69
##  8 Tr8      S8       train     0   -35  33.7  22.1     2     0     2     3  1.31
##  9 Tr9      S9       train     0   -14  32.9  19.8     2     2     1     2  1.84
## 10 Tr10     S10      train     1   -99  28.5  29.6     1     3     2     2  1.40
## # ... with 231 more rows, 227 more variables: X2 <dbl>, X3 <dbl>, X4 <dbl>,
## #   X5 <dbl>, X6 <dbl>, X7 <dbl>, X8 <dbl>, X9 <dbl>, X10 <dbl>, X11 <dbl>,
## #   X12 <dbl>, X13 <dbl>, X14 <dbl>, X15 <dbl>, X16 <dbl>, X17 <dbl>,
## #   X18 <dbl>, X19 <dbl>, X20 <dbl>, X21 <dbl>, X22 <dbl>, X23 <dbl>,
## #   X24 <dbl>, X25 <dbl>, X26 <dbl>, X27 <dbl>, X28 <dbl>, X29 <dbl>,
## #   X30 <dbl>, X31 <dbl>, X32 <dbl>, X33 <dbl>, X34 <dbl>, X35 <dbl>,
## #   X36 <dbl>, X37 <dbl>, X38 <dbl>, X39 <dbl>, X40 <dbl>, X41 <dbl>, ...
```

```
res2 = TransImput(PID=res$PID,
                  Group="T",
                  Vars="all.x",
                  Method="lod")

res3 = DelNearZeroVar(PID = res$PID)

res4 = DelMiss(PID = res$PID)

res5 = TransType(PID=res$PID,
                 Vars="Y1",
```

```
                To="factor")

res6 = TransClass(PID=res$PID,
                  Group="F",
                  Vars="X1",
                  LevelTo="4")

res7 = TransScale(PID=res$PID,
                  Group="T",
                  Vars="X4,X5",
                  Method="normal")

res8 = TransDistr(PID=res$PID,
                  Vars="X6,X7",
                  Method="log10")

res9 = TransGroup(PID=res$PID,
                  Vars="X4,X5",
                  ToGroup = "G1")

res10 = TransDummy(PID=res$PID,
                   Vars="default")

res10$Expo$Data
```

```
## # A tibble: 241 x 218
##    SampleID Subjec~1 Group Y1       Y2    C1    C2    C3    C4    C5    C6  X1.2
##    <chr>    <chr>    <chr> <fct> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
##  1 Tr1      S1       train 1      -101  26.9  25.4     3     0     1     3     0
##  2 Tr2      S2       train 0       -51  30.9  23.9     1     1     2     2     1
##  3 Tr3      S3       train 0       -37  25.8  23.0     2     3     1     2     0
##  4 Tr4      S4       train 1       -61  38.0  21.2     1     2     2     3     0
##  5 Tr5      S5       train 0       -28  31.6  19.5     1     0     2     2     0
##  6 Tr6      S6       train 0        -8  25.9  20.8     3     3     1     2     0
##  7 Tr7      S7       train 1       -63  32.4  27.0     2     3     2     1     0
##  8 Tr8      S8       train 0       -35  33.7  22.1     2     0     2     3     1
##  9 Tr9      S9       train 0       -14  32.9  19.8     2     2     1     2     0
## 10 Tr10     S10      train 1       -99  28.5  29.6     1     3     2     2     1
## # ... with 231 more rows, 206 more variables: X1.3 <dbl>, X1.4 <dbl>, X4 <dbl>,
## #   X5 <dbl>, X6 <dbl>, X7 <dbl>, X8 <dbl>, X9 <dbl>, X10 <dbl>, X11 <dbl>,
## #   X12 <dbl>, X13 <dbl>, X14 <dbl>, X15 <dbl>, X16 <dbl>, X17 <dbl>,
## #   X18 <dbl>, X19 <dbl>, X20 <dbl>, X21 <dbl>, X22 <dbl>, X23 <dbl>,
## #   X24 <dbl>, X25 <dbl>, X26 <dbl>, X27 <dbl>, X28 <dbl>, X30 <dbl>,
## #   X32 <dbl>, X34 <dbl>, X35 <dbl>, X36 <dbl>, X37 <dbl>, X38 <dbl>,
## #   X39 <dbl>, X41 <dbl>, X43 <dbl>, X45 <dbl>, X46 <dbl>, X47 <dbl>, ...
```

```
FuncExit(PID = res$PID)
```

```
## [1] "Success to exit. Thanks for using ExposomeX platform!"
```

4 ExpoStat module

4.1 Application domain

Data statistics is designed for the statistical description of data. Before selecting the statistical algorithm to build the model, it is necessary to understand the distribution of the data and the socio-demographic characteristics of the population. The results of the statistical description provide the basis for selecting the statistical algorithm to build the model and the potential confounding to be adjusted during the modeling process.

The ExpoStat module provides functions for creating the basic descriptive statistics table, testing normality distribution of variable, screening extreme values, comparing the size between/among groups, and calculating correlation coefficient between variables. The visualization of the statistical results are also provided. It provides six main functions for users : StatTable1, StatNorm, StatExtre, StatDesc, StatComp, and StatCorr.

- StatTable1: Create a statistical description table of demographic characteristics for three common types of epidemiological studies(i.e., cohort, case control, and cross sectional).
- StatNorm: Normality test for continuous variables, and visualizations of whether variables conform to normal distribution are provided.
- StatExtre: Calculating The extreme value of the variable and output the result. The visualization of the result shows the frequency of the extreme value in each subject.
- StatDesc: Calculate the mean, standard deviation, median and quartile of continuous variables, and the count and frequency of discrete variables..
- StatComp: Performing inter-group comparison for continuous variables and visualize whether there is a statistical difference between the different groups.
- StatCorr: Calculate the correlation coefficient between each variable, output the correlation coefficient and P value, and visualize the results.

4.2 Theory

The different "ExpoStat" module is based on different statistics test. Users can easily get the modeling results and their visualization plots with high quality by following the detailed instructions in each step. Only typical statistical methods are included in our module for convenience. For example, the statistical method of normality test is Shapiro-Wilk test, the statistical method of inter-group comparison is Wilcoxon rank test, and the statistical method of calculating the correlation coefficient between each variable is Pearson correlation coefficient or Spearman rank correlation coefficient.

4.3 Work pipeline

Initialize package

Make sure that the required packages is already installed.

```
# The following two packages should be installed in advance
# devtools::install_github("ExposomeX/exstat", force = TRUE)
# devtools::install_github("ExposomeX/extidy", force = TRUE)

# library(exstat)
# library(extidy)
library(tidyverse)
```

```
# devtools::install_github("ExposomeX/exposomex", force = TRUE)
library(exposomex)
```

At first, you need to initialize the calculation environment using a series of initialization functions, e.g., InitCros, InitMo, InitTidy, InitViz, InitBiolink, etc. Here, we use the package "exstat" for data description for example. The detailed information about the functions and returned value will be introcuced in the following chapters.

```
res <- InitStat()
res
```

```
## <eSet>
##   Public:
##     AddCommand: function (x)
##     AddLog: function (x)
##     clone: function (deep = FALSE)
##     EpiDesign: NULL
##     ExcecutionLog: Complete initializing the ExpoStat module.2022.12.14 14. ...
##     Expo: list
##     FileDirIn: NULL
##     FileDirOut: /home/ubuntu/@changxin/R_Exposome_1.0/output_144651GJAFZA
##     PID: 144651GJAFZA
##     RCommandLog: eSet <- InitVisual(PID = Any ID your like, FileDirOut =  ...
##     VarsDel: NULL
```

Here, we can see that the returned value "res" is an R6 object. It contains an unique program ID of res$PID (e.g., "100737GJMWJA"), which is random generated by the system. Users need to use it in the following step for further data process.

Upload data

After initializing the calculation environment, the second step is to upload local data file for ExpoStat Module. `LoadStat` is provided for this. It has four parameters, PID, UseExample, DataPath and VocaPath. PID is Program ID, which must be the same with the PID generated by `InitStat`. UseExample is a character indicates whether uses example data for analyses, available option include "example#1" for using example data1 and "default" for using data uploaded. DataPath refer to the input file directory, e.g. "D:/test/eg_data.xlsx". It should be noted that the slash symbol is `/`, not `\`. For convenience, here we use example data one for the following step.

```
res1 <- LoadStat(res$PID,
                 UseExample = "example#1")

res1$Expo$Voca %>%
  dplyr::slice(1:20) %>%
  knitr::kable(format = "latex",
               align = "l") %>%
  kableExtra::kable_styling(full_width = F,
                            latex_options = "striped",
                            position = "left",
                            font_size = 10)
```

SerialNo	SerialNo_Raw	FullName	GroupName	Lod
Y1	Y1	Y_disc	Outcome	NA
Y2	Y2	Y_cont	Outcome	NA
C1	C1	Cov_1	Demography	NA
C2	C2	Cov_2	Demography	NA
C3	C3	Cov_3	Demography	NA
C4	C4	Cov_4	Demography	NA
C5	C5	Cov_5	Demography	NA
C6	C6	Cov_6	Demography	NA
X1	X1	TE_1	Chemical	0.5
X2	X2	TE_2	Chemical	0.5
X3	X3	TE_3	Chemical	0.5
X4	X4	TE_4	Chemical	0.5
X5	X5	TE_5	Chemical	0.5
X6	X6	TE_6	Chemical	0.5
X7	X7	TE_7	Chemical	0.5
X8	X8	TE_8	Chemical	0.5
X9	X9	CH1	Chemical	5.0
X10	X10	CH2	Chemical	5.0
X11	X11	CH3	Chemical	5.0
X12	X12	CH4	Chemical	5.0

```
res1$Expo$Data %>%
  dplyr::select(SampleID:C2) %>%
  dplyr::slice(1:20) %>%
  knitr::kable(format = "latex",
               align = "l") %>%
  kableExtra::kable_styling(full_width = F,
                            latex_options = "striped",
                            position = "left",
                            font_size = 10)
```

SampleID	SubjectID	Group	Y1	Y2	C1	C2
Tr1	S1	train	1	-101	26.86773	25.35056
Tr2	S2	train	0	-51	30.91822	23.94432
Tr3	S3	train	0	-37	25.82186	23.04579
Tr4	S4	train	1	-61	37.97640	21.21191
Tr5	S5	train	0	-28	31.64754	19.53762
Tr6	S6	train	0	-8	25.89766	20.77442
Tr7	S7	train	1	-63	32.43715	27.00009
Tr8	S8	train	0	-35	33.69162	22.13620
Tr9	S9	train	0	-14	32.87891	19.84672
Tr10	S10	train	1	-99	28.47306	29.60787
Tr11	S11	train	0	-60	37.55891	25.27530
Tr12	S12	train	0	-32	31.94922	23.28406
Tr13	S13	train	0	-73	26.89380	27.17545
Tr14	S14	train	0	-18	18.92650	26.65927
Tr15	S15	train	0	-48	35.62465	22.14227
Tr16	S16	train	0	-20	29.77533	30.61831
Tr17	S17	train	0	-9	29.91905	23.23492
Tr18	S18	train	1	-98	34.71918	19.72652
Tr19	S19	train	0	-70	34.10611	23.56680
Tr20	S20	train	0	-36	32.96951	24.62261

Tidy data

After initializing the calculation environment, and upload local data file for ExpoStat Module. Users can choose to clean up the data, including deleting missing values, scale the data, changing the type of the data, interpolation of missing values, and so on.

```
res2 <- DelMiss(res$PID)
res2$Expo$Voca
```

```
## # A tibble: 226 x 5
##    SerialNo SerialNo_Raw FullName GroupName    Lod
##    <chr>    <chr>        <chr>    <chr>      <dbl>
##  1 Y1       Y1           Y_disc   Outcome     NA
##  2 Y2       Y2           Y_cont   Outcome     NA
##  3 C1       C1           Cov_1    Demography  NA
##  4 C2       C2           Cov_2    Demography  NA
##  5 C3       C3           Cov_3    Demography  NA
##  6 C4       C4           Cov_4    Demography  NA
##  7 C5       C5           Cov_5    Demography  NA
##  8 C6       C6           Cov_6    Demography  NA
##  9 X2       X2           TE_2     Chemical     0.5
## 10 X3       X3           TE_3     Chemical     0.5
## # ... with 216 more rows
```

```
res3 <- DelNearZeroVar(res$PID)
res3$Expo$Voca
```

```
## # A tibble: 209 x 5
##    SerialNo SerialNo_Raw FullName GroupName    Lod
##    <chr>    <chr>        <chr>    <chr>      <dbl>
##  1 Y1       Y1           Y_disc   Outcome     NA
##  2 Y2       Y2           Y_cont   Outcome     NA
##  3 C1       C1           Cov_1    Demography  NA
##  4 C2       C2           Cov_2    Demography  NA
##  5 C3       C3           Cov_3    Demography  NA
##  6 C4       C4           Cov_4    Demography  NA
##  7 C5       C5           Cov_5    Demography  NA
##  8 C6       C6           Cov_6    Demography  NA
##  9 X7       X7           TE_7     Chemical     0.5
## 10 X8       X8           TE_8     Chemical     0.5
## # ... with 199 more rows
```

```
res4 <- TransClass(res$PID,
                   Group = F,
                   Vars = "X10",
                   LevelTo = 4)
res4$Expo$Voca
```

```
## # A tibble: 209 x 5
##    SerialNo SerialNo_Raw FullName GroupName    Lod
##    <chr>    <chr>        <chr>    <chr>      <dbl>
##  1 Y1       Y1           Y_disc   Outcome     NA
##  2 Y2       Y2           Y_cont   Outcome     NA
##  3 C1       C1           Cov_1    Demography  NA
##  4 C2       C2           Cov_2    Demography  NA
##  5 C3       C3           Cov_3    Demography  NA
##  6 C4       C4           Cov_4    Demography  NA
```

```
##  7 C5       C5           Cov_5    Demography  NA
##  8 C6       C6           Cov_6    Demography  NA
##  9 X7       X7           TE_7     Chemical     0.5
## 10 X8       X8           TE_8     Chemical     0.5
## # ... with 199 more rows
```

StatTable1

StatTable1 provides sociodemographic information tables for three common epidemiological studies, returning the information to the user based on epidemiological design. EpiDesign/Group/VarsY/VarsC/Missing parameters must be entered. Attention please, PID must be got from the return result of InitStat(). StatTable1 can only run successfully after successfully running InitStat and LoadStat functions.

EpiDesign/Group/VarsY/VarsC/Missing parameters can be a character, run `?StatTable1` to see more details.

```
res_table1<-StatTable1(res$PID,
                       EpiDesign = "case.control",
                       Group = F,
                       VarsY = "Y1",
                       VarsC = res1$Expo$Voca %>%
                         dplyr::filter(str_detect(SerialNo_Raw, "C")) %>%
                         .$SerialNo_Raw %>%
                         str_c(collapse = ","),
                       Missing = "ifany")
# res_table1

res_table1<-StatTable1(res$PID,
                       EpiDesign = "cohort",
                       Group = F,
                       VarsY = "Y1",
                       VarsC = "all.c",
                       Missing = "always")
# res_table1
```

StatNorm

StatNorm provides function of normality test for continuous variable.For the time being, only the typical Shapiro-Wilk test is provided to test the normality of variables, and other tests will be added gradually in the future. Group/Vars/Method/Layout/Brightness/Palette parameters must be entered. Attention please, PID must be got from the return result of InitStat(). StatNorm can only run successfully after successfully running InitStat and LoadStat functions.

Group/Vars/Method/Layout/Brightness/Palette parameters can be a character, run `?StatNorm` to see more details.

```
res_norm <- StatNorm(PID = res$PID,
                     Group = T,
                     Vars =  "all",
                     Method = "shapiro.test",
                     Layout = "rose.chart",
                     Brightness = "light",
                     Palette = "default3")
res_norm
```

```
## $normtable
```

```
## $normtable$StatNorm_by_shapiro.test
## # A tibble: 416 x 5
##    Group Vars       P Method       Normal
##    <chr> <fct>  <dbl> <chr>        <chr>
##  1 Train Y1     3     shapiro.test N
##  2 Train Y2     3     shapiro.test N
##  3 Train C1     0.103 shapiro.test Y
##  4 Train C2     0.0326 shapiro.test Y
##  5 Train C3     3     shapiro.test N
##  6 Train C4     3     shapiro.test N
##  7 Train C5     3     shapiro.test N
##  8 Train C6     3     shapiro.test N
##  9 Train X7     3     shapiro.test N
## 10 Train X8     3     shapiro.test N
## # ... with 406 more rows
##
##
## $NormPlot
## NULL
##
## $NormPlot_para
## NULL
```

StatExtre

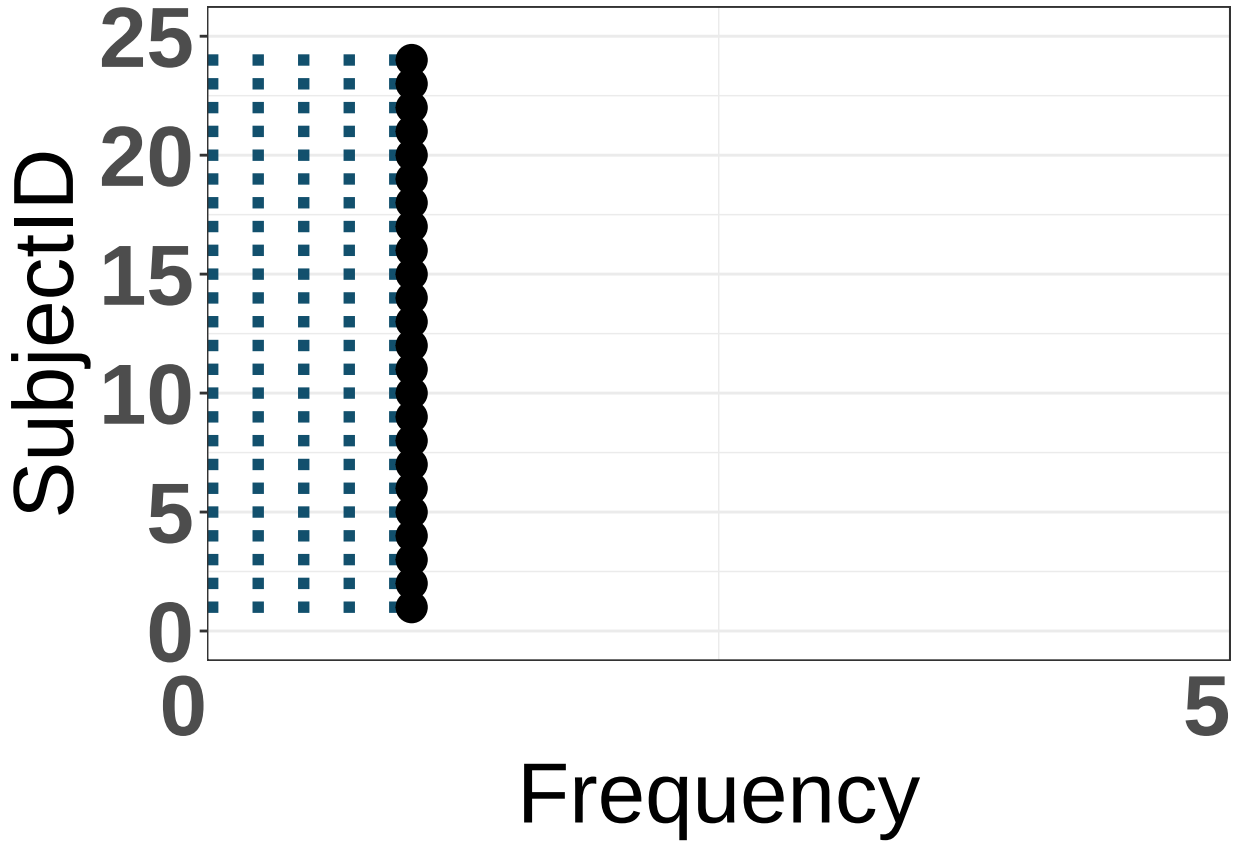
StatExtre provides the functions of calculating The extreme value for the continuous variable. The two parameters 'LimitLow' and 'LimitUpper' can be set to adjust the percentile of the variable. Group/Vars/LimitLow/LimitUpper/Layout/Brightness/Palette parameters must be entered. Attention please, PID must be got from the return result of InitStat(). StatExtre can only run successfully after successfully running InitStat and LoadStat functions.

Group/Vars/LimitLow/LimitUpper/Layout/Brightness/Palette parameters can be a character, run `?StatExtre` to see more details.

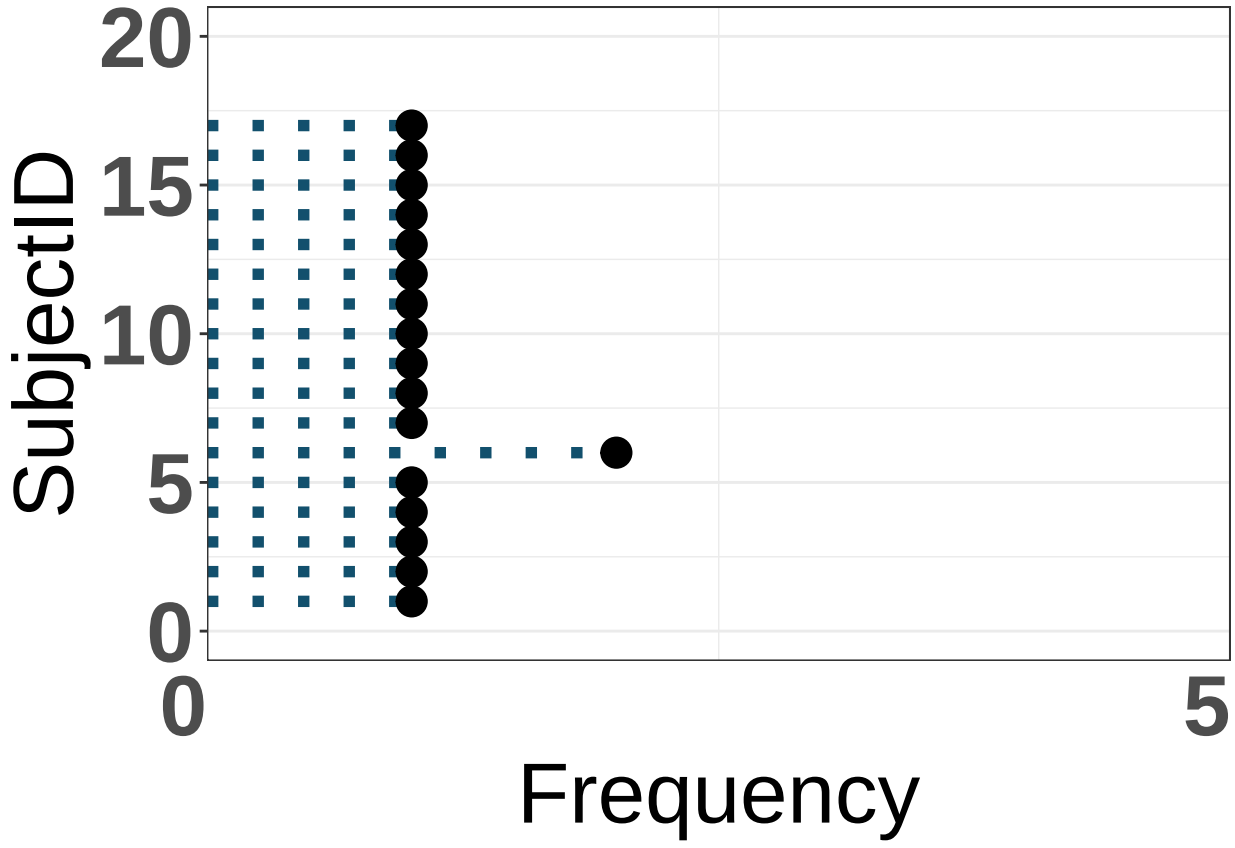
```
res_extre<-StatExtre(PID = res$PID,
                     Group = T,
                     Vars = "X5,X6,X7,X8,X9",
                     LimitLow = 0.025,
                     LimitUpper = 0.975,
                     Layout = "column.points",
                     Brightness = "dark",
                     Palette = "default2")
res_extre
```

```
## $Extretable
## # A tibble: 42 x 5
##    Group SerialNo value SubjectID SampleID
##    <chr> <chr>    <dbl> <chr>     <chr>
##  1 Train X7       -5    S106      Tr106
##  2 Train X7       -4.91 S150      Tr150
##  3 Train X7       -4.86 S102      Tr102
##  4 Train X7       -4.59 S5        Tr5
##  5 Train X7        4.59 S138      Tr138
##  6 Train X7        4.60 S38       Tr38
##  7 Train X7        4.78 S12       Tr12
```

```
##  8 Train X7        5    S1        Tr1
##  9 Train X8       -5    S49       Tr49
## 10 Train X8       -4.94 S116      Tr116
## # ... with 32 more rows
##
## $ExtrePlot
## $ExtrePlot$VizStatExtre_Train_ColumnPoints_dark_default2
```




```
##
## $ExtrePlot$VizStatExtre_Test_ColumnPoints_dark_default2
```




```
##
##
## $ExtrePlot_para
## $ExtrePlot_para$VizStatExtre_Train_ColumnPoints_dark_default2
## $ExtrePlot_para$VizStatExtre_Train_ColumnPoints_dark_default2$width
## [1] 37
##
## $ExtrePlot_para$VizStatExtre_Train_ColumnPoints_dark_default2$height
## [1] 56
##
##
## $ExtrePlot_para$VizStatExtre_Test_ColumnPoints_dark_default2
## $ExtrePlot_para$VizStatExtre_Test_ColumnPoints_dark_default2$width
## [1] 37
##
## $ExtrePlot_para$VizStatExtre_Test_ColumnPoints_dark_default2$height
## [1] 54
```

StatDesc

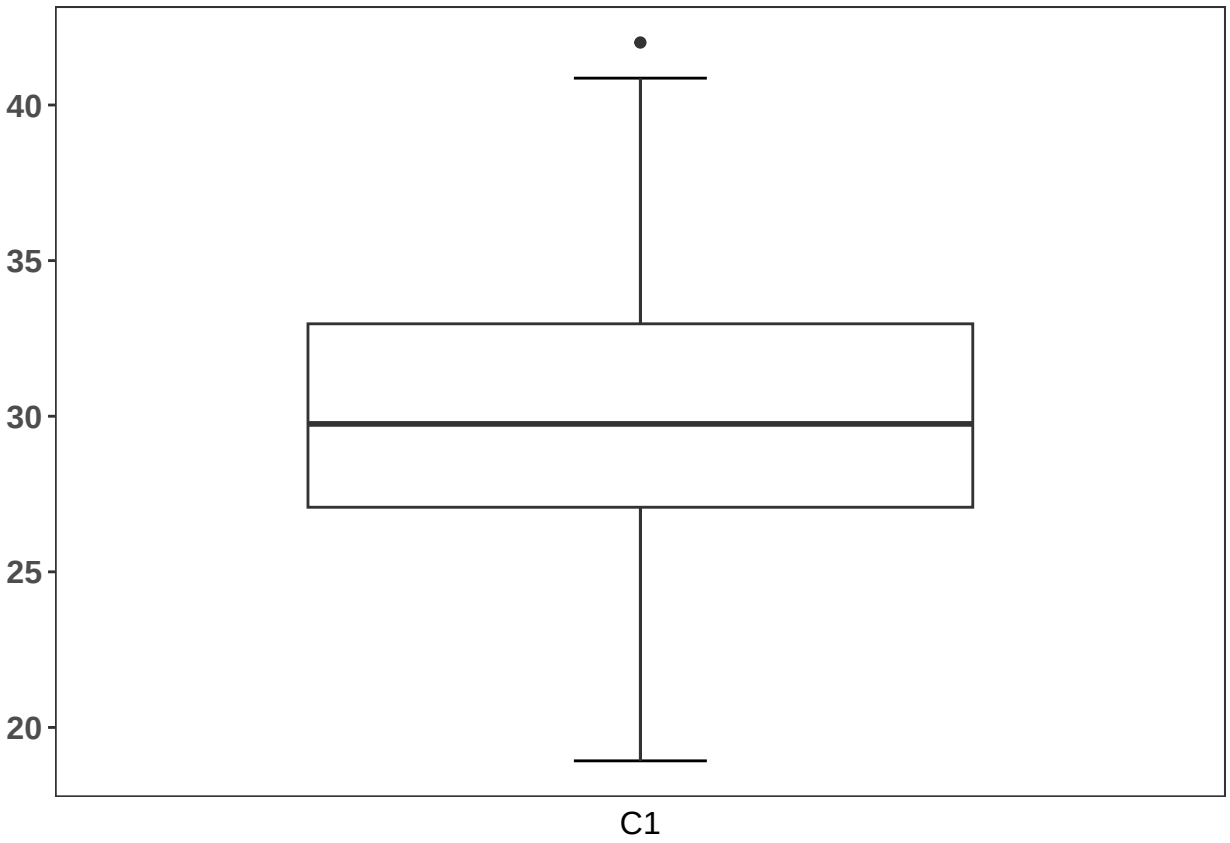
StatDesc provides the description of the size of a variable. Different types of variables have different size descriptions.The module calculates the mean, standard deviation, median, quartile and other information of continuous variables, the count and frequency of discrete variables. The visualization of this part is different from the visualization of the result statistics of other functions. This part only provides the box diagram and violin diagram to represent the median quartile of continuous variables. Group/Vars/VarsBy/Layout/Brightness/Palette parameters must be entered. Attention please, PID must be

got from the return result of InitStat(). StatDesc can only run successfully after successfully running InitStat and LoadStat functions.

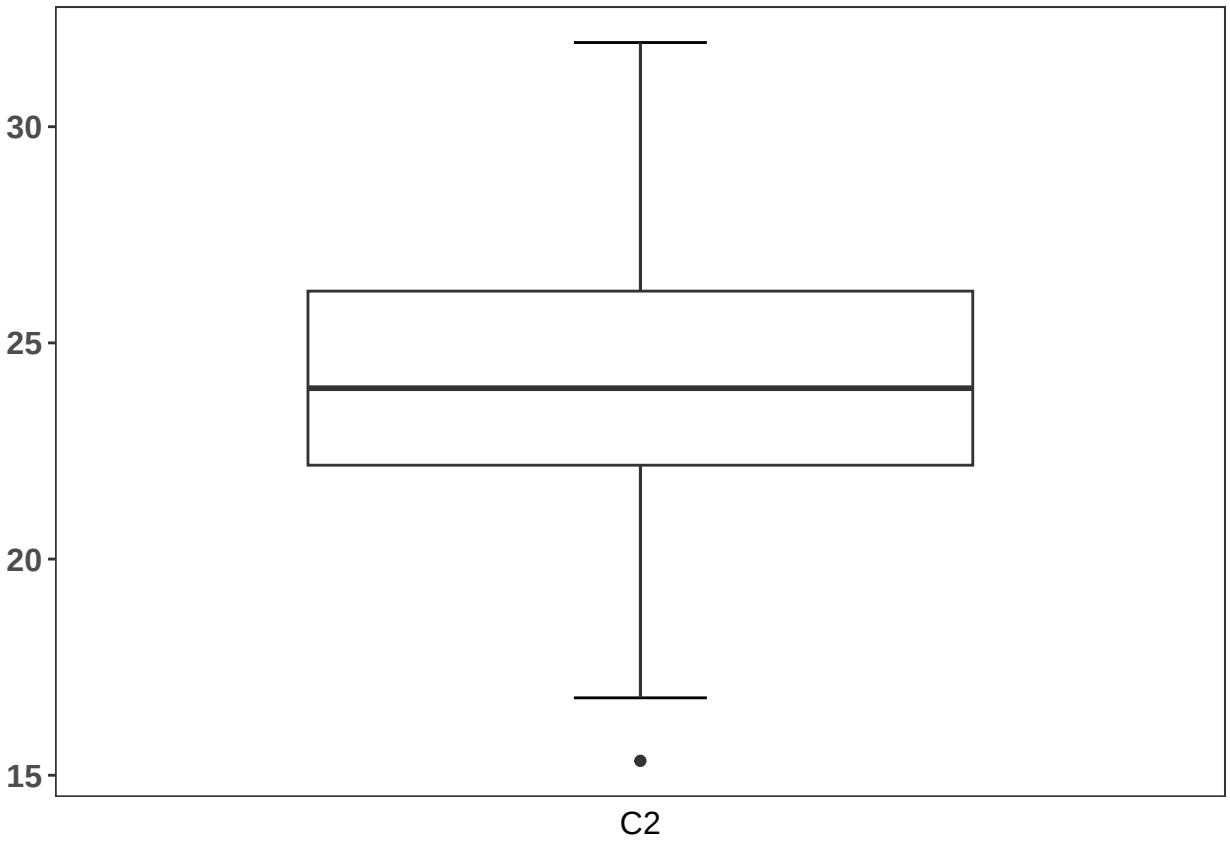
Group/Vars/VarsBy/Layout/Brightness/Palette parameters can be a character, run `?StatDesc` to see more details.

```
res_desc <- StatDesc(PID = res$PID,
                     Group = T,
                     Vars = "C1,C2,X5,X6,X7,X8,X9",
                     VarsBy = NULL,
                     Layout = "box",
                     Brightness = "dark",
                     Palette = "default2")
res_desc
```

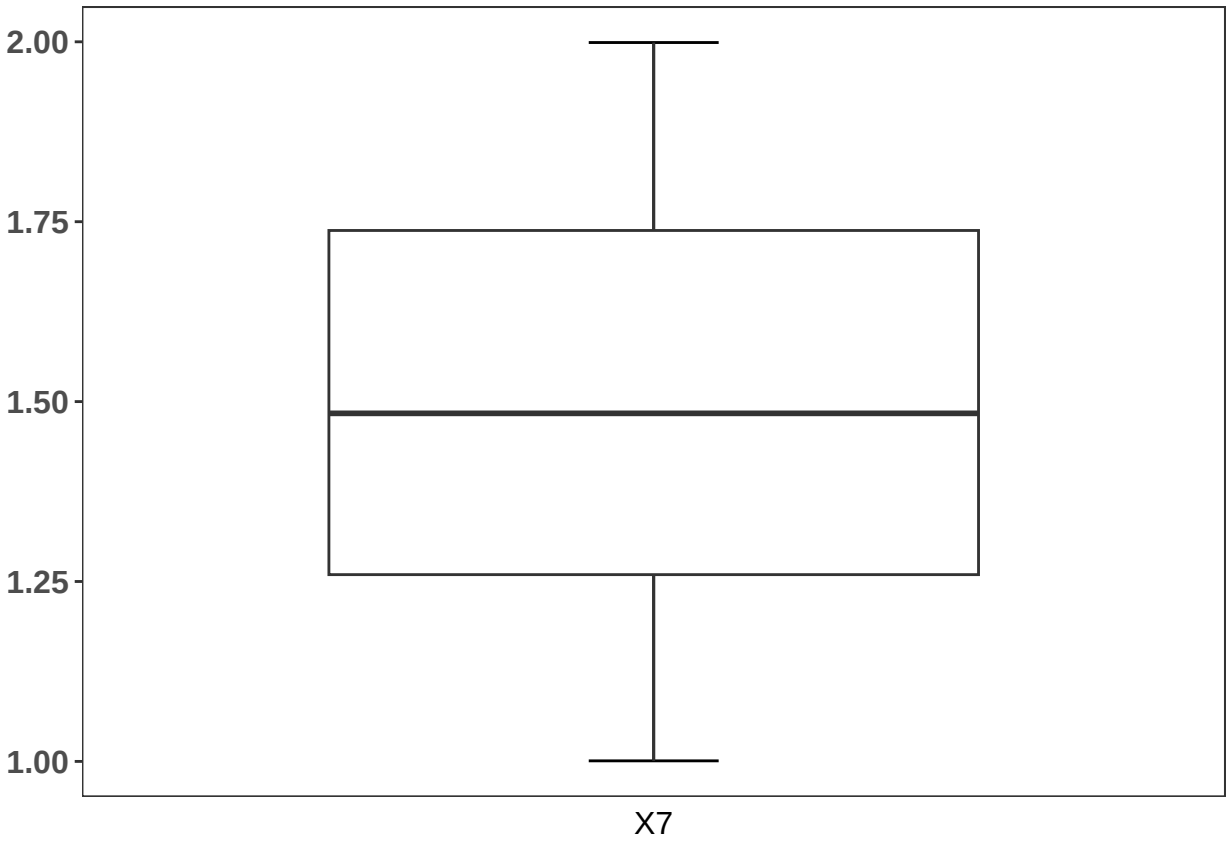
```
## $Desctable
## $Desctable$`/Stat_DescNum_by_`
##    Group variable   n    min    max median     q1     q3   iqr   mad   mean
## 1  Train       C1 150 18.927 42.008 29.753 27.075 32.970 5.895 4.326 30.109
## 2   Test       C1  91 18.927 40.863 29.919 27.145 33.300 6.155 4.479 30.142
## 3  Train       C2 150 15.333 31.948 23.952 22.171 26.198 4.026 2.914 24.136
## 4   Test       C2  91 17.133 31.948 24.006 22.123 25.715 3.592 2.773 24.130
## 5  Train       X7 150  1.001  1.999  1.484  1.260  1.738 0.478 0.367  1.500
## 6   Test       X7  91  1.015  1.953  1.515  1.292  1.702 0.410 0.312  1.504
## 7  Train       X8 150  1.000  1.992  1.460  1.256  1.748 0.492 0.360  1.486
## 8   Test       X8  91  1.018  1.992  1.499  1.305  1.734 0.429 0.309  1.508
## 9  Train       X9 150  4.493 31.199  8.174  6.782 10.386 3.604 2.158 10.095
## 10  Test       X9  91  5.147 24.346  7.973  6.704  9.268 2.564 1.898  9.624
##       sd    se    ci
## 1  4.521 0.369 0.729
## 2  4.765 0.500 0.992
## 3  3.068 0.250 0.495
## 4  2.983 0.313 0.621
## 5  0.289 0.024 0.047
## 6  0.272 0.029 0.057
## 7  0.291 0.024 0.047
## 8  0.265 0.028 0.055
## 9  5.188 0.424 0.837
## 10 4.818 0.505 1.003
##
##
## $DescPlot
## $DescPlot$VizStatDesc_Train_Box_C1
```



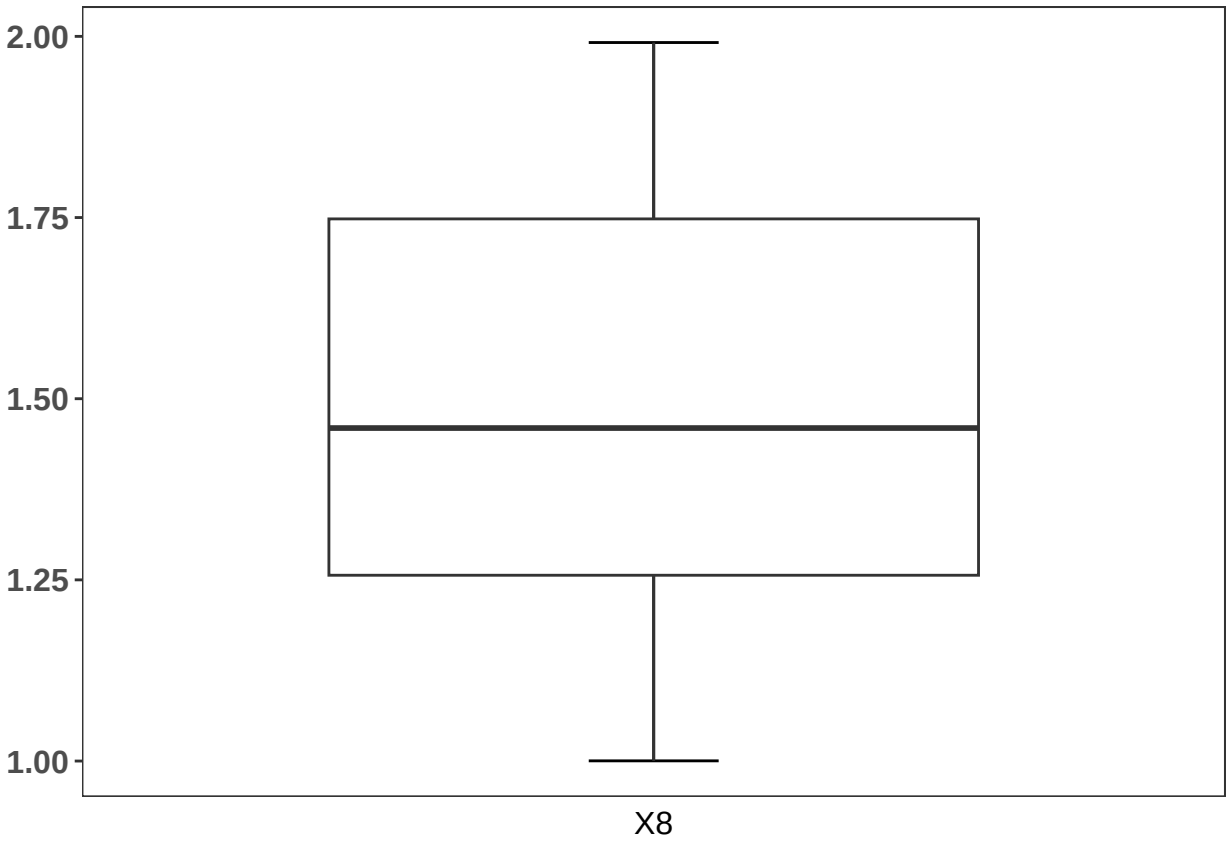

```
##
## $DescPlot$VizStatDesc_Train_Box_C2
```



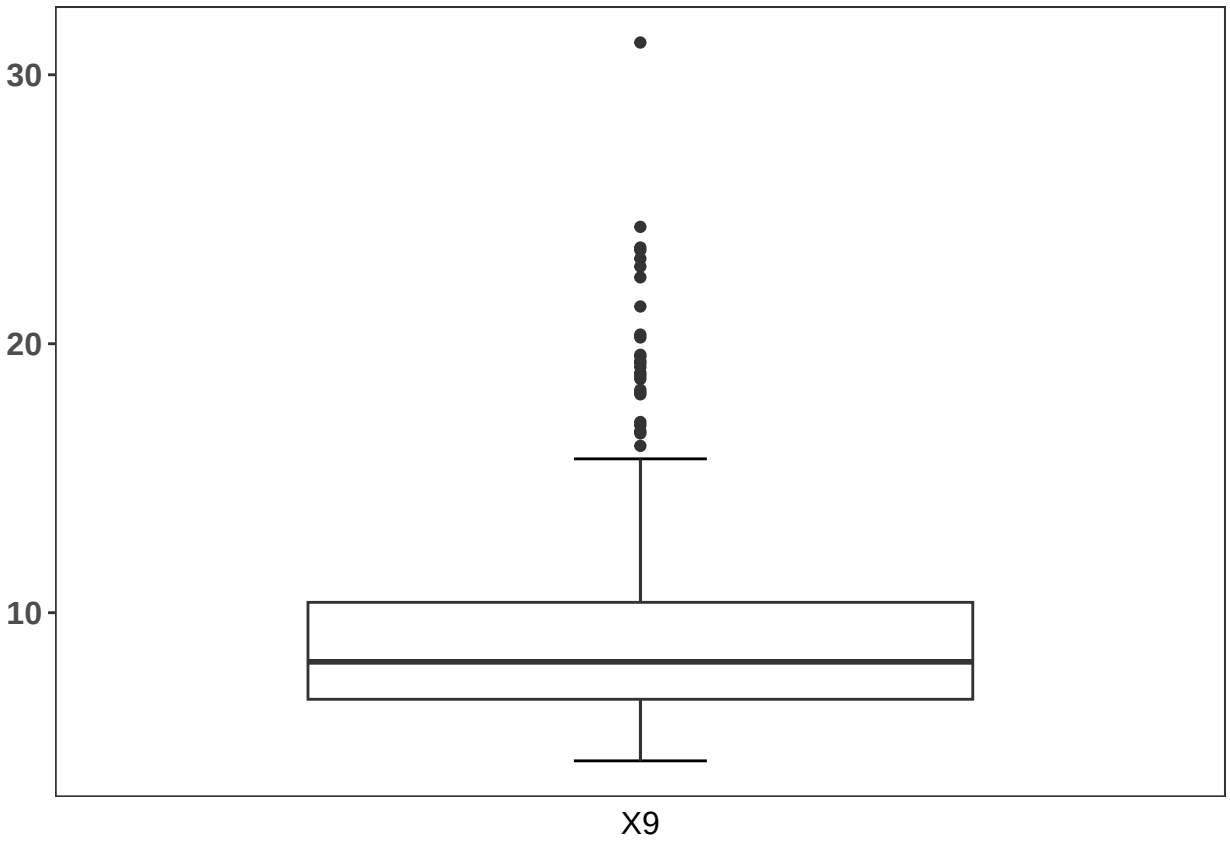

```
##
## $DescPlot$VizStatDesc_Train_Box_X7
```



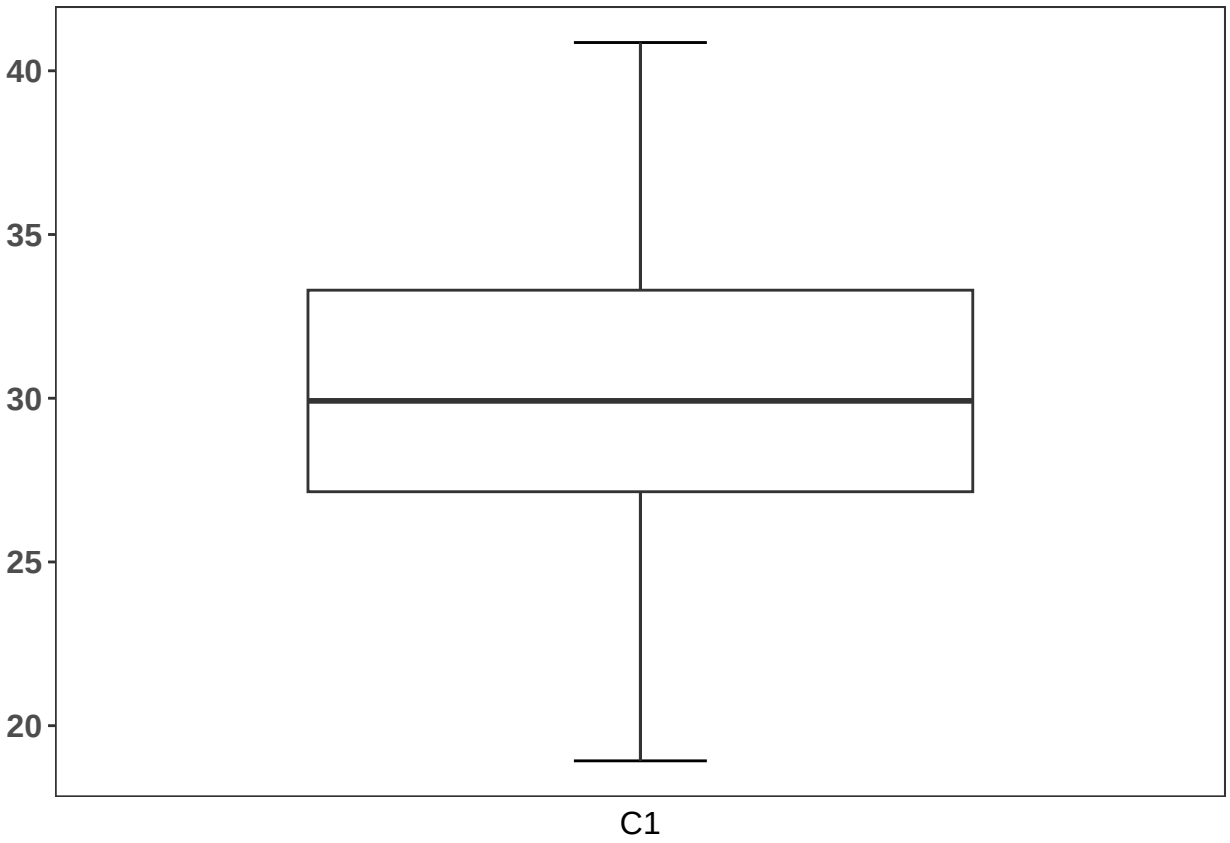

```
##
## $DescPlot$VizStatDesc_Train_Box_X8
```



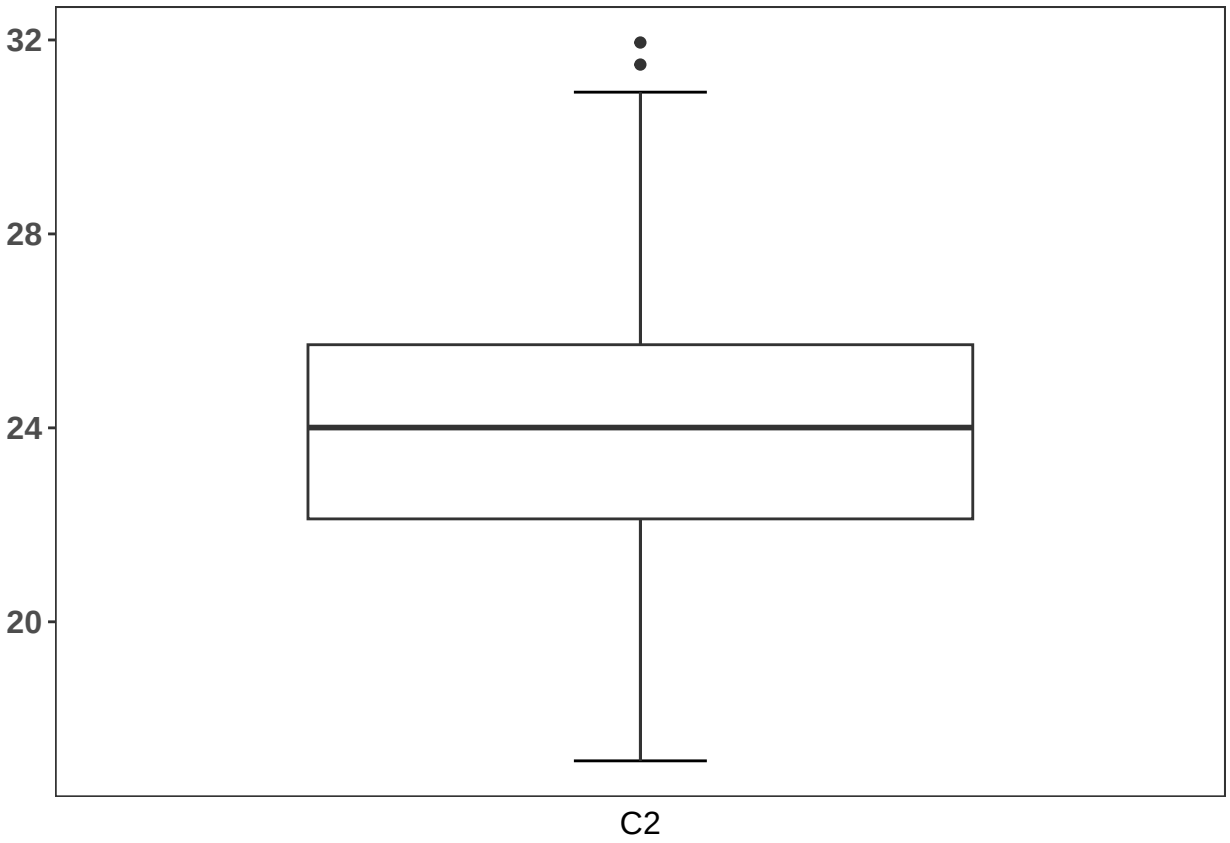

```
##
## $DescPlot$VizStatDesc_Train_Box_X9
```



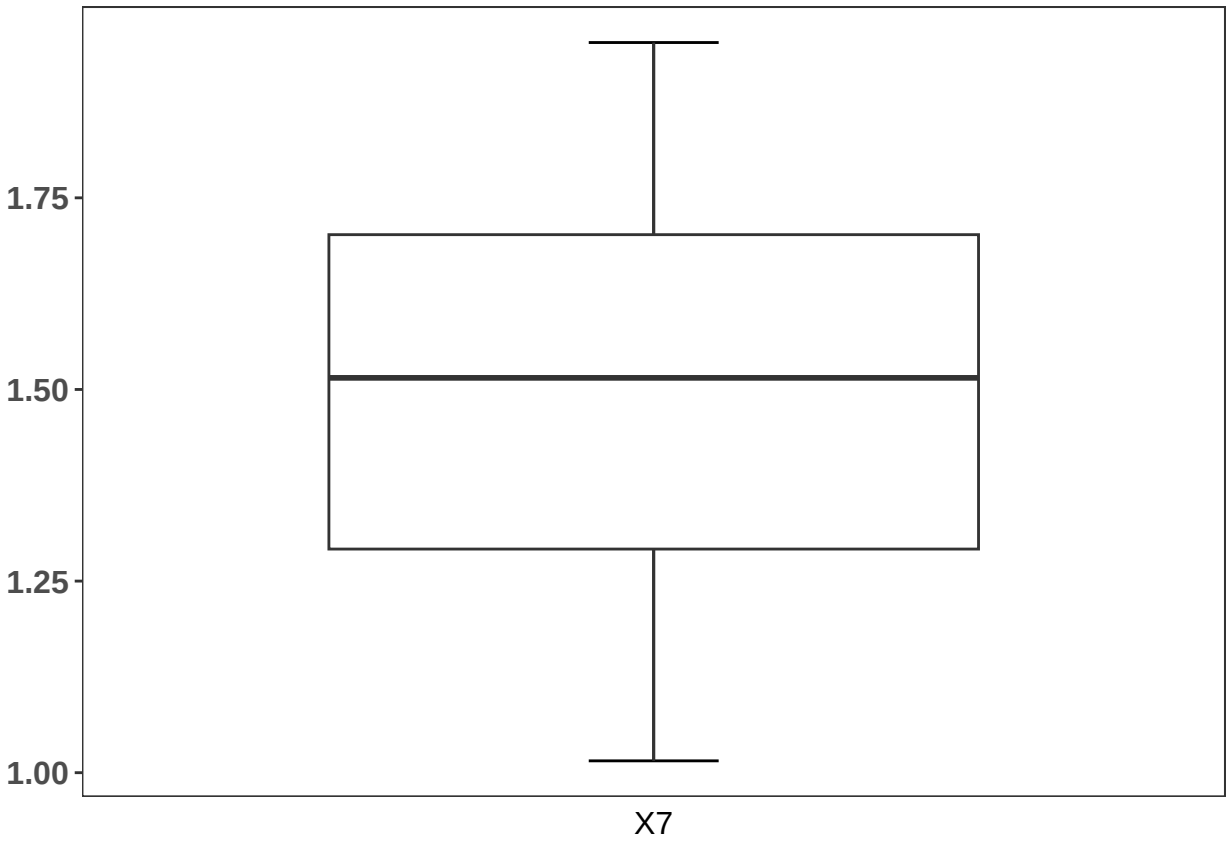

```
##
## $DescPlot$VizStatDesc_Test_Box_C1
```



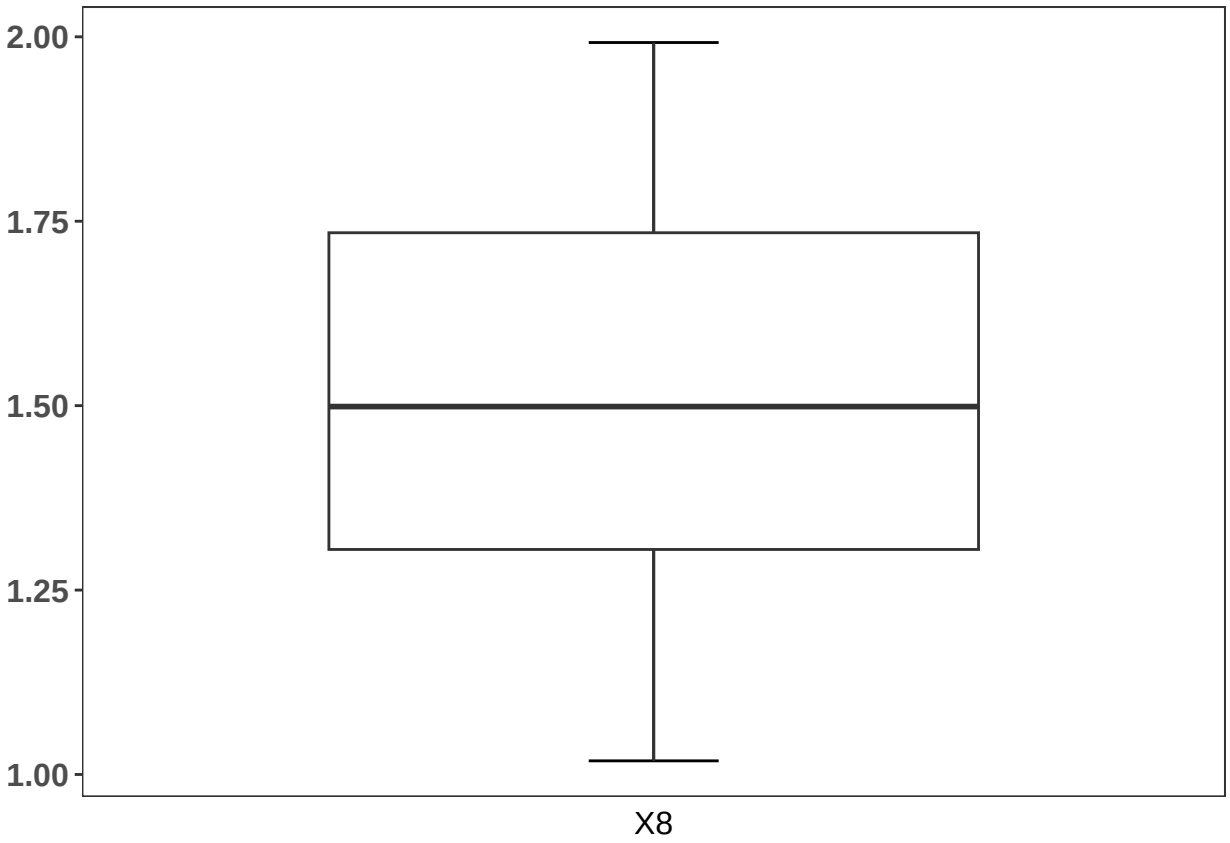

```
##
## $DescPlot$VizStatDesc_Test_Box_C2
```



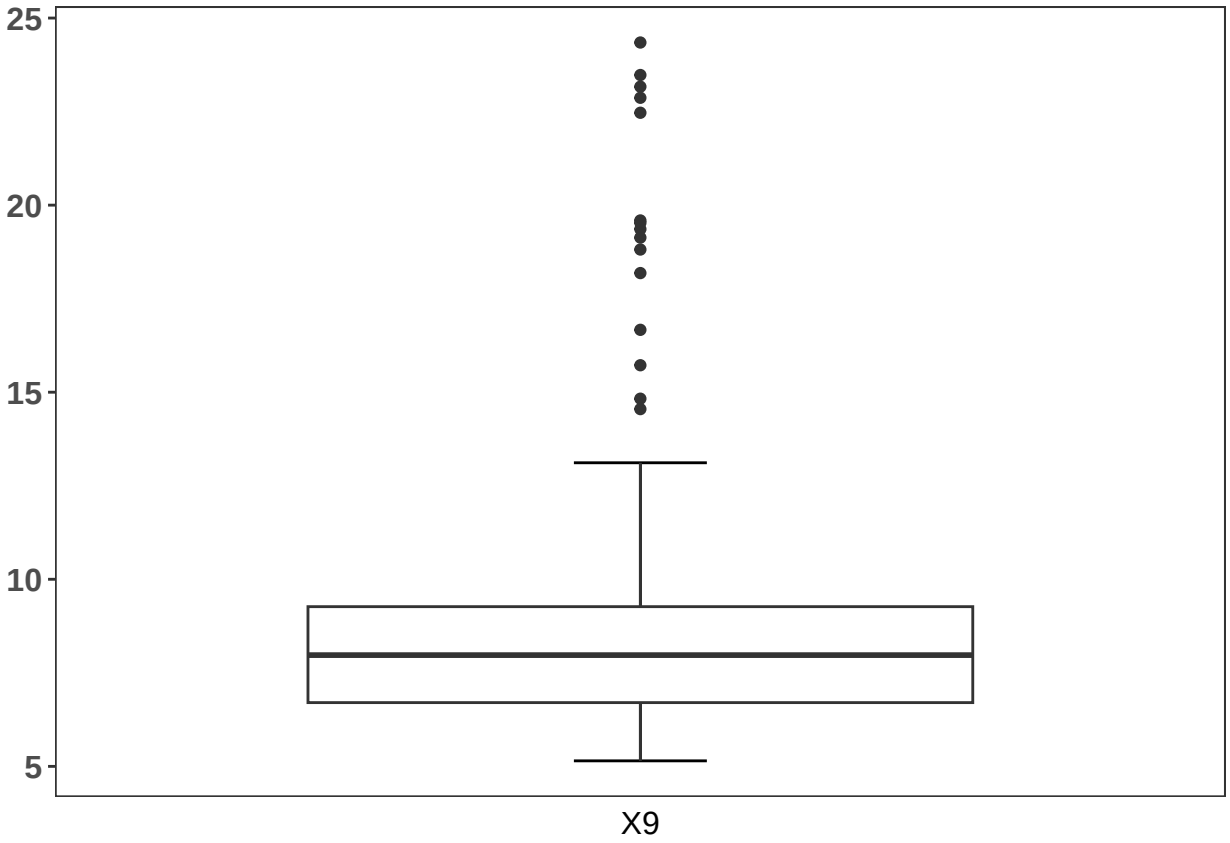

```
##
## $DescPlot$VizStatDesc_Test_Box_X7
```




```
##
## $DescPlot$VizStatDesc_Test_Box_X8
```




```
##
## $DescPlot$VizStatDesc_Test_Box_X9
```




```
##
##
## $DescPlot_para
## $DescPlot_para$VizStatDesc_Train_Box_C1
## $DescPlot_para$VizStatDesc_Train_Box_C1$width
## [1] 12
##
## $DescPlot_para$VizStatDesc_Train_Box_C1$height
## [1] 15
##
##
## $DescPlot_para$VizStatDesc_Train_Box_C2
## $DescPlot_para$VizStatDesc_Train_Box_C2$width
## [1] 12
##
## $DescPlot_para$VizStatDesc_Train_Box_C2$height
## [1] 15
##
##
## $DescPlot_para$VizStatDesc_Train_Box_X7
## $DescPlot_para$VizStatDesc_Train_Box_X7$width
## [1] 12
##
## $DescPlot_para$VizStatDesc_Train_Box_X7$height
## [1] 15
##
```

```
##
## $DescPlot_para$VizStatDesc_Train_Box_X8
## $DescPlot_para$VizStatDesc_Train_Box_X8$width
## [1] 12
##
## $DescPlot_para$VizStatDesc_Train_Box_X8$height
## [1] 15
##
##
## $DescPlot_para$VizStatDesc_Train_Box_X9
## $DescPlot_para$VizStatDesc_Train_Box_X9$width
## [1] 12
##
## $DescPlot_para$VizStatDesc_Train_Box_X9$height
## [1] 15
##
##
## $DescPlot_para$VizStatDesc_Test_Box_C1
## $DescPlot_para$VizStatDesc_Test_Box_C1$width
## [1] 12
##
## $DescPlot_para$VizStatDesc_Test_Box_C1$height
## [1] 15
##
##
## $DescPlot_para$VizStatDesc_Test_Box_C2
## $DescPlot_para$VizStatDesc_Test_Box_C2$width
## [1] 12
##
## $DescPlot_para$VizStatDesc_Test_Box_C2$height
## [1] 15
##
##
## $DescPlot_para$VizStatDesc_Test_Box_X7
## $DescPlot_para$VizStatDesc_Test_Box_X7$width
## [1] 12
##
## $DescPlot_para$VizStatDesc_Test_Box_X7$height
## [1] 15
##
##
## $DescPlot_para$VizStatDesc_Test_Box_X8
## $DescPlot_para$VizStatDesc_Test_Box_X8$width
## [1] 12
##
## $DescPlot_para$VizStatDesc_Test_Box_X8$height
## [1] 15
##
##
## $DescPlot_para$VizStatDesc_Test_Box_X9
## $DescPlot_para$VizStatDesc_Test_Box_X9$width
## [1] 12
##
## $DescPlot_para$VizStatDesc_Test_Box_X9$height
```

```
## [1] 15
```

StatComp

StatComp provides the function of size comparison between groups. For the time being, only the typical Wilcoxon rank test is provided to test the difference between groups, and other tests will be added gradually in the future. Group/Task/Vars/VarsBy/Method/Layout/Brightness/Palette parameters must be entered. Attention please, PID must be got from the return result of InitStat(). StatDesc can only run successfully after successfully running InitStat and LoadStat functions.

Group/Task/Vars/VarsBy/Method/Layout/Brightness/Palette parameters can be a character, run `?StatComp` to see more details.

```
#res_comp <- StatComp(PID = res$PID,
#                      Group = T,
#                      Task = "mean",
#                      Vars = "X5,X6,X7,X8,X9",
#                      VarsBy = "Y1",
#                      Method = "wilcox",
#                      Layout = "density",
#                      Brightness = "dark",
#                      Palette = "default1")
```

StatCorr

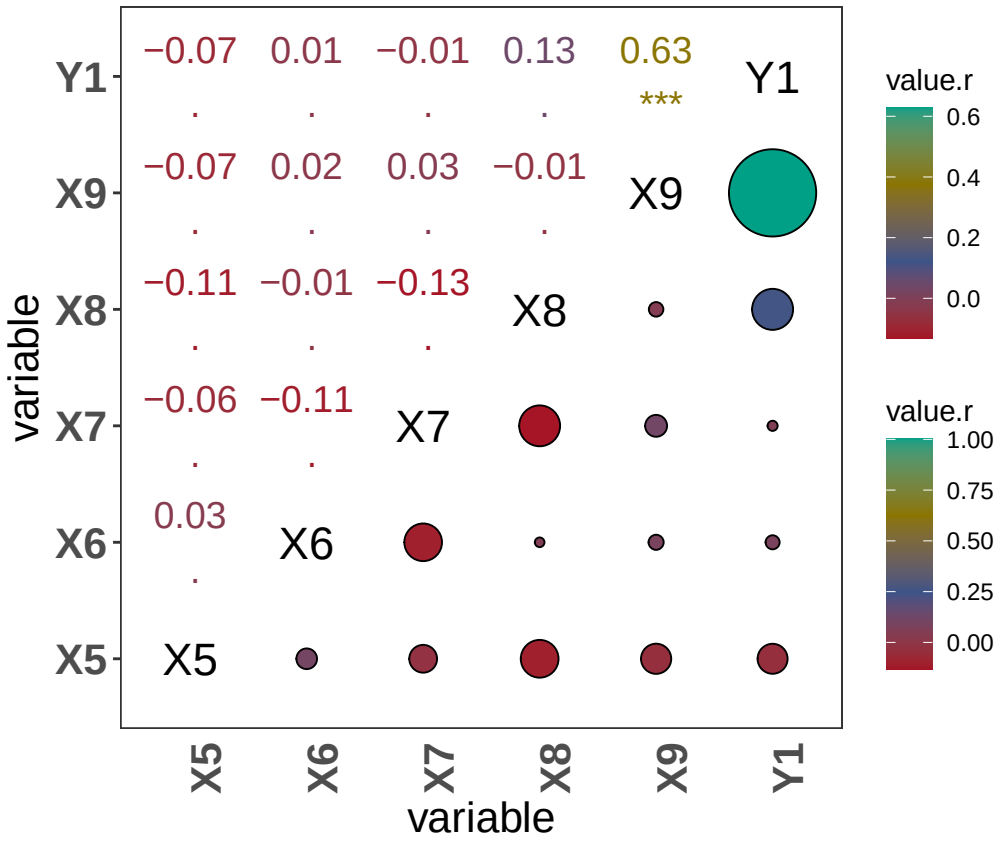
StatCorr provides the function of calculating of correlation coefficient between variables. For the time being, there are two typical statistics method Pearson correlation coefficient or Spearman rank correlation coefficient is provided to calculating of correlation, and statistics method will be added gradually in the future. Group/VarsX/VarsBy/Method/Layout/Brightness/Palette parameters must be entered. Attention please, PID must be got from the return result of InitStat(). StatDesc can only run successfully after successfully running InitStat and LoadStat functions.

Group/VarsX/VarsBy/Method/Layout/Brightness/Palette parameters can be a character, run `?StatCorr` to see more details.

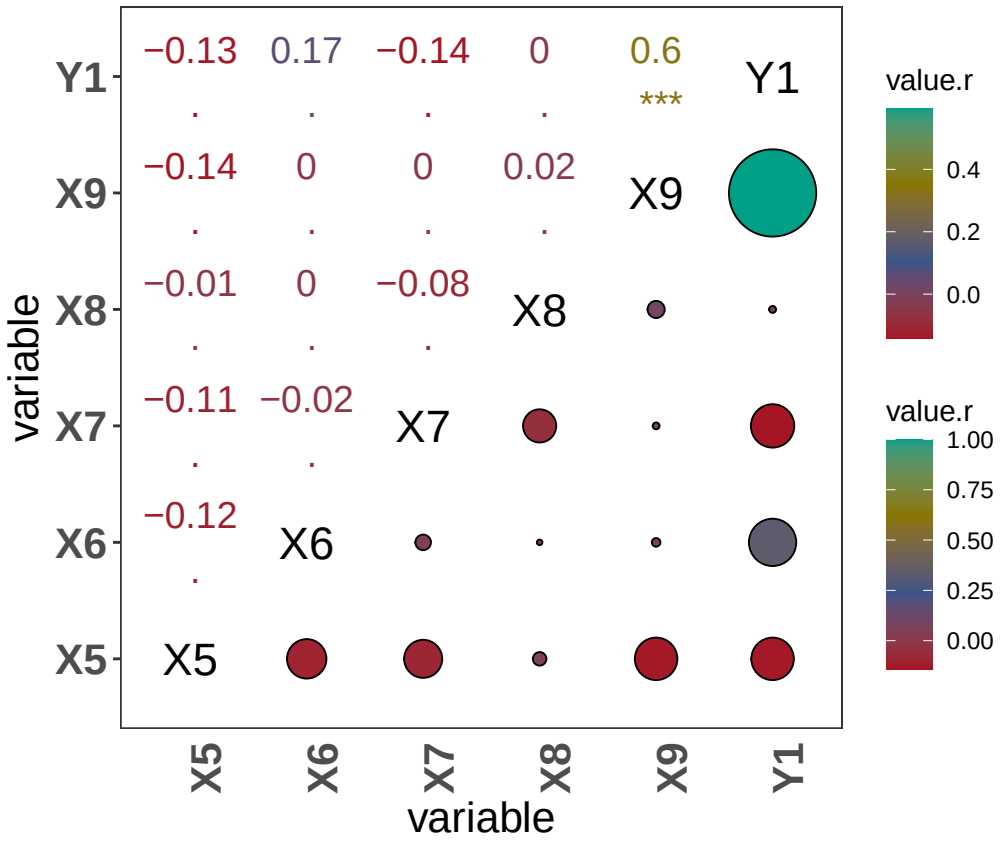
```
res = InitStat()

res1 = LoadStat(PID = res$PID,
                UseExample = "example#1")

res2 = StatCorr(PID = res$PID,
                Group = T,
                VarsX = "X5,X6,X7,X8,X9",
                VarsY = "Y1",
                VarsBy = "Y1",
                Method = "pearson",
                Layout = "bubble",
                Brightness = "dark",
                Palette = "nature")
res2
```

```
## $Corrtable
## $Corrtable$StatCorr_pearson
## # A tibble: 24 x 9
##    Terms Group Vars       X5       X6       X7       X8      X9       Y1
##    <chr> <chr> <chr>   <dbl>    <dbl>    <dbl>    <dbl>   <dbl>    <dbl>
```

```
##  1 r     Train X5     1       0.0303  -0.0578  -0.111   -0.0689 -0.0680
##  2 r     Train X6     0.0303  1       -0.110   -0.00523  0.0151  0.0127
##  3 r     Train X7    -0.0578 -0.110    1       -0.131    0.0345 -0.00585
##  4 r     Train X8    -0.111  -0.00523 -0.131    1       -0.0138  0.132
##  5 r     Train X9    -0.0689  0.0151   0.0345  -0.0138   1       0.626
##  6 r     Train Y1    -0.0680  0.0127  -0.00585  0.132    0.626   1
##  7 P     Train X5    NA       0.717    0.486    0.181    0.406   0.411
##  8 P     Train X6     0.717  NA        0.184    0.950    0.856   0.878
##  9 P     Train X7     0.486   0.184   NA        0.111    0.675   0.943
## 10 P     Train X8     0.181   0.950    0.111   NA        0.867   0.108
## # ... with 14 more rows
##
##
## $CorrPlot
## $CorrPlot$VizStatCorr_Bubble_Train_dark_nature
```




```
##
## $CorrPlot$VizStatCorr_Bubble_Test_dark_nature
```




```
##
##
## $CorrPlot_para
## $CorrPlot_para$VizStatCorr_Bubble_Train_dark_nature
## $CorrPlot_para$VizStatCorr_Bubble_Train_dark_nature$width
## [1] 11.6
##
## $CorrPlot_para$VizStatCorr_Bubble_Train_dark_nature$height
## [1] 11
##
##
## $CorrPlot_para$VizStatCorr_Bubble_Test_dark_nature
## $CorrPlot_para$VizStatCorr_Bubble_Test_dark_nature$width
## [1] 11.6
##
## $CorrPlot_para$VizStatCorr_Bubble_Test_dark_nature$height
## [1] 11
```

After all the analysis is done, please run the "FuncExit()" function to delete the data uploaded to the server.

```
FuncExit(PID = res$PID)
```

```
## [1] "Success to exit. Thanks for using ExposomeX platform!"
```

5 ExpoViz module

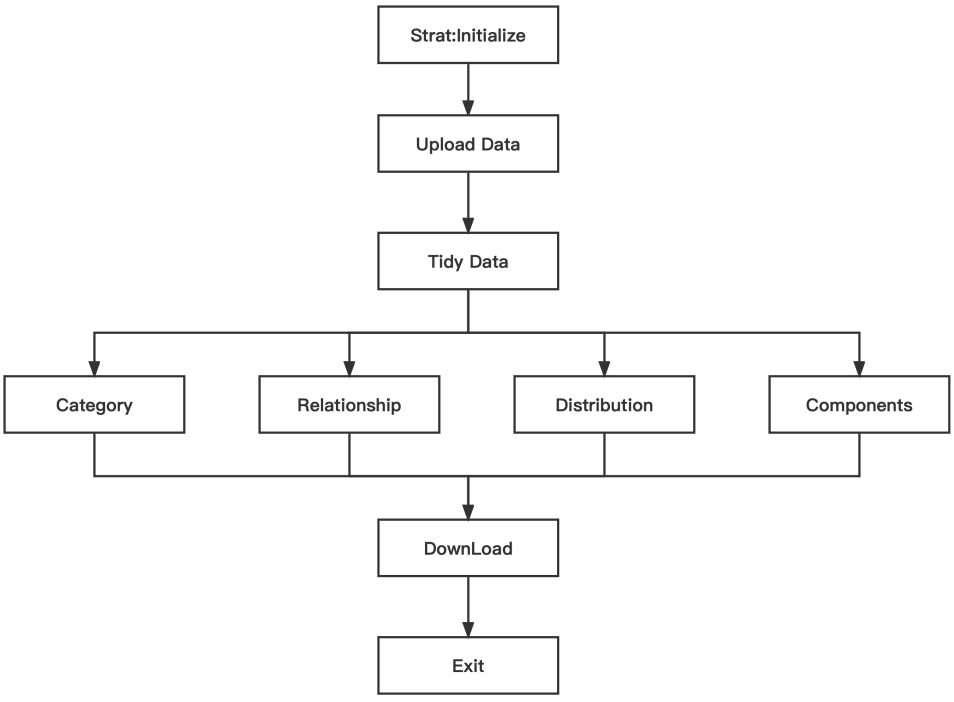
5.1 Application domain

The ***exviz*** package is designed for the data visualization of different statistical and biological analysis in a user friendly and easy way, including four typical classes of visualization. The visualization of the high dimension data is also very useful for the users in the field. The data visualization of different statistical analysis as well as the biological interaction can save much time for the data interpretation. Here, we mainly classify all the visualization task into four types, including category (distinguishing the characteristics of various groups), relationship (statistical relationship between various features), distribution (the distribution characters of the single or multiple factors), and components (the inclusion relationship between the part components and the whole). For each type, users only need to provide the original dataset by following the template and the target types, the ***exviz*** module can generate various potential visualization types.

5.2 Theory

The ***exviz*** package is based on "ggplot2" package and its extension packages used for four typical classes of data visualization.Accurate and beautiful visualization can make readers understand the paper presentation easily.The main functions of the visualization include: (1) To show the data truthfully,accurately and comprehensively; (2) To carry more information in a smaller space; (3) To reveal the essence,relationship and rule of data.

5.3 Work pipeline



Initialize package

Make sure that the packages needed is already installed.

```
# The following two packages should be installed in advance
```

```
# devtools::install_github("ExposomeX/exviz", force = TRUE)
# devtools::install_github("ExposomeX/extidy", force = TRUE)

#library(exviz)
# library(extidy)
library(tidyverse)

# devtools::install_github("ExposomeX/exposomex", force = TRUE)
library(exposomex)
```

Besides, we also strongly recommended the users to install the package *extidy* for data processing.

At first, you need to initialize the calculation environment using a series of initialization functions, e.g., InitCros, InitMo, InitTidy, InitViz, InitBiolink, etc. Here, we use the package "exviz" for data visualization for example. The detailed information about the functions and returned value will be introcuced in the following chapters.

```
res = InitViz()
res
```

```
## <eSet>
##   Public:
##     AddCommand: function (x)
##     AddLog: function (x)
##     clone: function (deep = FALSE)
##     ExcecutionLog: Complete initializing the Exoverse module.2022.12.14 14. ...
##     Expo: list
##     ExpoDel: list
##     FileDirIn: NULL
##     FileDirOut: /home/ubuntu/@changxin/R_Exposome_1.0/output_144722MBWWKV
##     PID: 144722MBWWKV
##     RCommandLog: eSet <- InitVisual(PID = Any ID your like, FileDirOut =  ...
```

Here, we can see that the returned value "res" is an R6 object. It contains an unique program ID of res$PID (e.g., "100737GJMWJA"), which is random generated by the system. Users need use it in the following step for further data process.

Upload data

Secondly, you need to load data file for visualization. The PID Parameter, you can enter res$PID which is random generated by the system when you run the InitViz function. If you want to try this package at first, you can input the UseExample Parameter with "example#1" to use our example data. Or you can enter *UseExample = "default"* to use your own data, you can enter *DataPath =* and *VocaPath =* to choose you own input data file and vocabulary file directories. The demand of these Parameters can see in the help of *LoadViz* function.

You should note that the format of the input data file and vocabulary file is ruled. You can see the demand of the data format by visiting the following website http://www.exposomex.cn/#/expoviz.

The details for each parameter are listed below:

PID

chr. Program ID. It must be the same with the PID generated by InitViz.

UseExample

chr. Method of uploading data. If "default", user should upload their own data files,or use "example#1" provided by this module.

ExdataPath

chr. Input data file directory, e.g. "D:/test/eg_expoviz_data.xlsx". It should be noted that the slash symbol is "/", not "\".

VodataPath

chr. Input vocabulary file directory, e.g. "D:/test/eg_expoviz_voca.xlsx". It should be noted that the slash symbol is "/", not "\".

```
res1 <- LoadViz(res$PID,
                UseExample = "example#1")

res1$Expo$Voca %>%
  dplyr::slice(1:20) %>%
  knitr::kable(format = "latex",
               align = "l") %>%
  kableExtra::kable_styling(full_width = F,
                            latex_options = "striped",
                            position = "left",
                            font_size = 10)
```

SerialNo	SerialNo_Raw	FullName	GroupName	Lod
Y1	Y1	Y_disc	Outcome	NA
Y2	Y2	Y_cont	Outcome	NA
C1	C1	Cov_1	Demography	NA
C2	C2	Cov_2	Demography	NA
C3	C3	Cov_3	Demography	NA
C4	C4	Cov_4	Demography	NA
C5	C5	Cov_5	Demography	NA
C6	C6	Cov_6	Demography	NA
X1	X1	TE_1	Chemical	0.5
X2	X2	TE_2	Chemical	0.5
X3	X3	TE_3	Chemical	0.5
X4	X4	TE_4	Chemical	0.5
X5	X5	TE_5	Chemical	0.5
X6	X6	TE_6	Chemical	0.5
X7	X7	TE_7	Chemical	0.5
X8	X8	TE_8	Chemical	0.5
X9	X9	CH1	Chemical	5.0
X10	X10	CH2	Chemical	5.0
X11	X11	CH3	Chemical	5.0
X12	X12	CH4	Chemical	5.0

```
res1$Expo$Data %>%
  dplyr::select(SampleID:C2) %>%
  dplyr::slice(1:20) %>%
  knitr::kable(format = "latex",
               align = "l") %>%
  kableExtra::kable_styling(full_width = F,
                            latex_options = "striped",
                            position = "left",
                            font_size = 10)
```

SampleID	SubjectID	Group	Y1	Y2	C1	C2
Tr1	S1	train	1	-101	26.86773	25.35056
Tr2	S2	train	0	-51	30.91822	23.94432
Tr3	S3	train	0	-37	25.82186	23.04579
Tr4	S4	train	1	-61	37.97640	21.21191
Tr5	S5	train	0	-28	31.64754	19.53762
Tr6	S6	train	0	-8	25.89766	20.77442
Tr7	S7	train	1	-63	32.43715	27.00009
Tr8	S8	train	0	-35	33.69162	22.13620
Tr9	S9	train	0	-14	32.87891	19.84672
Tr10	S10	train	1	-99	28.47306	29.60787
Tr11	S11	train	0	-60	37.55891	25.27530
Tr12	S12	train	0	-32	31.94922	23.28406
Tr13	S13	train	0	-73	26.89380	27.17545
Tr14	S14	train	0	-18	18.92650	26.65927
Tr15	S15	train	0	-48	35.62465	22.14227
Tr16	S16	train	0	-20	29.77533	30.61831
Tr17	S17	train	0	-9	29.91905	23.23492
Tr18	S18	train	1	-98	34.71918	19.72652
Tr19	S19	train	0	-70	34.10611	23.56680
Tr20	S20	train	0	-36	32.96951	24.62261

Here, we can see that the returned value "res1" is an R6 object. It contains two data frames which are your input data. Users need use it in the following step for further data visualization.

Tidy data

The third step is to tidy your input data into appropriate form. You should install our expotidy package before use it. If users have installed the *exposomex* package before, you can ignore this. By the way, this step is not necessary, users need to decide whether to do this step based on their own data. Noted that almost most of the functions below in the tidy module are from the package *extidy*, where the users can access for detailed information.

```
#Imputation for variables that have missing values
res2 = TransImput(PID = res$PID,
                  Group = F,
                  Vars = "all.x",
                  Method ="lod")

#Delete variables with low variance
res3 = DelNearZeroVar(PID = res$PID)

#Delete variables with missing values
res4 = DelMiss(PID = res$PID)

#Transform datatype
res5 = TransType(PID = res$PID,
                 Vars = "Y1",
                 To="factor")

#Scale
res6 = TransScale(PID = res$PID,
```

```
                  Group = F,
                  Vars = "all.x",
                  Method = "normal")

#Distribution
res7 = TransDistr(PID = res$PID,
                  Vars = "all.x",
                  Method = "ln")
```

By now, we have prepared our dataset ready for the following visualization.

Category Visualization

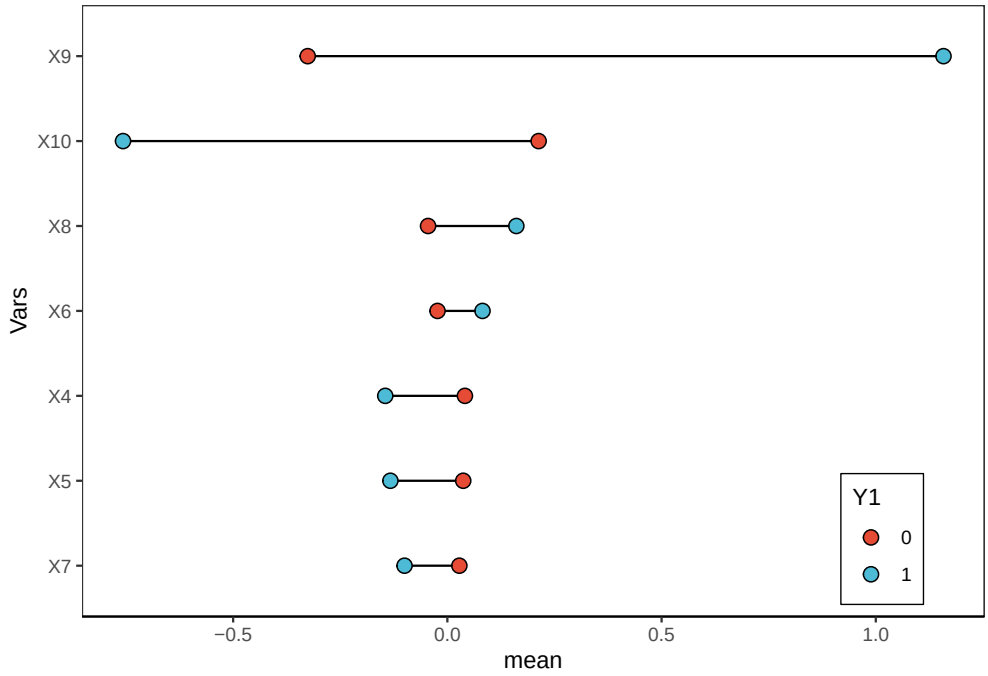
A category comparison chart generally contains two types of data: numerical and categorical, often used to compare the size of data. The input data must contain a categorical variable to use this function. We have a function to visualize data via dot plot.

The *Vars* Parameter is necessarily needed, other Parameters have default option. Or users can choose to enter other options.

The details for each parameter are listed below:
PID
chr. Program ID. It must be the same with the PID generated by InitViz.
OutPath
chr. Output file directory, e.g. "D:/test". It should be noted that the slash symbol is "/", not "\". If "default", the current working directory will be set.
Group
lgl.Whether to separate dataset into train and test datasets for data imputation,including T or F.The default option is F.
Vars
chr. Specifying the variables.Available options include:
"all.x", all independent variables;
"all.c", all covariate variables;
"all.cx", combination of All x and All x;
or input a character string specifying the variables,separated by comma "," without space(e.g."X4,X5,X6,X7,X8,X9,X10").No more than 50 variables be entered is recommended (< 50 variables).
Parameter
chr. Specifying which parameter of the data to be the ordinate of the output plot. Available options include: "mean", "median", "min", "max", "mad" or "sd".Default is "mean".
Brightness
chr. Visualization brightness. Available options include "light" and "dark".
Palette
chr. Visualization palette. Available options include "default1", "default2" and "Journal". The "Journal" option provides several journal preference styles including cell, nature, science, lancet, nejm, and jama.

```
res8 = VizCateDot(PID = res$PID,
                  OutPath = "default",
                  Group = "F",
                  Vars = "X4,X5,X6,X7,X8,X9,X10",
                  Parameter = "mean",
                  Brightness = "light",
                  Palette = "default1")
```

```
res8$light_default1
```



Here, we can see that the returned value "res8" is a ggplot2 plot. The dot plot is used to display the relative position of two data points in the same time period, or compare the difference between the two categorical variables. The vertical coordinates are ordered by dividing the absolute value of the difference by the mean value.

Relationship Visualization

Data relational chart includes relatioanal, hierarchical and network relational charts, which respectively show the relationships between two or more variabels, the hierarchical relationships between data individuals and the visualization of relational data without hierarchical structures.

Network Visualization

The *VarsY* Parameter is necessarily needed, other Parameters have default option. Or users can choose to enter other options.

The details for each parameter are listed below:

PID

chr. Program ID. It must be the same with the PID generated by InitViz.

OutPath

chr. Output file directory, e.g. "D:/test". It should be noted that the slash symbol is "/", not "\". If "default", the current working directory will be set.

VarsY

chr. Specifying the dependent variables(e.g."Y2").

VarsC

chr. Specifying the covariate variable.Available options include:

"all.c", all covariate variables;

or input a character string specifying the variables,separated by comma "," without space(e.g."C1,C2").

VarsX

chr. Specifying the independent variables. Available options include:

"all.x", all independent variables;

or input a character string specifying the variables,separated by comma "," without space(e.g."X4,X5,X6,X7,X8,X9,X10").

Family

chr. Specifying the data distribution in order to determine the visualization method. Available options include:"gaussian","poisson" and "logistic".Notice that the family are determined by data type of an outcome, or the plot can not be visualized.

Layout

chr. Visualization layout. Available options include "force-directed" and "degree-circle".

CutOff

num. Partial outcomes to visualize which is determined by correlation coefficient r. The range must between 0 and 1.

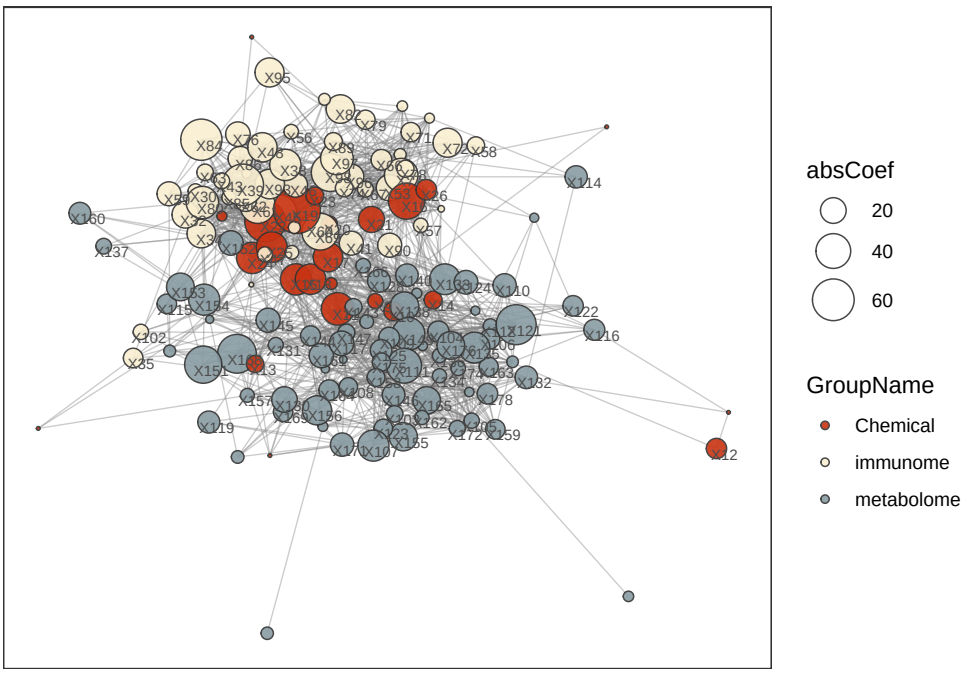
Brightness

chr. Visualization brightness. Available options include "light" and "dark".

Palette

chr. Visualization palette. Available options include "default1", "default2" and "Journal". The "Journal" option provides several journal preference styles including cell, nature, science, lancet, nejm, and jama.

```
res9 = VizRelatNetwork(PID = res$PID,
                       OutPath = "default",
                       VarsY = "Y2",
                       VarsC = "all.c",
                       VarsX = "all.x",
                       Family = "gaussian",
                       Layout = "force-directed",
                       CutOff = 0.8,
                       Brightness = "light",
                       Palette = "default1")
res9$light_default1
```



Here, we can see that the returned value "res9 is a ggplot2 plot. The network plot is used to visualize the relationship between the input variables.The point color is defined by the coefficient which is calculate by the

generalized linear model, the point size is defined by the p-value which is determined by the relationship between the input variables.

Edge Bundling Visualization

The *VarsY* Parameter is necessarily needed, other Parameters have default option. Or users can choose to enter other options.

The details for each parameter are listed below:

PID

chr. Program ID. It must be the same with the PID generated by InitViz.

OutPath

chr. Output file directory, e.g. "D:/test". It should be noted that the slash symbol is "/", not "\". If "default", the current working directory will be set.

VarsY

chr. Specifying the dependent variables(e.g."Y2").

VarsC

chr. Specifying the covariate variable.Available options include:

"all.c", all covariate variables;

or input a character string specifying the variables,separated by comma "," without space(e.g."C1,C2").

VarsX

chr. Specifying the independent variables. Available options include:

"all.x", all independent variables;

or input a character string specifying the variables,separated by comma "," without space(e.g."X4,X5,X6,X7,X8,X9,X10").

Family

chr. Specifying the data distribution in order to determine the visualization method. Available options include:"gaussian","poisson" and "logistic".Notice that the family are determined by data type of an outcome, or the plot can not be visualized.

SizeFor

chr. Parameter to represent the size of the points in the output plot. Available options include "pvalue" and "beta".The default option is "pvalue".

CutOff

num. Partial outcomes to visualize which is determined by correlation coefficient r. The range must between 0 and 1.

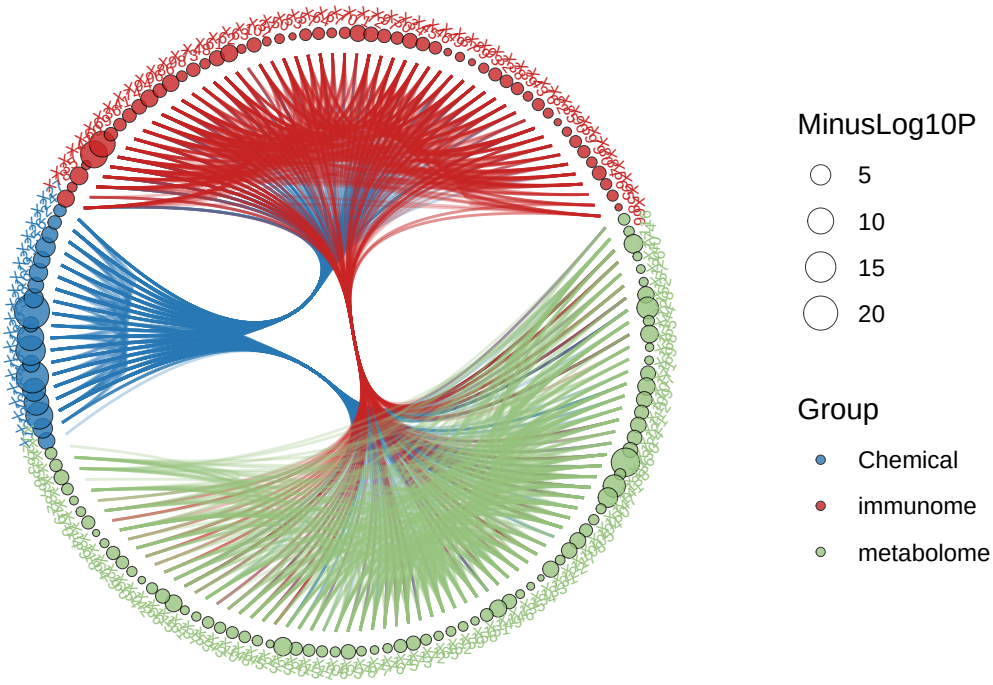
Brightness

chr. Visualization brightness. Available options include "light" and "dark".

Palette

chr. Visualization palette. Available options include "default1", "default2" and "Journal". The "Journal" option provides several journal preference styles including cell, nature, science, lancet, nejm, and jama.

```
res10 = VizRelatEdgeBundling(PID = res$PID,
                             OutPath = "default",
                             VarsY = "Y2",
                             VarsC = "all.c",
                             VarsX = "all.x",
                             Family = "gaussian",
                             SizeFor = "pvalue",
                             Brightness = "light",
                             Palette = "default1")
res10$light_default1
```



Here, we can see that the returned value "res10" is a ggplot2 plot. The edge bundling plot is used to bundle the edges closely in order to reduce complexity. The point color is defined by the group of the input independent variables, the point size is defined by the p-value which is determined by the relationship between the input variables.

Heatmap Visualization

The *VarsY* and *VarsX* Parameter are necessarily needed, other Parameters have default option. Or users can choose to enter other options.

The details for each parameter are listed below:

PID

chr. Program ID. It must be the same with the PID generated by InitViz.

OutPath

chr. Output file directory, e.g. "D:/test". It should be noted that the slash symbol is "/", not "\". If "default", the current working directory will be set.

Group

lgl.Whether to separate dataset into train and test datasets for data imputation,including T or F.The default option is F.

VarsY

chr. Specifying the dependent variables(e.g."Y2").

VarsX

chr. Specifying the independent variables. Available options include:

"all.x", all independent variables;

or input a character string specifying the variables,separated by comma "," without space(e.g."X4,X5,X6,X7,X8,X9,X10").

Method

chr. Method to calculate the correlation. Default option is "spearman".

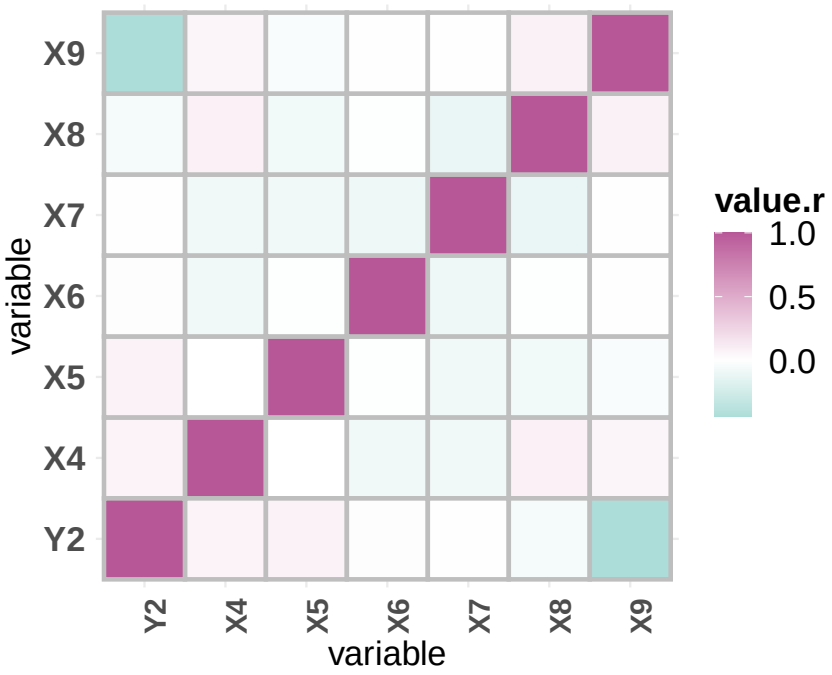
Brightness

chr. Visualization brightness. Available options include "light" and "dark".

Palette

chr. Visualization palette. Available options include "default1", "default2" and "Journal". The "Journal" option provides several journal preference styles including cell, nature, science, lancet, nejm, and jama.

```
res11 = VizRelatHeatmap(PID = res$PID,
                        OutPath = "default",
                        Group = "F",
                        VarsY = "Y2",
                        VarsX = "X4,X5,X6,X7,X8,X9",
                        Method = "spearman",
                        Brightness = "light",
                        Palette = "default1")

res11$light_default1
```



Here, we can see that the returned value "res11" is a ggplot2 plot. The heatmap plot is used to display data in color changes as a matrix. The point color is defined by the coefficient r which is determined by the relationship between the input variables.

Matrix Visualization

The *VarsY* and *VarsX* Parameter are necessarily needed, other Parameters have default option. Or users can choose to enter other options.

The details for each parameter are listed below:

PID

chr. Program ID. It must be the same with the PID generated by InitViz.

OutPath

chr. Output file directory, e.g. "D:/test". It should be noted that the slash symbol is "/", not "\". If "default", the current working directory will be set.

Group

lgl.Whether to separate dataset into train and test datasets for data imputation,including T or F.The default option is F.

VarsY

chr. Specifying the dependent variables(e.g."Y2").

VarsX

chr. Specifying the independent variables. Available options include:

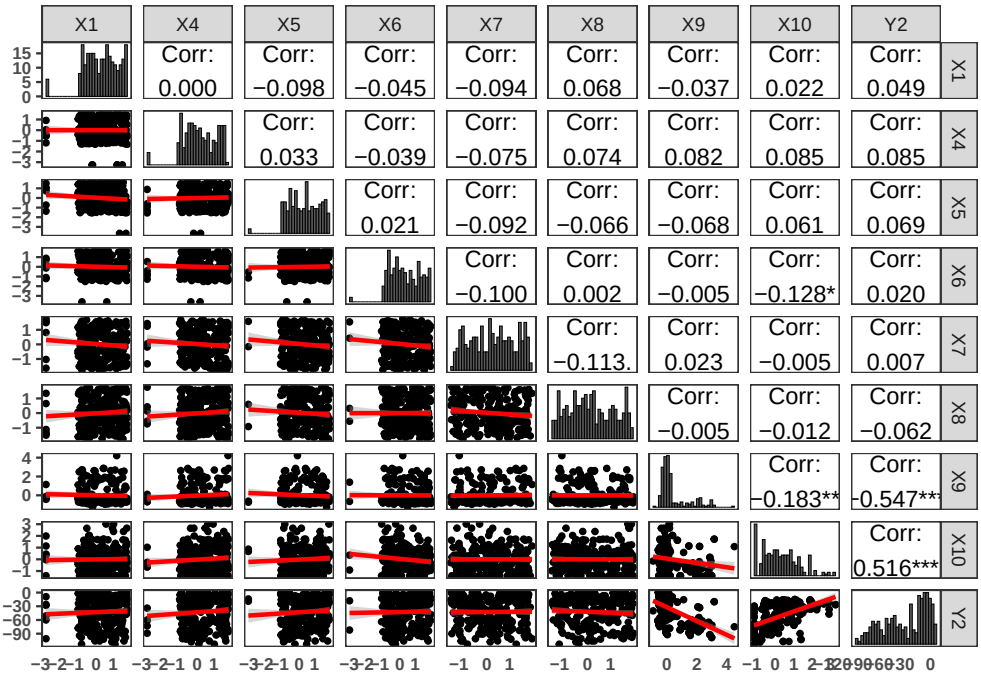
"all.x", all independent variables;

or input a character string specifying the variables,separated by comma "," without space(e.g."X4,X5,X6,X7,X8,X9,X10").

Method

chr. Method to calculate the correlation. Default option is "spearman".

```
res12 = VizRelatMatrix(PID = res$PID,
                       OutPath = "default",
                       Group = "F",
                       VarsY = "Y2",
                       VarsX = "X1,X4,X5,X6,X7,X8,X9,X10",
                       Method = "spearman")

res12$ALL
```



Here, we can see that the returned value "res12" is a ggplot2 plot. The matrix plot is used to make a matrix of plots with a given data set.

Distribution Visualization

Data distribution type chart mainly shows the values in the data set and their frequency or distribution rule. Generally, the horizontal axis represents the data type and the vertical axis represents the distribution.We have a function to visualize data via sierra plot.

The *Vars* Parameter is necessarily needed, other Parameters have default option. Or users can choose to enter other options.

The details for each parameter are listed below:

PID

chr. Program ID. It must be the same with the PID generated by InitViz.

OutPath

chr. Output file directory, e.g. "D:/test". It should be noted that the slash symbol is "/", not "\". If "default", the current working directory will be set.

Group

lgl.Whether to separate dataset into train and test datasets for data imputation,including T or F.The default option is F.

Vars

chr. Specifying the variables.Available options include:

"all.x", all independent variables;

"all.c", all covariate variables;

"all.cx", combination of All x and All x;

or input a character string specifying the variables,separated by comma "," without space(e.g."X4,X5,X6,X7,X8,X9,X10").No more than 50 variables be entered is recommended (< 50 variables).

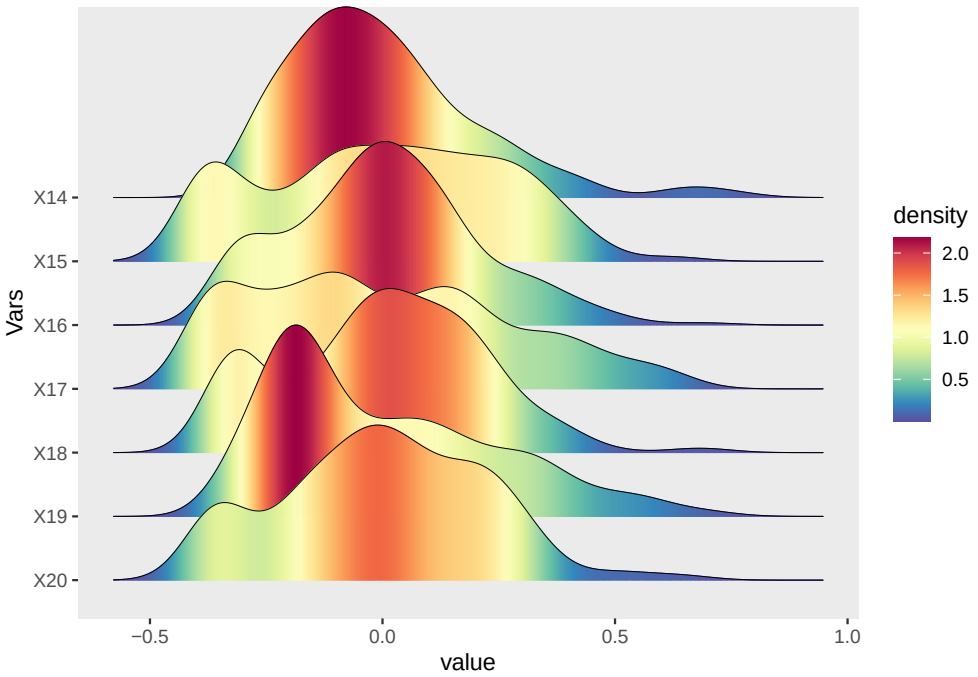
Brightness

chr. Visualization brightness. Available options include "light" and "dark".

Palette

chr. Visualization palette. Available options include "default1", "default2" and "Journal". The "Journal" option provides several journal preference styles including cell, nature, science, lancet, nejm, and jama.

```
res13 = VizDistrSierra(PID = res$PID,
                       OutPath = "default",
                       Group = "F",
                       Vars = "X14,X15,X16,X17,X18,X19,X20",
                       Brightness = "light",
                       Palette = "default1")
res13$light_default1
```



Here, we can see that the returned value "res13" is a ggplot2 plot. The sierra plot is used to visualize the kernel density estimation of data.

Components Visualization

Local integrity chart can show the proportion information of the local component and the whole. We have a function to visualize data via dendrogram plot.

The *Vars* Parameter is necessarily needed, other Parameters have default option. Or users can choose to enter other options.

The details for each parameter are listed below:

PID

chr. Program ID. It must be the same with the PID generated by InitViz.

OutPath

chr. Output file directory, e.g. "D:/test". It should be noted that the slash symbol is "/", not "\". If "default", the current working directory will be set.

Group

lgl.Whether to separate dataset into train and test datasets for data imputation,including T or F.The default option is F.

Vars

chr. Specifying the variables.Available options include:

"all.x", all independent variables;

"all.c", all covariate variables;

"all.cx", combination of All x and All x;

or input a character string specifying the variables,separated by comma "," without space(e.g."X4,X5,X6,X7,X8,X9,X10").No more than 50 variables be entered is recommended (< 50 variables).

Parameter

chr. Specifying which parameter of the data to be the ordinate of the output plot. Available options include: "mean", "median", "min", "max", "mad" or "sd".Default is "mean".

DistMethod

chr.The distance measure. This must be one of "euclidean", "maximum" or "manhattan".Default is "euclidean".

ClusterMethod

chr.The agglomeration method. This should be one of "ward.D", "ward.D2" or "single".Default is "ward.D".

ClusterNum

num. The number of groups for cutting the tree.Default is 4.

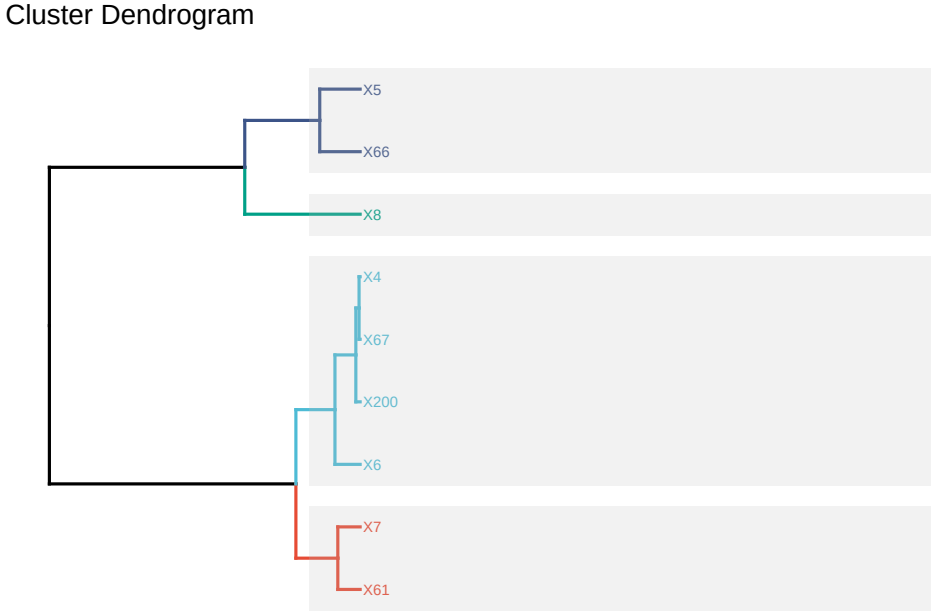
Brightness

chr. Visualization brightness. Available options include "light" and "dark".

Palette

chr. Visualization palette. Available options include "default1", "default2" and "Journal". The "Journal" option provides several journal preference styles including cell, nature, science, lancet, nejm, and jama.

```
res14 = VizCompoDendrogram(PID = res$PID,
                           OutPath = "default",
                           Group = "T",
                           Vars = "X4,X5,X6,X7,X8,X61,X66,X67,X200",
                           Parameter = "median",
                           DistMethod = "euclidean",
                           ClusterMethod = "ward.D2",
                           ClusterNum = "4",
                           Brightness = "light",
                           Palette = "default1")
res14$all_light_default1
```



Here, we can see that the returned value "res9" is a list containing three ggplot2 plots. Here we show one of them. The dendrogram plot is used to plot beautiful dendrograms.

```
FuncExit(PID = res$PID)
```

```
## [1] "Success to exit. Thanks for using ExposomeX platform!"
```

6 ExpoCros module

6.1 Application domain

ExpoCros module was designed to analyze the cross-sectional data from exposome-wide association study (EWAS). This data structure can be obtained from the epidemiological designs of cross-section, case-control, and cohort. Users can easily get the modeling results and their visualization plots with high quality by following the detailed instructions in each step. For association, the model is chosen according to the data type of health outcome; while for prediction, most of the frequently-used models are evaluated for users' reference.

6.2 Theory

6.3 Work pipeline

Initialize package

Make sure that the required packages is already installed.

```
# The following two packages should be installed in advance
# devtools::install_github("ExposomeX/excros", force = TRUE)
# devtools::install_github("ExposomeX/extidy", force = TRUE)

# library(excros)
# library(extidy)
library(tidyverse)

# devtools::install_github("ExposomeX/exposomex", force = TRUE)
library(exposomex)
```

At first, you need to initialize the calculation environment using a series of initialization functions, e.g., InitCros, InitMo, InitTidy, InitViz, InitBiolink, etc. Here, we use the package "excros" for cross-sectional data analysis for example. The detailed information about the functions and returned value will be introcuced in the following chapters.

```
res = InitCros()
res
```

```
## <eSet>
##   Public:
##     AddCommand: function (x)
##     AddLog: function (x)
##     clone: function (deep = FALSE)
##     EpiDesign: NULL
##     ExcecutionLog: Complete initializing the ExpoStat module.2022.12.14 14. ...
##     Expo: list
##     FileDirIn: NULL
##     FileDirOut: /home/ubuntu/@changxin/R_Exposome_1.0/output_144957LDYDKS
##     PID: 144957LDYDKS
##     RCommandLog: eSet <- InitCros(PID = Any ID your like, FileDirOut = An ...
##     VarsDel: NULL
```

Here, we can see that the returned value "res" is an R6 object. It contains an unique program ID of res$PID (e.g., "100737GJMWJA"), which is random generated by the system. Users need use it in the following step for further data process.

Upload data

```
res1 <- LoadCros(res$PID,
                 UseExample = "example#1")

res1$Expo$Voca %>%
  dplyr::slice(1:20) %>%
  knitr::kable(format = "latex",
               align = "l") %>%
  kableExtra::kable_styling(full_width = F,
                            latex_options = "striped",
                            position = "left",
                            font_size = 10)
```

SerialNo	SerialNo_Raw	FullName	GroupName	Lod
Y1	Y1	Y_cont	Outcome	NA
Y2	Y2	Y_disc	Outcome	NA
C1	C1	Cov_1	Demography	NA
C2	C2	Cov_2	Demography	NA
C3	C3	Cov_3	Demography	NA
C4	C4	Cov_4	Demography	NA
C5	C5	Cov_5	Demography	NA
C6	C6	Cov_6	Demography	NA
X1	X1	TE_1	Chemical	0.5
X2	X2	TE_2	Chemical	0.5
X3	X3	TE_3	Chemical	0.5
X4	X4	TE_4	Chemical	0.5
X5	X5	TE_5	Chemical	0.5
X6	X6	TE_6	Chemical	0.5
X7	X7	TE_7	Chemical	0.5
X8	X8	TE_8	Chemical	0.5
X9	X9	CH1	Chemical	5.0
X10	X10	CH2	Chemical	5.0
X11	X11	CH3	Chemical	5.0
X12	X12	CH4	Chemical	5.0

```
res1$Expo$Data %>%
  dplyr::select(SampleID:C2) %>%
  dplyr::slice(1:20) %>%
  knitr::kable(format = "latex",
               align = "l") %>%
  kableExtra::kable_styling(full_width = F,
                            latex_options = "striped",
                            position = "left",
                            font_size = 10)
```

SampleID	SubjectID	Group	Y1	Y2	C1	C2
Tr1	S1	train	-101	1	26.86773	25.35056
Tr2	S2	train	-51	0	30.91822	23.94432
Tr3	S3	train	-37	0	25.82186	23.04579
Tr4	S4	train	-61	1	37.97640	21.21191
Tr5	S5	train	-28	0	31.64754	19.53762
Tr6	S6	train	-8	0	25.89766	20.77442
Tr7	S7	train	-63	1	32.43715	27.00009
Tr8	S8	train	-35	0	33.69162	22.13620
Tr9	S9	train	-14	0	32.87891	19.84672
Tr10	S10	train	-99	1	28.47306	29.60787
Tr11	S11	train	-60	0	37.55891	25.27530
Tr12	S12	train	-32	0	31.94922	23.28406
Tr13	S13	train	-73	0	26.89380	27.17545
Tr14	S14	train	-18	0	18.92650	26.65927
Tr15	S15	train	-48	0	35.62465	22.14227
Tr16	S16	train	-20	0	29.77533	30.61831
Tr17	S17	train	-9	0	29.91905	23.23492
Tr18	S18	train	-98	1	34.71918	19.72652
Tr19	S19	train	-70	0	34.10611	23.56680
Tr20	S20	train	-36	0	32.96951	24.62261

Tidy data

```
res2 <- DelMiss(res$PID)

res2$Expo$Voca %>%
  dplyr::slice(1:20) %>%
  knitr::kable(format = "latex",
               align = "l") %>%
  kableExtra::kable_styling(full_width = F,
                            latex_options = "striped",
                            position = "left",
                            font_size = 10)
```

SerialNo	SerialNo_Raw	FullName	GroupName	Lod
Y1	Y1	Y_cont	Outcome	NA
Y2	Y2	Y_disc	Outcome	NA
C1	C1	Cov_1	Demography	NA
C2	C2	Cov_2	Demography	NA
C3	C3	Cov_3	Demography	NA
C4	C4	Cov_4	Demography	NA
C5	C5	Cov_5	Demography	NA
C6	C6	Cov_6	Demography	NA
X1	X1	TE_1	Chemical	0.5
X2	X2	TE_2	Chemical	0.5
X3	X3	TE_3	Chemical	0.5
X4	X4	TE_4	Chemical	0.5
X5	X5	TE_5	Chemical	0.5
X6	X6	TE_6	Chemical	0.5
X7	X7	TE_7	Chemical	0.5
X8	X8	TE_8	Chemical	0.5
X9	X9	CH1	Chemical	5.0
X10	X10	CH2	Chemical	5.0
X11	X11	CH3	Chemical	5.0
X12	X12	CH4	Chemical	5.0

```
res2$Expo$Data %>%
  dplyr::select(SampleID:C2) %>%
  dplyr::slice(1:20) %>%
  knitr::kable(format = "latex",
               align = "l") %>%
  kableExtra::kable_styling(full_width = F,
                            latex_options = "striped",
                            position = "left",
                            font_size = 10)
```

SampleID	SubjectID	Group	Y1	Y2	C1	C2
Tr1	S1	train	-101	1	26.86773	25.35056
Tr2	S2	train	-51	0	30.91822	23.94432
Tr3	S3	train	-37	0	25.82186	23.04579
Tr4	S4	train	-61	1	37.97640	21.21191
Tr5	S5	train	-28	0	31.64754	19.53762
Tr6	S6	train	-8	0	25.89766	20.77442
Tr7	S7	train	-63	1	32.43715	27.00009
Tr8	S8	train	-35	0	33.69162	22.13620
Tr9	S9	train	-14	0	32.87891	19.84672
Tr10	S10	train	-99	1	28.47306	29.60787
Tr11	S11	train	-60	0	37.55891	25.27530
Tr12	S12	train	-32	0	31.94922	23.28406
Tr13	S13	train	-73	0	26.89380	27.17545
Tr14	S14	train	-18	0	18.92650	26.65927
Tr15	S15	train	-48	0	35.62465	22.14227
Tr16	S16	train	-20	0	29.77533	30.61831
Tr17	S17	train	-9	0	29.91905	23.23492
Tr18	S18	train	-98	1	34.71918	19.72652
Tr19	S19	train	-70	0	34.10611	23.56680
Tr20	S20	train	-36	0	32.96951	24.62261

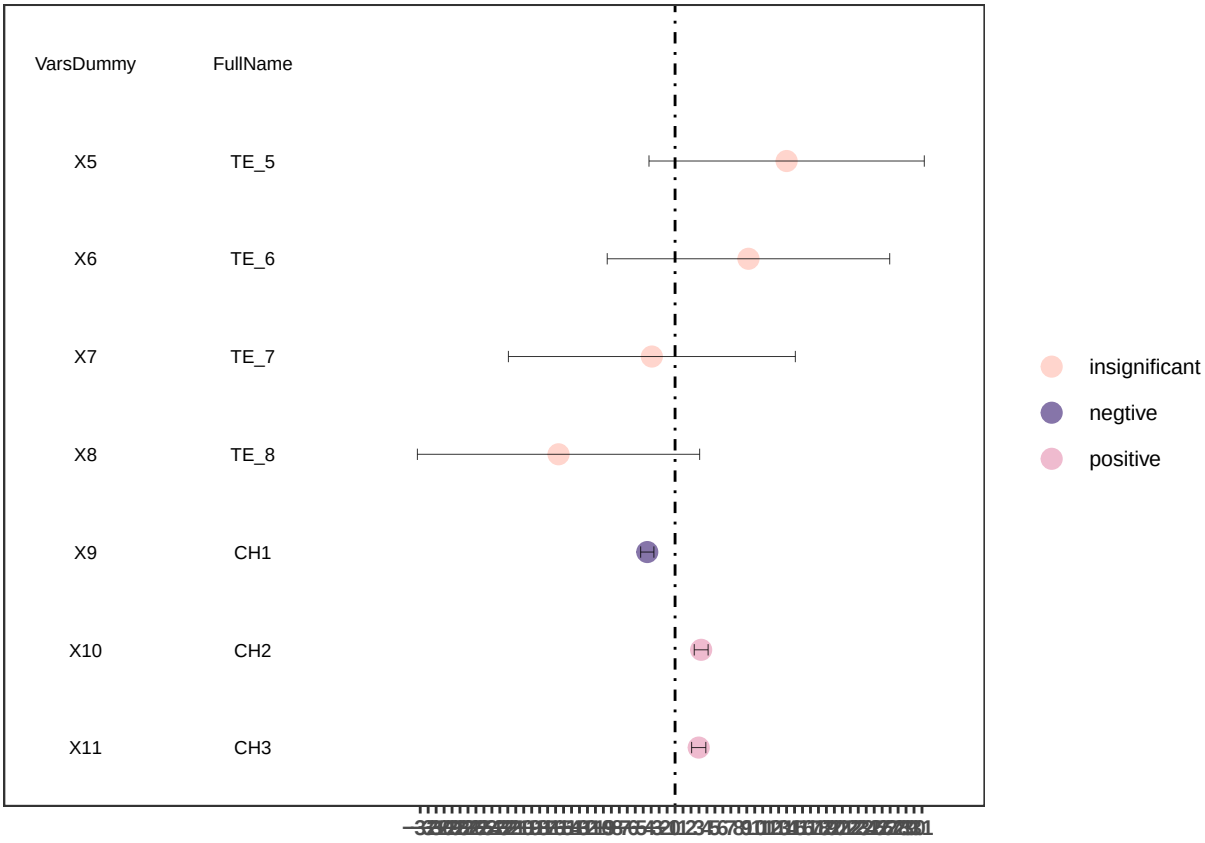
Modeling

```
res3 = CrosAsso(PID = res$PID,
                EpiDesign = "cohort",
                VarsY = "Y1",
                VarsX = "X5,X6,X7,X8,X9,X10,X11",
                VarsN = "single.factor" ,
                VarsSel = F,
                VarsSelThr = 0.1,
                IncCova = T,
                Family = "gaussian",
                RepMsr = F,
                Corstr = "ar1")

res3$Y1_single.factor_cohort_gaussian %>%
  dplyr::select(SerialNo:ci_h) %>%
  dplyr::slice(1:20) %>%
  knitr::kable(format = "latex",
               align = "l") %>%
  kableExtra::kable_styling(full_width = F,
                            latex_options = "striped",
                            position = "left",
                            font_size = 10)
```

SerialNo	Vars.dummy	beta.value	ci_l	ci_h
X5	X5	14.005401	-3.277425	31.288227
X6	X6	9.214098	-8.517261	26.945457
X7	X7	-2.915070	-20.927230	15.097090
X8	X8	-14.634061	-32.354215	3.086094
X9	X9	-3.489654	-4.319254	-2.660055
X10	X10	3.275350	2.404960	4.145739
X11	X11	2.968116	2.078754	3.857479

Visualize model

```
res4 = VizCrosAsso(PID = res$PID,
                   VarsN="single.factor",
                   VarsY="Y1",
                   Layout = "forest",
                   Brightness = "dark",
                   Palette = "default1")
res4$Y1_single.factor_forest_dark_default1
```




```
FuncExit(PID = res$PID)
```

```
## [1] "Success to exit. Thanks for using ExposomeX platform!"
```

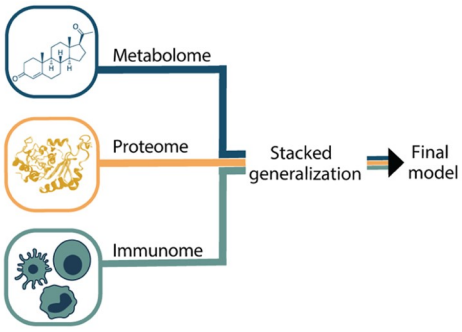
7 ExpoMo module

7.1 Application domain

Multi-omics research is a method of exploring the interactions between multiple substances in biological systems, including genomics, proteomics, metabolomics, etc., which together affect the phenotype and traits of organisms. With the rapid development of high-throughput sequencing and multiomics, we can develop more comprehensive predictive models and identify important features by integrating multi-omic data, which facilitates a deeper understanding of the complexity of biology. ***exmo*** package is designed to integrate the multi-omic data. It mainly aims to construct various stacked generalization models to predict the response or probability of outcome incidence, as well as providing the statistical explanation. Additionly, the package can provide visualization plots with high quality of the final calculation results to make it easier for users to understand.

7.2 Theory

The theory of the package ***exmo*** is based on "Integrated trajectories of the maternal metabolome, proteome, and immunome predict labor onset". And the code for machine learning and visualization of the results are based on "mlr3" package and "ggplot2" package.



Stelzer IA, Ghaemi MS, Han X, et al. Integrated trajectories of the maternal metabolome, proteome, and immunome predict labor onset. Sci Transl Med. 2021;13(592):eabd9898. doi:10.1126/scitranslmed.abd9898

7.3 Work pipeline

Initialize package

Make sure that the required packages is already installed.

```
# The following two packages should be installed in advance
# devtools::install_github("ExposomeX/exmo", force = TRUE)
# devtools::install_github("ExposomeX/extidy", force = TRUE)

# library(exmo)
# library(extidy)
library(tidyverse)

# devtools::install_github("ExposomeX/exposomex", force = TRUE)
library(exposomex)
```

The first step,you need to initialize the environment using "InitMO".

```
res = InitMO()
```

Here, we can see that the returned value "res" is an R6 object. It contains an unique program ID of res$PID (e.g., "100737GJMWJA"), which is random generated by the system. Users need use it in the

following step for further data process.

Upload data

```
res1 = LoadMO(PID = res$PID,
              UseExample="example#1")
```

Tidy data

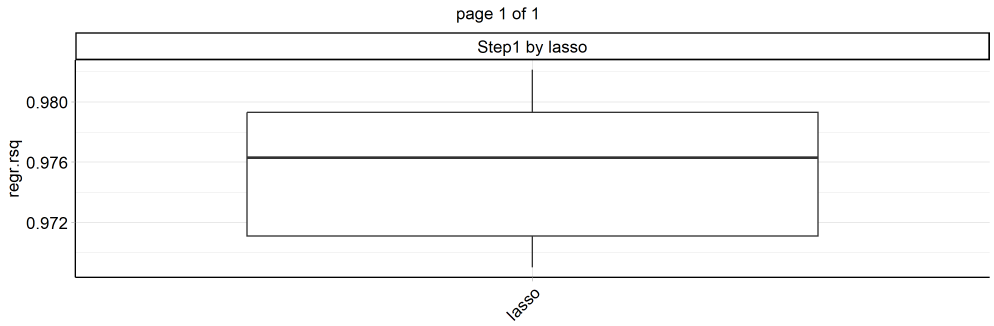
```
res2 = TransImput(PID = res$PID,
                  Group="T",
                  Vars="all.x",
                  Method="lod")

res3 = DelNearZeroVar(PID = res$PID)

res4 = DelMiss(PID = res$PID)

res5 = TransType(PID = res$PID,
                 Vars="all.x",
                 To="numeric")

res6 = TransClass(PID = res$PID,
                  Group="F",
                  Vars="C1",
                  LevelTo="4")

res7 = TransScale(PID = res$PID,
                  Group="T",
                  Vars="all.x",
                  Method="normal")

res8 = TransDistr(PID = res$PID,
                  Vars="C2",
                  Method="log10")

res10 = TransDummy(PID = res$PID,
                   Vars="default")
```

Modeling This step is the most critical part of the entire module. we will build multiomics model with function "MulOmicsCros".You can select one or more arbitrary learning methods in parameter SG_Lrns. Here we choose lasso(least absolute shrinkage and selection operator) and rf(random forest) as examples. The calculation time depends on the characteristics of your data, the number of learning methods, and the tuning method. For parameter TuneMethod, the default option can provide faster calculations but less accurate results than other autotune methods. If you want to train a better model, choose other auto-tune method and increase the number of tuning times.

```
res11 = MulOmicsCros(PID = res$PID,
                     OutPath = "default",
                     OmicGroups = "immunome,metabolome,proteome",
                     VarsY = "Y1",
                     VarsC = "all.c",
                     TuneMethod = "default",
                     TuneNum = 5,
                     RsmpMethod = "cv",
```

```
                        Folds = 5,
                        Ratio = 0.67,
                        Repeats = 5,
                        VarsImpThr = 0.85,
                        SG_Lrns ="lasso")
```

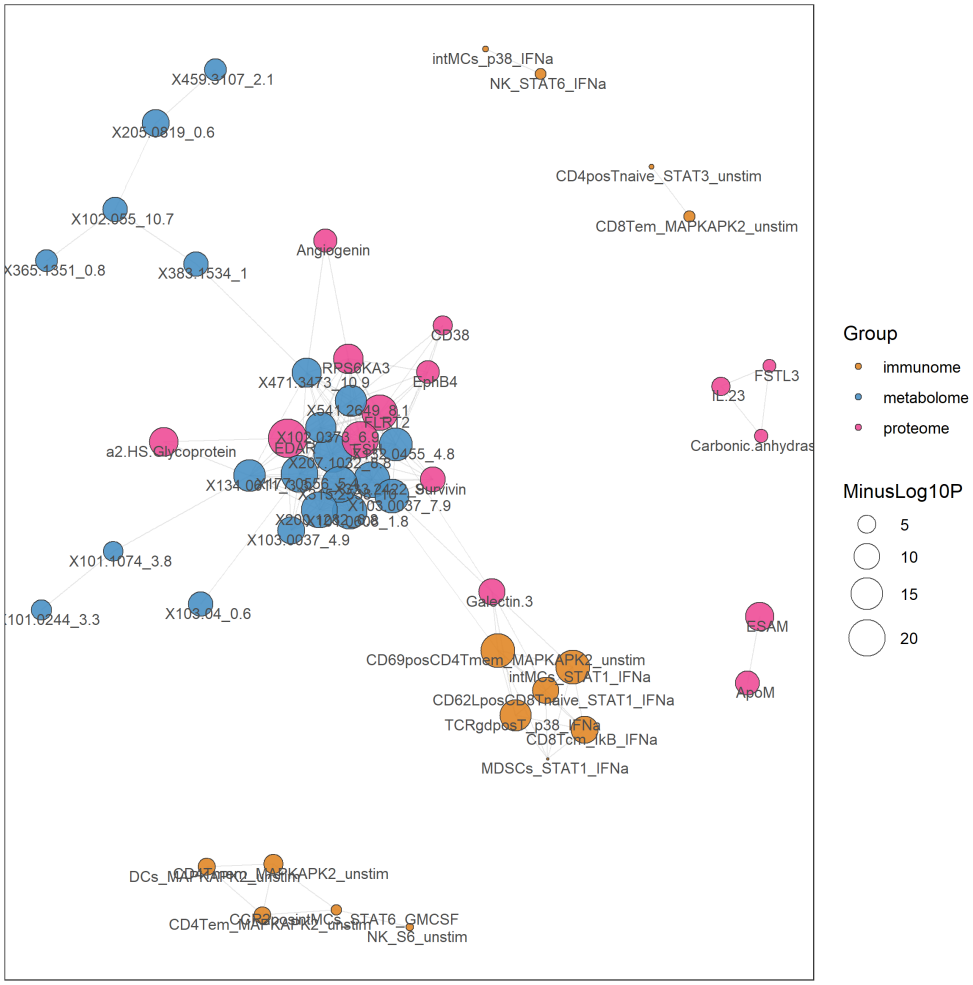
```
knitr::include_graphics("@Figures/Model assessment_boxplot.png")
```



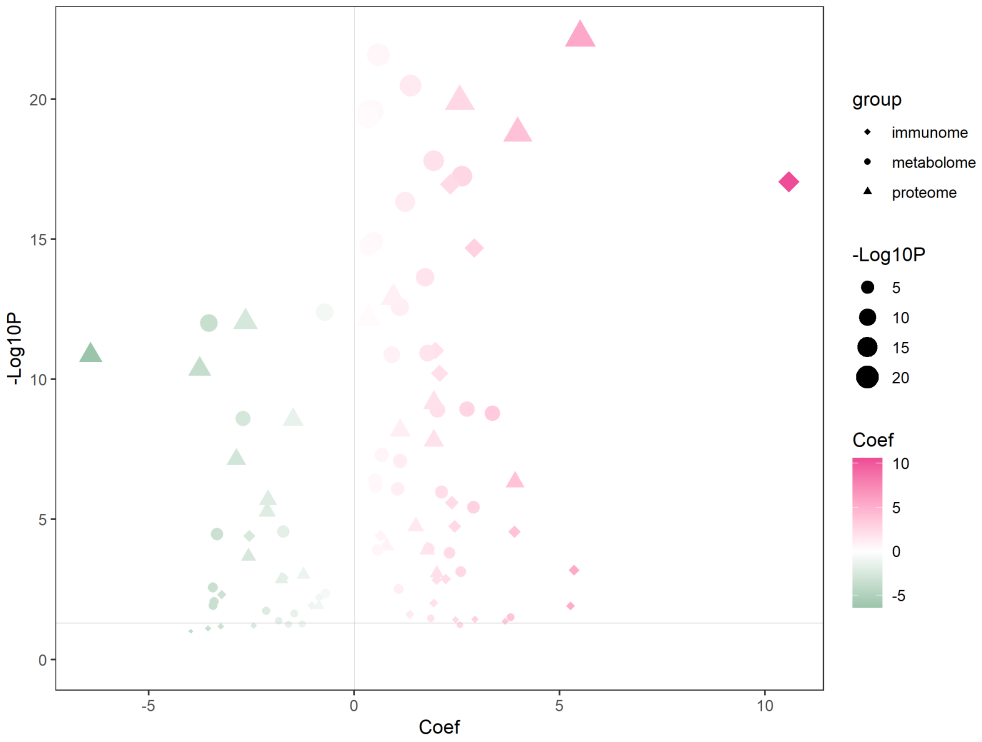
Visualize model This step will visualize multi-omic data with function "VizMulOmicsCros". You can get different styles of images by selecting different parameters.

```
res12 = VizMulOmicCros(PID = res$PID,
                       OutPath = "default",
                       VarsY = "Y1",
                       NodeNum = 100,
                       EdgeThr= 0.45,
                       Layout = "force-directed",
                       Brightness = "light",
                       Palette = 'nejm')
```

```
knitr::include_graphics("@Figures/NetworkPlot_lasso_force-directed_light_nejm.png")
```



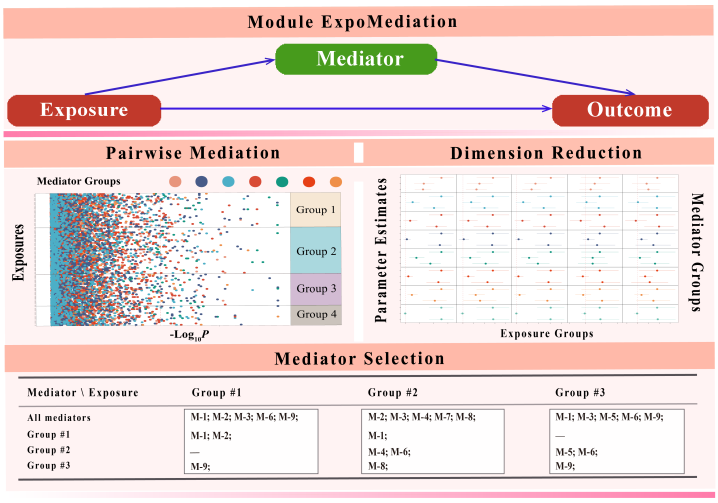

```
knitr::include_graphics("@Figures/NodePlot_lasso_light_nejm.png")
```




```
FuncExit(PID = res$PID)
```

```
## [1] "Success to exit. Thanks for using ExposomeX platform!"
```

8 ExpoMedt module



8.1 Application domain

The **ExpoMedt** module is designed for estimating and screening potential mediation pathways of indigenous biomarkers (i.e., mediator) between external environmental exposure and health outcome in a user friendly and efficient way. Especially, when the number and category of exposures and mediators are of high dimension. Please see the website (http://www.exposomex.cn/#/expomediation) for more information.

8.2 Theory

The module **ExpoMedt** is realized by the package "*exmedt*". The core theory of the *exmedt* package contains two parts: pairwise mediation modelling and exposure/mediator dimension reduction. Pairwise mediation modelling allows the users to estimate the mediation effects of all potential combination of exposure-mediator pairs. For given m exposures and n mediators, an exhaustive rule will be executed and $m * n$ pair-wised modelling are fitted. In most cases or scenarios, the users may deal with datasets with numerous exposures and mediators. Using *exmedt* might be a good choice to efficiently establish batch mediaton modelling.

Dimension reduction for exposure/mediator is also an important module in *exmedt* package. By providing subgroup information for each exposure/mediator, users can conduct dimension reduction for user-specified variables or for all variables automatically (default). In *exmedt* package, various alternative methods are provided for dimension reduction. Users can choose one that may be suitable for their data or try all possible methods to obtain results for sensitive analyses.

8.3 Work pipeline

8.3.1 Initialize package

Before we start using *exmedt*, Make sure that the required packages is already installed.

```
# The following two packages should be installed in advance
# devtools::install_github("ExposomeX/exmedt", force = TRUE)
# devtools::install_github("ExposomeX/extidy", force = TRUE)

# library(exmedt)
# library(extidy)
library(tidyverse)
```

```
# devtools::install_github("ExposomeX/exposomex", force = TRUE)
library(exposomex)
```

At first, you need to initialize the calculation environment using the initialization function, e.g., *InitMedt*.

```
res = InitMedt()
res
```

```
## <eSet>
##   Public:
##     AddCommand: function (x)
##     AddLog: function (x)
##     clone: function (deep = FALSE)
##     DataStr: NULL
##     EpiDesign: NULL
##     ExcecutionLog: Complete initializing the mediation module.
##     2022-12-14  ...
##     Expo: list
##     ExpoDel: list
##     FileDirOut: /home/ubuntu/@changxin/R_Exposome_1.0/output_145226LTXFBP
##     PID: 145226LTXFBP
##     RCommandLog: eSet <- InitExpoData(PID = Any ID your like, FileDirOut  ...
```

Here, we can see that the returned value "res" is an R6 object. It contains an unique program ID of res$PID (e.g., "100737GJMWJA"), which is random generated by the system. Users need use it in the following step for further data process.

8.3.2 Upload data

In *exmedt*, we use the function *LoadMedt* to upload data file as well as vocabulary file for the following ExpoMediation analyses.

```
res1 <- LoadMedt(res$PID,
                 UseExample = "example#1")

res1$Expo$Voca %>%
  dplyr::select(SerialNo:Lod) %>%
  dplyr::slice(1:20) %>%
  knitr::kable(format = "latex",
               align = "l") %>%
  kableExtra::kable_styling(full_width = F,
                            latex_options = "striped",
                            position = "left",
                            font_size = 10)
```

SerialNo	SerialNo_Raw	FullName	GroupName	SubgroupName	DataType	Lod
Y1	Y1	Outcome1	outcome	none	numeric	NA
Y2	Y2	Outcome2	outcome	none	numeric	NA
C1	C1	Race	demography	none	factor	NA
C2	C2	Age	demography	none	numeric	NA
C3	C3	BMI	demography	none	numeric	NA
X1	X1	NAP	chemical	polycyclic_aromatic_hydrocarbons	numeric	NA
X2	X2	ANY	chemical	polycyclic_aromatic_hydrocarbons	numeric	NA
X3	X3	ACE	chemical	polycyclic_aromatic_hydrocarbons	numeric	0.01
X4	X4	FLU	chemical	polycyclic_aromatic_hydrocarbons	numeric	NA
X5	X5	PHE	chemical	polycyclic_aromatic_hydrocarbons	numeric	NA
X6	X6	Pb	chemical	trace_metals	numeric	NA
X7	X7	Hg	chemical	trace_metals	numeric	NA
X8	X8	As	chemical	trace_metals	numeric	NA
X9	X9	Cd	chemical	trace_metals	numeric	NA
X10	X10	Zn	chemical	trace_metals	numeric	NA
M1	M1	medi_1	mediator	Pathway1	numeric	NA
M2	M2	medi_2	mediator	Pathway1	numeric	NA
M3	M3	medi_3	mediator	Pathway1	numeric	NA
M4	M4	medi_4	mediator	Pathway1	numeric	NA
M5	M5	medi_5	mediator	Pathway1	numeric	NA

```
res1$Expo$Data %>%
  dplyr::select(SampleID:C2) %>%
  dplyr::slice(1:20) %>%
  knitr::kable(format = "latex",
               align = "l") %>%
  kableExtra::kable_styling(full_width = F,
                            latex_options = "striped",
                            position = "left",
                            font_size = 10)
```

SampleID	SubjectID	Group	Y1	Y2	C1	C2
1	1	train	36.08893	1	1	30.41476
2	2	train	37.77839	0	1	35.94014
3	3	train	37.12210	0	2	47.89022
4	4	train	38.31941	0	1	35.02661
5	5	train	35.68648	1	2	32.83606
6	6	train	37.50690	0	1	32.34126
7	7	train	39.01246	0	3	36.94808
8	8	train	38.20825	0	1	38.20296
9	9	train	36.19353	1	1	31.94458
10	10	train	38.56214	0	3	25.56944
11	11	train	37.46305	0	1	29.73365
12	12	train	38.78437	0	1	36.90168
13	13	train	36.67348	1	1	34.98749
14	14	train	39.96011	0	1	36.42101
15	15	train	38.81336	0	2	28.54855
16	16	train	39.70540	0	3	33.47661
17	17	train	40.86476	0	1	31.10082
18	18	train	38.72371	0	1	30.72760
19	19	train	39.09666	0	3	31.82517
20	20	train	37.41684	0	3	36.24989

8.3.3 Tidy data

Noted that almost most of the functions below in the tidy module are from the package *extidy*, where the users can access for detailed information.

```
#Imputation for variables that have missing values
res3 = TransImput(PID = res$PID,
                  Group="F",
                  Vars="all.x",
                  Method="lod")

#Transform datatype
res4 = TransType(PID = res$PID,
                 Vars="C1",
                 To="factor")

#Scale
res5 = TransScale(PID = res$PID,
                  Group="F",
                  Vars="all.x",
                  Method="normal")
res6 = TransScale(PID = res$PID,
                 Group="F",
                 Vars="all.m",
                 Method="normal")

#Distribution
res7 = TransDistr(PID = res$PID,
                  Vars="all.x",
                  Method="ln")
res8 = TransDistr(PID = res$PID,
                 Vars="all.m",
                 Method="ln")

#Transform factor variables into dummy variables
res9 = TransDummy(PID = res$PID,
                   Vars="default")

#Divide exposures and mediators into their different subgroups
res10 <- XMlists(PID = res$PID)
res10$ExpoList
```

```
## $polycyclic_aromatic_hydrocarbons
## # A tibble: 161 x 5
##        X1     X2     X3       X4     X5
##     <dbl>  <dbl>  <dbl>    <dbl>  <dbl>
##  1 -0.191 -0.310 -0.300 -0.239   -0.364
##  2 -0.318 -0.290 -0.273 -0.210   -0.217
##  3 -0.267 -0.388 -0.359 -0.303   -0.356
##  4 -0.304 -0.251 -0.232 -0.268   -0.102
##  5 -0.112 -0.326 -0.326 -0.295   -0.418
##  6  0.593  0.817  0.548  0.752    0.287
##  7 -0.298 -0.471 -0.462 -0.372   -0.313
```

```
##  8 -0.352 -0.444 -0.436 -0.338   -0.321
##  9 -0.264 -0.450 -0.360 -0.0239  -0.174
## 10 -0.161 -0.461 -0.294 -0.00260 -0.370
## # ... with 151 more rows
##
## $trace_metals
## # A tibble: 161 x 5
##        X6      X7      X8      X9    X10
##     <dbl>   <dbl>   <dbl>   <dbl>  <dbl>
##  1 -0.707  0.0617 -0.698  -0.114  -0.198
##  2 -0.297 -0.406   0.198  -0.0726 -0.541
##  3 -0.629 -0.206  -0.701  -0.140  -0.510
##  4 -0.420  0.0298  0.341  -0.120  -0.679
##  5 -0.214 -0.433  -0.0128 -0.163  -0.716
##  6  6.29  -0.344  -0.829  -0.0181 -0.422
##  7 -0.111 -0.420  -0.659  -0.115  -0.192
##  8 -0.489 -0.0383  0.278   0.0397  0.785
##  9 -0.655 -0.228   1.17   -0.117  -0.618
## 10 -0.759 -0.530  -0.333  -0.0567 -0.469
## # ... with 151 more rows
```

```
res10$MediList
```

```
## $Pathway1
## # A tibble: 161 x 5
##        M1     M2     M3      M4     M5
##     <dbl>  <dbl>  <dbl>   <dbl>  <dbl>
##  1  3.36  -0.366 -0.201  0.434  -0.378
##  2  2.38   2.04   1.04   1.31    2.14
##  3 -0.460 -0.528 -0.372 -0.112  -0.265
##  4 -0.498 -0.485 -0.634 -0.651  -0.499
##  5  1.61   1.33   2.14   0.218  -0.343
##  6  2.58   2.16   1.41  -0.0330 -0.479
##  7 -0.498 -0.525 -0.389 -0.328  -0.352
##  8 -0.539 -0.405 -0.176 -0.234  -0.441
##  9 -0.535 -0.561 -0.352 -0.478  -0.356
## 10 -0.388 -0.284 -0.705 -0.155  -0.430
## # ... with 151 more rows
##
## $Pathway2
## # A tibble: 161 x 5
##         M6     M7      M8       M9     M10
##      <dbl>  <dbl>   <dbl>    <dbl>   <dbl>
##  1  0.0647 -0.963  1.04    0.549    0.857
##  2  1.03   -1.24   0.589  -0.120   -0.0478
##  3 -0.438  -1.14   0.194   1.06    -0.286
##  4 -0.584  -1.11  -0.334  -0.0634  -0.405
##  5  7.61   -1.03  -0.795  -0.551   -0.238
##  6 -0.0329 -1.25  -0.363  -0.403   -0.548
##  7 -0.536  -0.609 -0.129   0.0613  -0.167
##  8 -0.721  -1.20   0.260  -0.0634  -0.191
##  9 -0.443  -0.792 -0.0408 -0.00676 -0.500
## 10 -0.511  -0.878 -0.436  -0.596    0.452
## # ... with 151 more rows
```

By now, we have prepared our dataset ready for the following mediation modelling.

8.3.4 Modelling

8.3.4.1 Pairwise mediation modelling

First, we proposed a pairwise mediation modelling via the function *Pairwise*, where each given exposure (named as the form of "X*,X*"; i.e., "X1,X2") and mediator (named as the form of "M*,M*"; i.e., "M1,M2") will be selected to constructe mediation model.

In *Pairwise*, several parameters need to be specified before running. The details for each parameter are listed below:

PID:

A character indicating Program ID. It must be the same with the PID generated by *InitMedt*.

VarsY:

A character indicating the outcome variable. The outcome variables should be in the form of "Y*"; i.e., "Y1".Either continuous, binary or count variable is available in function *Pairwise*.

VarsX:

A character indicating the exposure variables. The exposure variables should be in the form of "X*,X*"; i.e., "X1,X2". Users can also specify "default" to include all exposure variables into the function.

VarsM:

A character indicating the mediator variables. The mediator variables should be in the form of "M*,M*"; i.e., "M1,M2". Users can also specify "default" to include all mediator variables into the function.

VarsC:

A character indicating the confounder variables. The confounder variables should be in the form of "C*,C*"; i.e., "C1,C2". Users can also specify "default" to include all confounder variables into the function.

Family:

A character indicating the family of the mediation modellings. Available options include "linear" and "gaussian" for continuous outcome variables, and "binomial" as well as "poisson" options for binary and count outcome variables.

Iter:

A numeric value indicating the number of iteration times. Default is 500. To get a more stable result, a minimum iteration of 1,000 is recommended. To obtain more digits of P value of parameter estimation, a minimum iteration of 5,000 is suggested.

Noted that if the number of Iter, or the number of exposure-mediator pair, is large, the total execution time of the function might be time consumed. Users should be patient for waiting the outputs. The following code generate a set of 40 mediation models (10 exposures * 4 mediators) with 500 iterations for each model. The outcome variable "Y1" is a continuous variable and "linear" family was used.

In this example, we also used function *VizMedtPair* to visualize our pairwise mediation results. Noted that before using function *VizMedtPair*, make sure that the function *Pairwise* have been successfully executed. For details about this function, run "*?exmedt::VizMedtPair*" to see the help mannuals.

```
#1. Pairwise mediation modelling
res11 <- Pairwise(PID = res$PID,
                  VarsY = "Y1",
                  VarsX = "default",
                  VarsM = "M1,M2,M8,M9",
                  VarsC = "C1,C2,C3",
                  Family = "linear",
                  Iter = 500)
res11$MedtPairWise_Stats
```

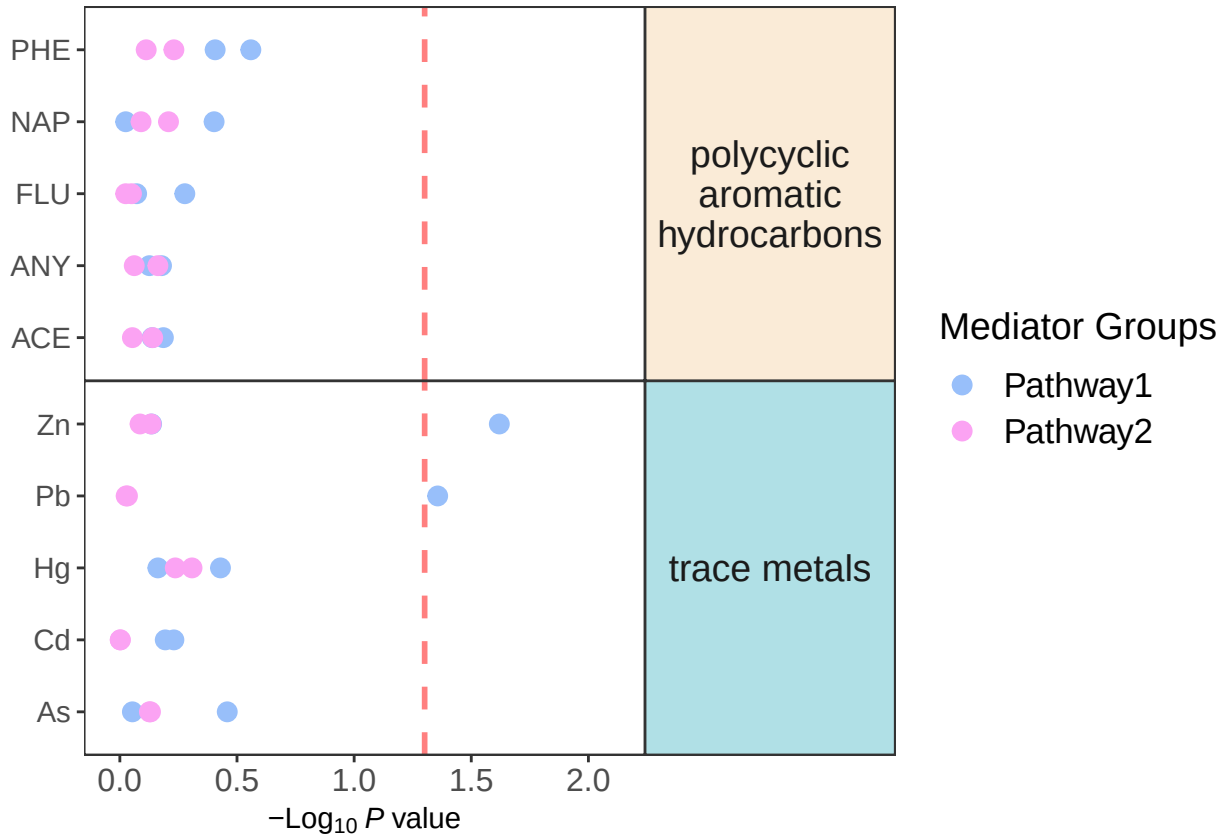
```
## # A tibble: 40 x 17
##    Pairs       TE TE_LCL  TE_UCL TE_Pv~1       IE  IE_LCL  IE_UCL IE_Pv~2     DE
```

```
##    <chr>    <dbl>  <dbl>   <dbl>   <dbl>    <dbl>   <dbl>   <dbl>   <dbl>  <dbl>
##  1 X1 * M1 -0.347 -0.653 -0.0774   0.02   0.00106 -0.0714  0.0724   0.944 -0.348
##  2 X1 * M2 -0.364 -0.667 -0.0674   0.016 -0.0365  -0.124   0.0459   0.396 -0.328
##  3 X1 * M8 -0.349 -0.637 -0.0729   0.024 -0.00896 -0.0554  0.0267   0.62  -0.340
##  4 X1 * M9 -0.350 -0.646 -0.0575   0.016 -0.00372 -0.0353  0.0257   0.812 -0.346
##  5 X10 * ~ -0.382 -0.687 -0.0906   0.008 -0.0127  -0.0846  0.0587   0.732 -0.370
##  6 X10 * ~ -0.371 -0.653 -0.0781   0.016 -0.0925  -0.202  -0.0123   0.024 -0.278
##  7 X10 * ~ -0.382 -0.658 -0.0406   0.024 -0.00769 -0.0635  0.0290   0.736 -0.374
##  8 X10 * ~ -0.378 -0.660 -0.108    0.016  0.00406 -0.0289  0.0409   0.82  -0.382
##  9 X2 * M1 -0.388 -0.645 -0.0907   0.02   0.0154  -0.0576  0.111    0.664 -0.403
## 10 X2 * M2 -0.411 -0.726 -0.137    0.008 -0.0140  -0.108   0.0739   0.748 -0.397
## # ... with 30 more rows, 7 more variables: DE_LCL <dbl>, DE_UCL <dbl>,
## #   DE_Pvalue <dbl>, PctIE <dbl>, PctIE_LCL <dbl>, PctIE_UCL <dbl>,
## #   PctIE_Pvalue <dbl>, and abbreviated variable names 1: TE_Pvalue,
## #   2: IE_Pvalue
```

```
#2. Visualize pairwise mediation modelling result
res12 <- VizMedtPair(PID = res$PID,
                     Brightness = "light",
                     Pallette = "default1")

res12$plotdata%>%
  dplyr::slice(1:10) %>%
  knitr::kable(format = "latex",
               align = "l") %>%
  kableExtra::kable_styling(full_width = F,
                            latex_options = "striped",
                            position = "left",
                            font_size = 10)
```

Mediator	SubgrpM	NameM	Exposure	SubgrpX	NameX	IE_Pvalue
M1	Pathway1	medi_1	X1	polycyclic aromatic hydrocarbons	NAP	0.944
M1	Pathway1	medi_1	X2	polycyclic aromatic hydrocarbons	ANY	0.664
M1	Pathway1	medi_1	X3	polycyclic aromatic hydrocarbons	ACE	0.728
M1	Pathway1	medi_1	X4	polycyclic aromatic hydrocarbons	FLU	0.848
M1	Pathway1	medi_1	X5	polycyclic aromatic hydrocarbons	PHE	0.392
M1	Pathway1	medi_1	X6	trace metals	Pb	0.004
M1	Pathway1	medi_1	X7	trace metals	Hg	0.688
M1	Pathway1	medi_1	X8	trace metals	As	0.884
M1	Pathway1	medi_1	X9	trace metals	Cd	0.588
M1	Pathway1	medi_1	X10	trace metals	Zn	0.732

```
grid::grid.draw(res12$plot[[1]])
```



8.3.4.2 Exposure dimension reduction

In this part, we will implement an example of exposure dimension reduction. Exposure dimension reduction is realized by *RedX* function in *exmedt* function. In *RedX*, exposure variables in the datasets will be shrinkaged into different subgroups that users have defined in vocabulary file of SubGroupName column. For each shinkaged expoure variable, pairwise mediation with given mediators will be fitted at one step.

Same as *Pairwise*, several parameters are required to specify to run *RedX*. Among them, we recommended keeping the parameters of PID, VarsY, VarsC and Family in *RedX* as same as those in *Pairwise*, for better conprehension and explanation of the results. Noted that the parameter "VarsM" in *RedX* is only used for constructing mediation models, that is, the process of exposure dimension reduction in *RedX* is **independent** of "VarsM". Besides, one input parameter is unique in *RedX* function: Method.

Method:

A character indicating the method for exposure dimension reduction. Available options include "mean" and "gcdnet". We recommended choosing "gcdnet" for dimension reduction, inspite of its long time of execution.

```
#3. Exposures dimension reduction
res13 <- RedX(PID = res$PID,
              VarsY = "Y1",
              VarsC = "C1,C2,C3",
              VarsM = "M1,M2,M3",
              Method = "mean",
              Family = "linear",
              Iter = 500)
res13$MedtRedX_ERSall
```

```
## # A tibble: 161 x 1
##    ERS_all
##      <dbl>
##  1  -0.306
##  2  -0.242
##  3  -0.386
##  4  -0.201
##  5  -0.302
##  6   0.768
##  7  -0.341
##  8  -0.132
##  9  -0.172
## 10  -0.344
## # ... with 151 more rows
```

```
res13$MedtRedX_ERSlist
```

```
## # A tibble: 161 x 2
##    polycyclic_aromatic_hydrocarbons trace_metals
##                               <dbl>        <dbl>
##  1                           -0.281      -0.331
##  2                           -0.261      -0.224
##  3                           -0.335      -0.437
##  4                           -0.232      -0.170
##  5                           -0.296      -0.308
##  6                            0.600       0.936
##  7                           -0.383      -0.300
##  8                           -0.378       0.115
##  9                           -0.254      -0.0887
## 10                           -0.258      -0.430
## # ... with 151 more rows
```

```
res13$MedtRedX_Stats
```

```
## # A tibble: 9 x 17
##   Pairs        TE TE_LCL TE_UCL TE_Pv~1       IE  IE_LCL   IE_UCL IE_Pv~2     DE
##   <chr>     <dbl>  <dbl>  <dbl>   <dbl>    <dbl>   <dbl>    <dbl>   <dbl>  <dbl>
## 1 ERS_all~ -0.984 -1.57  -0.428   0     -2.77e-2 -0.187   1.16e-1   0.696 -0.956
## 2 ERS_all~ -0.975 -1.54  -0.398   0     -1.04e-1 -0.311   6.27e-2   0.26  -0.872
## 3 ERS_all~ -0.997 -1.58  -0.329   0.004 -1.27e-1 -0.312   7.99e-4   0.056 -0.870
## 4 polycyc~ -0.551 -0.923 -0.161   0.004  6.32e-4 -0.0909  9.83e-2   0.976 -0.552
## 5 polycyc~ -0.545 -0.927 -0.140   0.008 -4.66e-2 -0.185   7.06e-2   0.412 -0.498
## 6 polycyc~ -0.552 -0.914 -0.133   0.012 -8.31e-2 -0.190  -4.18e-3   0.032 -0.469
## 7 trace_m~ -0.749 -1.32  -0.216   0.004 -5.20e-2 -0.207   8.82e-2   0.44  -0.697
## 8 trace_m~ -0.762 -1.31  -0.193   0.012 -1.20e-1 -0.306   6.47e-2   0.16  -0.642
## 9 trace_m~ -0.752 -1.31  -0.206   0.008 -9.84e-2 -0.266   5.56e-3   0.072 -0.654
## # ... with 7 more variables: DE_LCL <dbl>, DE_UCL <dbl>, DE_Pvalue <dbl>,
## #   PctIE <dbl>, PctIE_LCL <dbl>, PctIE_UCL <dbl>, PctIE_Pvalue <dbl>, and
## #   abbreviated variable names 1: TE_Pvalue, 2: IE_Pvalue
```

8.3.4.3 Mediator dimension reduction

Similar to exposure dimension reduction, *exmedt* also provides dimension reducton for mediators via *RedM* function. In *RedM* function, two methods are available for dimension reduction including "mean" as well as "pdm1"(recommended but time consumed). Likewise, we recommended keeping parameters "VarsY",

"VarsC", "Family" as same as those in former steps.

However, it should be mentioned that when you choose using "pdm1" to execute mediator dimension reduction, the process of mediator dimension reduction is **NOT** independent of exposure variables. That is, the exposure variables specified in "VarsX" take effects in both dimension reduction and pairwise modedlling process in "pdm1" method settings. Therefore, we strongly recommended the users to specify **VarsX="default"** when **Method="pdm1"**.

```
#4. Mediators dimension reduction
res14 <- RedM(PID = res$PID,
              VarsY = "Y1",
              VarsX = "X1,X2,X3",
              VarsC = "C1,C2,C3",
              Method = "mean",
              Family = "linear",
              Iter = 500)
res14$MedtRedM_all
```

```
## # A tibble: 3 x 18
##   Expo  Mediator   TE    TE_LCL TE_UCL TE_Pv~1 IE    IE_LCL IE_UCL IE_Pv~2 DE
##   <chr> <chr>      <chr> <chr>  <chr>  <chr>   <chr> <chr>  <chr>  <chr>   <chr>
## 1 X1    All media~ -0.3~ -0.67~ -0.05~ 0.032   -0.0~ -0.21~ 0.007~ 0.076   -0.2~
## 2 X2    All media~ -0.3~ -0.72~ -0.10~ 0.016   -0.0~ -0.20~ 0.017~ 0.116   -0.3~
## 3 X3    All media~ -0.3~ -0.67~ -0.05~ 0.032   -0.0~ -0.20~ 0.017~ 0.116   -0.2~
## # ... with 7 more variables: DE_LCL <chr>, DE_UCL <chr>, DE_Pvalue <chr>,
## #   PctIE <chr>, PctIE_LCL <chr>, PctIE_UCL <chr>, PctIE_Pvalue <chr>, and
## #   abbreviated variable names 1: TE_Pvalue, 2: IE_Pvalue
```

```
res14$MedtRedM_list
```

```
## # A tibble: 6 x 18
##   Expo  Mediator TE      TE_LCL TE_UCL TE_Pv~1 IE    IE_LCL IE_UCL IE_Pv~2 DE
##   <chr> <chr>    <chr>   <chr>  <chr>  <chr>   <chr> <chr>  <chr>  <chr>   <chr>
## 1 X1    Pathway1 -0.352~ -0.67~ -0.05~ 0.028   -0.0~ -0.14~ 0.061~ 0.452   -0.3~
## 2 X2    Pathway1 -0.396~ -0.71~ -0.10~ 0.016   -0.0~ -0.12~ 0.085~ 0.776   -0.3~
## 3 X3    Pathway1 -0.351~ -0.67~ -0.05~ 0.028   -0.0~ -0.12~ 0.081~ 0.736   -0.3~
## 4 X1    Pathway2 -0.352~ -0.67~ -0.06~ 0.024   -0.0~ -0.16~ -0.00~ 0.024   -0.2~
## 5 X2    Pathway2 -0.395~ -0.71~ -0.10~ 0.016   -0.0~ -0.16~ -0.00~ 0.028   -0.3~
## 6 X3    Pathway2 -0.350~ -0.67~ -0.05~ 0.024   -0.0~ -0.16~ -0.00~ 0.016   -0.2~
## # ... with 7 more variables: DE_LCL <chr>, DE_UCL <chr>, DE_Pvalue <chr>,
## #   PctIE <chr>, PctIE_LCL <chr>, PctIE_UCL <chr>, PctIE_Pvalue <chr>, and
## #   abbreviated variable names 1: TE_Pvalue, 2: IE_Pvalue
```

8.3.4.4 Exposure and mediator dimension reduction

In this part, we use *RedXM* function to realize exposure and mediator dimension reduction. It's almost the same as what in *RedM*. The only difference between *RedM* and *RedXM* is that: in *RedXM*, the parameter VarsX is fixed as all shrinkaged exposure variables. Therefore, before using *RedXM*, make sure that you have executed *RedX* in your work directory.

In this example, we also used function *VizRedXM* to visualize our dimension reduction results. Noted that before using function *VizRedXM*, make sure that the function *RedXM* have been successfully executed. For details about this function, run "*?exmedt::VizRedXM*" to see the help mannuals.

```
#5. Exposure and mediator dimension reduction
res15 <- RedXM(PID = res$PID,
               VarsY = "Y1",
               VarsC = "C1,C2,C3",
```

```
                Method = "mean",
                Family = "linear",
                Iter = 500)
res15$MedtRedXM_all
```

```
## # A tibble: 3 x 18
##   Expo     Media~1 TE    TE_LCL TE_UCL TE_Pv~2 IE    IE_LCL IE_UCL IE_Pv~3 DE
##   <chr>    <chr>   <chr> <chr>  <chr>  <chr>   <chr> <chr>  <chr>  <chr>   <chr>
## 1 ERS_all  All me~ -1.0~ -1.59~ -0.49~ 0       -0.1~ -0.42~ -0.01~ 0.032   -0.8~
## 2 polycyc~ All me~ -0.5~ -0.96~ -0.22~ 0       -0.1~ -0.28~ -0.00~ 0.032   -0.4~
## 3 trace_m~ All me~ -0.8~ -1.40~ -0.26~ 0.008   -0.1~ -0.35~ 0.085~ 0.324   -0.6~
## # ... with 7 more variables: DE_LCL <chr>, DE_UCL <chr>, DE_Pvalue <chr>,
## #   PctIE <chr>, PctIE_LCL <chr>, PctIE_UCL <chr>, PctIE_Pvalue <chr>, and
## #   abbreviated variable names 1: Mediator, 2: TE_Pvalue, 3: IE_Pvalue
```

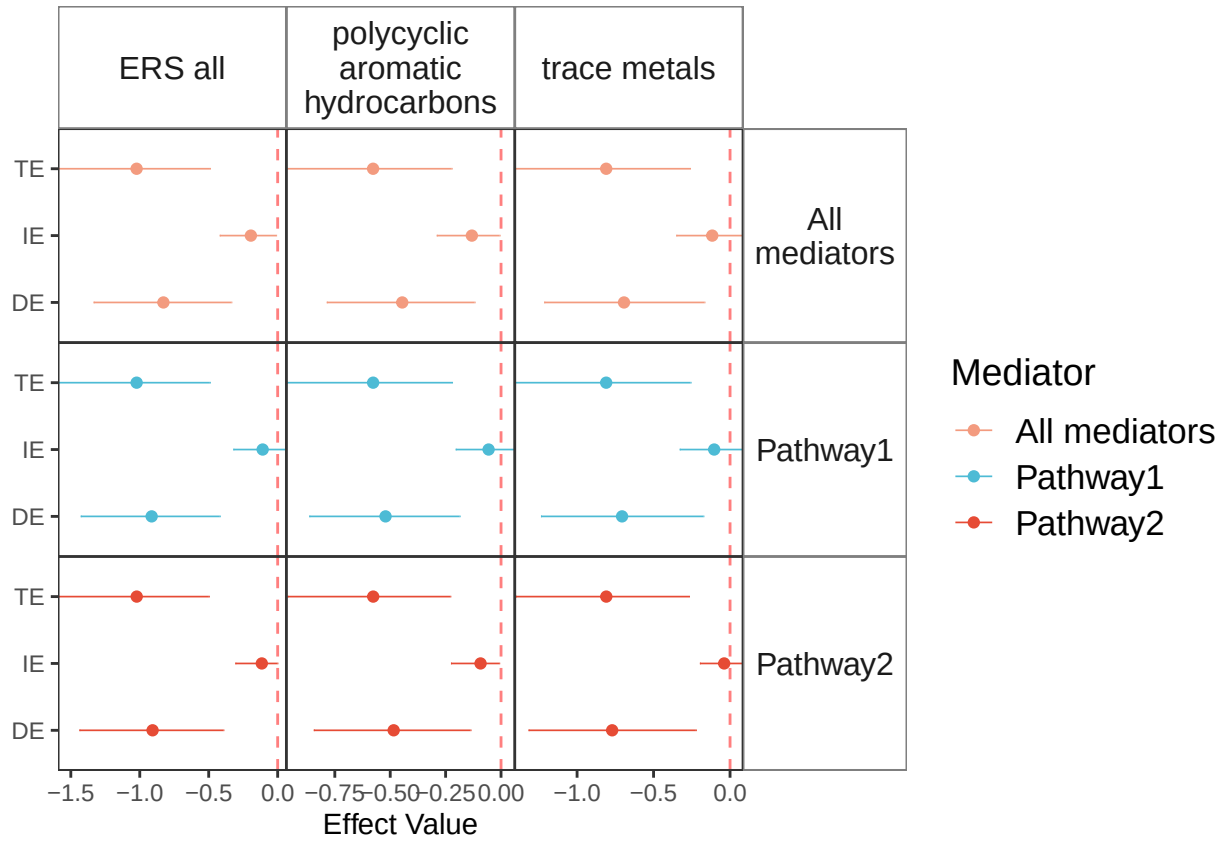
```
res15$MedtRedXM_list
```

```
## # A tibble: 6 x 18
##   Expo     Media~1 TE    TE_LCL TE_UCL TE_Pv~2 IE    IE_LCL IE_UCL IE_Pv~3 DE
##   <chr>    <chr>   <chr> <chr>  <chr>  <chr>   <chr> <chr>  <chr>  <chr>   <chr>
## 1 ERS_all  Pathwa~ -1.0~ -1.59~ -0.49~ 0       -0.1~ -0.32~ 0.063~ 0.28    -0.9~
## 2 polycyc~ Pathwa~ -0.5~ -0.96~ -0.22~ 0       -0.0~ -0.20~ 0.061~ 0.416   -0.5~
## 3 trace_m~ Pathwa~ -0.8~ -1.40~ -0.25~ 0.008   -0.1~ -0.32~ 0.079~ 0.336   -0.7~
## 4 ERS_all  Pathwa~ -1.0~ -1.59~ -0.49~ 0       -0.1~ -0.30~ -0.00~ 0.048   -0.9~
## 5 polycyc~ Pathwa~ -0.5~ -0.96~ -0.22~ 0       -0.0~ -0.22~ -0.00~ 0.04    -0.4~
## 6 trace_m~ Pathwa~ -0.8~ -1.40~ -0.26~ 0.008   -0.0~ -0.19~ 0.083~ 0.584   -0.7~
## # ... with 7 more variables: DE_LCL <chr>, DE_UCL <chr>, DE_Pvalue <chr>,
## #   PctIE <chr>, PctIE_LCL <chr>, PctIE_UCL <chr>, PctIE_Pvalue <chr>, and
## #   abbreviated variable names 1: Mediator, 2: TE_Pvalue, 3: IE_Pvalue
```

```
#6. Visualize MedtRedXM mediation modelling result
res16 <- VizRedXM(PID = res$PID,
                  Brightness = "light",
                  Pallette = "nature")

res16$MedtRedXM_plotadta %>%
  dplyr::slice(1:10) %>%
  knitr::kable(format = "latex",
               align = "l") %>%
  kableExtra::kable_styling(full_width = F,
                            latex_options = "striped",
                            position = "left",
                            font_size = 10)
```

Expo	Mediator	Effect	Effect_Value	LowerCL	UpperCL
ERS all	Pathway1	IE	-0.1087993	-0.3213227	0.0634497
ERS all	Pathway1	DE	-0.9137781	-1.4278846	-0.4197218
ERS all	Pathway1	TE	-1.0225774	-1.5911547	-0.4917670
polycyclic aromatic hydrocarbons	Pathway1	IE	-0.0560722	-0.2044135	0.0618910
polycyclic aromatic hydrocarbons	Pathway1	DE	-0.5210358	-0.8650161	-0.1865901
polycyclic aromatic hydrocarbons	Pathway1	TE	-0.5771080	-0.9663433	-0.2211435
trace metals	Pathway1	IE	-0.1036513	-0.3290675	0.0795817
trace metals	Pathway1	DE	-0.7060029	-1.2359356	-0.1742808
trace metals	Pathway1	TE	-0.8096542	-1.4056128	-0.2578012
ERS all	Pathway2	IE	-0.1150173	-0.3068455	-0.0017912

```
res16$MedtRedXM_plot
```



8.3.4.5 Estimation the importace of mediators

ImptM function estimates the importance of mediators among given exposures. The estimation procedure is mainly realized by the hima function in HIMA package. *ImptM* also provides another evaluation method by the bama function in bama package. The users can search for these packages for further information.

For better comprehension and explanation of the result, we recommend that the parameter "VarsY" and "VarsC" unchanged. *ImptM* also supports shrinkaged exposure variables in VarsX, but *RedX* should be executed before including them into *ImptM*.

```
#7. Estimation the importance of mediators
res17  <- ImptM(PID = res$PID,
                VarsY = "Y1",
                VarsX = "X1,X2,X3",
                VarsC = "C1")

res17$MedtImptM_all %>%
  dplyr::select(exposure:pip,Bonferroni.p,BH.FDR) %>%
  dplyr::slice(1:10) %>%
  knitr::kable(format = "latex",
               align = "l") %>%
  kableExtra::kable_styling(full_width = F,
                            latex_options = "striped",
                            position = "left",
                            font_size = 10)
```

exposure	mediator	mediator_group	pip	Bonferroni.p	BH.FDR
X1	M1	All mediators	0.000	NA	NA
X1	M2	All mediators	0.002	1	0.9146337
X1	M3	All mediators	0.028	NA	NA
X1	M4	All mediators	0.002	1	0.9146337
X1	M5	All mediators	0.004	NA	NA
X1	M6	All mediators	0.010	1	0.9146337
X1	M7	All mediators	0.002	1	0.9146337
X1	M8	All mediators	0.006	NA	NA
X1	M9	All mediators	0.002	NA	NA
X1	M10	All mediators	0.134	NA	NA

```
res17$MedtImptM_list %>%
  dplyr::select(exposure:pip,Bonferroni.p,BH.FDR) %>%
  dplyr::slice(1:10) %>%
  knitr::kable(format = "latex",
               align = "l") %>%
  kableExtra::kable_styling(full_width = F,
                            latex_options = "striped",
                            position = "left",
                            font_size = 10)
```

exposure	mediator	mediator_group	pip	Bonferroni.p	BH.FDR
X1	M1	Pathway1	0.026	NA	NA
X1	M2	Pathway1	0.022	0.9815088	0.8977833
X1	M3	Pathway1	0.150	NA	NA
X1	M4	Pathway1	0.016	1.0000000	0.8977833
X1	M5	Pathway1	0.024	NA	NA
X2	M1	Pathway1	0.026	NA	NA
X2	M2	Pathway1	0.032	1.0000000	0.8044372
X2	M3	Pathway1	0.136	NA	NA
X2	M4	Pathway1	0.020	1.0000000	0.8044372
X2	M5	Pathway1	0.056	NA	NA

```
res17$MedtImptM_table1 %>%
  knitr::kable(format = "latex",
               align = "l") %>%
  kableExtra::kable_styling(full_width = F,
                            latex_options = "striped",
                            position = "left",
                            font_size = 10)
```

mediator_group	X1	X2	X3
All mediators	M2,M4,M6,M7	M2,M4,M6,M7	M2,M4,M6,M7
Pathway1	M2,M4	M2,M4	M2,M4
Pathway2	M6	M6	M6

```
res17$MedtImptM_table2 %>%
  knitr::kable(format = "latex",
               align = "l") %>%
  kableExtra::kable_styling(full_width = F,
                            latex_options = "striped",
                            position = "left",
                            font_size = 10)
```

mediator_group	X1	X2	X3
All mediators	M2,M4,M6,M7	M2,M4,M6,M7	M2,M4,M6,M7
Pathway1	M2,M4	M2,M4	M2,M4
Pathway2	M6	M6	M6

8.3.4.6 Exit exposome mediation analyses

When finishing your analyses task and saving all the results needed (**That is vital for the users.**), remenber run the exit function provided in that package, which will release the memory of R environment and is of helpful for both the users and the exposome platform.

```
FuncExit(PID = res$PID)
```

```
## [1] "Success to exit. Thanks for using ExposomeX platform!"
```

9 ExpoMixEffect module

9.1 Application domain

ExpoMixEffect module is designed to analyze mixture effect of the various exposure factors. It mainly aims to screen the representative features with high contribution to the health outcome, as well as their potential interaction effect.

9.2 Theory

9.3 Work pipeline

Users can easily get the modeling results and their visualization plots with high quality by following the detailed instructions in each step. Three frequently used mixture-effect models are provided including multiple linear regression with regulation algorithms (e.g., stepwise, LASSO and elastic network), weighted quantile sum regression and Bayesian kernel machine regression.

```
# The following two packages should be installed in advance
# devtools::install_github("ExposomeX/exmix", force = TRUE)
# devtools::install_github("ExposomeX/extidy", force = TRUE)

# library(exmix)
# library(extidy)
library(tidyverse)

# devtools::install_github("ExposomeX/exposomex", force = TRUE)
library(exposomex)
```

```
res = InitMix()

res1 = LoadMix(PID = res$PID,
               UseExample = "example#1")

res2 = FindCovaMix(PID = res$PID,
                   OutPath= "default",
                   VarsY = "Y1",
                   VarsC_Prior = "default",
                   VarsC_Fixed = "C5,C6",
                   Method = "single.factor",
                   Thr = 0.1)
res2$MixCovariables
```

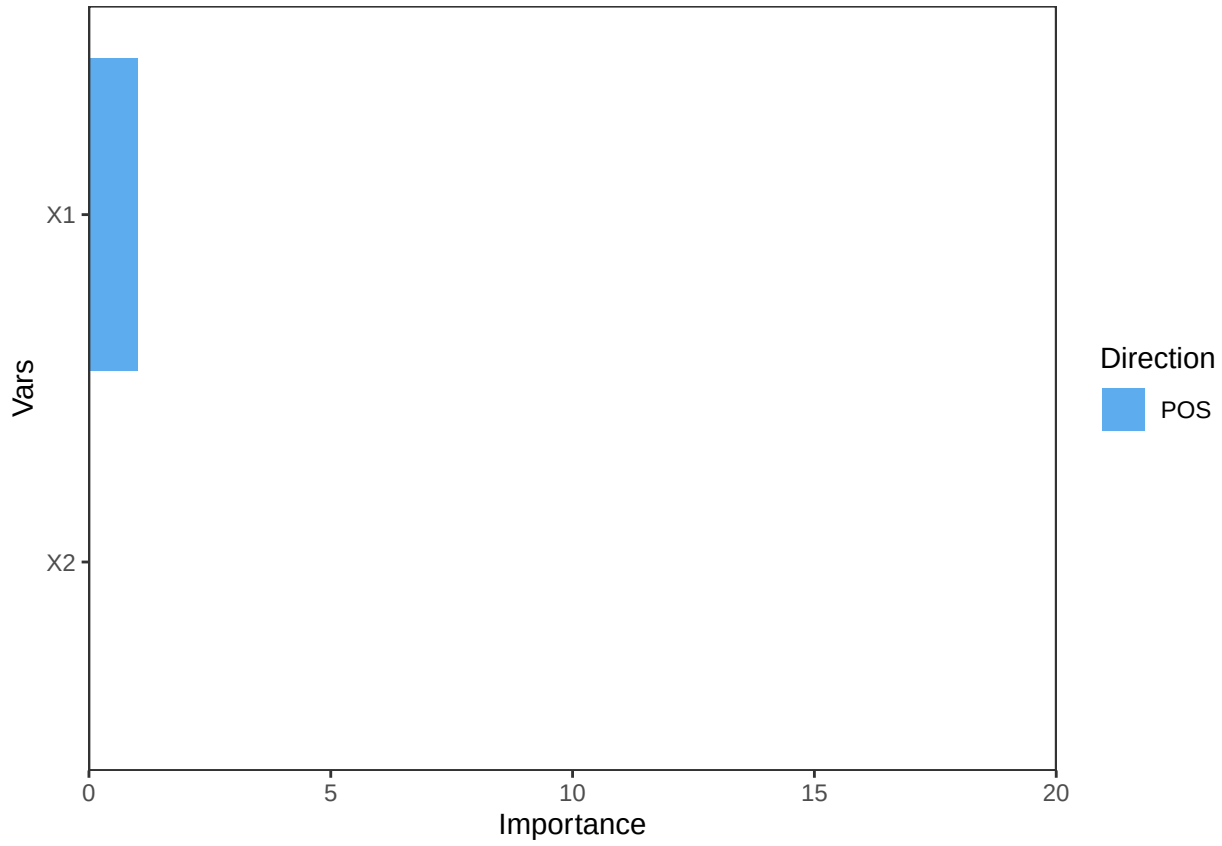
```
## [1] "C4" "C5" "C6"
```

```
res3 = MixMLR(PID = res$PID,
              OutPath= "default",
              VarsY = "Y1",
              VarsX = "all.x",
              IncCova = F,
              SelMethod = "lasso" ,
              PredType = "response" ,
              Family = "gaussian")
res3$Y1_lasso_vip
```

```
## # A tibble: 2 x 3
##   Variable Importance Sign
##   <chr>         <dbl> <chr>
```

```
## 1 X1           1.00   POS
## 2 X2           0.0209 POS
```

```
res4 = VizMixMLR(PID=res$PID,
                 OutPath= "default",
                 VarsY = "Y1",
                 SelMethod = 'lasso',
                 Brightness = "light",
                 Palette = "default1")
res4$Y1_lasso_importance_light_default1
```

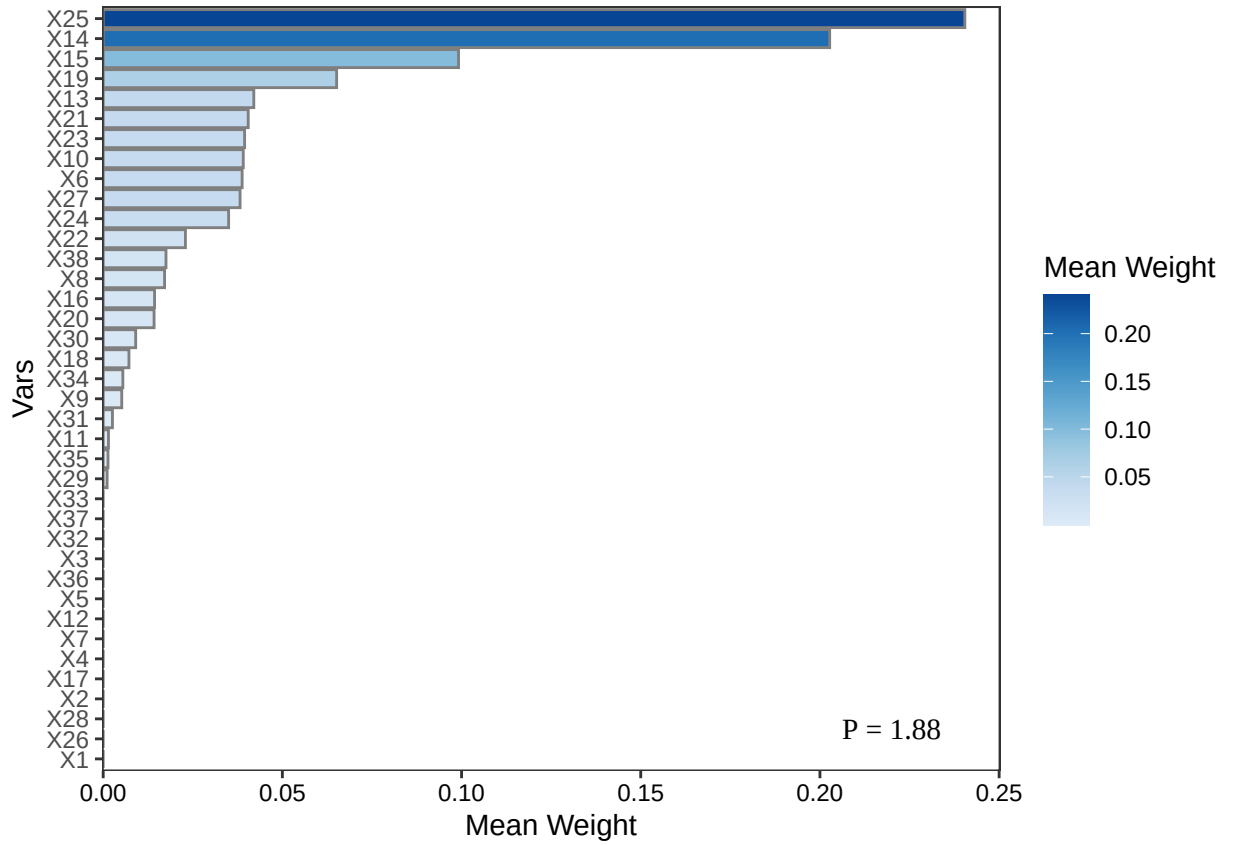



```
res5 = MixWQS(PID=res$PID,
              OutPath= "default",
              VarsY = "Y1",
              VarsX = "all.x" ,
              IncCova = F,
              Family = "gaussian" ,
              VarStrat = "none",
              RatioValidat = 0.3,
              q = 10,
              b=100,
              b1_pos = F,
              b1_constr = F)
res5$Y1_gaussian_Imp
```

```
##     mix_name  mean_weight
## X25      X25 2.404228e-01
```

```
## X14      X14 2.027116e-01
## X15      X15 9.916052e-02
## X19      X19 6.513869e-02
## X13      X13 4.202749e-02
## X21      X21 4.046160e-02
## X23      X23 3.945876e-02
## X10      X10 3.908749e-02
## X6        X6 3.876949e-02
## X27      X27 3.819521e-02
## X24      X24 3.500169e-02
## X22      X22 2.294915e-02
## X38      X38 1.752412e-02
## X8        X8 1.712935e-02
## X16      X16 1.434207e-02
## X20      X20 1.419808e-02
## X30      X30 9.069463e-03
## X18      X18 7.190387e-03
## X34      X34 5.481655e-03
## X9        X9 5.182469e-03
## X31      X31 2.585260e-03
## X11      X11 1.476624e-03
## X35      X35 1.341644e-03
## X29      X29 1.094139e-03
## X33      X33 4.314870e-08
## X37      X37 3.205410e-08
## X32      X32 3.190042e-08
## X3        X3 3.182883e-08
## X36      X36 3.145175e-08
## X5        X5 2.473607e-08
## X12      X12 2.222508e-08
## X7        X7 1.959676e-08
## X4        X4 7.793889e-09
## X17      X17 4.956954e-09
## X2        X2 4.868430e-09
## X28      X28 3.122512e-09
## X26      X26 1.322164e-09
## X1        X1 1.560702e-10
```

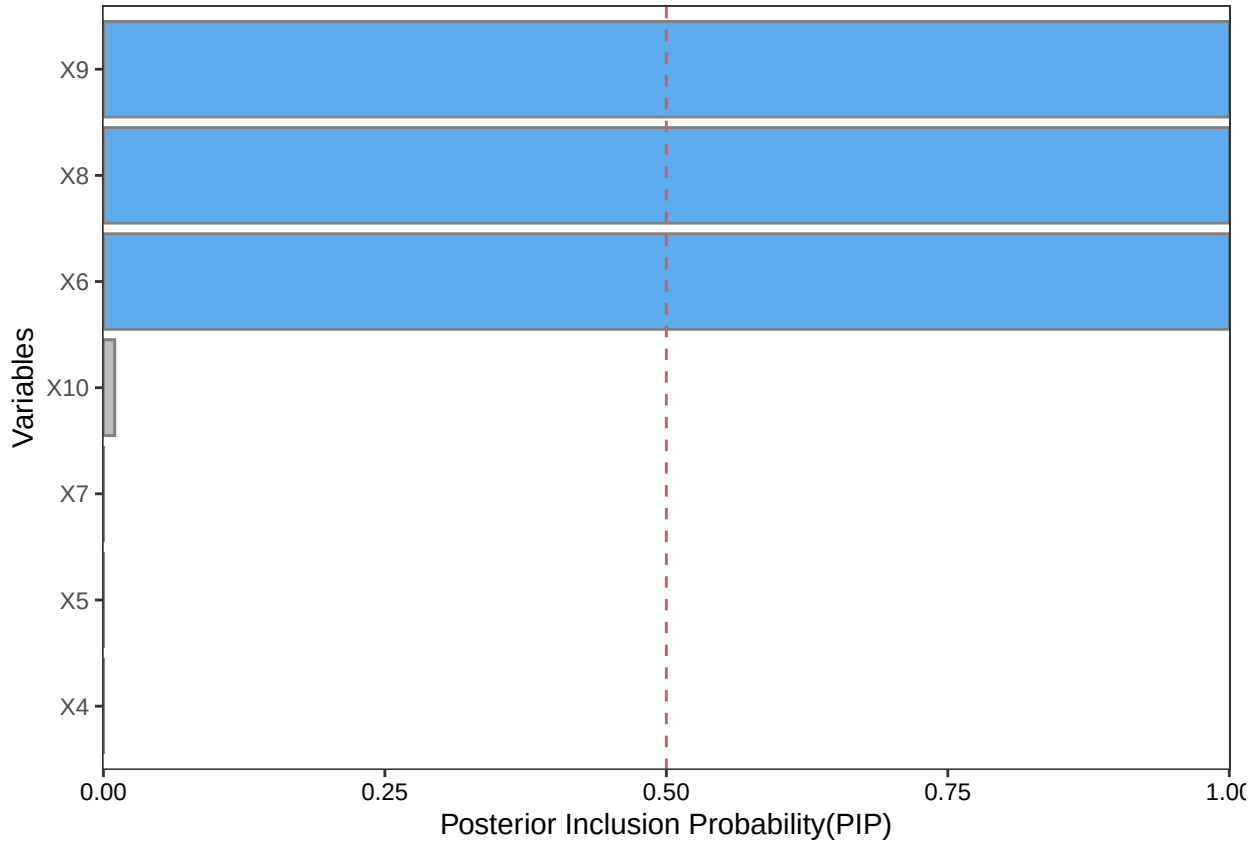
```
res6 = VizMixWQS(PID = res$PID,
                 OutPath= "default",
                 VarsY = "Y1",
                 Brightness = "dark",
                 Palette = "default1")
res6$Y1_gaussian_Imp_dark_default1
```




```
res7 = MixBKMR(PID = res$PID,
               OutPath= "default",
               VarsY = "Y1",
               VarsX = "X4,X5,X6,X7,X8,X9,X10",
               IncCova = F,
               Family = "gaussian" ,
               Group = F,
               Iter = 2000,
               qfixed = 0.5,
               qsbivar = "default" ,
               qsoverall = "default",
               qsdiff = "default")
res7$Y1_risks.overall
```

```
##    quantile        est       sd
## 1      0.25 -19.342570 49.27176
## 2      0.30 -12.972066 47.47622
## 3      0.35 -13.670789 48.16824
## 4      0.40  -2.537996 48.37987
## 5      0.45 -20.554091 70.10191
## 6      0.50   0.000000  0.00000
## 7      0.55 -10.558709 53.27296
## 8      0.60 -11.722467 51.65752
## 9      0.65 -25.440104 76.20959
## 10     0.70  -4.165882 78.07037
## 11     0.75 -15.774105 82.97824
```

```
res8 = VizMixBKMR(PID=res$PID,
                  OutPath= "default",
                  VarsY = "Y1",
                  Brightness = "dark",
                  Palette = "default1")
res8$Y1_pip.univar_dark_default1
```




```
FuncExit(PID = res$PID)
```

```
## [1] "Success to exit. Thanks for using ExposomeX platform!"
```

10 ExpoNTA module

10.1 Application domain

ExpoNontarget module is designed to conduct the analysis of the features from the high-resolution mass spectrometry. It mainly aims to screen and annotate the significant features associated with the health outcomes.

10.2 Theory

10.3 Work pipeline

Users can easily get the modeling results and their visualization plots with high quality by following the detailed instructions in each step. Three methods are adopted to screen the important features, including "Stepwise" "LASSO", and "Random forest". At present, the annotation is only applied for the MS1 features acquired from high-resolution liquid chromatography–mass spectrometry.

```
# The following two packages should be installed in advance
# devtools::install_github("ExposomeX/exnta", force = TRUE)
# devtools::install_github("ExposomeX/extidy", force = TRUE)

# library(exnta)
# library(extidy)
library(tidyverse)

# devtools::install_github("ExposomeX/exposomex", force = TRUE)
library(exposomex)
```

```
res = InitNTA()

res1 = LoadNTA(PID = res$PID,
               UseExample = "example#1")

res1$Expo$Data
```

```
## # A tibble: 150 x 206
##    SampleID SubjectID    Y1    Y2    C1    C2     X1     X2     X3     X4     X5
##    <chr>        <dbl> <dbl> <dbl> <dbl> <dbl>  <dbl>  <dbl>  <dbl>  <dbl>  <dbl>
##  1 001_26_A         1  -101     0    35     7 8.12e3 3.62e4 6.12e4 1.56e5 4.38e4
##  2 001_33_B         1   -51     0    51   779 9.57e3 2.61e4 5.90e4 2.21e5 6.41e4
##  3 001_35_C         1   -37     1    30   613 7.95e3 3.06e4 1.09e5 1.87e5 5.31e4
##  4 003_25_A         3   -61     0    50   454 1.12e5 3.57e5 6.15e4 9.91e4 8.96e4
##  5 003_30_B         3   -28     1    39   766 1.16e5 4.60e5 9.56e4 3.85e5 9.88e4
##  6 003_32_C         3    -8     1    76   768 1.46e4 6.24e4 3.15e4 2.10e5 1.12e5
##  7 005_25_A         5   -63     0    27   418 2.69e4 7.84e4 5.33e4 1.80e5 9.93e4
##  8 005_29_B         5   -35     1    50   484 8.48e3 3.24e4 6.11e4 6.50e4 8.28e4
##  9 005_32_C         5   -14     1    16   647 7.92e3 3.92e4 7.10e4 1.75e5 8.92e4
## 10 006_25_A         6   -99     0    49   329 1.51e4 4.80e4 3.67e4 1.99e5 5.22e4
## # ... with 140 more rows, and 195 more variables: X6 <dbl>, X7 <dbl>, X8 <dbl>,
## #   X9 <dbl>, X10 <dbl>, X11 <dbl>, X12 <dbl>, X13 <dbl>, X14 <dbl>, X15 <dbl>,
## #   X16 <dbl>, X17 <dbl>, X18 <dbl>, X19 <dbl>, X20 <dbl>, X21 <dbl>,
## #   X22 <dbl>, X23 <dbl>, X24 <dbl>, X25 <dbl>, X26 <dbl>, X27 <dbl>,
## #   X28 <dbl>, X29 <dbl>, X30 <dbl>, X31 <dbl>, X32 <dbl>, X33 <dbl>,
## #   X34 <dbl>, X35 <dbl>, X36 <dbl>, X37 <dbl>, X38 <dbl>, X39 <dbl>,
## #   X40 <dbl>, X41 <dbl>, X42 <dbl>, X43 <dbl>, X44 <dbl>, X45 <dbl>, ...
```

```
res2 = DelNearZeroVar(OutPath = "default",
                      PID = res$PID)

res3 = DelMiss(PID = res$PID)

res4 = TransType(PID=res$PID,
                 OutPath = "default",
                 Vars="Y2",
                 To="factor")

res5 = TransScale(PID=res$PID,
                  OutPath = "default",
                  Group= F,
                  Vars="all.x",
                  Method="normal")
res5$Expo$Data
```

```
## # A tibble: 150 x 206
##    SampleID Subje~1    Y1 Y2       C1    C2     X1     X2     X3       X4     X5
##    <chr>      <dbl> <dbl> <fct> <dbl> <dbl>  <dbl>  <dbl>  <dbl>    <dbl>  <dbl>
##  1 001_26_A       1  -101 0        35     7 -0.655 -0.577 -0.312 -0.258   -1.28
##  2 001_33_B       1   -51 0        51   779 -0.643 -0.600 -0.320  0.114   -0.560
##  3 001_35_C       1   -37 1        30   613 -0.657 -0.590 -0.135 -0.0815  -0.949
##  4 003_25_A       3   -61 0        50   454  0.240  0.155 -0.311 -0.590    0.346
##  5 003_30_B       3   -28 1        39   766  0.275  0.390 -0.184  1.07     0.669
##  6 003_32_C       3    -8 1        76   768 -0.600 -0.517 -0.422  0.0549   1.13
##  7 005_25_A       5   -63 0        27   418 -0.493 -0.480 -0.341 -0.120    0.689
##  8 005_29_B       5   -35 1        50   484 -0.652 -0.586 -0.312 -0.787    0.103
##  9 005_32_C       5   -14 1        16   647 -0.657 -0.570 -0.275 -0.151    0.329
## 10 006_25_A       6   -99 0        49   329 -0.595 -0.550 -0.403 -0.00932 -0.982
## # ... with 140 more rows, 195 more variables: X6 <dbl>, X7 <dbl>, X8 <dbl>,
## #   X9 <dbl>, X10 <dbl>, X11 <dbl>, X12 <dbl>, X13 <dbl>, X14 <dbl>, X15 <dbl>,
## #   X16 <dbl>, X17 <dbl>, X18 <dbl>, X19 <dbl>, X20 <dbl>, X21 <dbl>,
## #   X22 <dbl>, X23 <dbl>, X24 <dbl>, X25 <dbl>, X26 <dbl>, X27 <dbl>,
## #   X28 <dbl>, X29 <dbl>, X30 <dbl>, X31 <dbl>, X32 <dbl>, X33 <dbl>,
## #   X34 <dbl>, X35 <dbl>, X36 <dbl>, X37 <dbl>, X38 <dbl>, X39 <dbl>,
## #   X40 <dbl>, X41 <dbl>, X42 <dbl>, X43 <dbl>, X44 <dbl>, X45 <dbl>, ...
```

```
res6 = FindCovaNta(PID=res$PID,
                   OutPath = "default",
                   VarsY = "Y1",
                   VarsC_Prior = "default",
                   VarsC_Fixed = "C2",
                   Method = "single.factor",
                   Thr = 0.1)
res6$Covariables
```

```
## [1] "C2"
```

```
res7 = NtaCros(PID=res$PID,
               OutPath = "default",
               VarsY = "Y1",
               VarsX = "all.x",
               VarsN = "single.factor",
               FdrCorrect = "T",
```

```
               SelMethod = "lasso",
               StepwizeThr = 0.1,
               RF_ImpThr = 0.9,
               IncCova = F,
               Family = "gaussian",
               RepMsr = F,
               Corstr = "ar1")
res7$Y1_single.factor
```

```
## # A tibble: 15 x 11
##    SerialNo Vars.dummy Importance FullN~1 Exact~2 beta.~3   ci_l   ci_h  p.value
##    <chr>    <chr>           <dbl> <chr>     <dbl>   <dbl>  <dbl>  <dbl>    <dbl>
##  1 X196     X196            185.  X132.0~    132.  -16.8  -21.2  -12.3  9.73e-12
##  2 X20      X20              82.0 X103.0~    103.  -12.9  -17.6   -8.10 4.29e- 7
##  3 X16      X16              78.3 X102.0~    102.   12.7    7.89  17.5  6.65e- 7
##  4 X163     X163             65.0 X130.0~    130.   11.9    7.06  16.7  3.46e- 6
##  5 X164     X164             56.8 X130.0~    130.   11.4    6.49  16.2  1.00e- 5
##  6 X135     X135             50.8 X128.0~    128.   10.9    6.04  15.8  2.25e- 5
##  7 X90      X90              49.4 X119.0~    119.   10.8    5.92  15.7  2.73e- 5
##  8 X173     X173             42.4 X130.0~    130.   10.3    5.33  15.2  7.33e- 5
##  9 X17      X17              41.5 X102.0~    102.   10.2    5.25  15.1  8.41e- 5
## 10 X104     X104             40.9 X121.0~    121.  -10.1  -15.1   -5.19 9.17e- 5
## 11 X56      X56              40.8 X114.0~    114.   10.1    5.19  15.1  9.24e- 5
## 12 X161     X161             36.2 X130.0~    130.    9.70   4.75  14.7  1.85e- 4
## 13 X98      X98              35.9 X120.0~    120.    9.68   4.72  14.6  1.93e- 4
## 14 X176     X176             34.9 X130.9~    131.    9.58   4.61  14.5  2.25e- 4
## 15 X191     X191             34.5 X131.1~    131.    9.54   4.57  14.5  2.41e- 4
## # ... with 2 more variables: std.error <dbl>, formula <chr>, and abbreviated
## #   variable names 1: FullName, 2: ExactMass, 3: beta.value
```

```
res8 = VizNtaCros(PID=res$PID,
                  OutPath = "default",
                  VarsY = "Y1",
                  VarsN = "single.factor",
                  Layout = "forest",
                  Brightness = "light",
                  Palette = "default1")
res8$Y1_single.factor_forest_light_default1
```

```
## [[1]]
## TableGrob (1 x 1) "arrange": 1 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
##
## attr(,"class")
## [1] "arrangelist" "list"
```

```
res9 = NtaCros(PID=res$PID,
               OutPath = "default",
               VarsY = "Y2",
               VarsX = "all.x",
               VarsN = "multiple.factor",
               FdrCorrect = "T",
               SelMethod = "lasso",
               StepwizeThr = 0.1,
```

```
               RF_ImpThr = 0.9,
               IncCova = F,
               Family = "binomial",
               RepMsr = F,
               Corstr = "ar1")

res10 = VizNtaCros(PID=res$PID,
                   OutPath = "default",
                   VarsY = "Y2",
                   VarsN = "multiple.factor",
                   Layout = "forest",
                   Brightness = "light",
                   Palette = "default1")
res10$Y2_multiple.factor_forest_light_default1
```

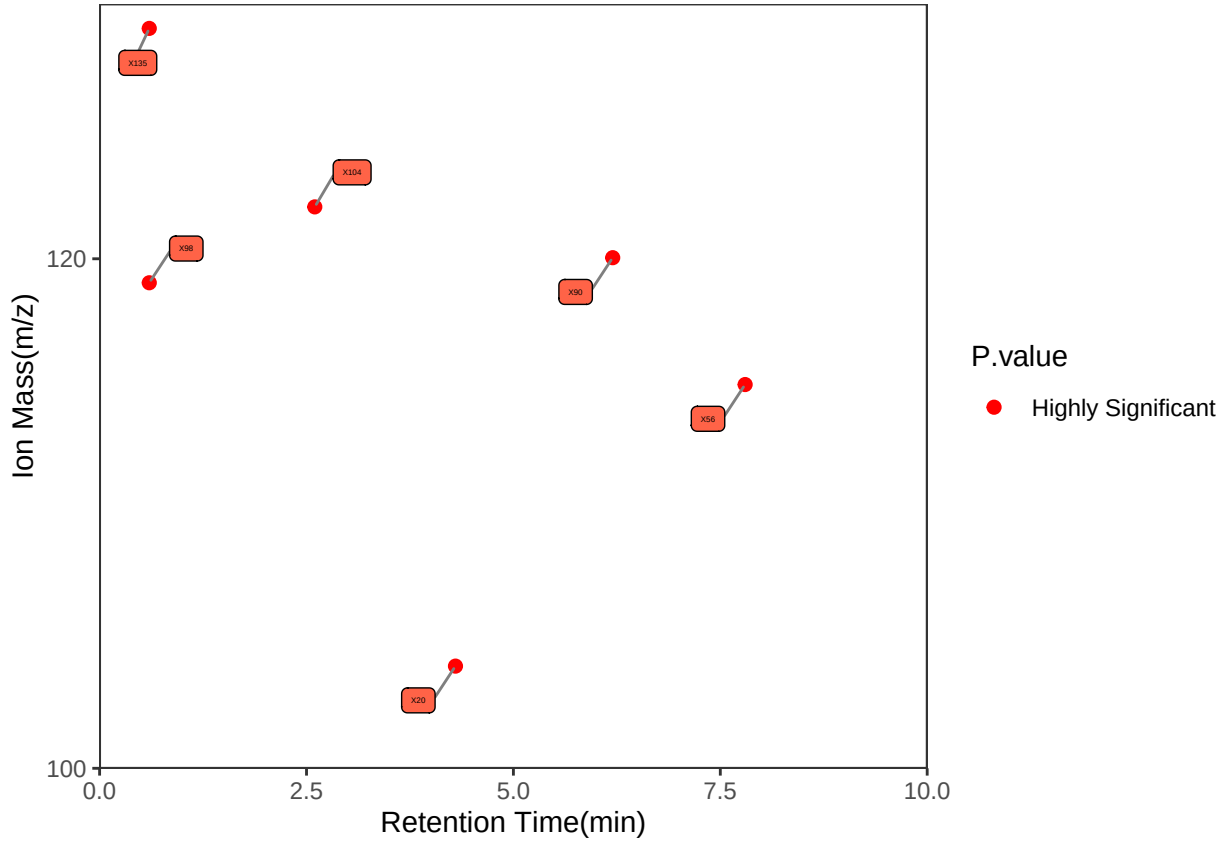
```
## [[1]]
## TableGrob (1 x 1) "arrange": 1 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
##
## attr(,"class")
## [1] "arrangelist" "list"
```

```
res11 = NtaAnno(PID=res$PID,
                OutPath = "default",
                VarsY = "Y1",
                VarsX = "default",
                VarsN = "single.factor",
                FdrCorrect = F,
                AdductPos = "M+H",
                AdductNeg = "M-H",
                Accuracy = 1)
res11$NtaAnno_Y1_single.factor_1
```

```
## # A tibble: 218 x 9
##    SerialNo    RT ExactMass Name             SMILES Monoi~1 IonMode Adduct Group
##    <chr>    <dbl>     <dbl> <chr>            <chr>    <dbl> <chr>   <chr>  <chr>
##  1 X98        0.6      120. 2-Methyl-2-nitr~ CC(C)~    119. positi~ M+H    pare~
##  2 X98        0.6      120. L-Threonine      C[C@@~    119. positi~ M+H    pare~
##  3 X98        0.6      120. 2-Nitro-1-butan~ CCC(C~    119. positi~ M+H    pare~
##  4 X98        0.6      120. 4-Amino-3-hydro~ NCC(O~    119. positi~ M+H    pare~
##  5 X98        0.6      120. N-Butylnitrite   CC(CC~    119. positi~ M+H    prec~
##  6 X98        0.6      120. Dibutyl sulfide  CC(CC~    119. positi~ M+H    prec~
##  7 X98        0.6      120. Isobutyl nitrite CC(C)~    119. positi~ M+H    prec~
##  8 X98        0.6      120. 2-Heptanol       CC(C)~    119. positi~ M+H    prec~
##  9 X98        0.6      120. Dichlorvos       CC(CC~    119. positi~ M+H    prec~
## 10 X98        0.6      120. Ethyl N-methylc~ CN=C(~    119. positi~ M+H    prec~
## # ... with 208 more rows, and abbreviated variable name 1: Monoisotopic_Mass
```

```
res12 = VizNtaAnno(PID=res$PID,
                   OutPath = "default",
                   VarsY = "Y1",
                   VarsN = "single.factor",
                   Accuracy = 1,
                   Brightness = "light",
```

```
                    Palette = "default1")
res12$VizNtaAnno_Y1_single.factor_1_light_default1
```

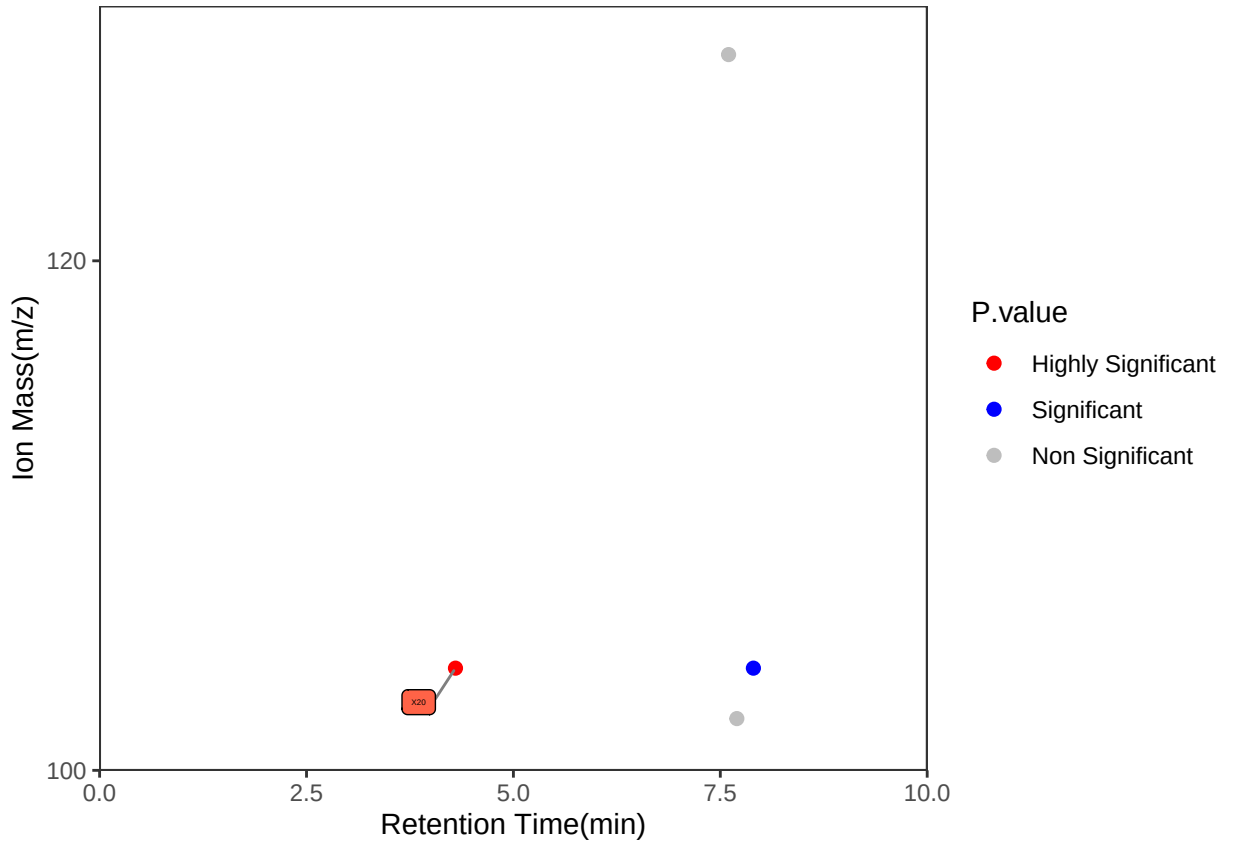



```
res13 = NtaAnno(PID=res$PID,
                OutPath = "default",
                VarsY = "Y2",
                VarsX = "default",
                VarsN = "multiple.factor",
                FdrCorrect = F,
                AdductPos = "M+H,M+2H,M+3H,M+3Na",
                AdductNeg = "M-H,M-2H,M+Cl,M-3H,2M-H",
                Accuracy = 1)
res13$NtaAnno_Y2_multiple.factor_1
```

```
## # A tibble: 43 x 9
##    SerialNo    RT ExactMass Name             SMILES Monoi~1 IonMode Adduct Group
##    <chr>    <dbl>     <dbl> <chr>            <chr>    <dbl> <chr>   <chr>  <chr>
##  1 X20        4.3      103. Propanedioic ac~ OC(=O~    104. negati~ M-H    pare~
##  2 X20        4.3      103. Hydroxypyruvic ~ OCC(=~    104. negati~ M-H    pare~
##  3 X196       4.8      132. alpha-Hydroxybe~ OC(C#~    133. negati~ M-H    pare~
##  4 X196       4.8      132. 4-Anisonitrile   COC1=~    133. negati~ M-H    pare~
##  5 X196       4.8      132. Benzoxazole, 2-~ CC1=N~    133. negati~ M-H    pare~
##  6 X196       4.8      132. 4-Methoxyphenyl~ c1cc(~    133. negati~ M-H    prec~
##  7 X196       4.8      132. (2-Methoxypheny~ c1ccc~    133. negati~ M-H    prec~
##  8 X196       4.8      132. Hydroxy(2-metho~ c1ccc~    133. negati~ M-H    prec~
##  9 X196       4.8      132. (3-Methoxypheny~ c1cc(~    133. negati~ M-H    prec~
```

```
## 10 X196       4.8      132. (3-Methyl-2-fur~ c1cc(~    133. negati~ M-H    prec~
## # ... with 33 more rows, and abbreviated variable name 1: Monoisotopic_Mass
```

```
res14 = VizNtaAnno(PID=res$PID,
                   OutPath = "default",
                   VarsY = "Y2",
                   VarsN = "multiple.factor",
                   Accuracy = 1,
                   Brightness = "light",
                   Palette = "default1")
res14$VizNtaAnno_Y2_multiple.factor_1_light_default1
```




```
FuncExit(PID = res$PID)
```

```
## [1] "Success to exit. Thanks for using ExposomeX platform!"
```

11 ExpoPanel module

11.1 Application domain

ExpoPanel module is designed to conduct the analysis of the panel data. It mainly aims to evaluate the associations between exposure factors and the health outcome.

11.2 Theory

11.3 Work pipeline

Users can easily get the modeling results and their visualization plots with high quality by following the detailed instructions in each step. Linear mixed effect model is adopted to evaluate the association.

```
# The following two packages should be installed in advance
# devtools::install_github("ExposomeX/expanel", force = TRUE)
# devtools::install_github("ExposomeX/extidy", force = TRUE)

# library(expanel)
# library(extidy)
library(tidyverse)

# devtools::install_github("ExposomeX/exposomex", force = TRUE)
library(exposomex)
```

```
res = InitPanel()
res$PID
```

```
## [1] "145800XZKZHT"
```

```
res1 = LoadPanel(PID = res$PID,
                 UseExample = "example#1")

res1$Expo$Data
```

```
## # A tibble: 150 x 208
##    SampleID SubjectID Group    Y1    Y2     C1    C2    C3    X1      X2     X3
##    <chr>        <dbl> <chr> <dbl> <dbl>  <dbl> <dbl> <dbl> <dbl>   <dbl>  <dbl>
##  1 Tr1              1 train     1   106 0.0473 0.623 0.593 0.234 0       0
##  2 Tr2              1 train     0    56 0.0392 0.276 0.186 0.189 0       0
##  3 Tr3              1 train     0    42 0.0526 0.423 0.343 0.225 0       0
##  4 Tr4             16 train     1    66 0.0860 0.557 0.608 0.290 0       0
##  5 Tr5             16 train     0    33 0.109  0.405 0.385 0.322 0       0
##  6 Tr6             16 train     0    13 0.0886 0.433 0.458 0.307 0.0103  0.0310
##  7 Tr7             28 train     1    68 0.0826 0.509 0.536 0.291 0.0168  0.0880
##  8 Tr8             28 train     0    40 0.0979 0.382 0.392 0.328 0.00480 0.0588
##  9 Tr9             28 train     0    19 0.0904 0.346 0.338 0.305 0.00966 0.0878
## 10 Tr10            36 train     1   104 0.122  0.378 0.529 0.280 0.0840  0.212
## # ... with 140 more rows, and 197 more variables: X4 <dbl>, X5 <dbl>, X6 <dbl>,
## #   X7 <dbl>, X8 <dbl>, X9 <dbl>, X10 <dbl>, X11 <dbl>, X12 <dbl>, X13 <dbl>,
## #   X14 <dbl>, X15 <dbl>, X16 <dbl>, X17 <dbl>, X18 <dbl>, X19 <dbl>,
## #   X20 <dbl>, X21 <dbl>, X22 <dbl>, X23 <dbl>, X24 <dbl>, X25 <dbl>,
## #   X26 <dbl>, X27 <dbl>, X28 <dbl>, X29 <dbl>, X30 <dbl>, X31 <dbl>,
## #   X32 <dbl>, X33 <dbl>, X34 <dbl>, X35 <dbl>, X36 <dbl>, X37 <dbl>,
## #   X38 <dbl>, X39 <dbl>, X40 <dbl>, X41 <dbl>, X42 <dbl>, X43 <dbl>, ...
```

```
res2 = TransImput(PID=res$PID,
                  Group="T",
                  Vars="all.x",
                  Method="lod")

res3 = DelNearZeroVar(PID = res$PID)

res4 = DelMiss(PID = res$PID)

res5 = TransType(PID=res$PID,
                 Vars="Y1",
                 To="factor")

res6 = TransClass(PID=res$PID,
                  Group="F",
                  Vars="X1",
                  LevelTo="4")

res7 = TransScale(PID=res$PID,
                  Group="T",
                  Vars="all.x",
                  Method="normal")

res8 = TransDummy(PID=res$PID,
                  Vars="default")

res7$Expo$Data
```

```
## # A tibble: 150 x 208
##    SampleID SubjectID Group Y1       Y2     C1    C2    C3 X1        X2     X3
##    <chr>        <dbl> <chr> <fct> <dbl>  <dbl> <dbl> <dbl> <fct>  <dbl>  <dbl>
##  1 Tr1              1 train 1       106 0.0473 0.623 0.593 1     -0.944 -1.14
##  2 Tr2              1 train 0        56 0.0392 0.276 0.186 1     -0.944 -1.14
##  3 Tr3              1 train 0        42 0.0526 0.423 0.343 1     -0.944 -1.14
##  4 Tr4             16 train 1        66 0.0860 0.557 0.608 2     -0.944 -1.14
##  5 Tr5             16 train 0        33 0.109  0.405 0.385 3     -0.944 -1.14
##  6 Tr6             16 train 0        13 0.0886 0.433 0.458 2     -0.673 -0.691
##  7 Tr7             28 train 1        68 0.0826 0.509 0.536 2     -0.501  0.124
##  8 Tr8             28 train 0        40 0.0979 0.382 0.392 3     -0.817 -0.294
##  9 Tr9             28 train 0        19 0.0904 0.346 0.338 2     -0.689  0.122
## 10 Tr10            36 train 1       104 0.122  0.378 0.529 2      1.27   1.91
## # ... with 140 more rows, and 197 more variables: X4 <dbl>, X5 <dbl>, X6 <dbl>,
## #   X7 <dbl>, X8 <dbl>, X9 <dbl>, X10 <dbl>, X11 <dbl>, X12 <dbl>, X13 <dbl>,
## #   X14 <dbl>, X15 <dbl>, X16 <dbl>, X17 <dbl>, X18 <dbl>, X19 <dbl>,
## #   X20 <dbl>, X21 <dbl>, X22 <dbl>, X23 <dbl>, X24 <dbl>, X25 <dbl>,
## #   X26 <dbl>, X27 <dbl>, X28 <dbl>, X29 <dbl>, X30 <dbl>, X31 <dbl>,
## #   X32 <dbl>, X33 <dbl>, X34 <dbl>, X35 <dbl>, X36 <dbl>, X37 <dbl>,
## #   X38 <dbl>, X39 <dbl>, X40 <dbl>, X41 <dbl>, X42 <dbl>, X43 <dbl>, ...
```

```
res7$Expo$Voca
```

```
## # A tibble: 205 x 5
##    SerialNo SerialNo_Raw FullName                GroupName Lod
##    <chr>    <chr>        <chr>                   <chr>     <lgl>
```

```
##  1 Y1       Y1           TL_cat2                   Outcome   NA
##  2 Y2       Y2           TL                        Outcome   NA
##  3 C1       C1           cMCs_NFkB_unstim          immunome  NA
##  4 C2       C2           CCR2poscMCs_STAT3_unstim immunome  NA
##  5 C3       C3           MDSCs_STAT3_unstim        immunome  NA
##  6 X1       X1           DCs_S6_unstim             immunome  NA
##  7 X2       X2           DCs_MAPKAPK2_unstim       immunome  NA
##  8 X3       X3           DCs_NFkB_unstim           immunome  NA
##  9 X4       X4           CCR2negncMCs_NFkB_unstim immunome  NA
## 10 X5       X5           NK_STAT5_unstim           immunome  NA
## # ... with 195 more rows
```

```
res9 = FindCovaPanel(PID=res$PID,
                     OutPath= "default",
                     VarsY = "Y1",
                     VarsC_Prior = "default",
                     VarsC_Fixed ="C1",
                     Method = "single.factor",
                     Thr = 0.1)

res9_1 = PanelAsso(PID=res$PID,
                   VarsY = "Y1",
                   VarsX = "X1,X2,X3,X4,X5,X6,X7,X8,X9,X10,X11,X12",
                   VarsN = "multiple.factor",
                   VarsRandomIpt = "SubjectID",
                   VarsRandomSlp = "none",
                   IncCova = "F")

res9_2 = PanelAsso(PID=res$PID,
                   OutPath= "default",
                   VarsY = "Y2",
                   VarsX = "X1,X2,X3,X4,X5,X6,X7,X8,X9,X10,X11,X12",
                   VarsN = "multiple.factor",
                   VarsRandomIpt = "SubjectID",
                   VarsRandomSlp = "none",
                   IncCova = "F")

res10 = VizPanelAsso(PID=res$PID,
                     OutPath= "default",
                     VarsY = "Y1",
                     VarsN = "multiple.factor" ,
                     EffectThr = 0.5,
                     Layout = "forest",
                     Brightness = "dark",
                     Palette = "default1")

res11_1 = PanelAsso(PID=res$PID,
                    OutPath= "default",
                    VarsY = "Y1",
                    VarsX = "X1,X2,X3,X4,X5,X6,X7,X8,X9,X10,X11,X12",
                    VarsN = "single.factor" ,
                    VarsRandomIpt = "SubjectID",
                    VarsRandomSlp = "none",
                    IncCova = "F")
```

```
res11_2 = PanelAsso(PID=res$PID,
                    OutPath= "default",
                    VarsY = "Y2",
                    VarsX = "X1,X2,X3,X4,X5,X6,X7,X8,X9,X10,X11,X12",
                    VarsN = "single.factor" ,
                    VarsRandomIpt = "SubjectID",
                    VarsRandomSlp = "none",
                    IncCova = "F")

res12 = VizPanelAsso(PID = res$PID,
                     OutPath= "default",
                     VarsY = "Y1",
                     VarsN = "single.factor",
                     EffectThr = 0.5,
                     Layout = "forest",
                     Brightness = "dark",
                     Palette = "default1")


FuncExit(PID = res$PID)
```

```
## [1] "Success to exit. Thanks for using ExposomeX platform!"
```

12 ExpoSurv module

12.1 Application domain

ExpoSurvival module is designed to conduct the survival analysis of the censored data. It mainly aims to evaluate the associations between exposure factors and health outcome, as well as predicting the survival probability. Users can easily get the modeling results and their visualization plots with high quality by following the detailed instructions in each step.

12.2 Theory

In "SurvAsso()" function, Cox proportional hazards regression model is used to calculate the hazard ratio (HR). In "SurvPred()" function, various machine learners are used to conduct the prediction analysis, including coxph, coxboost, xgboost, ranger, cforest, ctree, nelson, etc., from mlr3 R package.

12.3 Work pipeline

Install packages

To use Exposurv module, users can install the "exposomex" package, or "exposurv" and "expotidy" packages separately.

```
# The following two packages should be installed in advance
# devtools::install_github("ExposomeX/exsurv", force = TRUE)
# devtools::install_github("ExposomeX/extidy", force = TRUE)

# library(exsurv)
# library(extidy)
library(tidyverse)

# devtools::install_github("ExposomeX/exposomex", force = TRUE)
library(exposomex)
```

Initialize

At first, you need to initialize the calculation environment using initialization function: "InitSurv()".

```
res = InitSurv()
res
```

```
## <eSet>
##   Public:
##     AddCommand: function (x)
##     AddLog: function (x)
##     clone: function (deep = FALSE)
##     ExcecutionLog: Complete initializing the Exoverse module.2022-12-14 14: ...
##     Expo: list
##     ExpoDel: list
##     FileDirOut: /home/ubuntu/@changxin/R_Exposome_1.0/output_145843FDEPNS
##     Func: list
##     PID: 145843FDEPNS
##     RCommandLog: eSet <- InitExpoData(PID = Any ID your like, FileDirOut  ...
```

Here, we can see that the returned value "res" is an R6 object. It contains an unique program ID of res$PID (e.g., "100737GJMWJA"), which is random generated by the system. Users need use it in the following step for further data process.

Upload data

Then, you need to upload data for analysis by runing "LoadSurv()" function. The function includes four parameters:
1. PID: chr. Program ID. It must be the same with the PID generated by InitSurv.
2. UseExample: chr. Method of uploading data. If "default", user should upload their own data files, or use "example#1" provided by this module.
3. DataPath: chr. Input data file directory, e.g. "D:/test/eg_Surv_data.xlsx".
4. VocaPath: chr. Input vocabulary file directory, e.g. "D:/test/eg_Surv_voca.xlsx".

Noted that there are several data format requirments for the data and vocabulary file. For data file,the first three columns should be named as "SampleID", "SubjectID", and "Group", respectively. For the "Group" variable, only two values can be used, i.e. "train" and "test". If there is no data for test, all values should be set as "train". For outcome variables, their initials must be set as "Y" and serialized by adding Arabic numerals if needed, e.g., Y1, Y2, Y3. In this module, the survival time (Y1) and status (Y2) must be provided.For exposure variables, their initials must be set as "X" and serialized by adding Arabic numerals if needed, e.g., X1, X2, X3. For covariate variables, their initials must be set as "C" and serialized by adding Arabic numerals if needed, e.g., C1, C2, C3. It should be noted the covariates are not required if users don't have. For vocabulary file, the first two columns must be named as "SerialNo" and "FullName", respectively. The list of SerialNo of outcomes, exposure, and covariates should be the same with the column names of "Data file". The list of the FullName is prepared as users' like.

Here, we use "example#1" data for demonstration:

```
res1 <- LoadSurv(PID = res$PID, UseExample = "example#1", DataPath = NULL,
   VocaPath = NULL)
res1$Expo$Data
```

```
## # A tibble: 394 x 13
##    SampleID Subjec~1 Group    Y1    Y2    C1    C2    C3    X1    X2    X3    X4
##       <dbl>    <dbl> <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
##  1        1        1 train   306     2 209.     74     1     1    90   100 1175
##  2        2        2 train   455     2 213.     68     1     0    90    90 1225
##  3        3        3 train  1010     1 105.     56     1     0    90    90  485.
##  4        4        4 train   210     2 196.     57     1     1    90    60 1150
##  5        5        5 train   883     2 108.     60     1     0   100    90  504.
##  6        6        6 train  1022     1 103.     74     1     1    50    80  513
##  7        7        7 train   310     2  85.1    68     2     2    70    60  384
##  8        8        8 train   361     2 108.     71     2     2    60    80  538
##  9        9        9 train   218     2 149.     53     1     1    70    80  825
## 10       10       10 train   166     2  72.7    61     1     2    70    70  271
## # ... with 384 more rows, 1 more variable: X5 <dbl>, and abbreviated variable
## #   name 1: SubjectID
```

```
res1$Expo$Voca
```

```
## # A tibble: 10 x 11
##    SerialNo Serial~1 FullN~2 Group~3 Lod   CasRN Inchi~4 KeggC~5 Uniprot Disea~6
##    <chr>    <chr>    <chr>   <chr>   <lgl> <lgl> <lgl>   <lgl>   <lgl>   <lgl>
##  1 Y1       Y1       time    outcome NA    NA    NA      NA      NA      NA
##  2 Y2       Y2       status  outcome NA    NA    NA      NA      NA      NA
##  3 C1       C1       conf    chemic~ NA    NA    NA      NA      NA      NA
##  4 C2       C2       age     chemic~ NA    NA    NA      NA      NA      NA
##  5 C3       C3       sex     chemic~ NA    NA    NA      NA      NA      NA
##  6 X1       X1       ph.ecog chemic~ NA    NA    NA      NA      NA      NA
##  7 X2       X2       ph.kar~ chemic~ NA    NA    NA      NA      NA      NA
##  8 X3       X3       pat.ka~ chemic~ NA    NA    NA      NA      NA      NA
```

```
##  9 X4       X4       meal.c~ chemic~ NA    NA    NA      NA      NA      NA
## 10 X5       X5       wt.loss chemic~ NA    NA    NA      NA      NA      NA
## # ... with 1 more variable: PhenotypeID <lgl>, and abbreviated variable names
## #   1: SerialNo_Raw, 2: FullName, 3: GroupName, 4: Inchikey, 5: KeggChemEntry,
## #   6: DiseaseID
```

Tidy data

See ExpoTidy module for more information. You can skip some steps of the "Tidy" part, if you make sure that the data can satisfy the modeling requirement. We recommend that you run 'TransDummy()' to prevent errors like the one below:

If X1 in your data is a categorical variable and you have an X11 variable, then name conflicts will occur during modeling. Because model result will turn X1 into X11, X12, X12...

Find Covariates

Users can run "FindCovaSurv()" to find out which covariates should be included in the model before modeling. The function includes eight parameters:
1. PID: chr. Program ID. It must be the same with the PID generated by InitSurv.
2. OutPath: chr. Output file directory, e.g. "D:/test". If "default", the current working directory will be set.
3. TimeY: chr. Outcome variable of survival time used for modelling. Only one variable can be entered.
4. EventY: chr. Outcome variable of status used for modelling. Only one variable can be entered.
5. VarsC_Prior: chr. Potential covariates needing further statistical test. The default value is all covariate variables listed in the data file.
6. VarsC_Fixed: chr. Covariate variables fixed in the model by users.
7. Method: chr. Methods for screening the covariates, including two options, i.e. "single.factor" and "two.stage".
8. Thr: num. Threshold of the P-value for screening the covariates. It isranging 0.05-0.25. The default value is 0.1.

```
res2 = FindCovaSurv(PID = res$PID,
                    TimeY = "Y1",
                    EventY= 'Y2',
                    VarsC_Prior = "default",
                    VarsC_Fixed = NULL,
                    Method = "single.factor",
                    Thr = 0.1)

res2$FindCovaSurv
```

```
## # A tibble: 2 x 1
##   covariates
##   <chr>
## 1 C2
## 2 C3
```

Association

Now, users can calculate the hazard ratio(HR) of exposures variables by running "SurvAsso()" function. The function includes eight parameters:
1. PID: chr. Program ID. It must be the same with the PID generated by InitSurv.
2. OutPath: chr. Output file directory, e.g. "D:/test". If "default", the current working directory will be set.
3. TimeY: chr. Outcome variable of survival time used for modelling. Only one variable can be entered.
4. EventY: chr. Outcome variable of status used for modelling. Only one variable can be entered.
5. VarsX: chr. Exposure variable used for modeling. The default option is "all.x" (All exposure variables are

included). Users can also choose available variables. It should be noted that there is fixed format for the entering characters separated with comma and without space, e.g., "X1,X2,X3"
6. VarsN: chr. Choose the single factor or multiple factor model. Available options include "single.factor" and "multiple.factor"
7. VarsSel: lgl. T (or TRUE) and F (or FALSE). Whether to select the significant variable for the final model. Available options includes T and F.
8. IncCova: lgl. T (or TRUE) and F (or FALSE). Whether to include the covariate selected in the function "FindCovaSurv".

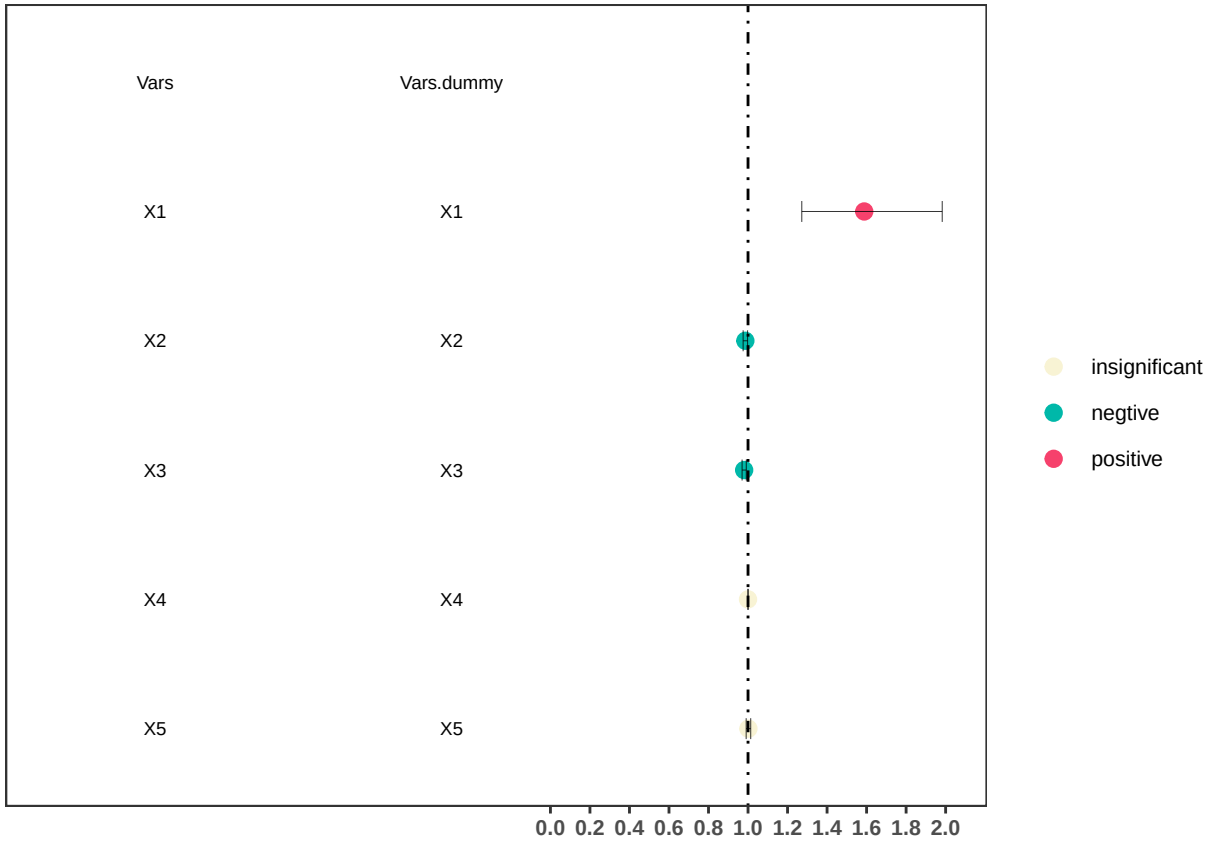
```
res3 = SurvAsso(PID = res$PID,
                TimeY = "Y1",
                EventY= 'Y2',
                VarsX='all.x',
                VarsN="single.factor",
                VarsSel=T,
                IncCova = T)
res3$coxph_res
```

```
## # A tibble: 5 x 8
##   Vars  Vars.dummy    HR  CI_L  CI_H   p.value cindex logloss
##   <chr> <chr>      <dbl> <dbl> <dbl>     <dbl>  <dbl>   <dbl>
## 1 X1    X1         1.59  1.27  1.98  0.0000442  0.637    4.17
## 2 X2    X2         0.986 0.975 0.998 0.0201     0.640    4.21
## 3 X3    X3         0.980 0.970 0.991 0.000327   0.643    4.16
## 4 X4    X4         1.00  1.00  1.00  0.502      0.607    4.23
## 5 X5    X5         1.00  0.990 1.01  0.741      0.601    4.24
```

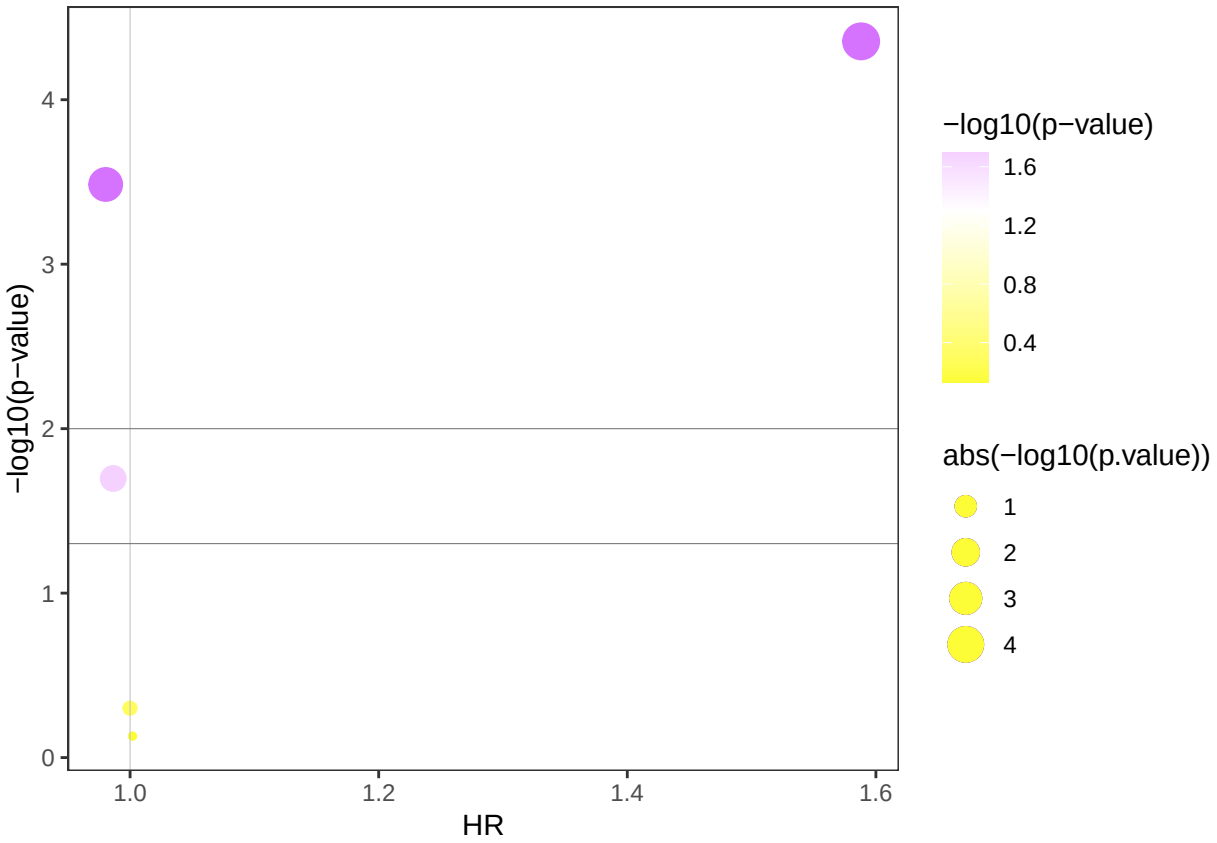
Viz Association

After running association model, users can visualize the results by running "VizSurvAsso()" function. The function includes eight parameters:
1. PID: chr. Program ID. It must be the same with the PID generated by InitSurv.
2. OutPath: chr. Output file directory, e.g. "D:/test". If "default", the current working directory will be set.
3. VarsN: chr. Choose the single factor or multiple factor model. Available options include "single.factor" and "multiple.factor". It must be same as the 'VarsN' in Association function.
4. Layout: chr. Visualization layout. Available options include "forest" and "volcano".
5. Brightness: chr. Visualization brightness. Available options include "light" and "dark".
6. Palette: chr. Visualization palette. Available options include "default1", "default2" and several journal preference styles (i.e., "cell", "nature", "science", "lancet", "nejm", and "jama").
7. ColorFor: chr. Volcano plot dot color. Available options include "p.value" and "hr".
8. SizeFor: chr. Volcano plot dot size. Available options include "p.value" and "hr".

```
res5 = VizSurvAsso(PID = res$PID,
                   VarsN="single.factor",
                   Layout="forest",
                   Brightness= "light",
                   Palette = "default1")
res5$ForestPlot
```




```
res6 = VizSurvAsso(PID = res$PID,
                   VarsN="single.factor",
                   Layout="volcano",
                   Brightness= "light",
                   Palette = "default1",
                   ColorFor= "p.value",
                   SizeFor= "p.value")
res6$VolcanoPlot
```



Prediction

In addition, users can predict the survival curve by "SurvPred()" function. The function includes ten parameters:
1. PID: chr. Program ID. It must be the same with the PID generated by InitSurv.
2. OutPath: chr. Output file directory, e.g. "D:/test". If "default", the current working directory will be set.
3. TimeY: chr. Outcome variable of survival time used for modelling. Only one variable can be entered.
4. EventY: chr. Outcome variable of status used for modelling. Only one variable can be entered.
5. VarsX: chr. Exposure variable used for modeling. The default option is "all.x" (All exposure variables are included). Users can also choose available variables. It should be noted that there is fixed format for the entering characters separated with comma and without space, e.g., "X1,X2,X3" 6. IncCova: lgl. Whether to include the covariate selected in the function of "FindCovaSurv". Available options include T (or TRUE) and F (or FALSE).
7. RsmpMethod: chr. Three resampling methods options for internal validation, including "cv" (i.e.,Cross validation), "bootstrap", and "holdout".
8. Folds: num. Folds of Cross-validation resampling. It is ranging 2-10.
9. Ratio: num. Ratio of Bootstrap resampling. It is ranging 0.4-0.9.
10. Repeats: num. Number of Bootstrap resampling. It is ranging 2-20.

```
res7 = SurvPred(PID = res$PID,
                TimeY = "Y1",
                EventY = 'Y2',
                VarsX ='all.x',
                IncCova = T,
                RsmpMethod ="cv",
                Folds = 3)
```

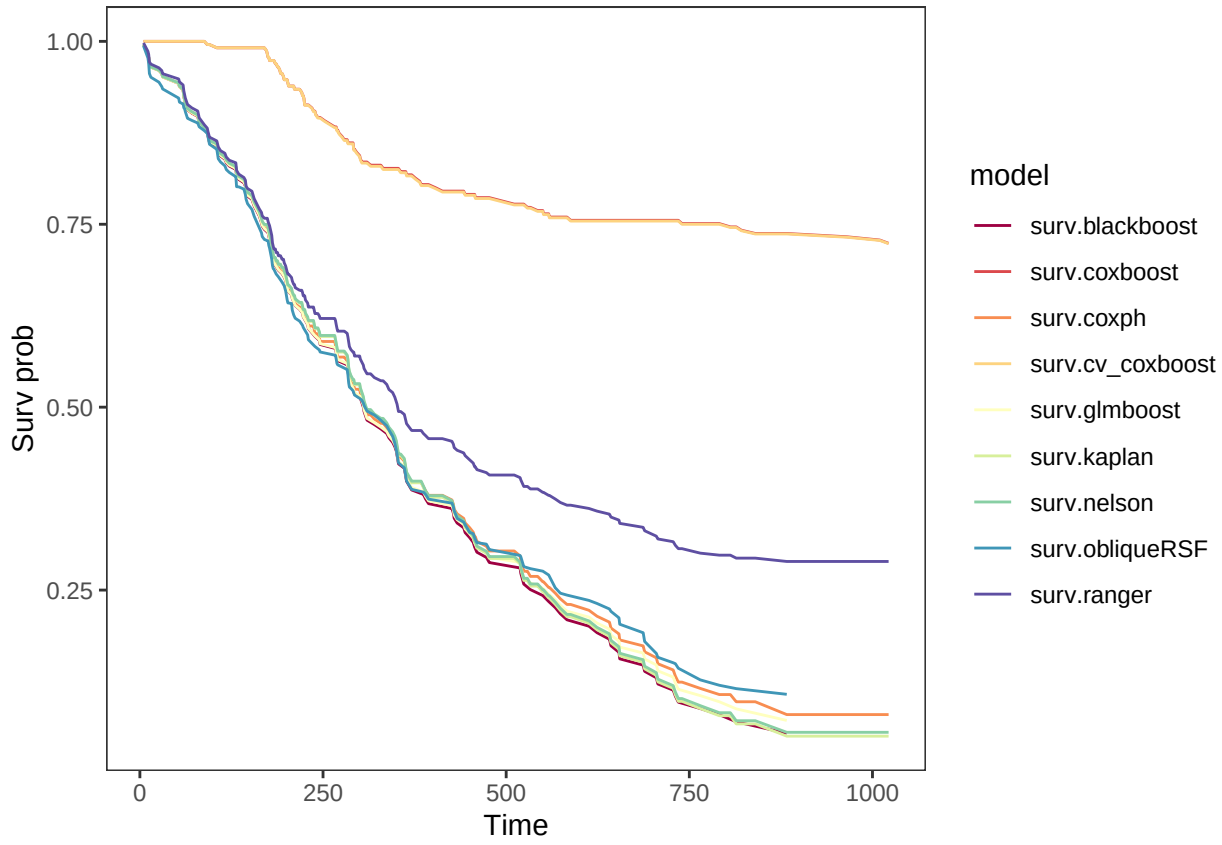
```
res7$bmrout
```

```
## # A tibble: 14 x 7
##    learner_id       cindex_rsmp_train_~1 cinde~2 cinde~3 cinde~4 cinde~5 cinde~6
##    <chr>                           <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
##  1 surv.blackboost                 0.737   0.598 0.00614 0.0302    0.709   0.718
##  2 surv.cforest                    0.723   0.602 0.00688 0.00907   0.723   0.729
##  3 surv.coxboost                   0.338   0.388 0.00483 0.0213    0.343   0.332
##  4 surv.coxph                      0.663   0.609 0.00802 0.0226    0.651   0.660
##  5 surv.ctree                      0.605   0.520 0.00468 0.0135    0.622   0.617
##  6 surv.cv_coxboost                0.391   0.418 0.0384  0.0477    0.357   0.346
##  7 surv.gbm                        0.701   0.580 0.00248 0.0142    0.679   0.684
##  8 surv.glmboost                   0.669   0.612 0.00591 0.0145    0.660   0.671
##  9 surv.kaplan                     0.5     0.5   0       0         0.5     0.5
## 10 surv.nelson                     0.5     0.5   0       0         0.5     0.5
## 11 surv.obliqueRSF                 0.703   0.580 0.0175  0.0226    0.711   0.715
## 12 surv.parametric                 0.663   0.609 0.00956 0.0214    0.651   0.660
## 13 surv.ranger                     0.862   0.569 0.00498 0.0382    0.857   0.866
## 14 surv.xgboost                    0.432   0.467 0.0386  0.0547    0.415   0.395
## # ... with abbreviated variable names 1: cindex_rsmp_train_mean,
## #   2: cindex_rsmp_test_mean, 3: cindex_rsmp_train_sd, 4: cindex_rsmp_test_sd,
## #   5: cindex_train, 6: cindex_test
```

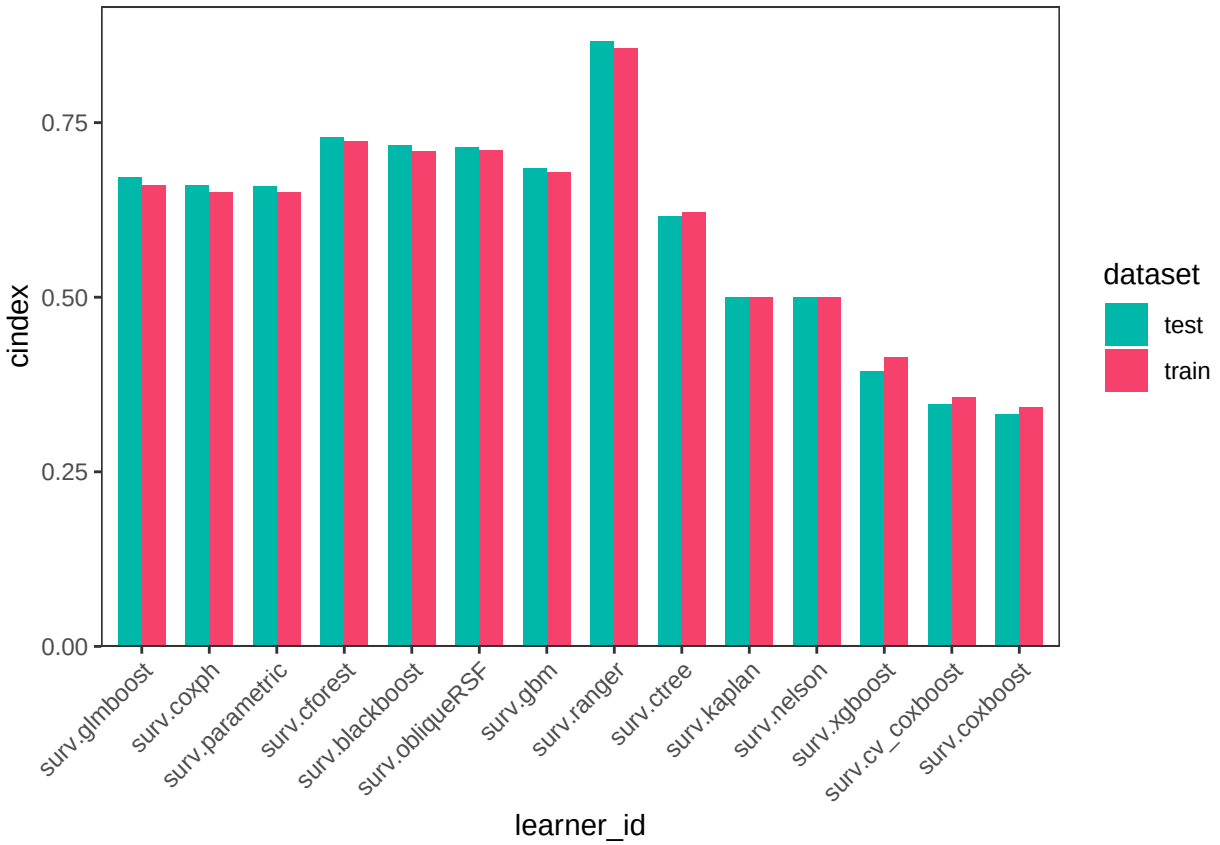
Viz Prediction

After running SurvPred model, users can visualize the results by running "VizSurvPred()". The function includes five parameters:
1. PID: chr. Program ID. It must be the same with the PID generated by InitSurv.
2. OutPath: chr. Output file directory, e.g. "D:/test". If "default", the current working directory will be set.
3. Layout: chr. chr. Visualization layout. Available options include "curve" , "bar" and 'all'.
4. Brightness: chr. Visualization brightness. Available options include "light" and "dark".
5. Palette: chr. Visualization palette. Available options include "default1", "default2" and several journal preference styles (i.e., "cell", "nature", "science", "lancet", "nejm", and "jama").

```
res8 = VizSurvPred(PID = res$PID,
                   Layout="all",
                   Brightness="light",
                   Palette='default1')
res8$PredSurvCurvPlot
```




```
res8$BmrBarPlot
```

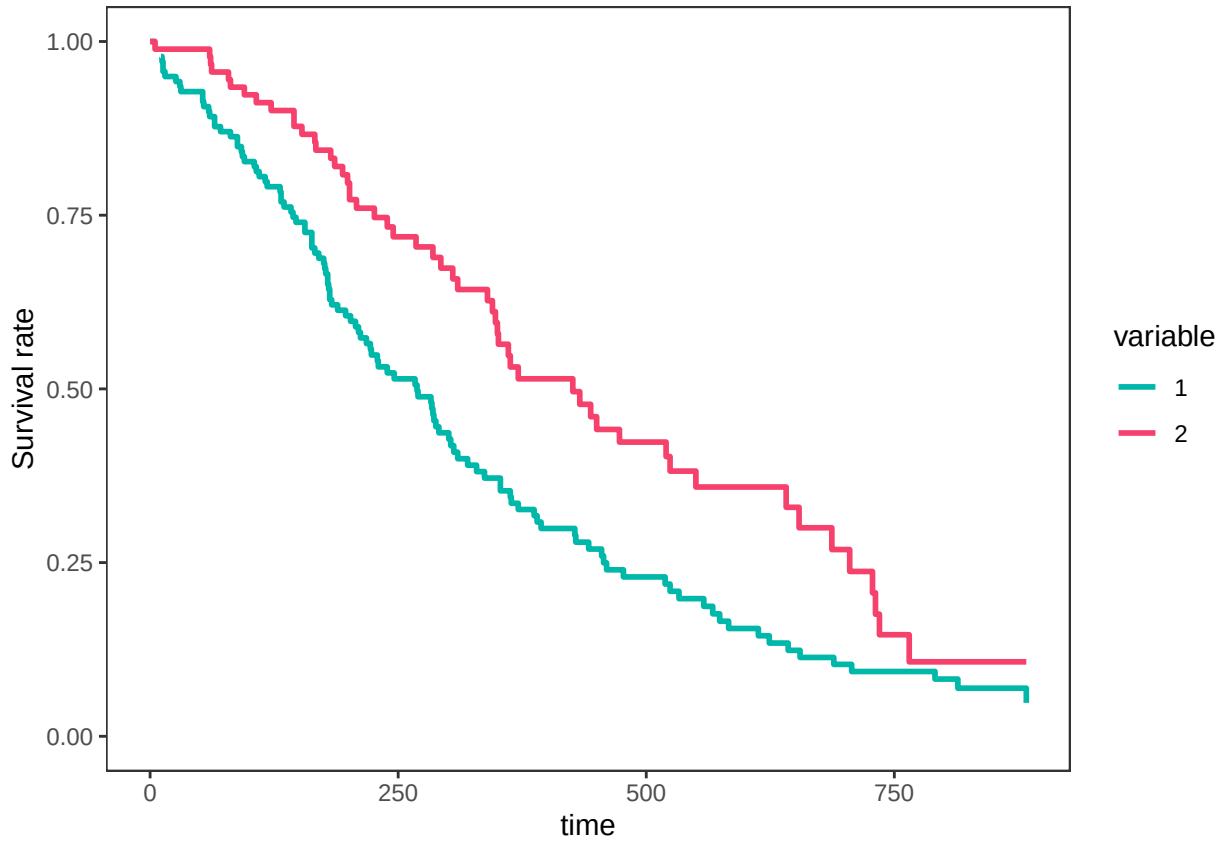


VizSurvCompGroup

We also provide a function to compare the survival curves of two groups. This function does not depend on the previous functions result. Users can run it after "LoadSurv()". The function includes ten parameters:
1. PID: chr. Program ID. It must be the same with the PID generated by InitSurv.
2. OutPath: chr. Output file directory, e.g. "D:/test". If "default", the current working directory will be set.
3. TimeY: chr. Outcome variable of survival time used for modelling. Only one variable can be entered.
4. EventY: chr. Outcome variable of status used for modelling. Only one variable can be entered.
5. VarsG: chr. Grouping variable, must be binary variables. It should be noted that there is fixed format for the entering characters separated with comma and without space, e.g., "X1,X2,X3" 6. Model: chr. Methods to depict the survival curve. Options include 'km' (Kaplan-Meier estimate) and "coxph" (Cox proportional hazards regression mode).
7. VarsAdj: chr. If you choose the cox model, co-variables used for modelling. It should be noted that there is fixed format for the entering characters separated with comma and without space, e.g., "X1,X2,X3".
8. AdjMethod: chr. If you choose the coxph model, method for adjusting model, includes: "average","single","margin" and "conditional".
9. Brightness: chr. Visualization brightness. Available options include "light" and "dark".
10. Palette: chr. Visualization palette. Available options include "default1", "default2" and several journal preference styles (i.e., "cell", "nature", "science", "lancet", "nejm", and "jama").

```
res9 = VizSurvCompGroup(PID = res$PID,
                        TimeY="Y1",
                        EventY="Y2",
                        VarsG="C3",
                        Model="coxph",
                        VarsAdj='X1,X2,X3,X4,X5',
```

```
                        AdjMethod='average',
                        Brightness="light",
                        Palette='default1')
res9$Comp_Coxph_Plot[[1]]
```



Exit

After all the analysis is done, please run the "FuncExit()" function to delete the data uploaded to the server.

```
FuncExit(PID = res$PID)
```

```
## [1] "Success to exit. Thanks for using ExposomeX platform!"
```

13 ExpoStatLink module

13.1 Application domain

ExpoStatLink module is designed to find the statistical relationships between exposure factors and health outcome.

13.2 Theory

13.3 Work pipeline

Users can easily get the modeling results and their visualization plots with high quality by following the detailed instructions in each step. The statistical interpretations from the perspectives of subjects and whole dataset are both provided.

```
# The following two packages should be installed in advance
# devtools::install_github("ExposomeX/exstat", force = TRUE)
# devtools::install_github("ExposomeX/extidy", force = TRUE)

# library(exstat)
# library(extidy)
library(tidyverse)

# devtools::install_github("ExposomeX/exposomex", force = TRUE)
library(exposomex)
```

```
res = InitStatLink()

res1 = LoadStatLink(PID=res$PID,
                    UseExample="example#1")
res1$Expo$Data
```

```
## # A tibble: 20 x 14
##    SampleID Subjec~1    Y1    Y2    X1    X2    X3    X4    X5    X6    X7    X8
##    <chr>    <chr>    <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
##  1 Tr1      S1           1  -101  7.76 10.2  24.2   1.47  1.06  1.43  2.00  1.87
##  2 Tr2      S2           0   -51 10.1  11.6   3.39  1.03  1.46  1.32  1.69  1.18
##  3 Tr3      S3           0   -37  8.54  9.52 15.8   1.23  1.42  1.64  1.52  1.95
##  4 Tr4      S4           1   -61 14.2  14.9  11.3  NA     1.75  1.23  1.91  1.29
##  5 Tr5      S5           0   -28 11.0  16.4   1.43  1.31  1.70  1.69  1.04  1.68
##  6 Tr6      S6           0    -8 11.3  12.7   8.37  1.29  1.25  1.38  1.26  1.17
##  7 Tr7      S7           1   -63  7.66  7.73 17.1   1.49  1.19  1.83  1.46  1.08
##  8 Tr8      S8           0   -35 11.3   8.25 13.3   1.48  1.91  1.07  1.30  1.09
##  9 Tr9      S9           0   -14 14.5  11.1  11.4   1.29 NA     1.30  1.46  1.23
## 10 Tr10     S10          1   -99  6.26  6.04  8.03  1.92  1.53  1.08  1.92  1.85
## 11 Tr11     S11          0   -60  3.43 10.3   8.45  1.76  1.79  1.13  1.73  1.07
## 12 Tr12     S12          0   -32  6.75  6.86 28.0   1.03  1.10  1.58  1.98  1.06
## 13 Tr13     S13          0   -73  7.71  9.47 22.3  NA     1.20  1.34  1.45  1.53
## 14 Tr14     S14          0   -18 12.6   8.97  9.23  1.23  1.13 NA     1.62  1.58
## 15 Tr15     S15          0   -48 11.7   8.60  9.31  1.20  1.33  1.99  1.85  1.94
## 16 Tr16     S16          0   -20  7.10 12.5   2.52  1.83  1.19  1.65  1.47  1.48
## 17 Tr17     S17          0    -9  6.32 49.1   8.52  1.26  1.94  1.53  1.95  1.18
## 18 Tr18     S18          1   -98  4.66  8.49 24.2   1.07  1.25  1.35  1.70  1.98
## 19 Tr19     S19          0   -70  2.44 -2.27 16.0   1.43  1.77  1.21  1.87  1.48
## 20 Tr20     S20          0   -36 10.6   7.07 10.0   1.99  1.61  1.65  1.84  1.94
## # ... with 2 more variables: X9 <dbl>, X10 <dbl>, and abbreviated variable name
```

```
## #   1: SubjectID
```

```
res1$Expo$Voca
```

```
## # A tibble: 12 x 5
##    SerialNo SerialNo_Raw FullName GroupName   Lod
##    <chr>    <chr>        <chr>    <chr>     <dbl>
##  1 Y1       Y1           Y_disc   Outcome    NA
##  2 Y2       Y2           Y_cont   Outcome    NA
##  3 X1       X1           TE_1     Chemical    0.5
##  4 X2       X2           TE_2     Chemical    0.5
##  5 X3       X3           TE_3     Chemical    0.5
##  6 X4       X4           TE_4     Chemical    0.5
##  7 X5       X5           TE_5     Chemical    0.5
##  8 X6       X6           TE_6     Chemical    0.5
##  9 X7       X7           TE_7     Chemical    0.5
## 10 X8       X8           TE_8     Chemical    0.5
## 11 X9       X9           CH1      Chemical    5
## 12 X10      X10          CH2      Chemical    5
```

```
res2 = StatLinkCros(PID=res$PID,
                    OutPath = "default",
                    VarsY = "Y1" ,
                    VarsX = "all.x" ,
                    LinkModel = "ranger",
                    ObsrPartType = "raw" ,
                    ObsrPartNum  = "50",
                    ObsrProfType  = "partial" ,
                    ObsrProfNum  = "100",
                    ObsrProfVars = "all.x" ,
                    ObsrProfGeom  = "profiles",
                    SubjPredSeq = "S1,S2,S3",
                    SubjPartType = "break_down")
res2$CrosVipPlot
```

```
## $Y1_importance_ranger_raw
## [[1]]
## TableGrob (1 x 1) "arrange": 1 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
##
## attr(,"class")
## [1] "arrangelist" "list"
```

```
res2$CrosProfPlot
```

```
## $Y1_ranger_partial_profiles_by_X1
## [[1]]
## TableGrob (1 x 1) "arrange": 1 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
##
## attr(,"class")
## [1] "arrangelist" "list"
##
## $Y1_ranger_partial_profiles_by_X2
```

```
## [[1]]
## TableGrob (1 x 1) "arrange": 1 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
##
## attr(,"class")
## [1] "arrangelist" "list"
##
## $Y1_ranger_partial_profiles_by_X3
## [[1]]
## TableGrob (1 x 1) "arrange": 1 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
##
## attr(,"class")
## [1] "arrangelist" "list"
##
## $Y1_ranger_partial_profiles_by_X4
## [[1]]
## TableGrob (1 x 1) "arrange": 1 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
##
## attr(,"class")
## [1] "arrangelist" "list"
##
## $Y1_ranger_partial_profiles_by_X5
## [[1]]
## TableGrob (1 x 1) "arrange": 1 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
##
## attr(,"class")
## [1] "arrangelist" "list"
##
## $Y1_ranger_partial_profiles_by_X6
## [[1]]
## TableGrob (1 x 1) "arrange": 1 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
##
## attr(,"class")
## [1] "arrangelist" "list"
##
## $Y1_ranger_partial_profiles_by_X7
## [[1]]
## TableGrob (1 x 1) "arrange": 1 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
##
## attr(,"class")
## [1] "arrangelist" "list"
##
## $Y1_ranger_partial_profiles_by_X8
```

```
## [[1]]
## TableGrob (1 x 1) "arrange": 1 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
##
## attr(,"class")
## [1] "arrangelist" "list"
##
## $Y1_ranger_partial_profiles_by_X9
## [[1]]
## TableGrob (1 x 1) "arrange": 1 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
##
## attr(,"class")
## [1] "arrangelist" "list"
##
## $Y1_ranger_partial_profiles_by_X10
## [[1]]
## TableGrob (1 x 1) "arrange": 1 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
##
## attr(,"class")
## [1] "arrangelist" "list"
```

```
res2$CrosSubjPlot$Y1_ranger_break_down_S1
```

```
## [[1]]
## TableGrob (1 x 1) "arrange": 1 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
##
## attr(,"class")
## [1] "arrangelist" "list"
```

```
FuncExit(PID = res$PID)
```

```
## [1] "Success to exit. Thanks for using ExposomeX platform!"
```

14 ExpoBioLink module

14.1 Application domain

ExpoBioLink module is designed to find the biological relationships between exposure factors and health outcome. This module adopts the most frequently-used and authoritative databases, e.g., T3DB, CTD, ToxCast, StringDB, STITCH, KEGG, and GO.

14.2 Theory

14.3 Work pipeline

Users can easily get the modeling results and their visualization plots of high quality by following the detailed instructions in each step. The biological relationships between exposures and health outcome are interpreted from the perspectives of protein-protein interaction (PPI) and gene ontology (GO).

```
# The following two packages should be installed in advance
# devtools::install_github("ExposomeX/exbiolink", force = TRUE)
# devtools::install_github("ExposomeX/extidy", force = TRUE)

# library(exbiolink)
# library(extidy)
library(tidyverse)

# devtools::install_github("ExposomeX/exposomex", force = TRUE)
library(exposomex)
```

```
res = InitBioLink()

res1 = LoadBioLink(PID=res$PID,
                   UseExample="example#1")
res1$Expo$Data
```

```
## # A tibble: 221 x 7
##    SerialNo FullName GroupName DiseaseID   ExposureID MetabolomeID ProteomeID
##    <chr>    <chr>    <chr>     <chr>       <chr>      <chr>        <chr>
##  1 Y1       CSNU     disease   OMIM:220100 <NA>       <NA>         <NA>
##  2 Y2       SCZD     disease   OMIM:181500 <NA>       <NA>         <NA>
##  3 Y3       TGCT     disease   OMIM:273300 <NA>       <NA>         <NA>
##  4 Y4       RASJ     disease   OMIM:604302 <NA>       <NA>         <NA>
##  5 Y5       HH3      disease   OMIM:244200 <NA>       <NA>         <NA>
##  6 Y6       DECRD    disease   OMIM:616034 <NA>       <NA>         <NA>
##  7 Y7       PMDS1    disease   OMIM:261550 <NA>       <NA>         <NA>
##  8 Y8       CRMCC2   disease   OMIM:617341 <NA>       <NA>         <NA>
##  9 Y9       FSHD1    disease   OMIM:158900 <NA>       <NA>         <NA>
## 10 Y10      RA       disease   OMIM:180300 <NA>       <NA>         <NA>
## # ... with 211 more rows
```

```
res2 = ConvToExpoID(PID = res$PID,
                    OutPath = "default")
res2
```

```
## # A tibble: 221 x 8
##    SerialNo FullName GroupName DiseaseID   ExposureID MetabolomeID Prote~1 EX
##    <chr>    <chr>    <chr>     <chr>       <chr>      <chr>        <chr>   <chr>
##  1 Y1       CSNU     disease   OMIM:220100 <NA>       <NA>         <NA>    EX:D~
##  2 Y2       SCZD     disease   OMIM:181500 <NA>       <NA>         <NA>    EX:D~
```

```
##  3 Y3       TGCT     disease   OMIM:273300 <NA>       <NA>         <NA>    EX:D~
##  4 Y4       RASJ     disease   OMIM:604302 <NA>       <NA>         <NA>    EX:D~
##  5 Y5       HH3      disease   OMIM:244200 <NA>       <NA>         <NA>    EX:D~
##  6 Y6       DECRD    disease   OMIM:616034 <NA>       <NA>         <NA>    EX:D~
##  7 Y7       PMDS1    disease   OMIM:261550 <NA>       <NA>         <NA>    EX:D~
##  8 Y8       CRMCC2   disease   OMIM:617341 <NA>       <NA>         <NA>    EX:D~
##  9 Y9       FSHD1    disease   OMIM:158900 <NA>       <NA>         <NA>    EX:D~
## 10 Y10      RA       disease   OMIM:180300 <NA>       <NA>         <NA>    EX:D~
## # ... with 211 more rows, and abbreviated variable name 1: ProteomeID
```

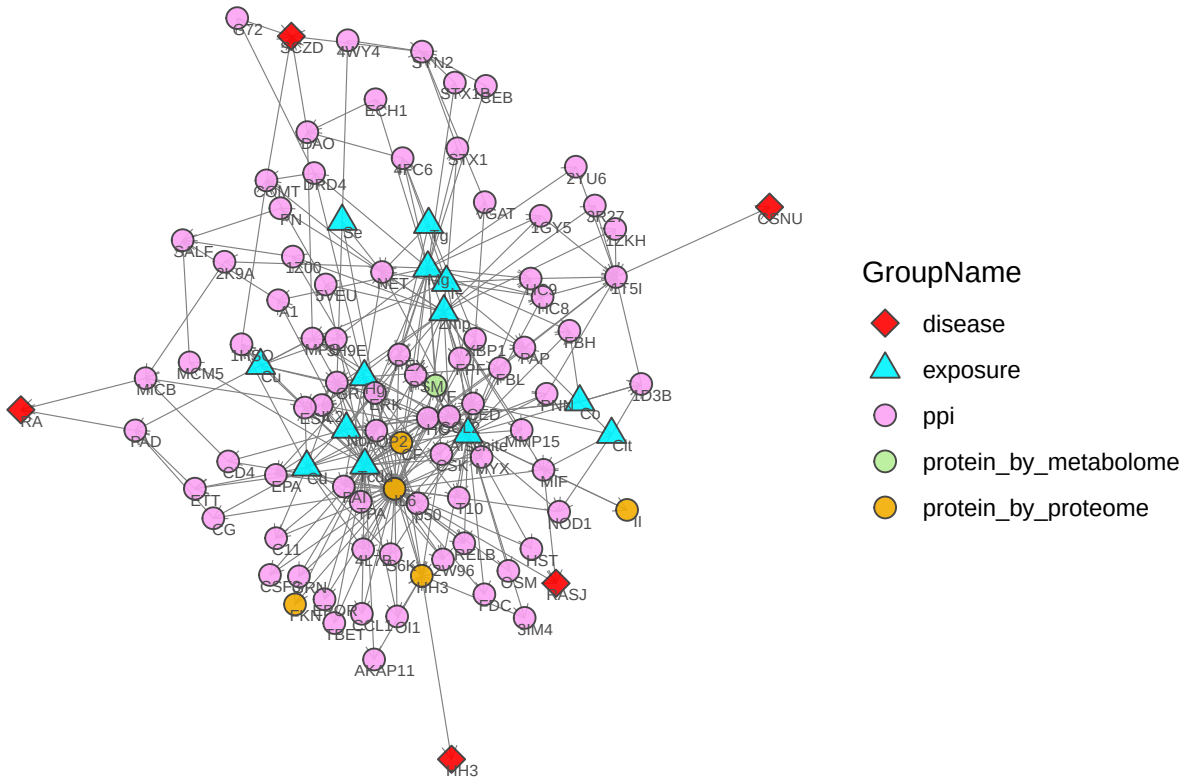
```
res3 = BioLink(PID = res$PID,
               OutPath = "default",
               Mode = "PPI",
               ChemCas = "default",
               ChemInchikey = "default",
               DiseaseID = "default",
               MetabolomeID = "default",
               MetBiospec = "blood",
               ProteomeID = "default")
res3$Edges
```

```
## # A tibble: 310 x 9
##    source    target     intera~1 sourc~2 targe~3 datab~4 edge_~5 sourc~6 targe~7
##    <chr>     <chr>      <chr>    <chr>   <chr>   <chr>     <dbl> <chr>   <chr>
##  1 EX:E00321 EX:P000667 Active   chemic~ protein toxcast       1 EX:E00~ EX:P00~
##  2 EX:E00321 EX:P000608 Active   chemic~ protein toxcast       1 EX:E00~ EX:P00~
##  3 EX:E00321 EX:P000493 Active   chemic~ protein toxcast       1 EX:E00~ EX:P00~
##  4 EX:E00321 EX:P000687 Active   chemic~ protein toxcast       1 EX:E00~ EX:P00~
##  5 EX:E03544 EX:P000608 Active   chemic~ protein toxcast       1 EX:E03~ EX:P00~
##  6 EX:E00321 EX:P000420 Active   chemic~ protein toxcast       1 EX:E00~ EX:P00~
##  7 EX:E47866 EX:P000466 Active   chemic~ protein toxcast       1 EX:E47~ EX:P00~
##  8 EX:E04534 EX:P000298 Active   chemic~ protein toxcast       1 EX:E04~ EX:P00~
##  9 EX:E04534 EX:P166812 Active   chemic~ protein toxcast       1 EX:E04~ EX:P16~
## 10 EX:E00321 EX:P000153 Active   chemic~ protein toxcast       1 EX:E00~ EX:P00~
## # ... with 300 more rows, and abbreviated variable names 1: interaction,
## #   2: source.class, 3: target.class, 4: database, 5: edge_type,
## #   6: source.label, 7: target.label
```

```
res3$Nodes
```

```
## # A tibble: 126 x 3
##    id         label group
##    <chr>      <chr> <chr>
##  1 EX:E00321  Hg    exposure
##  2 EX:P000667 CCL2  ppi
##  3 EX:P000608 PAI   ppi
##  4 EX:P000493 TPA   ppi
##  5 EX:P000687 T10   ppi
##  6 EX:E03544  Tcdd  exposure
##  7 EX:P000420 EPA   ppi
##  8 EX:E47866  Zmp   exposure
##  9 EX:P000466 NET   ppi
## 10 EX:E04534  Tg    exposure
## # ... with 116 more rows
```

```
res4 = VizBioLink(PID = res$PID,
                  OutPath = "default",
                  Mode = 'PPI',
                  Layout = "force-directed",
                  Brightness = "light",
                  Palette = "default1")
res4
```




```
res5 = BioLink(PID = res$PID,
               OutPath = "default",
               Mode = "GO",
               ChemCas = "default",
               ChemInchikey = "default",
               DiseaseID = "default",
               MetabolomeID = "default",
               MetBiospec = "blood",
               ProteomeID = "default")
res5$Edges
```

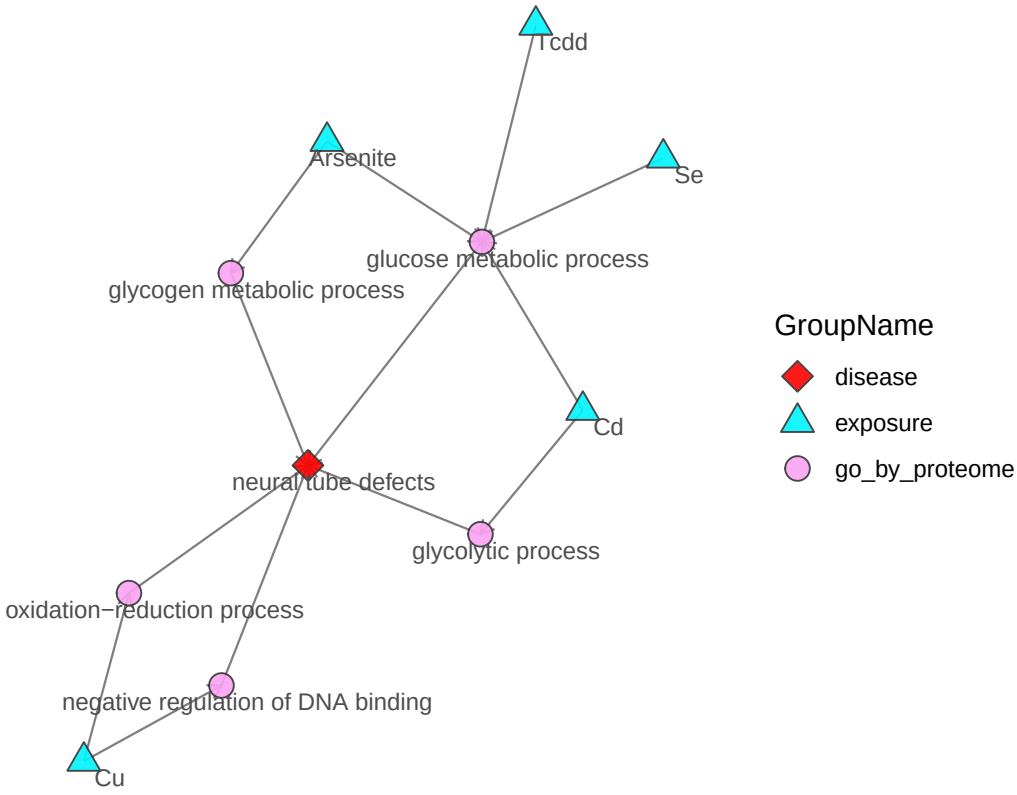
```
## # A tibble: 13 x 7
##    source     target     interaction source.class target.class database edge_t~1
##    <chr>      <chr>      <chr>       <chr>        <chr>        <chr>       <dbl>
##  1 EX:E07343  GO:0006006 association chemical     GO           ctd             1
##  2 EX:E07343  GO:0005977 association chemical     GO           ctd             1
##  3 EX:E26544  GO:0006006 association chemical     GO           ctd             1
##  4 EX:E26544  GO:0006096 association chemical     GO           ctd             1
```

```
##  5 EX:E00033  GO:0043392 association chemical     GO           ctd             1
##  6 EX:E00033  GO:0055114 association chemical     GO           ctd             1
##  7 EX:E38567  GO:0006006 association chemical     GO           ctd             1
##  8 EX:E03544  GO:0006006 association chemical     GO           ctd             1
##  9 GO:0006006 EX:D12164  association GO           disease      ctd             2
## 10 GO:0005977 EX:D12164  association GO           disease      ctd             2
## 11 GO:0006096 EX:D12164  association GO           disease      ctd             2
## 12 GO:0043392 EX:D12164  association GO           disease      ctd             2
## 13 GO:0055114 EX:D12164  association GO           disease      ctd             2
## # ... with abbreviated variable name 1: edge_type
```

```
res5$Nodes
```

```
## # A tibble: 11 x 3
##    id         label                                group
##    <chr>      <chr>                                <chr>
##  1 EX:E07343  Arsenite                              exposure
##  2 GO:0006006 glucose metabolic process            go_by_proteome
##  3 GO:0005977 glycogen metabolic process           go_by_proteome
##  4 EX:E26544  Cd                                   exposure
##  5 GO:0006096 glycolytic process                   go_by_proteome
##  6 EX:E00033  Cu                                   exposure
##  7 GO:0043392 negative regulation of DNA binding   go_by_proteome
##  8 GO:0055114 oxidation-reduction process          go_by_proteome
##  9 EX:E38567  Se                                   exposure
## 10 EX:E03544  Tcdd                                 exposure
## 11 EX:D12164  neural tube defects                  disease
```

```
res6 = VizBioLink(PID = res$PID,
                  OutPath = "default",
                  Mode = 'GO',
                  Layout = "force-directed",
                  Brightness = "dark",
                  Palette = "default1")
res6
```




```
FuncExit(PID = res$PID)
```

```
## [1] "Success to exit. Thanks for using ExposomeX platform!"
```

15 ExpoMeta module

15.1 Application domain

Meta-analyses can be performed when there are multiple scientific studies addressing the same question, with each individual study reporting measurements that are expected to have some degree of error. The aim then is to use approaches from statistics to derive a pooled estimate closest to the unknown common truth based on how this error is perceived. Meta-analytic results are considered the most trustworthy source of evidence by the evidence-based medicine literature.

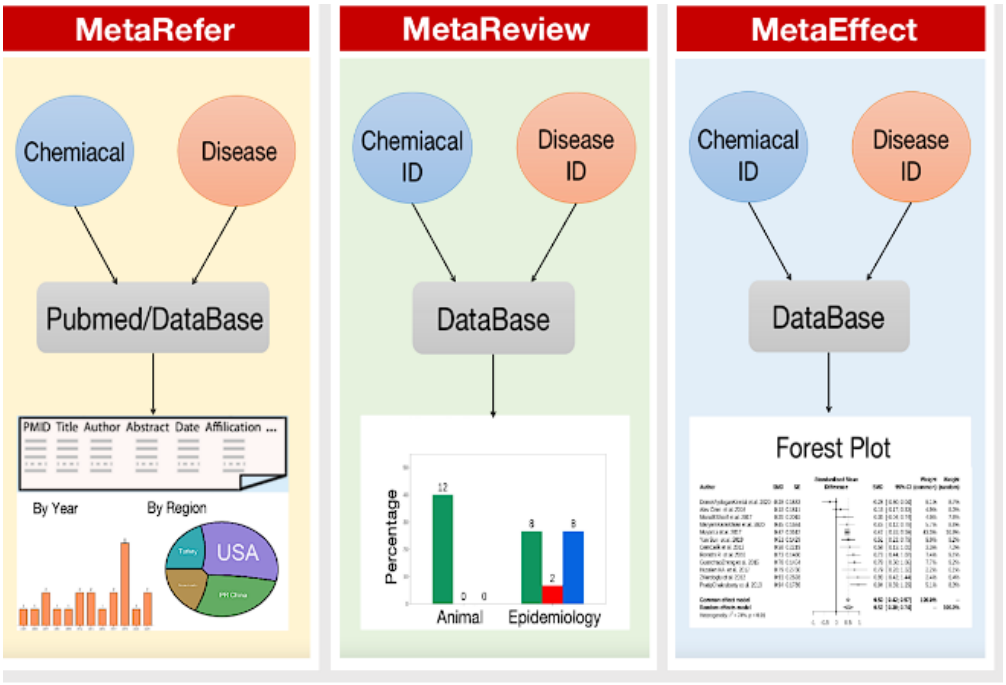
The ExpoMeta module mainly provides users preliminary information retrieval and screening for meta-analysis. It can also summarize the conclusions and effect values of previous studies (available in our meta database), which can help users form a preliminary view of the research topic and further validate the result from the present study. It provides four main functions for users : MetaRefer, VizRef, MetaReview, and MetaEffect.

- MetaRefer: Auto-search all the relative publications from the our meta database and PubMed based on the defined keyword, which has been summarized by the machine learning method and further conveyed to the users.
- VizRefer: VizRefer provides the visual function of papers information searchced or downloaded by `MetaRefer`, showing the year and region distribution directly.
- MetaReview: We have been building a standardized database to summarize the up-to-now knowledge about the relationship between environmental exposure and specific diseases. For each topic, the viewpoint from the animal and epidemiological studies are comprehensively summarized by well-trained researchers.
- MetaEffect: For some issues, if their epidemiological studies have been well-conducted, e.g. association between PM2.5 exposure and mortality, the meta-analysis can be conducted to summarize the effect value [e.g. odds ratio (OR), relative risk (RR), hazardous risk (HR)]. The related publications are also summarized in the same database with the "MetaReiew".

15.2 Theory

The "ExpoMeta" module is based on "A 24-step guide on how to design, conduct, and successfully publish a systematic review and meta-analysis in medical research",which can be summaried by the flow chart below.

```
knitr::include_graphics("@Figures/exmeta1.png")
```



Muka T, Glisic M, Milic J, Verhoog S, Bohlius J, Bramer W, Chowdhury R, Franco OH. A 24-step guide on how to design, conduct, and successfully publish a systematic review and meta-analysis in medical research. Eur J Epidemiol. 2020 Jan;35(1):49-60. doi: 10.1007/s10654-019-00576-5. Epub 2019 Nov 13. PMID: 31720912.

15.3 Work pipeline

Initialize package

Make sure that the required packages is already installed.

```
# The following two packages should be installed in advance
# devtools::install_github("ExposomeX/exmeta", force = TRUE)
# devtools::install_github("ExposomeX/extidy", force = TRUE)

# library(exmeta)
# library(extidy)
library(tidyverse)

# devtools::install_github("ExposomeX/exposomex", force = TRUE)
library(exposomex)
```

At first, you need to initialize the calculation environment using a series of initialization functions, e.g., InitCros, InitMo, InitTidy, InitViz, InitBiolink, etc. Here, we use the package "exmeta" for meta analysis for example. The detailed information about the functions and returned value will be introcuced in the following chapters.

```
res <- InitMeta()
res
```

```
## <eSet>
##   Public:
##     AddCommand: function (x)
##     AddLog: function (x)
```

```
##     clone: function (deep = FALSE)
##     ExcecutionLog: Complete initializing the ExpoMeta module.2022.12.14 15. ...
##     Expo: list
##     FileDirIn: NULL
##     FileDirOut: /home/ubuntu/@changxin/R_Exposome_1.0/output_150303TKYFFP
##     PID: 150303TKYFFP
##     RCommandLog: eSet <- InitMeta(PID = Any ID your like, FileDirOut = An ...
```

Here, we can see that the returned value "res" is an R6 object. It contains an unique program ID of res$PID (e.g., "100737GJMWJA"), which is random generated by the system. Users need to use it in the following step for further data process.

Upload data

After initializing the calculation environment, the second step is to load upload local data file for ExpoMeta Module. `LoadMeta` is provided for this. It has three parameters, PID, UseExample and DataPath. PID is Program ID, which must be the same with the PID generated by `InitMeta`. UseExample is a character indicates whether uses example data for analyses, available option include "example#1" for using example data1 and "default" for using data uploaded. DataPath refer to the input file directory, e.g. "D:/test/eg_meta.xlsx". It should be noted that the slash symbol is `/`, not \. For convenience, here we use example data one for the following step.

```
res1 <- LoadMeta(res$PID,
                 UseExample = "example#1")
```

MetaRefer

MetaRefer provides the functions of paper retrieval and relevance sorting, returning the information to the user based on keywords. Please attention, PID must be got from the return result of InitMeta(). MetaRefer can only run successfully after successfully running InitMeta and LoadMeta functions.

Two modes are provided in MetaRefer. "Search" for paper retrieval by keywords VarX/VarY/VarM/YearFrom/YearEnd, and "Download" for downloading information (main information only) for specified PMID. Set the Mode parameter as you like. OutPath refers to the output file directory, e.g. "D:/output". If "default", the current working directory will be set. It should be noted that the slash symbol is `/`, not \.

VarX/VarY/VarM/YearFrom/YearEnd/PMID parameters can be default" or a character, run `?MetaRefer` to see more details. For convenience, we set them "default" in this example.

```
res2 <- MetaRefer(PID = res$PID,
                  OutPath = "default",
                  Mode = "search",
                  VarX = "default",
                  VarY = "default",
                  VarM = "default",
                  YearFrom = "default",
                  YearEnd = "default",
                  PMID = "default")
```

VizRefer

VizRefer function can visualize the articles' main information after MetaRefer function. Which has only two parameters, PID and OutPath. PID is Program ID, which must be the same with the PID generated by `InitMeta` and OutPath refers to the output file directory.

```
res3 <- VizRefer(PID = res$PID,
                 OutPath = "default")
```

MetaReview

MetaReview provides the functions of literature review. In the publishd papers, how many recorded that X is a protective/risky factor for Y? This questions can be solved by MetaReview function. (It can only search the papers available in our database).

MetaReview has four parameters, PID, CID, DID and OutPath. PID and OutPath are the same as introduced above. CID can be "default" or a chemical ID character (separate different values by ","). If "default", the function will use the Chemical_ID values in the file loaded by LoadMeta. If a character (separate different values by ","), the function will use the chemical ID in the character instead. Chemical_ID refers to the target chemical ID which can be inchikey (eg. JIAARYAFYJHUJI-UHFFFAOYSA-L), cas.rn (eg. 7784-42-1) or our EXC ID (eg. EX:C01631). DID can be "default" or a disease ID character (separate different values by ","). If "default", the function will use the Disease_ID values in the file loaded by LoadMeta. If a character (separate different values by ","), the function will use the disease ID in the character instead. Disease_ID refers to the target disease ID which can be MESH ID (format like MESH:D006973), OMIM ID (format like OMIM:182940) or our EXD ID (eg. EX:D16243).

```
res2 <- MetaReview(PID = res$PID,
                   OutPath = "default",
                   CID = "default",
                   DID = "default")
```

MetaEffect

MetaEffect provides the functions of effect value pooling. In the publishd papers, what is the effect value between X and Y? This questions can be solved by MetaEffect function. It can provide the combined results of fixed effect model and random effect model. (It can only search the papers available in ExpoMeta database DB_Meta).

MetaEffect has four parameters, PID, CID, DID and OutPath, which are all the same as MetaReview.

```
res2 <- MetaEffect(PID = res$PID,
                   OutPath = "default",
                   CID = "default",
                   DID = "default")
```

```
## [[1]]
## TableGrob (2 x 1) "arrange": 2 grobs
##   z     cells    name                 grob
## 1 1 (2-2,1-1) arrange gTree[GRID.gTree.580]
## 2 2 (1-1,1-1) arrange   text[GRID.text.581]
##
## attr(,"class")
## [1] "arrangelist" "list"
```

After all the analysis is done, please run the "FuncExit()" function to delete the data uploaded to the server.

```
FuncExit(PID = res$PID)
```

```
## [1] "Success to exit. Thanks for using ExposomeX platform!"
```