LUCIDus: An R Package For Implementing Latent Unknown Clustering By Integrating Multi-omics Data (LUCID) With Phenotypic Traits

by Yinqi Zhao, Qiran Jia, Jesse A. Goodrich, David V. Conti

Abstract Many studies are leveraging current technologies to obtain multiple omics measurements on the same individuals. These measurements are usually cross-sectional, and methods developed and commonly used focus on omic integration at a single time point. More unique, and a growing area of interest, are studies that leverage biology or the temporal sequence of measurements to relate long-term exposures or germline genetics to intermediate measures capturing transitional processes that ultimately result in an outcome. In this context, we have previously introduced an integrative model named Latent Unknown Clustering by Integrating multi-omics Data (LUCID) aiming to distinguish unique effects of environmental exposures or germline genetics and informative omic effects while jointly estimating subgroups of individuals relevant to the outcome of interest. This multiple omics analysis consists of early integration (concatenation of omic layers to estimate common subgroups); intermediate integration (omic-layer-specific estimation of subgroups that are all related to the outcome); and late integration (omic-layer-specific estimation of subgroups that are then interrelated by a priori structures). In this article, we introduce LUCIDus version 3, an R package to implement the LUCID model. We review the statistical background of the model and introduce the workflow of LUCIDus, including model fitting, model selection, interpretation, inference, and prediction. Throughout, we use a realistic but simulated dataset based on an ongoing study, the Human Early Life Exposome Study (HELIX), to illustrate the workflow.

1 Introduction

The rapid advancement in high-throughput technology has made it possible to measure multiple omics (referred to as "multi-omics") data on the same person in many cohort studies. These datasets include DNA genome sequences (Goodwin et al., 2016), RNA expression data (Ozsolak and Milos, 2011), metabolites in biofluids (Beger, 2013), proteins in cells or tissues (Aslam et al., 2017), as well as chemical exposures in the environment (Wild, 2005). For example, the Human Early-Life Exposome project (HELIX) measured molecular omics signatures including RNA expression data, metabolites, plasma proteins, etc. from 1300 children at the age of 6-11 in six European countries (Vrijheid et al., 2014). Guided by biology or the temporal sequence of measurements, these studies often share a common structure that relates germline genetics or environmental exposures to intermediate factors capturing transitional processes that ultimately result in an outcome. While these suspected causal pathways can be measured with current omic technology, the analysis often focuses on cross-sectional relationships using cluster analysis or models focused on feature selection. This is in part due to the analytic and computational challenges resulting from the complex structure of the data collected from various sources, limited sample sizes, and the high dimensionality of multi-omics information (Tini et al., 2019). Existing methods do not consider risk factors that precede the omic measurements and link those factors with disease or trait outcomes while using multi-omics data to characterize the underlying mechanism. Thus, there is a great need to develop and easily implement approaches leveraging these mediating relationships or latent structures for the integration of multi-omics data (Subramanian et al., 2020). Conventional analysis tools range from clustering approaches to variable selection approaches (González and Cáceres, 2019). Clustering is a fundamental method for analyzing both single omic and multi-omics data (Rappoport and Shamir, 2018). Clustering aims to divide observations into several groups (called clusters) so that observations within a group are similar while samples in different groups are dissimilar. Clustering has many important applications in the field of biological science and medicine, such as defining gene sets (Hejblum et al., 2015), identifying subtypes of cancer, and performing diagnostic predictions (Curtis et al., 2012; Khan et al., 2001), defining cell types in flow cytometry and scRNA-seq experiments (Chan et al., 2008; Hejblum et al., 2019; Prabhakaran et al., 2016), and estimating protein localization (Crook et al., 2018). Conventional clustering methods applied to multi-omics data either concatenate multiple datasets (often called early integration) or analyze each dataset independently (i.e., late integration). However, both approaches fall short of adequately capturing the variation across omic levels or reflecting the heterogeneity of the integrated datasets. To overcome these limitations, several specific integrated clustering methods for analyzing multi-omics data are available in the community. Pierre-Jean et al. (2020) conducted a thorough review comparing 13 unsupervised methods for multi-omics data integration via clustering. Methods include SGCCA (Tenenhaus et al., 2014), SNF (Wang et al., 2014), and iCluster plus (Mo et al., 2013), etc. Building upon the foundation of Principal Component Analysis (PCA), Joint and Individual Variation Explained (JIVE) conducts integrated clustering of multi-omics data via a variance decomposition framework to summarize information across multiple omic layers (Lock et al., 2013). Within the Bayesian framework, Kirk et al. (2012) proposed an unsupervised approach, Bayesian correlated clustering, that simultaneously models each omic layer via a mixture model while considering the correlation between each model. In a similar fashion, Bayesian consensus clustering can flexibly describe the dependency and the heterogeneity of the multi-omics data (Lock and Dunson, 2013). An alternative approach to clustering includes dimension reduction or variable selection to reduce noise, improve model interpretability, and avoid problems with overfitting. For example, the least absolute shrinkage and selection operator (LASSO) induces sparsity in model coefficients based on L_1 norm regularization (Tibshirani, 1996). An extension of LASSO, Group LASSO, allows for variable selection on both individual and pre-specified grouped variables (Yuan and Lin, 2006). L_1 norm regularization is integrated into many clustering approaches to reduce data dimensionality, such as SGCCA and iCluster plus. Extensions of variable selection approaches exist for mediation as well and include both high-dimensional and latent variable approaches. HIMA first implements pre-screening of the high-dimensional mediators and then utilizes an LASSO-type penalty to obtain a sparse solution (Zhang et al., 2016), while BAMA is a Bayesian shrinkage approach that employs continuous shrinkage priors on the key coefficients to select active mediators with large effects only (Song et al., 2020). Both Albert et al. (2016) and Derkach et al. (2019) proposed statistical models within the causal mediation framework that summarize the information from high-dimensional multi-omics data into a latent variable to facilitate interpretation. In the context of linking multi-omics to health outcomes and identifying key omic features that drive the outcomes, Singh et al. (2019) proposed Data Integration Analysis for Biomarker discovery using Latent cOmponents (DIABLO), which extends the unsupervised sGCCA to a supervised framework. DIABLO achieves summarizing common information across different omic layers and conducting variable selection while discriminating the outcome of interest. Nonetheless, DIABLO fails to take into account the impact of risk factors related to the outcome and their interplay with multi-omics data. Finally, Peng et al. (2020) proposed a model called Latent Unknown Clustering by Integrating multi-omics Data (LUCID) that incorporates both clustering and variable selection to distinguish unique effects of germline genetics or environmental exposures and informative omic effects while jointly estimating subgroups of individuals relevant to the outcome of interest. LUCID is a novel 'quasi-mediation' approach that uses latent variable analysis to estimate subgroups of individuals characterized by key omic factors and with differential associations to the outcome and exposures. They demonstrated the performance of LUCID through extensive simulation studies and real data applications to highlight the integration of genomic, exposomic, and metabolomic data. **LUCIDus**, the R package to implement the LUCID model, provides an integrated clustering framework and has numerous downloads (around 19,000 times since it was first introduced according to **dlstats** (Yu, 2022)). It has also been applied in several environmental epidemiological studies (Jin et al., 2020; Stratakis et al., 2020; Matta et al., 2022). In this paper, we introduce **LUCIDus** version 3, a major update and enhancement from the original release (Jia et al., 2024). To account for the heterogeneity of the integrated datasets based on the study design, **LUCIDus** version 3 incorporates three integration strategies into the LUCID framework: (1) Early integration (Figure 1 (a)) concatenates all multi-omics data matrices into a single matrix prior to model estimation; (2) Intermediate integration (Figure 1 (b)) estimates latent clusters for each omic layer, and the resulting clusters are then modeled in parallel in a joint model linking exposures and the outcome; and (3) Late integration (Figure 1 (c)) in which each omic layer is used to estimate omic-specific clusters that are then linked via *apriori* relationships with each other and the exposure and outcome. **LUCIDus** version 3 also includes model selection, model visualization, and inference based on bootstrap resampling. It also incorporates an integrated imputation approach to deal with missingness in multi-omics data. The paper is organized as follows: we first briefly introduce the statistical model of LUCID; we then go through the details of functions in the **LUCIDus** package; we illustrate the workflow of fitting LUCID models by using a dataset simulated based on real cases from the HELIX study (Vrijheid et al., 2014; Maitre et al., 2022).

2 Model and software

2.1 Overview of the original LUCID model (early integration)

In the LUCID model, genomic/exposomic exposures $\boldsymbol{G}$, other multi-omics data $\boldsymbol{Z}$, and phenotype trait $\boldsymbol{Y}$ are integrated through a latent categorical variable $\boldsymbol{X}$. To model the complex correlation structure between each layer of the multi-omics data, we propose three types of LUCID models based on

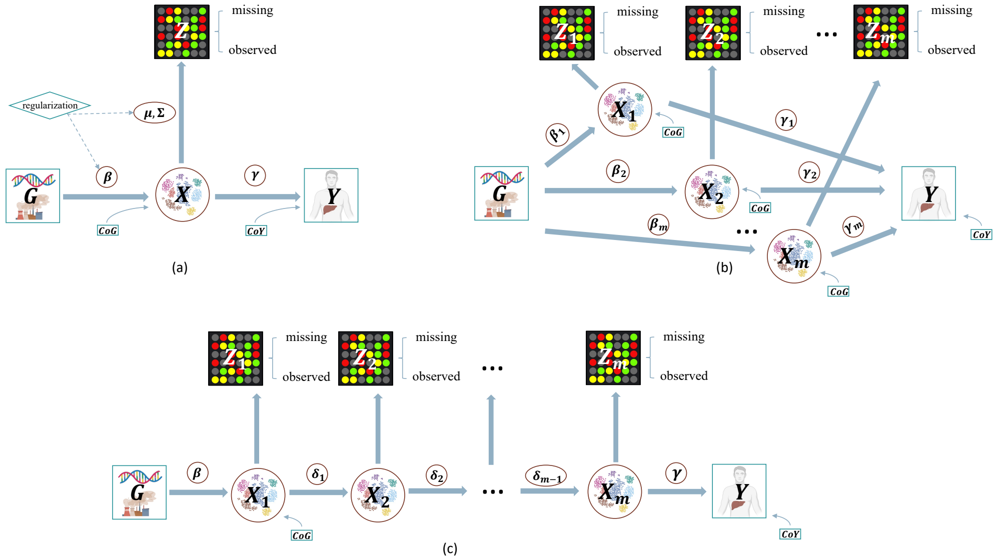


Figure 1: Three DAGs represent how three different types of the LUCID model integrate genetic/environmental exposures ($\boldsymbol{G}$), other multi-omics data ($\boldsymbol{Z}$), and the phenotype trait ($\boldsymbol{Y}$). (a) LUCID early integration; (b) LUCID in parallel; (c) LUCID in serial. The squares represent observed data, the circles represent unobserved latent variables (clusters) and model parameters, and the diamond refers to L_1 penalty terms for regularization for (a). $\boldsymbol{CoG}$ and $\boldsymbol{CoY}$ represent covariates to be adjusted in the LUCID model. Missingness is allowed in multi-omics data. $\boldsymbol{Z}$ is divided into subsets of observations with complete measurements and observations with missingness. For (b) and (c), $\boldsymbol{Z}$ is partitioned into m layers.

different integration strategies: (1) LUCID early integration; (2) LUCID in parallel or intermediate integration; and (3) LUCID in serial or late integration. Figure 1 illustrates the joint relationship between $\boldsymbol{G}$, $\boldsymbol{Z}$, $\boldsymbol{X}$, and $\boldsymbol{Y}$ for each LUCID model. We will discuss LUCID in parallel and LUCID in serial in detail under Sections LUCID in Parallel and LUCID in Serial as two extensions, and here we focus on LUCID early integration. Because $\boldsymbol{X}$ is an unobserved categorical variable, each category of $\boldsymbol{X}$ is interpreted as a latent cluster in the data, jointly defined by $\boldsymbol{G}$, $\boldsymbol{Z}$, and $\boldsymbol{Y}$. Let $\boldsymbol{G}$ be a $N \times P$ matrix with columns representing genetic/environmental exposures and rows representing the observations; $\boldsymbol{Z}$ be a $N \times M$ matrix of omics data (for example, gene expression data, DNA methylation profiles, and metabolomic data, etc.) and $\boldsymbol{Y}$ be a N-length vector of phenotype traits. We further assume $\boldsymbol{G}$, $\boldsymbol{Z}$, and $\boldsymbol{Y}$ are measured through a prospective sampling procedure, so we do not model the distribution of $\boldsymbol{G}$. All three measured components ($\boldsymbol{G}$, $\boldsymbol{Z}$, and $\boldsymbol{Y}$) are linked by a latent variable $\boldsymbol{X}$ consisting of K categories. The directed acyclic graph (DAG) in Figure 1 (a) implies that the distributions of $\boldsymbol{X}$ given $\boldsymbol{G}$, $\boldsymbol{Z}$ given $\boldsymbol{X}$, and $\boldsymbol{Y}$ given $\boldsymbol{X}$ are conditionally independent of each other. Let $f(\cdot)$ denote the probability mass functions (PMFs) for categorical random variables or the probability density functions (PDFs) for continuous random variables. The joint log-likelihood of the LUCID model is constructed as:

$$\begin{aligned}\log L(\boldsymbol{\Theta}) &= \sum_{i=1}^{N} \log f(\boldsymbol{Z}_i, Y_i|\boldsymbol{G}_i; \boldsymbol{\Theta}) \\ &= \sum_{i=1}^{N} \log \sum_{j=1}^{K} f(X_i = j|\boldsymbol{G}_i; \boldsymbol{\Theta}) f(\boldsymbol{Z}_i|X_i = j; \boldsymbol{\Theta}) f(Y_i|X_i = j; \boldsymbol{\Theta})\end{aligned} \tag{1}$$

where $\boldsymbol{\Theta}$ is a generic notation for all parameters in the LUCID model. Since $\boldsymbol{X}$ is a discrete variable with K categories, we assume $\boldsymbol{X}$ follows a multinomial distribution conditioning on $\boldsymbol{G}$, denoted by the softmax function $S(\cdot)$. We assume omics data $\boldsymbol{Z}$ follows a multivariate Gaussian distribution conditioning on $\boldsymbol{X}$, denoted by $\phi(\boldsymbol{Z}|\boldsymbol{X} = j; \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$, where $\boldsymbol{\mu}_j$ and $\boldsymbol{\Sigma}_j$ are cluster-specific means and variance-covariance matrices for $\boldsymbol{Z}$. For illustrative purposes, we assume $\boldsymbol{Y}$ is a continuous outcome following a univariate Gaussian distribution, denoted by $\phi(\boldsymbol{Y}|\boldsymbol{X} = j; \gamma_j, \sigma_j^2)$ (γ_j and σ_j^2 are also interpreted as the cluster-specific means and variances for the effect of $\boldsymbol{X}$ on $\boldsymbol{Y}$). Similar development for a binary outcome is detailed in (Peng et al., 2020). Because the latent cluster $\boldsymbol{X}$ is unobserved, the maximum likelihood estimates (MLE) of the parameters associated with the LUCID model based on Equation 1 are not readily estimated. Therefore, we use the Expectation-Maximization (EM) algorithm to obtain the MLE of model parameters. We define $I(X_i = j)$ as an indicator function representing that observation i belongs to latent cluster j. Then, the log-likelihood function of model parameters in Equation 1 becomes:

$$\log L(\boldsymbol{\Theta}) = \sum_{i=1}^{N}\sum_{j=1}^{K} I(X_i = j)\left(\log S(X_i = j|\boldsymbol{G}_i;\boldsymbol{\Theta}) + \log\phi(\boldsymbol{Z}_i|X_i = j;\boldsymbol{\Theta}) + \log\phi(Y_i|X_i = j;\boldsymbol{\Theta})\right) \tag{2}$$

We denote the observed data as $\boldsymbol{D} = \{\boldsymbol{G},\boldsymbol{Z},\boldsymbol{Y}\}$, and define the responsibility r as the inclusion probability (IP) of belonging to the latent cluster j given observed data and current estimation of model parameters at iteration t, which is:

$$\begin{aligned} r_{ij}^{(t)} &= P(X_i = j|\boldsymbol{D};\boldsymbol{\Theta}^{(t)}) \\ &= \frac{S\left(X_i = j|\boldsymbol{G}_i;\boldsymbol{\beta}^{(t)}\right)\phi\left(\boldsymbol{Z}_i|X_i = j;\boldsymbol{\mu}_j^{(t)},\boldsymbol{\Sigma}_j^{(t)}\right)\phi\left(Y_i|X_i = j;\gamma_j^{(t)},\sigma_j^{2(t)}\right)}{\sum_{j=1}^{K} S\left(X_i = j|\boldsymbol{G}_i;\boldsymbol{\beta}^{(t)}\right)\phi\left(\boldsymbol{Z}_i|X_i = j;\boldsymbol{\mu}_j^{(t)},\boldsymbol{\Sigma}_j^{(t)}\right)\phi\left(Y_i|X_i = j;\gamma_j^{(t)},\sigma_j^{2(t)}\right)} \end{aligned} \tag{3}$$

At each iteration t, in the E-step, we compute the expectation of the complete data likelihood, which is:

$$\begin{aligned} Q(\boldsymbol{\Theta}|\boldsymbol{D},\boldsymbol{\Theta}^{(t)}) = &\sum_{i=1}^{N}\sum_{j=1}^{K} r_{ij}\log S\left(X_i = j|\boldsymbol{G}_i;\boldsymbol{\beta}\right) + \sum_{i=1}^{N}\sum_{j=1}^{K} r_{ij}\log\phi(\boldsymbol{Z}_i|X_i = j;\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j) \\ &+ \sum_{i=1}^{N}\sum_{j=1}^{K} r_{ij}\log\phi\left(Y|X_i = j;\gamma_j,\sigma_j^2\right) \end{aligned} \tag{4}$$

In the M-step, we update parameter estimates by maximizing Equation 4 in terms of $\boldsymbol{\Theta}$, which results in the following:

$$\boldsymbol{\beta}^{(t+1)} = \arg\max_{\boldsymbol{\beta}} \sum_{i=1}^{N}\sum_{j=1}^{K} r_{ij}^{(t)}\log S(X_i = j|\boldsymbol{G}_i;\boldsymbol{\beta}_j) \tag{5}$$

$$\boldsymbol{\mu}_j^{(t+1)} = \frac{\sum_{i=1}^{N} r_{ij}^{(t)}\boldsymbol{Z}_i}{\sum_{i=1}^{N} r_{ij}^{(t)}} \tag{6}$$

$$\boldsymbol{\Sigma}_j^{(t+1)} = \frac{\sum_{i=1}^{N} r_{ij}^{(t)}\left(\boldsymbol{Z}_i - \boldsymbol{\mu}_j^{(t+1)}\right)\left(\boldsymbol{Z}_i - \boldsymbol{\mu}_j^{(t+1)}\right)^T}{\sum_{i=1}^{N} r_{ij}^{(t)}} \tag{7}$$

$$\gamma_j^{(t+1)} = \frac{\sum_{i=1}^{N} r_{ij}^{(t)} Y_i}{\sum_{i=1}^{N} r_{ij}^{(t)}} \tag{8}$$

$$\sigma_j^{2(t+1)} = \frac{\sum_{i=1}^{N} r_{ij}^{(t)}\left(Y_i - \gamma_j^{(t+1)}\right)^2}{\sum_{i=1}^{N} r_{ij}^{(t)}} \tag{9}$$

Although maximization of $\boldsymbol{\beta}_j^{(t+1)}$ in Equation 5 does not have a closed-form solution, it is equivalent to fitting a multinomial logistic regression by treating r_{ij} as the outcome and $\boldsymbol{G}_i$ as the exposure. For **LUCIDus**, we include a more flexible geometric feature of latent clusters, such as volume, shape, and orientation determined by $\boldsymbol{\Sigma}_j$. We use the parameterization of covariance matrices by means of an eigenvalue decomposition in the form below (Banfield and Raftery, 1993):

$$\boldsymbol{\Sigma}_j = \lambda_j\boldsymbol{D}_j\boldsymbol{A}_j\boldsymbol{D}_j^T \tag{10}$$

where λ_j is a scalar, $\boldsymbol{D}_j$ is the orthogonal matrix of eigenvectors, and $\boldsymbol{A}_j$ is a diagonal matrix whose values are proportional to eigenvalues. A detailed discussion of maximizing $\boldsymbol{\Sigma}_j$ parameterized by Equation 10 is provided by Celeux and Govaert (1995). Their algorithm is implemented in the R package **mclust** (Scrucca et al., 2016), and we leverage **mclust** to update $\boldsymbol{\Sigma}_j$ in the M-step at each iteration of the EM algorithm for LUCID.

2.2 General workflow of LUCIDus

The **LUCIDus** package includes five main functions and two auxiliary functions to implement the analysis framework based on LUCID. Brief descriptions of each function are listed in Table 1. The workflow of the **LUCIDus** package is shown in Figure 2. Below we describe the typical workflow

Main function	Description
`lucid()`	Main function to fit LUCID models, specified by giving integrated data, a distribution of the outcome, and the type of the LUCID model (early, parallel, serial). It also conducts model selection and variable selection for LUCID early integration and model selection for candidate models with different numbers of latent clusters for LUCID in parallel and LUCID in serial.
`summary()`	S3 method for LUCID. Create tables to summarize a LUCID model.
`plot()`	S3 method for LUCID. Visualize LUCID models through a Sankey diagram.
`boot_lucid()`	Derive confidence intervals based on bootstrap resampling.
`predict_lucid()`	Predict latent cluster assignment and outcome using integrated data.

Workhorse function	Description
`estimate_lucid()`	Fit a LUCID model to estimate latent clusters by using integrated data.
`tune_lucid()`	Fits a series of LUCID models over different combinations of tuning parameters and determines an optimal model with the minimum BIC.

Table 1: Functions in the **LUCIDus** package. `lucid()` calls `estimate_lucid()` and `tune_lucid()` in the backend. The two workhorse functions are not normally called directly, but they can be useful when a user wants to examine the model fitting process in more detail.

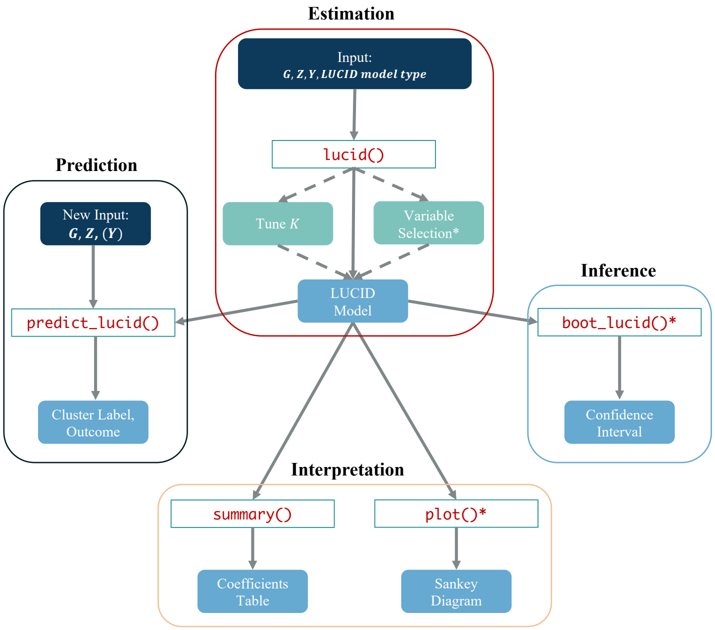


Figure 2: The workflow of the **LUCIDus** package. Dark blue nodes represent input data, light blue nodes represent output results. Green nodes and dashed arrows are optional steps for model estimation. Red texts correspond to 5 key functions in **LUCIDus**. Steps and functions marked with an asterisk currently work for LUCID early integration only.

of analyzing integrated data using the LUCID early integration model. The function `lucid()` is the primary function in the package, which fits a LUCID model based on an exposure matrix (argument `G`), multi-omics data (argument `Z`), outcome data (argument `Y`), the number of latent clusters (`K`; default is 2), the type of the LUCID model (argument `lucid_model`; here it is '`early`' as we focus on the LUCID early integration model in this section), and the family of the outcome (argument `family`; default is '`normal`'). If a vector of `K` and/or L_1 penalties are supplied, `lucid()` will automatically conduct model selection on the number of clusters K, select informative variables in $\boldsymbol{G}$ or $\boldsymbol{Z}$, or both, and return a LUCID early integration model (an R object of class '`lucid_early`') with optimal K and selected variables in $\boldsymbol{G}$ and/or $\boldsymbol{Z}$. Several additional functions can then be applied to a fitted LUCID object. `summary()` summarizes the fitted LUCID model by producing summary tables of parameter estimation and interpretation. Visualization is performed via the `plot()` function to create a Sankey diagram showing the interplay among the three components ($\boldsymbol{G}$, $\boldsymbol{Z}$, and $\boldsymbol{Y}$). In addition, statistical inference can be accomplished for LUCID by constructing confidence intervals (CIs) based on bootstrap resampling. This is achieved by the function `boot_lucid()`. Finally, predictions on the cluster assignment and the outcome can be obtained by calling the function `predict_lucid()`. In practice, it might not be necessary to implement the entire workflow above. For instance, if we have prior knowledge of the number of latent clusters K, model selection for the number of clusters can be skipped. If a given dataset has limited variables (for example, variables selected based on biological annotations), then variable selection may not be necessary. Note that for a LUCID in parallel model or a LUCID in serial model (the argument `lucid_model` is '`parallel`' or '`serial`', respectively), specification of `K` is different, and the features of variable selection, constructing confidence intervals (CIs) based on bootstrap resampling, and plotting are not yet available (see Sections LUCID in Parallel and LUCID in Serial). In the following sections, we show example code and demonstrate the usage of the LUCID early integration model using the simulated HELIX data.

3 Illustration

3.1 Fitting a LUCID model by `lucid()`

To illustrate integrative clustering with the LUCID early integration model, we use simulated data based on real cases from the HELIX study based on the correlation structure. A subset of the simulated data is incorporated in the **LUCIDus** package to replicate the workflow presented here. The dataset is a list of `dataframe` containing data from 420 children, with 1 variable measuring maternal exposure to utero mercury (referred to as the exposome), 10 methylomes measured in children (referred to as the methylomics), 10 transcriptomes measured in children (referred to as the transcriptomics), 10 miRNA measured in children (referred to as the miRNA), childhood cytokeratin 18 level (referred to as ck18, a continuous outcome as an indicator of metabolic-dysfunction-associated fatty liver disease (MAFLD), childhood cytokeratin 18 category (a binary outcome referred to as ck18_cat) and 2 covariates, including child's sex and child's age. To organize the data into separate `dataframes` for each type of data and to better illustrate functionality within **LUCIDus**, we use the following code:

```
> library(LUCIDus)
> # load data
> data("simulated_HELIX_data")
> simulated_data <- simulated_HELIX_data
> exposome <- as.matrix(simulated_data[["phenotype"]]$hs_hg_m_scaled)
> colnames(exposome) <- "hs_hg_m_scaled"
> methylomics <- simulated_data$methylome
> ck18 <- as.matrix(simulated_data[["phenotype"]]$ck18_scaled)
> colnames(ck18) <- "ck18_scaled"
> ck18_cat <- ifelse(ck18 > mean(ck18), 1, 0)
> covars <- c("hs_child_age_yrs_None","e3_sex_None")
> covs <- simulated_data[["phenotype"]][covars]
> covs$e3_sex_None <- ifelse(covs$e3_sex_None == "male", 1, 0)
```

Our goal is to conduct an integrated clustering of the exposome and methylome and to relate the estimated latent clusters to the childhood cytokeratin 18 level. The LUCID model is fitted using the function `lucid()`. The input data are specified by arguments `G`, `Z`, and `Y`, corresponding to the exposome, the methylome, and the childhood cytokeratin 18 level, respectively. In practice, scaling of multi-omics data is highly recommended to obtain more stable estimates; the methylomics data used in this example are already scaled. If multi-omics data are included in the LUCID analysis as the omic intermediate $\boldsymbol{Z}$, the user can concatenate them one by one into a single data matrix and then input the concatenated matrix as $\boldsymbol{Z}$ for early integration. For illustrative purposes, we assume that, in the example data, the optimal number of latent clusters is 2 (determining the optimal number of clusters is a major question prior to performing clustering analysis; more details are discussed in the later Section Model selection and variable selection). For illustration, we fit two LUCID models with continuous outcome ck18 and binary outcome ck18_cat, respectively. The parameter `family` specifies the outcome type. The default setting is '`normal`', corresponding to a continuous outcome. For a binary outcome, `family` should be specified as '`binary`'. `lucid()` returns an object of class '`lucid_early`' providing the MLE of the LUCID model as well as IPs.

```
> # Fit a LUCID model with a continuous outcome
> fit1 <- lucid(G = exposome, Z = methylomics, Y = ck18, init_omic.data.model = NULL,
+               lucid_model = "early", family = "normal", K = 2)
> # MLE of the LUCID model
> # fit1$res_Beta
> # fit1$res_Mu
> # fit1$res_Sigma
> # fit1$res_Gamma
> # IPs of the sample
> # fit1$inclusion.p
> # Fit LUCID model with a binary outcome
> fit2 <- lucid(G = exposome, Z = methylomics, Y = ck18_cat, init_omic.data.model = NULL,
+               lucid_model = "early", family = "binary", K = 2)
```

Initializing the parameters of LUCID

$\boldsymbol{\beta}$ is initiated by randomly drawing from a uniform distribution. By default, `init_par` is '`mclust`' and `init_omic.data.model` is '`EEV`', so `lucid()` calls `Mclust()` to fit the mixture model of '`EEV`' on omics data $\boldsymbol{Z}$ and use the mixture model estimates to initiate $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$, and then implement regression to initiate γ. Alternatively, the user can also specify `init_omic.data.model` to choose a certain mixture model. The available mixture models are listed in `mclust::mclustModelNames` (Scrucca et al., 2016). Note that `init_omic.data.model` can be '`NULL`' to let `Mclust()` automatically choose the optimal mixture model. If not using `Mclust()` for initiation, the user can also specify `init_par` to be '`random`' to initiate $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ by randomly drawing them from a uniform distribution and then implement regression to initiate γ. We recommend using the default setting to initiate the parameters by calling `Mclust()` for quick convergence and stable performance.

```
> # Fit LUCID model with spherical shape, equal volume
> fit3.1 <- lucid(G = exposome, Z = methylomics, Y = ck18,
+                 lucid_model = "early", family = "normal", K = 2,
+                 init_par = "mclust",init_omic.data.model = "EII")
> # Fit LUCID model with random guess
> fit3.2 <- lucid(G = exposome, Z = methylomics, Y = ck18,
+                 lucid_model = "early", family = "normal", K = 2,
+                 init_par = "random")
> # Fit LUCID model with ellipsoidal shape, varying volume, shape, and orientation
> fit4 <- lucid(G = exposome, Z = methylomics, Y = ck18,
+               lucid_model = "early", family = "normal", K = 2,
+               init_omic.data.model = "VVV")
```

Supervised LUCID versus unsupervised LUCID

In Equation 1, the latent clusters $\boldsymbol{X}$ are jointly defined by genomic/environmental exposure $\boldsymbol{G}$, other multi-omics data $\boldsymbol{Z}$, and outcome $\boldsymbol{Y}$. Because the outcome is explicitly incorporated in the likelihood function, the estimation process based on Equation 1 is similar to a supervised learning process. **LUCIDus** also allows for an unsupervised version of LUCID. In this unsupervised LUCID, the latent clusters are estimated only by $\boldsymbol{G}$ and $\boldsymbol{Z}$. Specifically, the joint likelihood of unsupervised LUCID is written as

$$\log L(\boldsymbol{\Theta}) = \sum_{i=1}^{N}\sum_{j=1}^{K} I(X_i = j)\left(\log S(X_i = j|\boldsymbol{G}_i;\boldsymbol{\Theta}) + \log\phi(\boldsymbol{Z}_i|X_i = j;\boldsymbol{\Theta})\right) \tag{11}$$

and the corresponding responsibility based on Equation 11 is derived as

$$r_{ij}^{(t)} = \frac{S\left(X_i = j|\boldsymbol{G}_i;\boldsymbol{\beta}^{(t)}\right)\phi\left(\boldsymbol{Z}_i|X_i = j;\boldsymbol{\mu}_j^{(t)},\boldsymbol{\Sigma}_j^{(t)}\right)}{\sum_{j=1}^{K} S\left(X_i = j|\boldsymbol{G}_i;\boldsymbol{\beta}^{(t)}\right)\phi\left(\boldsymbol{Z}_i|X_i = j;\boldsymbol{\mu}_j^{(t)},\boldsymbol{\Sigma}_j^{(t)}\right)} \tag{12}$$

Estimations for unsupervised LUCID are also obtained by the EM algorithm discussed previously. The parameter `useY` in `lucid()` is a flag indicating a supervised or unsupervised LUCID model. By default, `useY = TRUE`, and `lucid()` fits supervised LUCID. To fit an unsupervised LUCID model, a user should set `useY = FALSE`.

```
> # Fit an unsupervised LUCID model
> fit5 <- lucid(G = exposome, Z = methylomics, Y = ck18, lucid_model = "early",
+               family = "normal", K = 2, useY = FALSE)
```

The choice of a supervised LUCID or unsupervised LUCID model depends both on the study design and the research question. A supervised LUCID analysis may be appropriate, for example, if: (1) there is the belief that the cluster structure is defined jointly by $\boldsymbol{G}$, $\boldsymbol{Z}$, and $\boldsymbol{Y}$ (for example, data collected from a cross-sectional design); or (2) the goal is to build a predictive model with data from one cohort to then apply directly to another cohort. If the aim is to obtain an unbiased estimate for the association between the latent cluster $\boldsymbol{X}$ and the outcome $\boldsymbol{Y}$, an unsupervised LUCID model is more appropriate.

Adjusting for covariates

LUCIDus also allows for the adjustment of covariates. According to the DAG in Figure 1 (a), covariates may be included for the association between the exposure and the latent cluster (referred to as $\boldsymbol{G}$ to $\boldsymbol{X}$ covariate) or for the association between the latent cluster and the outcome (referred to as $\boldsymbol{X}$ to $\boldsymbol{Y}$ covariate). Covariates in the $\boldsymbol{G}$ to $\boldsymbol{X}$ relationship act more like predictors of the latent cluster while covariates in the $\boldsymbol{X}$ to $\boldsymbol{Y}$ relationship may be interpreted more in the context of an adjustment for a potential confounding effect. A variable can serve as both a $\boldsymbol{G}$ to $\boldsymbol{X}$ covariate and a $\boldsymbol{X}$ to $\boldsymbol{Y}$ covariate. In the `lucid()` function, $\boldsymbol{G}$ to $\boldsymbol{X}$ covariates are specified by parameter `CoG` and $\boldsymbol{X}$ to $\boldsymbol{Y}$ covariates are specified by parameter `CoY`.

```
> # Include covariates as G to X covariates
> fit6 <- lucid(G = exposome, Z = methylomics, Y = ck18, lucid_model = "early", K = 2,
+               family = "normal", CoG = covs)
> # Include covariates as X to Y covariates
> fit7 <- lucid(G = exposome, Z = methylomics, Y = ck18, lucid_model = "early", K = 2,
+               family = "normal", CoY = covs)
```

3.2 Interpreting LUCID

Since LUCID is an integrated analysis framework consisting of clustering, regression, and multiple data components, we provide two utility functions, `summary()` and `plot()` to summarize the results of LUCID and to facilitate interpretation.

Summarizing LUCID in tables by `summary()`

A direct call of `summary()` with the input of a model returned by `lucid()` prints three tables in the console, corresponding to associations among $\boldsymbol{G}$, $\boldsymbol{Z}$, and $\boldsymbol{Y}$. We take the LUCID model fitted with the continuous BMI as an example.

```
> # summarize a simple lucid model with a continuous outcome
> summary(fit1)
----------Summary of the LUCID Early Integration model----------
K =  2 , log likelihood = -6721.801 , BIC =  14808.7
(1) Y (continuous outcome): effect size of Y for each latent cluster
             Gamma
cluster1 0.0000000
cluster2 0.5040367
(2) Z: mean of omics data for each latent cluster
              mu_cluster1 mu_cluster2
cg_GRHL3        0.2452019  -0.3627827
cg_BTF3L4       0.1863510  -0.2798288
cg_AL358472.7   0.1954543  -0.3290283
cg_HDGF         0.1897587  -0.2614602
cg_TDRD5        0.1977435  -0.2186209
cg_CSRNP3       0.1742812  -0.1794667
cg_HSPD1        0.2829609  -0.3282916
cg_EPM2AIP1    -0.1565522   0.3031461
```

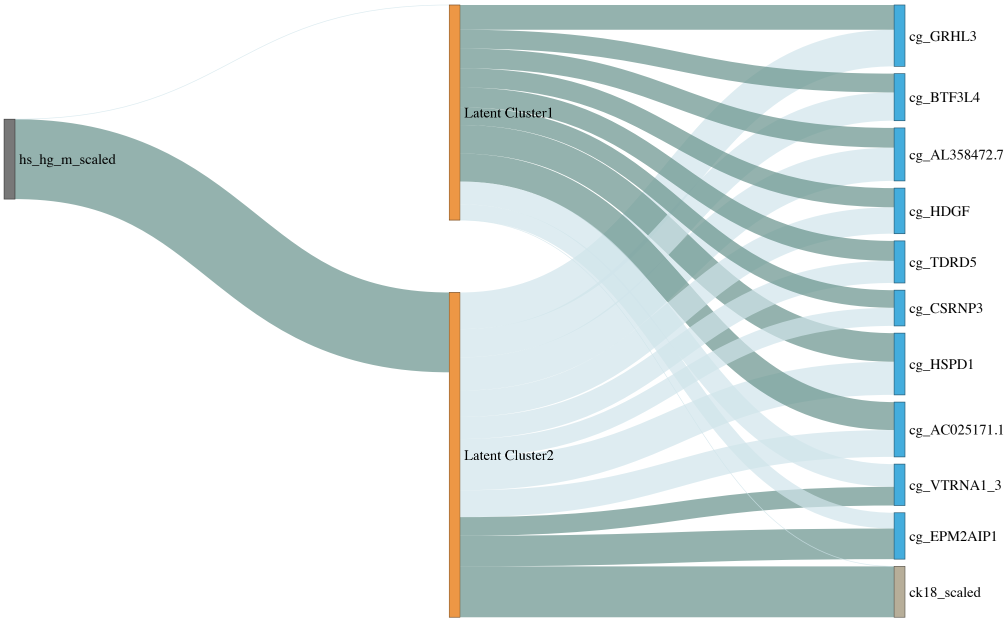


Figure 3: An example of using the Sankey diagram to visualize a LUCID model. The dark grey nodes represent the exposure, the light grey node represents the outcome, the blue nodes are omics data, and the orange nodes are latent clusters. The width of links and nodes corresponds to effect size. Light-colored links represent negative associations while dark-colored links indicate positive associations.

```
cg_AC025171.1   0.2769363  -0.2656729
cg_VTRNA1_3    -0.2250234   0.1847493
(3) E: odds ratio of being assigned to each latent cluster for each exposure
                              beta       OR
hs_hg_m_scaled.cluster2 0.7909168 2.205417
```

The first table summarizes the association between the latent cluster and the outcome (cytokeratin 18). The estimated effect size of cytokeratin 18 for cluster 1 is 0 since it is the reference cluster while that for cluster 2 is 0.504. In the case of a binary outcome, the coefficient in the first table is interpreted as the log odds for the variable for that specific cluster vs. the reference cluster. The second table characterizes each cluster by its omics signature. The omics signature is a cluster-specific mean for each variable in the omics data. For instance, latent cluster 1 is characterized by low levels of cg_EPM2AIP1 ($\mu_1(\text{cg\_EPM2AIP1}) = -0.157$) and cg_VTRNA1_3 ($\mu_1(\text{cg\_VTRNA1\_3}) = -0.225$), and high levels of all other omic features; for example, cg_GRHL3 ($\mu_1(\text{cg\_GRHL3}) = 0.2452019$). On the contrary, latent cluster 2 features high levels of cg_EPM2AIP1 ($\mu_2(\text{cg\_EPM2AIP1}) = 0.303$) and cg_VTRNA1_3 ($\mu_2(\text{cg\_VTRNA1\_3}) = 0.185$), and low levels of all other features including cg_GRHL3 ($\mu_2(\text{cg\_GRHL3}) = -0.363$). The third table relates the exposures to the latent clusters. For instance, 'hs_hg_m_scaled' represents the scaled level of maternal mercury exposure. The coefficient OR for 'hs_hg_m_scaled' is 0.791, meaning that for each doubling of the scaled level of maternal mercury exposure, the odds ratio of being assigned to latent cluster 2 is 2.205. Since latent cluster 2 is associated with a higher child level of cytokeratin 18, these results indicate that exposure to mercury is associated with a higher child level of cytokeratin 18 and thus higher risks of MAFLD.

Visualizing LUCID by plot()

Visualization is another imperative way to interpret statistical models. Here we use a Sankey diagram (Schmidt, 2008) for the visualization of the relationships among different components in LUCID. plot() takes a fitted LUCID model as input and creates a Sankey diagram in html format. It also accepts a user-defined color palette.

```
> # Visualize LUCID model via a Sankey diagram
> plot(fit1)
> # Change the node color
> plot(fit1, G_color = "yellow")
```

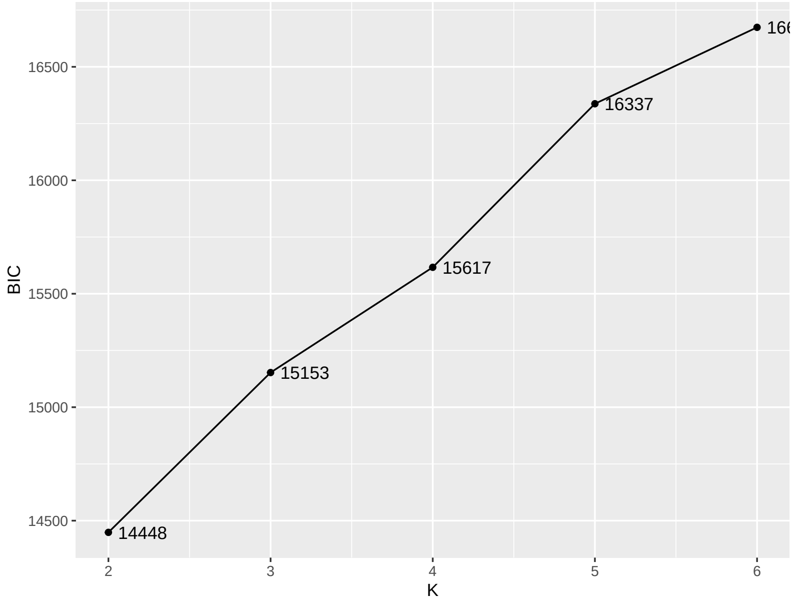


Figure 4: Choosing an optimal number of latent clusters K based on BIC.

```
> # Change the link color
> plot(fit1, pos_link_color = "red", neg_link_color = "green")
```

Figure 3 shows an example of a Sankey diagram for the example data. Each node represents a component in the LUCID model. Different components are labeled by user-defined colors. Each link represents the statistical association between two nodes. The width of the links reflects the effect size of the association, and the color of the links indicates the direction of the association. By default, dark-colored links represent positive associations while light-colored links indicate negative associations. The output of the Sankey diagram is an interactive html file in which the user can drag and rearrange the layout of the elements to avoid overlapping. In practice, we recommend limiting the number of variables in the Sankey diagram to approximately 10 to facilitate interpretation.

3.3 Model selection and variable selection

To determine the number of latent clusters, K, LUCID implements model selection by fitting a series of models, each with a different value for K, and then the Bayesian Information Criteria (BIC) is used to choose the optimal model (Fan and Tang, 2013). The optimal model is the one with the lowest BIC. The BIC for LUCID is defined as

$$\text{BIC} = -2\log L(\hat{\boldsymbol{\Theta}}) + D\log N \tag{13}$$

where $\hat{\boldsymbol{\Theta}}$ is the maximum likelihood estimation and D is the number of parameters in LUCID. Specifically, $D = P(K-1) + KM + KM(M+1)/2 + n_Y$, where n_Y is the number of parameters dependent upon the type of outcome $\boldsymbol{Y}$. If $\boldsymbol{Y}$ is a continuous outcome, then $n_Y = 2K$. If $\boldsymbol{Y}$ is a binary outcome, then $n_Y = K$. If a vector of K is used as input to the function `lucid()`, model selection is automatically performed and returns the model with optimal K.

```
> fit8 <- lucid(G = exposome, Z = methylomics, Y = ck18, lucid_model = "early",
+               family = "normal", K = 2:6)
> # Check the optimal K
> fit8$K
[1] 2
```

Separately, users can use the auxiliary function `tune_lucid()` to explore model selection in more detail. This function returns the optimal model as well as a table recording the tuning process. Below is an example of tuning K and visualizing the tuning process by using `tune_lucid()`.

```
> # Look into the tuning process in more detail
> tune_K <- tune_lucid(G = exposome, Z = methylomics, Y = ck18, lucid_model = "early",
+                     family = "normal", K = 2:6)
> fit9 <- tune_K$best_model
> ggplot(data = tune_K$tune_list, aes(x = K, y = BIC, label = round(BIC, 0))) +
+   geom_point() +
+   geom_line() +
+   geom_text(hjust = -0.2)
```

Figure 4 shows the tuning process with $K = 2$ resulting in the lowest BIC. Including additional or redundant variables in clustering models may increase model complexity, impair prediction accuracy, and boost computational time (Fop and Murphy, 2018). **LUCIDus** performs variable selection for both the genetic/environmental exposures $\boldsymbol{G}$ and for the other multi-omics data $\boldsymbol{Z}$ via L_1-norm penalty terms. For variable selection for $\boldsymbol{G}$, we apply the LASSO regression to obtain sparse solutions for Equation 5, which is

$$\boldsymbol{\beta}_{\text{LASSO}}^{(t+1)} = \arg\max_{\boldsymbol{\beta}} \left\{ \sum_{i=1}^{N} \sum_{j=1}^{K} r_{ij}^{(t)} \log S(X_i = j|\boldsymbol{G}_i; \boldsymbol{\beta}_j) - \lambda_{\boldsymbol{\beta}} \sum_{j=1}^{K} \sum_{l=1}^{P} |\beta_{jl}| \right\} \tag{14}$$

To obtain sparse estimation for $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$, we implement the penalized model-based method (Zhou et al., 2009). $\boldsymbol{\mu}$ is first updated by maximizing the following equation,

$$\boldsymbol{\mu}_j^{(t+1)} = \arg\max_{\boldsymbol{\mu}} \left\{ \sum_{i=1}^{N} \sum_{j=1}^{K} r_{ij}^{(t)} \log \phi(\boldsymbol{Z}_i|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j) - \lambda_{\boldsymbol{\mu}} \sum_{j=1}^{K} \sum_{l=1}^{M} |\mu_{jl}| \right\} \tag{15}$$

Denote $\boldsymbol{W} = \boldsymbol{\Sigma}^{-1}$. Then $\boldsymbol{\Sigma}$ is updated by maximizing its inverse penalized by L_1-norm

$$\boldsymbol{W}_j^{(t+1)} = \arg\max_{\boldsymbol{W}} \left\{ \sum_{i=1}^{N} \sum_{j=1}^{K} r_{ij}^{(t)} \left(\det \boldsymbol{W}_j - \text{trace}(\boldsymbol{S}_j^{(t)} \boldsymbol{W}_j) \right) - \lambda_{\boldsymbol{W}} \sum_{ls} |w_{jls}| \right\} \tag{16}$$

where $\boldsymbol{S}_j$ is the empirical covariance matrix at each iteration, defined as

$$\boldsymbol{S}_j^{(t)} = \frac{\sum_{i=1}^{N} r_{ij} \left(\boldsymbol{Z}_i - \boldsymbol{\mu}_j^{(t)} \right) \left(\boldsymbol{Z}_i - \boldsymbol{\mu}_j^{(t)} \right)^T}{\sum_{i=1}^{N} r_{ij}^{(t)}} \tag{17}$$

To choose the optimal combination of the three L_1-norm penalties, we implement a grid search by adopting a modified BIC (Pan and Shen, 2007), defined as

$$\text{BIC}_p = -2 \log L(\hat{\boldsymbol{\Theta}}) + (D - D_{\boldsymbol{G}} - D_{\boldsymbol{Z}}) \log N \tag{18}$$

Where $D_{\boldsymbol{G}}$ is the number of exposure variables whose effect estimates are 0 across all latent clusters and $D_{\boldsymbol{Z}}$ is the number of omic variables whose effect estimates are 0 across all latent clusters. We can tune $\lambda_{\boldsymbol{\beta}}$, $\lambda_{\boldsymbol{\mu}}$, and $\lambda_{\boldsymbol{W}}$ either separately or jointly. For instance, if only exposome data $\boldsymbol{G}$ is high-dimensional, we only need to tune $\boldsymbol{\lambda}_{\boldsymbol{\beta}}$. If variable selection is desired for both $\boldsymbol{G}$ and $\boldsymbol{Z}$, then the three L_1-norm penalties should be tuned simultaneously. $\lambda_{\boldsymbol{\beta}}$, $\lambda_{\boldsymbol{\mu}}$, and $\lambda_{\boldsymbol{W}}$ match the parameters Rho_G, Rho_Z_Mu, and Rho_Z_Cov in the lucid() function. Each parameter accepts a numeric vector or a scalar as input. For guidance, empirical experiments utilizing normalized multi-omics data suggest integer values in the range of $0 - 100$ for Rho_Z_Mu and values in the range of $0 - 1$ for Rho_G and Rho_Z_Cov. The higher the values of Rho_Z_Mu, Rho_G, and Rho_Z_Cov, the fewer omic features that will be selected. Below are examples for conducting variable selection for $\boldsymbol{G}$ and $\boldsymbol{Z}$, separately and jointly.

```
> # Variable selection for G
> # Add 10 more noise variables to the exposome
> noise <- matrix(rnorm(420 * 10), nrow = 420)
> exposome_noise <- cbind(exposome, noise)
> fit10 <- lucid(G = exposome_noise, Z = methylomics, Y = ck18,
+                lucid_model = "early", family = "normal", K = 2,
+                Rho_G = seq(0, 0.4, by = 0.01), seed = 1008)
1/11 exposures are selected
> # Summary of optimal lucid model
> # summary(fit10)
> # Variable selection for Z
> # add 10 more noise variables to the methylomics
> methylomics_noise <- cbind(methylomics, noise)
> fit11 <- lucid(G = exposome, Z = methylomics_noise, Y = ck18,
+               lucid_model = "early", family = "normal", K = 2,
+               Rho_Z_Mu = 5:10, Rho_Z_Cov = seq(0.1, 0.4, by = 0.1), seed = 1008)
```

```
10/20 omics variables are selected
> # Summary of optimal lucid model
> # summary(fit11)
> # Variable selection for G and Z jointly
> fit12 <- lucid(G = exposome_noise, Z = methylomics_noise, Y = ck18,
+               lucid_model = "early", family = "normal", K = 2,
+               Rho_G = 0.01, Rho_Z_Mu = 10, Rho_Z_Cov = 0.5, seed = 123)
1/11 exposures are selected
11/20 omics variables are selected
```

Oftentimes, the number of features M for omics data $\boldsymbol{Z}$ is in the hundreds or even thousands. With a high dimensionality of $\boldsymbol{Z}$, LUCID can still have stable performance and select the features with effect if the number of observations N is higher than M. Below is an example of conducting variable selection in $\boldsymbol{Z}$ with $M = 210$ features (including 200 noise features) while $N = 420$. We can see that with higher Rho_Z_Mu and Rho_Z_Cov, LUCID successfully selects the features with effect.

```
> # Variable selection for Z (high number of features M)
> set.seed(1008)
> # Add 200 more noise features to the methylomics
> noise <- matrix(rnorm(420 * 200), nrow = 420)
> methylomics_noise <- cbind(methylomics, noise)
> fit13 <- lucid(G = exposome, Z = methylomics_noise, Y = ck18,
+               family = "normal", lucid_model = "early", K = 2,
+               Rho_Z_Mu = 20, Rho_Z_Cov = 0.6)
17/210 omics variables are selected
> # Summary of optimal lucid model
> # summary(fit13)
```

Via simulations, the run time for LUCIDus increases linearly as the number of features increases. While the integration of regularization allows for a high number of features to be analyzed, when M is higher than N, LUCID might result in unstable estimation; therefore, pre-screening approaches for the omic features such as the "meet-in-the-middle" become imperative to reduce dimensionality before fitting the LUCID model (Cadiou et al., 2021).

3.4 Deriving confidence intervals

We use the nonparametric bootstrap approach to derive confidence intervals (CIs) for the MLE estimates from a LUCID model (Davison and Hinkley, 1997). Nonparametric bootstrap draws observations from a sample with replacement. We index the bootstrap samples by $b = 1, 2, \ldots, B$. For each bootstrap sample $\boldsymbol{D}^b$, we fit the LUCID model and calculate MLE $\boldsymbol{\Theta}^b$. Two types of bootstrap CIs are constructed based on the distribution of $\boldsymbol{\Theta}^b$: (1) normal CIs and (2) percentile CIs. Given the confidence level $1 - \alpha$, the normal CIs are constructed by

$$(1-\alpha)\%\text{CI}_{\text{normal}} = \boldsymbol{\Theta} - \text{bias} \pm Z_{1-\alpha/2}\sigma(\boldsymbol{\Theta}) \tag{19}$$

where bias $= \bar{\boldsymbol{\Theta}} - \boldsymbol{\Theta}$, $\bar{\boldsymbol{\Theta}}$ is the mean of the bootstrap estimators and $\sigma(\boldsymbol{\Theta})$ is the bootstrap standard error, which is

$$\sigma(\boldsymbol{\Theta}) = \sqrt{\frac{1}{B-1}\sum_{b=1}^{B}\left(\boldsymbol{\Theta}^b - \bar{\boldsymbol{\Theta}}\right)} \tag{20}$$

All calculations in Equations 19 and 20 are element-wise. In addition, the $1 - \alpha$ percentile CIs are calculated by ordering the bootstrap estimators from smallest to largest and selecting the estimates at $\alpha/2$ percentile and $(1 - \alpha/2)$ percentile as the CIs. boot_lucid() constructs the two bootstrap CIs described above for inference. Users can specify the confidence level (conf) and the number of bootstrap replicates (R). While boot_lucid() is running, a progress bar is displayed in the R console. We recommend R $\geq$ 200 to estimate the bootstrap standard error for normal CIs and R $\geq$ 800 to estimate quantiles to construct percentile CIs. Below are examples of deriving bootstrap CIs for LUCID. By default, LUCID calculates 95% CIs.

```
> # Bootstrap to obtain 95% CI (by default) for LUCID
> set.seed(123)
> boot1 <- boot_lucid(G = exposome, Z = methylomics, Y = ck18, lucid_model = "early",
+                     model = fit1, R = 200)
# 90% CIs
> boot2 <- boot_lucid(G = exposome, Z = methylomics, Y = ck18, lucid_model = "early",
+                     model = fit1, R = 200, conf = 0.9)
```

We can obtain a more comprehensive summary table with bootstrap CIs by integrating `summary()` with output from `boot_lucid()`. The resulting summary table has 5 columns: '`t0`' corresponds to parameters estimated from the observed data; '`norm_lower`' and '`norm_upper`' represent the lower and upper limit of normal CIs, respectively; '`perc_lower`' and '`perc_upper`' represent the percentile CIs.

```
> # Summary table with 95% bootstrap CIs
> summary(fit1, boot.se = boot1))
```

`boot_lucid()` is built upon the R library **boot** (Canty and Ripley, 2021). The original output from **boot** is also returned by `boot_lucid()` and can be used to evaluate the normality of the distribution of the bootstrap estimations.

3.5 Prediction

The prediction of LUCID includes two parts: prediction of the latent clusters $\boldsymbol{X}$ and prediction of the outcome $\boldsymbol{Y}$. The `predict_lucid()` can perform both tasks. Prediction for the latent cluster is determined by IP using the optimal rule, which is

$$\hat{X}_i = \arg\max_j P(X_i = j | \boldsymbol{D}, \boldsymbol{\Theta}) \tag{21}$$

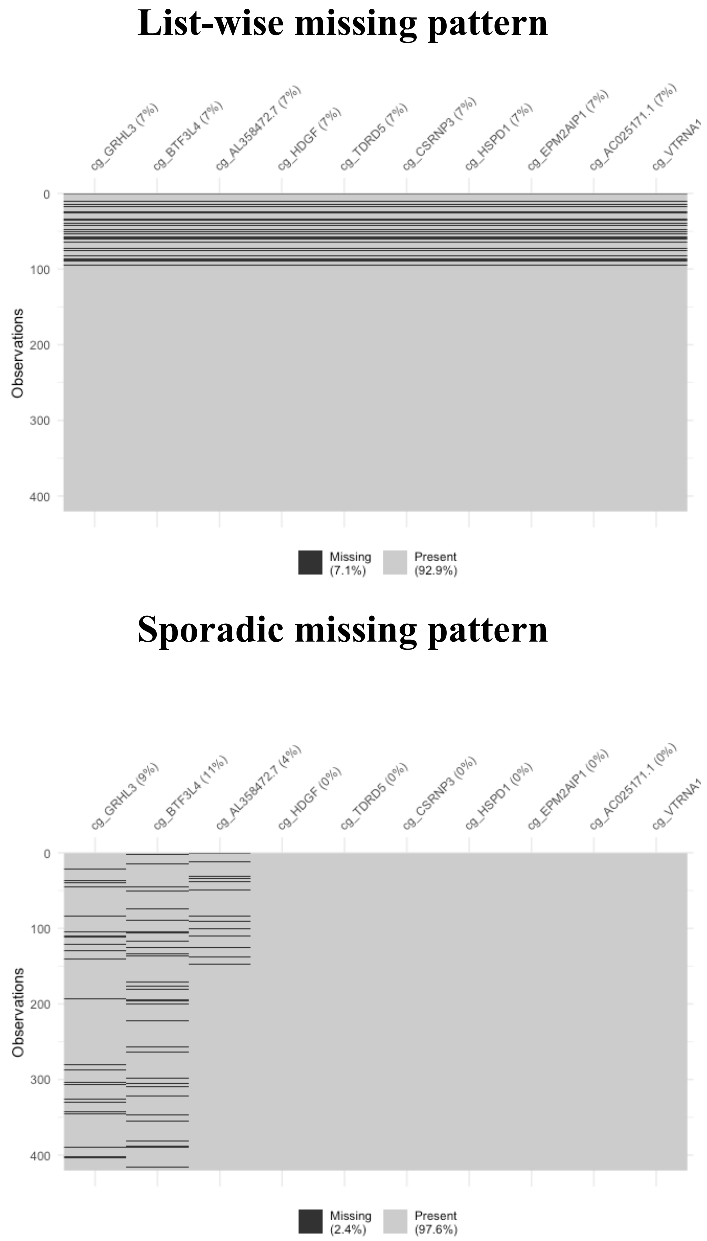
Prediction for the outcome follows a conventional regression framework. When making predictions, `predict_lucid()` for LUCID requires input of new $\boldsymbol{G}$ and $\boldsymbol{Z}$, and optional input of $\boldsymbol{Y}$. If $\boldsymbol{Y}$ is provided, we use Equation 3 to calculate IPs; otherwise, Equation 12 is applied. As a result, we can either fit a supervised LUCID model to make predictions in an unsupervised fashion or vice versa. This design allows for flexibility. For instance, we can build a LUCID model in one cohort in a supervised fashion and then make predictions on another new independent cohort, regardless of whether the outcome is measured or not. `predict_lucid()` for LUCID returns a `list` containing IPs for each observation, predicted cluster assignment, and predicted outcome. Below is an example code for prediction based on a supervised LUCID model, `fit1`.

```
> # Predict cluster with information of Y
> pred1 <- predict_lucid(model = fit1, G = exposome, Z = methylomics, Y = ck18)
> # Predict cluster without information of Y
> pred2 <- predict_lucid(model = fit1, G = exposome, Z = methylomics)
> # Predicted cluster label
> table(pred1$pred.x)
  1   2
223 197
> # Predicted outcome
> pred1$pred.y[1:5]
[1]  0.2580164 -0.3806054  0.2504817  0.4123169 -0.1898255
```

3.6 Incorporating missingness in multi-omics data

A major update in **LUCIDus** version 3 is the incorporation of missing data. Missingness is a major challenge in the integrative analysis of omics. In large cohort studies, it is common that some omics data are not available for all participants for various reasons such as budgetary constraints, low sample availability, or lack of consent for future use of biospecimens (Voillet et al., 2016). We refer to this type of missingness as a list-wise missingness pattern. In addition, even when omics data are measured for a given participant, it is common that some omic features are randomly missing due to the measurement process. We refer to this pattern as sporadic missingness. **LUCIDus** can deal with the two missing patterns mentioned above or the combination of the two. We assume the multi-omics data are missing completely at random (MCAR) for both missing patterns (i.e., for the list-wise missing pattern, the probability of multi-omics data being not available is the same for all subjects; for sporadic missing patterns, the probability of a certain omic feature being missing is the same for all omic features). For list-wise missingness, we use a modified LUCID model based on a likelihood partition. We divide the LUCID likelihood into two parts: observations with complete multi-omics data and observations without any measured multi-omics data. The likelihood of the former observations remains the same as Equation 2, denoted by $l_o(\boldsymbol{\Theta}|\boldsymbol{D})$. The joint likelihood of the latter observations becomes

$$l_m(\boldsymbol{\Theta}|\boldsymbol{D}) = \sum_{I_o=1}^{N_o} \sum_{j=1}^{K} I(X_{i_o} = j) \left(\log S(X_{i_o} = j | \boldsymbol{G}_{i_o}; \boldsymbol{\beta}_j) + \log \phi \left(Y_{i_o} | \gamma_j, \sigma_j^2 \right) \right) \tag{22}$$



List-wise missing pattern

Sporadic missing pattern

Figure 5: Two simulated missing patterns in methylomics data

with corresponding responsibility defined as

$$r_{ij}^{(t)} = \frac{S\left(X_i = j|\boldsymbol{G}_i; \boldsymbol{\beta}^{(t)}\right) \phi\left(Y_i|X_i = j; \gamma_j^{(t)}, \sigma_j^{2(t)}\right)}{\sum_{j=1}^{K} S\left(X_i = j|\boldsymbol{G}_i; \boldsymbol{\beta}^{(t)}\right) \phi\left(Y_i|X_i = j; \gamma_j^{(t)}, \sigma_j^{2(t)}\right)} \tag{23}$$

The joint likelihood of LUCID for all data becomes $l(\boldsymbol{\Theta}) = l_o(\boldsymbol{\Theta}|\boldsymbol{D}) + l_m(\boldsymbol{\Theta}|\boldsymbol{D})$. We then use the same EM algorithm described in Section Overview of the original LUCID model (early integration) to calculate the MLE of LUCID with list-wise missingness. For sporadic missingness, we use an integrated imputation method (Zhang et al., 2021). Each missing value in the omics data $\boldsymbol{Z}$ is treated as an unknown "parameter" to be optimized. Specifically in the M-step, after maximizing the parameters of the LUCID model with fixed $\boldsymbol{Z}$, we implement an additional *imputation step* which maximizes $\boldsymbol{Z}$ with fixed parameters within LUCID. Details of the statistical derivation can be found in Zhang et al. (2021). To illustrate this new feature, we first simulate missing values in the HELIX methylomics data by randomly setting values to 'NA'. Both list-wise and sporadic missing patterns are considered, as shown in Figure 5. We use the `vis_miss()` function from the R package **visdat** to visualize both missing patterns (Tierney, 2023).

```
> library(visdat)
> set.seed(1)
> methylomics_miss_sporadic <- methylomics_miss_listwise <- as.matrix(methylomics)
> index <- arrayInd(sample(1000, 0.1 * 1000), dim(methylomics))
> methylomics_miss_sporadic[index] <- NA # sporadic missing pattern
> methylomics_miss_listwise[sample(1:100, 30), ] <- NA # listwise missing pattern
> vis_miss(as.data.frame(methylomics_miss_sporadic))
> vis_miss(as.data.frame(methylomics_miss_listwise))
```

Below are examples of fitting the LUCID model with missingness in multi-omics data. `lucid()` will automatically examine missing patterns present in the multi-omics data and select a corresponding method to account for the missing data. For sporadic missing patterns, we use **mclust** to initialize missing values in the multi-omics data.

```
> fit14 <- lucid(G = exposome, Z = methylomics_miss_listwise, Y = ck18,
+                lucid_model = "early", K = 2, family = "normal")
> fit15 <- lucid(G = exposome, Z = methylomics_miss_sporadic, Y = ck18,
+                lucid_model = "early", K = 2, family = "normal")
> # summary(fit15)
```

3.7 LUCID in Parallel

In the previous sections, we focused on using the early integration strategy by concatenating each omic layer one by one and inputting a single data matrix into the LUCID model for estimation. However, researchers may be interested in modeling the correlation structure of each omic layer independently to investigate how multi-omics data may act in parallel with an outcome. In this section, we extend the LUCID model to incorporate multi-omics data by introducing multiple latent variables into the framework. Suppose we have a sample of size n, indexed by i = 1, 2, ..., n. It has a collection of m omic layers, denoted by Z_1, ..., Z_a, ..., Z_m with corresponding dimensions p_1, ..., p_a, ..., p_m. Each omic layer Z_a is summarized by a latent categorical variable X_a, which contains k_a categories. Each category is interpreted as a latent cluster (or subgroup) for that particular omic layer. All latent variables are linked to the same exposure matrix $\boldsymbol{G}$ and the outcome $\boldsymbol{Y}$ for LUCID in parallel, as shown in Figure 1 (b). Suppose the latent variable X_{ai} is observed for $a = 1, \ldots, m$. According to the DAG in Figure 1 (b), the distributions of $f\left(\boldsymbol{Z}_{ai} \mid \boldsymbol{G}\right)$ are conditionally independent of each other for $a = 1, \ldots, m$, and the distributions of the latent variable $f\left(X_{ai} \mid \boldsymbol{G}_i\right)$ are also conditionally independent of each other. Since $\boldsymbol{X}_{ai}$ is a discrete variable with k_i categories, we assume $\boldsymbol{X}_{ai}$ follows a multinomial distribution conditioning on $\boldsymbol{G}$, denoted by the softmax function $S(\cdot)$. We assume multi-omics data $\boldsymbol{Z}_{ai}$ follows a multivariate Gaussian distribution conditioning on $\boldsymbol{X}_{ai}$, denoted by $\phi\left(\boldsymbol{Z}_{ai} \mid \boldsymbol{X}_{ai} = j_a; \boldsymbol{\mu}_{ja}, \boldsymbol{\Sigma}_{ja}\right)$, where $\boldsymbol{\mu}_{ja}$ and $\boldsymbol{\Sigma}_{ja}$ are cluster-specific means and variance-covariance matrices for omic layer a. The outcome Y can be either a continuous outcome or a binary outcome. For illustration, here, we discuss the situation that Y is a continuous variable that follows a Gaussian distribution. The relationship between latent variables X_a and outcome Y is formulated as

$$EY_i = \gamma_0 + \sum_{j_1=2}^{k_1} \gamma_{j_1} I\left(X_{1i} = j_1\right) + \cdots + \sum_{j_a=2}^{k_a} \gamma_{j_a} I\left(X_{ai} = j_a\right) + \cdots + \sum_{j_m=2}^{k_m} \gamma_{j_m} I\left(X_{mi} = j_m\right) \tag{24}$$

where the intercept γ_0 is the expected value of Y_i given $X_{ai} = 1$ for $a = 1, 2, \ldots, m$ (the reference cluster for all combinations of latent cluster assignments); γ_{j_a} is the change of Y if the latent variable X_a becomes j_a instead of 1. Let $\boldsymbol{D}$ be the generic notation for all observed data. The log-likelihood of LUCID in Parallel is constructed below:

$$\begin{aligned}
\log L(\boldsymbol{\Theta} \mid \boldsymbol{D}) &= \sum_{i=1}^{n} \log f\left(\boldsymbol{Z}_{1i}, \ldots, \boldsymbol{Z}_{mi}, \boldsymbol{Y}_i \mid \boldsymbol{G}_i; \boldsymbol{\Theta}\right) \\
&= \sum_{i=1}^{n} \log \left[\prod_{j_1=1}^{k_1} \cdots \prod_{j_m=1}^{k_m} f\left(\boldsymbol{Z}_{1i}, \ldots, \boldsymbol{Z}_{mi}, X_{1i}, \ldots, X_{mi}, Y_i \mid \boldsymbol{G}_i; \boldsymbol{\Theta}\right)^{I(X_{1i}=j_1, \ldots, X_{mi}=j_m)} \right] \\
&= \sum_{i=1}^{n} \sum_{j_1=1}^{k_1} \cdots \sum_{j_m=1}^{k_m} I\left(X_{1i} = j_1, \ldots, X_{mi} = j_m\right) \log f\left(\boldsymbol{Z}_{1i}, \ldots, \boldsymbol{Z}_{mi}, X_{1i}, \ldots, X_{mi}, Y_i \mid \boldsymbol{G}_i; \boldsymbol{\Theta}\right) \\
&= \sum_{i=1}^{n} \sum_{j_1=1}^{k_1} \cdots \sum_{j_m=1}^{k_m} I\left(X_{1i} = j_1, \ldots, X_{mi} = j_m\right) \log \phi\left(Y_i \mid X_{1i}, \ldots, X_{mi}, \gamma, \sigma^2\right) \\
&+ \sum_{i=1}^{n} \sum_{a=1}^{m} \sum_{j_1=1}^{k_1} \cdots \sum_{j_m=1}^{k_m} I\left(X_{1i} = j_1, \ldots, X_{mi} = j_m\right) \log \phi\left(\boldsymbol{Z}_{ai} \mid X_{ai} = j_a, \boldsymbol{\mu}_{a,j_a}, \boldsymbol{\Sigma}_{a,j_a}\right) \\
&+ \sum_{i=1}^{n} \sum_{a=1}^{m} \sum_{j_1=1}^{k_1} \cdots \sum_{j_m=1}^{k_m} I\left(X_{1i} = j_1, \ldots, X_{mi} = j_m\right) \log S\left(\boldsymbol{X}_{ai} = j_a \mid \boldsymbol{G}_i, \boldsymbol{\beta}_a\right)
\end{aligned} \tag{25}$$

The log-likelihood of LUCID in parallel is similar to that with LUCID early integration. It is natural to follow the same principles of the EM algorithm for LUCID early integration and estimate the parameters of LUCID in parallel with targeted modifications. For illustration, we continue with the previous data example using maternal exposure to mercury as the exposome $\boldsymbol{G}$ and childhood cytokeratin 18 level as the outcome $\boldsymbol{Y}$. For a LUCID in parallel model, multi-omics data $\boldsymbol{Z}$ is a list of three $420 \times p_a$ matrices of omic layers where $a = 1, 2, 3$ (layer 1: methylome; layer 2: transcriptome; layer 3: miRNA). We fit LUCID in parallel models with continuous ck18 using `lucid()`. We specify the argument `lucid_model` to be '`parallel`', and the argument `K` should be specified as a list of three integers (or sequences of integers if tuning) indicating the number of latent clusters for each omic layer. The variable selection feature is not yet available for LUCID in parallel, and we can only tune for `K`. We can use argument `CoG` and `CoY` to adjust for covariates for the $\boldsymbol{G}$ to $\boldsymbol{X}$ association and the $\boldsymbol{X}$ to $\boldsymbol{Y}$ association, respectively. Similar to LUCID early integration, the parameter `family` specifies the outcome type, which is '`normal`' here, corresponding to a continuous outcome. For a binary outcome, `family` should be specified as '`binary`'. `lucid()` returns an object of class '`lucid_parallel`' providing the MLE of the LUCID in parallel model as well as IPs. We can call `summary()` with the input of a '`lucid_parallel`' object to print three tables of the model results. Similar to the summary table of a LUCID early integration model, the first table summarizes the association between the latent clusters for each omic layer and the outcome (cytokeratin 18). The second table characterizes each cluster by its omics signature within each layer. The third table relates the exposures to the latent clusters for each layer. The nuance is that the LUCID in parallel model has multiple independent omic layers, and each layer corresponds to a separate latent cluster variable, thus all the related results are presented in the summary tables. See the two examples of fitting a LUCID in parallel model below.

```
> # Create a list of multi-omics data
> omics_lst <- simulated_data[-which(names(simulated_data) == "phenotype")]
> Z = omics_lst[c(1:3)]
> # LUCID in parallel, adjusting for the covariates for the E-X and X-Y associations
> fit16 <- lucid(G = exposome, Z = Z, Y = ck18, K = list(2, 2, 2), family = "normal",
+               CoY = covs, CoG = covs,
+               lucid_model = "parallel", useY = TRUE)
> # print the summary of the LUCID in parallel model
> summary(fit16)
----------Summary of the LUCID in Parallel model----------
K =  2 2 2 , log likelihood = -16413.64 , BIC =  36892.36
(1) Y (continuous outcome): effects of each non-reference latent cluster
    for each layer of Y
(and effect of covariates if included)
                              Gamma
cluster2Layer1          2.139337462
cluster2Layer2         -0.976575066
cluster2Layer3          1.617148373
```

```
e3_sex_None            0.023397524
hs_child_age_yrs_None -0.002297037
(2) Z: mean of omics data for each latent cluster of each layer
Layer  1
              mu_cluster1 mu_cluster2
cg_GRHL3       0.09782065  -0.2827826
cg_BTF3L4      0.12271079  -0.3059186
cg_AL358472.7  0.10019649  -0.3164803
cg_HDGF        0.15388979  -0.3322801
cg_TDRD5       0.10778098  -0.1832600
cg_CSRNP3      0.08692918  -0.1300857
cg_HSPD1       0.21255103  -0.3855858
cg_EPM2AIP1   -0.11651397   0.3691065
cg_AC025171.1  0.24747122  -0.3750450
cg_VTRNA1_3   -0.13739036   0.1515515
Layer  2
                  mu_cluster1  mu_cluster2
tc_TC01006069_nc  0.098203482 -0.169267971
tc_SLC9A4        -0.025858454  0.046609093
tc_RAB6C_AS1      0.009567584 -0.087150028
tc_LOC100129029   0.032557846  0.078278431
tc_BRE           -0.039411493  0.073329913
tc_TC03001220_nc -0.068457436  0.013811537
tc_TC04002114_nc -0.030761261  0.033415671
tc_TC04002369_nc  0.297076410 -0.216123334
tc_BEND4         -0.128343956  0.074585000
tc_SLC9A3         0.139869536  0.002458175
Layer  3
              mu_cluster1  mu_cluster2
miR.101.3p     0.10748628  0.023648293
miR.125a.5p   -0.20581745  0.135670802
miR.125b.1.3p  0.07644648  0.019200103
miR.127.3p    -0.05542160  0.184175902
miR.140.5p     0.14979024  0.050720602
miR.142.3p     0.11157913 -0.017566242
miR.144.5p     0.07526727 -0.016027655
miR.19a.3p     0.09193308 -0.002059547
miR.19b.3p     0.08515311  0.032351416
miR.21.5p      0.05800681  0.029022417
(3) E: odds ratio of being assigned to each latent cluster for each exposure
    for each layer
Layer  1
                                     beta        OR
hs_hg_m_scaled.cluster2         0.7352341 2.0859703
e3_sex_None.cluster2           -0.4107395 0.6631596
hs_child_age_yrs_None.cluster2 -0.2657909 0.7665994
Layer  2
                                     beta        OR
hs_hg_m_scaled.cluster2         0.1313174 1.1403296
e3_sex_None.cluster2            0.1928708 1.2127261
hs_child_age_yrs_None.cluster2 -0.1066509 0.8988394
Layer  3
                                     beta        OR
hs_hg_m_scaled.cluster2        -0.4896515 0.6128399
e3_sex_None.cluster2            0.4855559 1.6250782
hs_child_age_yrs_None.cluster2  0.1042404 1.1098672
> # LUCID in parallel, tune for the number of clusters for methylomics
> fit17 <- lucid(G = exposome, Z = Z, Y = ck18, K = list(2, 2:3, 2), family = "normal",
+                lucid_model = "parallel", useY = TRUE)
> # summary(fit17)
```

3.8 LUCID in Serial

In a more late integration framework, LUCID can also be extended to incorporate multiple latent variables in a serial fashion if researchers believe that given an exposure, multi-omics data act serially through a multistep process towards the outcome, as illustrated in Figure 1 (c). This framework can accommodate the following situations: (1) longitudinal measurements on the same multi-omics data type and (2) biological relationships of multi-omics data. The estimation of latent clusters for each omic layer can be formulated as an unsupervised LUCID early integration or LUCID in parallel sub-model. For a LUCID in serial model, $\boldsymbol{Z}$ includes m ordered omic layers in a sequential fashion. For illustration, assuming all the sub-models are LUCID early integration models. For a certain omic component a, the cluster variable $\boldsymbol{X}_{a-1}$ serves as the "$\boldsymbol{G}$" in the LUCID early integration model framework, and $\boldsymbol{X}_a$ is jointly estimated by $\boldsymbol{X}_{a-1}$ and $\boldsymbol{Z}_a$. However, the special cases are for the first and the last omic layer, where the corresponding $\boldsymbol{X}_1$ is estimated jointly by $\boldsymbol{G}$ and $\boldsymbol{Z}_1$, and the corresponding $\boldsymbol{X}_m$ is estimated jointly by $\boldsymbol{X}_{m-1}$, $\boldsymbol{Z}_m$, and $\boldsymbol{Y}$ if supervised, respectively. Alternatively, for a certain omic component a, we can add another transition probability matrix component $\boldsymbol{p}$ in the joint likelihood relating $\boldsymbol{X}_{a-1}$ to $\boldsymbol{X}_a$ to describe the state change with $P(\boldsymbol{X}_a|\boldsymbol{X}_{a-1}, \boldsymbol{C}) = \boldsymbol{p}_a$, where $\boldsymbol{p}_a$ is a general notation for the inclusion probability of all the latent clusters for omic component a and $\boldsymbol{C}$ dynamically represents other variables that jointly estimate $\boldsymbol{X}_a$. Note that a list of m ordered omic components for $\boldsymbol{Z}$ indicates m sub-models in a LUCID in serial model, and each sub-model can be either a LUCID Early Integration model or a LUCID in Parallel model. Note that the DAG in Figure 1 (c) only shows the scenario where all the sub-models are LUCID early integration models. Bold $\boldsymbol{X}_a$ and omic component $\boldsymbol{Z}_a$ indicate that they may include more than 1 latent variable or omic layer for each omic component if the corresponding sub-model is LUCID in parallel. Let m be the number of sub-models in a LUCID in serial model. For each sub-model a, given $\boldsymbol{\Theta_a}$ and $\boldsymbol{D_a}$, its own $\log L(\Theta_a \mid D_a)$ follows the log-likelihood of a LUCID Early Integration or a LUCID in Parallel model defined in the previous sections. Except for the sub-model m, all sub-models are unsupervised. For sub-models 2 to m, the β parameter defined previously becomes δ. The log-likelihood of LUCID in Serial is constructed below,

$$\begin{aligned}
\log L(\boldsymbol{\Theta} \mid \boldsymbol{D}) &= \sum_{a=1}^{m} \log L(\boldsymbol{\Theta_a} \mid \boldsymbol{D_a}) \\
&= \sum_{i=1}^{n} \log f_1(\boldsymbol{Z}_{1i}, \boldsymbol{G}_i \mid \boldsymbol{\Theta_1}) \\
&+ \sum_{a=2}^{m-1} \sum_{i=1}^{n} \log f_a(\boldsymbol{Z}_{ai}, \boldsymbol{p}_{ai} \mid \boldsymbol{\Theta_a}) + \sum_{i=1}^{n} \log f_m(\boldsymbol{Z}_{mi}, \boldsymbol{p}_{ai}, \boldsymbol{Y_i} \mid \boldsymbol{\Theta_m}) \\
&= \sum_{i=1}^{n} \log f_1(\boldsymbol{Z}_{1i}, \boldsymbol{G}_i \mid \boldsymbol{\beta}, \boldsymbol{\mu_1}, \boldsymbol{\Sigma_1}) \\
&+ \sum_{a=2}^{m-1} \sum_{i=1}^{n} \log f_a(\boldsymbol{Z}_{ai}, \boldsymbol{p}_{ai} \mid \boldsymbol{\delta_a}, \boldsymbol{\mu_a}, \boldsymbol{\Sigma_a}) \\
&+ \sum_{i=1}^{n} \log f_m(\boldsymbol{Z}_{mi}, \boldsymbol{p}_{ai}, \boldsymbol{Y_i} \mid \boldsymbol{\delta_m}, \boldsymbol{\mu_m}, \boldsymbol{\Sigma_m}, \gamma)
\end{aligned} \tag{26}$$

Here, we assume sub-models are independent of each other. The EM algorithm for estimating the parameters of each sub-model is exactly the same as estimating a single LUCID early integration model or a LUCID in parallel model. Using the previous data example for illustration, for a LUCID in serial model, multi-omics data $\boldsymbol{Z}$ can be an ordered list of three $420 \times p_a$ matrices of omic layers where $a = 1, 2, 3$ (layer 1: methylome; layer 2: transcriptome; layer 3: miRNA), representing that all the sub-models are LUCID early integration models and `K` is a list of three integers (or sequences of integers if tuning) indicating the number of latent clusters for each successively linked omic layer. We specify `lucid_model` to be '`serial`'. The use of arguments `CoG`, `CoY`, `useY`, and `family` is exactly the same as for a LUCID in parallel model. `lucid()` constructs the model and returns an object of class '`lucid_serial`', and calling `summary()` with the input of a '`lucid_serial`' object prints the summarizing tables of the model estimates. We have introduced summarizing tables for the LUCID early integration model and the LUCID in parallel model. Since the LUCID in serial model is a combination of LUCID early integration and LUCID in parallel sub-models, the summarizing tables for each sub-model follow a similar structure to what we have discussed for the two types of sub-models, but with targeted modifications to conform to the setup of the LUCID in serial model. See the summarizing tables for the first LUCID in serial model example below. Alternatively, besides an integer, the element of the ordered list of $\boldsymbol{Z}$ can also be a list, which indicates that the corresponding sub-model is a LUCID in parallel model. Here, `K` should contain a list of lists of integers. See the second LUCID in serial model example below.

```
> # LUCID in serial, adjusting for the covariates of the E-X and X-Y associations
> # Tune the number of clusters for methylome and transcriptome
> # All 3 sub-models are LUCID early integration models and each with two latent clusters for each layer
> fit18 <- lucid(G = exposome, Z = Z, Y = ck18,
+                lucid_model = "serial", CoY = covs, CoG = covs,
+                K = list(2:3, 2:3, 2), useY = TRUE, family = "normal")
> # Print the summary of the LUCID in serial model
> summary(fit18)
----------Summary of the First Component of the LUCID in Serial Model----------
----------Summary of the LUCID Early Integration Sub Model----------
K =  2 , log likelihood = -5977.843 , BIC =  13320.78
(1) Z: mean of omics data for each latent cluster
               mu_cluster1  mu_cluster2
cg_GRHL3       0.295741860 -0.410975212
cg_BTF3L4      0.021123889 -0.090792176
cg_AL358472.7  0.301523381 -0.439813049
cg_HDGF        0.170527566 -0.234364618
cg_TDRD5       0.192293276 -0.207240424
cg_CSRNP3      0.235578302 -0.242839045
cg_HSPD1      -0.002680386 -0.003986538
cg_EPM2AIP1    0.033545336  0.086638565
cg_AC025171.1  0.303125436 -0.287725955
cg_VTRNA1_3   -0.090000154  0.029899266
(2) E: odds ratio of being assigned to each latent cluster for each exposure
                                     beta        OR
hs_hg_m_scaled.cluster2         1.8300519 6.2342102
e3_sex_None.cluster2           -0.1045747 0.9007075
hs_child_age_yrs_None.cluster2 -0.4683347 0.6260439
----------Summary of the Middle Component of the LUCID in Serial Model----------
----------Summary of the LUCID Early Integration Sub Model----------
K =  2 , log likelihood = -6035.014 , BIC =  13435.13
(1) Z: mean of omics data for each latent cluster
                 mu_cluster1  mu_cluster2
tc_TC01006069_nc -0.32822703  0.321670957
tc_SLC9A4         0.10853723 -0.106428335
tc_RAB6C_AS1     -0.24384843  0.195716238
tc_LOC100129029   0.10572458 -0.005932059
tc_BRE           -0.10399915  0.125807476
tc_TC03001220_nc -0.30053137  0.248463710
tc_TC04002114_nc  0.04931376 -0.060764599
tc_TC04002369_nc -0.38991227  0.586539770
tc_BEND4         -0.26952193  0.195503719
tc_SLC9A3         0.26488011 -0.110649505
(2) E: odds ratio of being assigned to each latent cluster for each cluster
    from the last sub model
               delta       OR
G1.cluster2 1.554258 4.731575
----------Summary of the Last Component of the LUCID in Serial Model----------
----------Summary of the LUCID Early Integration Sub Model----------
K =  2 , log likelihood = -4533.659 , BIC =  10444.5
(1) Y (continuous outcome): effect size of Y for each latent cluster
(and effect of covariates if included)
                            Gamma
cluster1               0.00000000
cluster2               1.64219533
e3_sex_None            0.06626907
hs_child_age_yrs_None -0.16063912
(2) Z: mean of omics data for each latent cluster
              mu_cluster1  mu_cluster2
miR.101.3p     0.12404049  0.028070959
miR.125a.5p   -0.19583201  0.088128200
miR.125b.1.3p -0.02249475  0.064270997
miR.127.3p    -0.21746619  0.215300195
miR.140.5p     0.27069168  0.017292491
miR.142.3p     0.15068775 -0.015944126
miR.144.5p     0.25253765 -0.071952430
miR.19a.3p     0.15974513 -0.015887823
miR.19b.3p     0.20886420 -0.008073847
miR.21.5p      0.18868583 -0.017110888
(3) E: odds ratio of being assigned to each latent cluster for each cluster
    from the last sub model
                delta       OR
G1.cluster2 0.9911389 2.694301
----------Overall Summary of the LUCID in Serial model----------
log likelihood = -16546.52 , BIC =  37200.41
> # LUCID in serial with the first sub-model being LUCID in parallel
  and the second sub-model being LUCID early integration
> # Rearrange the omics list to match the new structure of the LUCID in serial model
> Z_new = list(list(omics_lst$methylome, omics_lst$transcriptome), omics_lst$miRNA)
> # Fit the LUCID in serial model with the new omics list and the new K list
> fit19 <- lucid(G = exposome, Z = Z_new, Y = ck18,
+                lucid_model = "serial", CoY = covs, CoG = covs,
+                K = list(list(2, 2), 2), useY = TRUE, family = "normal")
> # summary(fit19)
```

4 Conclusion

In this paper, we have introduced the **LUCIDus** package, version 3, to perform estimation of the LUCID model in R (Jia et al., 2024). **LUCIDus** focuses on integrative clustering analysis using multi-omics data, both with and without phenotypic traits. It consists of a set of toolkits for modeling, interpretation, visualization, inference, and prediction. Compared to the previous version, **LUCIDus** version 3 is faster (120 times faster than the previous version for a dataset with 20,000 observations), more stable, and includes several features for more functionality, including options for early, intermediate, and late integration of multi-omics data. The **LUCIDus** package is a useful tool for performing integrated clustering analysis and can potentially provide greater insights into environmental epidemiology studies with measured multi-omics data.

References

J. M. Albert, C. Geng, and S. Nelson. Causal mediation analysis with a latent mediator. *Biometrical Journal*, 58(3):535–548, 2016. [p5]

B. Aslam, M. Basit, M. A. Nisar, M. Khurshid, and M. H. Rasool. Proteomics: technologies and their applications. *Journal of Chromatographic Science*, 55(2):182–196, 2017. [p4]

J. D. Banfield and A. E. Raftery. Model-based gaussian and non-gaussian clustering. *Biometrics*, pages 803–821, 1993. [p7]

R. D. Beger. A review of applications of metabolomics in cancer. *Metabolites*, 3(3):552–574, 2013. [p4]

S. Cadiou, X. Basagaña, J. R. Gonzalez, J. Lepeule, M. Vrijheid, V. Siroux, and R. Slama. Performance of approaches relying on multidimensional intermediary data to decipher causal relationships between the exposome and health: A simulation study under various causal structures. *Environment International*, 153:106509, 2021. ISSN 0160-4120. doi: https://doi.org/10.1016/j.envint.2021.106509. URL https://www.sciencedirect.com/science/article/pii/S0160412021001343. [p15]

A. Canty and B. Ripley. *boot: Bootstrap Functions (Originally by Angelo Canty for S)*, 2021. URL https://CRAN.R-project.org/package=boot. R package version 1.3-28. [p16]

G. Celeux and G. Govaert. Gaussian parsimonious clustering models. *Pattern Recognition*, 28(5): 781–793, 1995. [p7]

C. Chan, F. Feng, J. Ottinger, D. Foster, M. West, and T. B. Kepler. Statistical mixture modeling for cell subtype identification in flow cytometry. *Cytometry Part A: The Journal of the International Society for Analytical Cytology*, 73(8):693–701, 2008. [p4]

O. M. Crook, C. M. Mulvey, P. D. Kirk, K. S. Lilley, and L. Gatto. A bayesian mixture modelling approach for spatial proteomics. *PLoS Computational Biology*, 14(11):e1006516, 2018. [p4]

C. Curtis, S. P. Shah, S.-F. Chin, G. Turashvili, O. M. Rueda, M. J. Dunning, D. Speed, A. G. Lynch, S. Samarajiwa, Y. Yuan, et al. The genomic and transcriptomic architecture of 2,000 breast tumours reveals novel subgroups. *Nature*, 486(7403):346–352, 2012. [p4]

A. C. Davison and D. V. Hinkley. *Bootstrap methods and their application*. Cambridge University Press, 1997. [p15]

A. Derkach, R. M. Pfeiffer, T.-H. Chen, and J. N. Sampson. High dimensional mediation analysis with latent variables. *Biometrics*, 75(3):745–756, 2019. [p5]

Y. Fan and C. Y. Tang. Tuning parameter selection in high dimensional penalized likelihood. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 75(3):531–552, 2013. [p13]

M. Fop and T. B. Murphy. Variable selection methods for model-based clustering. *Statistics Surveys*, 12: 18–65, 2018. [p14]

J. R. González and A. Cáceres. *Omic association studies with R and Bioconductor*. CRC Press, 2019. [p4]

S. Goodwin, J. D. McPherson, and W. R. McCombie. Coming of age: ten years of next-generation sequencing technologies. *Nature Reviews Genetics*, 17(6):333–351, 2016. [p4]

B. P. Hejblum, J. Skinner, and R. Thiébaut. Time-course gene set analysis for longitudinal gene expression data. *PLoS Computational Biology*, 11(6):e1004310, 2015. [p4]

B. P. Hejblum, C. Alkhassim, R. Gottardo, F. Caron, and R. Thiébaut. Sequential dirichlet process mixtures of multivariate skew t-distributions for model-based clustering of flow cytometry data. 2019. [p4]

Q. Jia, Y. Zhao, D. Conti, and J. Goodrich. *LUCIDus: LUCID with Multiple Omics Data*, 2024. URL `https://CRAN.R-project.org/package=LUCIDus`. R package version 3.0.2. [p5, 23]

R. Jin, R. McConnell, C. Catherine, S. Xu, D. I. Walker, N. Stratakis, D. P. Jones, G. W. Miller, C. Peng, D. V. Conti, et al. Perfluoroalkyl substances and severity of nonalcoholic fatty liver in children: an untargeted metabolomics approach. *Environment International*, 134:105220, 2020. [p5]

J. Khan, J. S. Wei, M. Ringner, L. H. Saal, M. Ladanyi, F. Westermann, F. Berthold, M. Schwab, C. R. Antonescu, C. Peterson, et al. Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. *Nature Medicine*, 7(6):673–679, 2001. [p4]

P. Kirk, J. E. Griffin, R. S. Savage, Z. Ghahramani, and D. L. Wild. Bayesian correlated clustering to integrate multiple datasets. *Bioinformatics*, 28(24):3290–3297, 2012. [p5]

E. F. Lock and D. B. Dunson. Bayesian consensus clustering. *Bioinformatics*, 29(20):2610–2616, 2013. [p5]

E. F. Lock, K. A. Hoadley, J. S. Marron, and A. B. Nobel. Joint and individual variation explained (jive) for integrated analysis of multiple data types. *The Annals of Applied Statistics*, 7(1):523, 2013. [p5]

L. Maitre, J.-B. Guimbaud, C. Warembourg, N. Güil-Oumrait, T. E. D. C. P. Consortium, P. M. Petrone, M. Chadeau-Hyam, M. Vrijheid, J. R. Gonzalez, and X. Basagaña. State-of-the-art methods for exposure-health studies: results from the exposome data challenge event. *arXiv preprint arXiv:2202.01680*, 2022. [p5]

K. Matta, T. Lefebvre, E. Vigneau, V. Cariou, P. Marchand, Y. Guitton, A.-L. Royer, S. Ploteau, B. Le Bizec, J.-P. Antignac, et al. Associations between persistent organic pollutants and endometriosis: A multiblock approach integrating metabolic and cytokine profiling. *Environment International*, 158: 106926, 2022. [p5]

Q. Mo, S. Wang, V. E. Seshan, A. B. Olshen, N. Schultz, C. Sander, R. S. Powers, M. Ladanyi, and R. Shen. Pattern discovery and cancer gene identification in integrated cancer genomic data. *Proceedings of the National Academy of Sciences*, 110(11):4245–4250, 2013. [p5]

F. Ozsolak and P. M. Milos. Rna sequencing: advances, challenges and opportunities. *Nature Reviews Genetics*, 12(2):87–98, 2011. [p4]

W. Pan and X. Shen. Penalized model-based clustering with application to variable selection. *Journal of Machine Learning Research*, 8(May):1145–1164, 2007. [p14]

C. Peng, J. Wang, I. Asante, S. Louie, R. Jin, L. Chatzi, G. Casey, D. C. Thomas, and D. V. Conti. A latent unknown clustering integrating multi-omics data (lucid) with phenotypic traits. *Bioinformatics*, 36 (3):842–850, 2020. [p5, 6]

M. Pierre-Jean, J.-F. Deleuze, E. Le Floch, and F. Mauger. Clustering and variable selection evaluation of 13 unsupervised methods for multi-omics data integration. *Briefings in Bioinformatics*, 21(6): 2011–2030, 2020. [p5]

S. Prabhakaran, E. Azizi, A. Carr, and D. Pe'er. Dirichlet process mixture model for correcting technical variation in single-cell gene expression data. In *International Conference on Machine Learning*, pages 1070–1079. PMLR, 2016. [p4]

N. Rappoport and R. Shamir. Multi-omic and multi-view clustering algorithms: review and cancer benchmark. *Nucleic Acids Research*, 46(20):10546–10562, 2018. [p4]

M. Schmidt. The sankey diagram in energy and material flow management: part ii: methodology and current applications. *Journal of Industrial Ecology*, 12(2):173–185, 2008. [p12]

L. Scrucca, M. Fop, T. B. Murphy, and A. E. Raftery. mclust 5: clustering, classification and density estimation using gaussian finite mixture models. *The R Journal*, 8(1):289, 2016. [p7, 10]

A. Singh, C. P. Shannon, B. Gautier, F. Rohart, M. Vacher, S. J. Tebbutt, and K.-A. Lê Cao. Diablo: an integrative approach for identifying key molecular drivers from multi-omics assays. *Bioinformatics*, 35(17):3055–3062, 2019. [p5]

Y. Song, X. Zhou, M. Zhang, W. Zhao, Y. Liu, S. L. Kardia, A. V. D. Roux, B. L. Needham, J. A. Smith, and B. Mukherjee. Bayesian shrinkage estimation of high dimensional causal mediation effects in omics studies. *Biometrics*, 76(3):700–710, 2020. [p5]

N. Stratakis, D. V. Conti, R. Jin, K. Margetaki, D. Valvi, A. P. Siskos, L. Maitre, E. Garcia, N. Varo, Y. Zhao, et al. Prenatal exposure to perfluoroalkyl substances associated with increased susceptibility to liver injury in children. *Hepatology*, 72(5):1758–1770, 2020. [p5]

I. Subramanian, S. Verma, S. Kumar, A. Jere, and K. Anamika. Multi-omics data integration, interpretation, and its application. *Bioinformatics and Biology Insights*, 14:1177932219899051, 2020. [p4]

A. Tenenhaus, C. Philippe, V. Guillemot, K.-A. Le Cao, J. Grill, and V. Frouin. Variable selection for generalized canonical correlation analysis. *Biostatistics*, 15(3):569–583, 2014. [p5]

R. Tibshirani. Regression selection and shrinkage via the lasso. *Journal of the Royal Statistical Society: Series B*, 58(1):267–288, 1996. [p5]

N. Tierney. *visdat: Visualising Whole Data Frames Including Missing Values*, 2023. URL `https://cran.r-project.org/package=visdat`. R package version 0.5.3. [p18]

G. Tini, L. Marchetti, C. Priami, and M.-P. Scott-Boyer. Multi-omics integration—a comparison of unsupervised clustering methodologies. *Briefings in Bioinformatics*, 20(4):1269–1279, 2019. [p4]

V. Voillet, P. Besse, L. Liaubet, M. San Cristobal, and I. González. Handling missing rows in multi-omics data integration: multiple imputation in multiple factor analysis framework. *BMC Bioinformatics*, 17 (1):1–16, 2016. [p16]

M. Vrijheid, R. Slama, O. Robinson, L. Chatzi, M. Coen, P. Van den Hazel, C. Thomsen, J. Wright, T. J. Athersuch, N. Avellana, et al. The human early-life exposome (helix): project rationale and design. *Environmental Health Perspectives*, 122(6):535–544, 2014. [p4, 5]

B. Wang, A. M. Mezlini, F. Demir, M. Fiume, Z. Tu, M. Brudno, B. Haibe-Kains, and A. Goldenberg. Similarity network fusion for aggregating data types on a genomic scale. *Nature Methods*, 11(3): 333–337, 2014. [p5]

C. P. Wild. Complementing the genome with an "exposome": the outstanding challenge of environmental exposure measurement in molecular epidemiology. *Cancer Epidemiology, Biomarkers & Prevention*, 14(8):1847–1850, 2005. [p4]

G. Yu. *dlstats: Download Stats of R Packages*, 2022. URL `https://CRAN.R-project.org/package=dlstats`. R package version 0.1.5. [p5]

M. Yuan and Y. Lin. Model selection and estimation in regression with grouped variables. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 68(1):49–67, 2006. [p5]

H. Zhang, Y. Zheng, Z. Zhang, T. Gao, B. Joyce, G. Yoon, W. Zhang, J. Schwartz, A. Just, E. Colicino, et al. Estimating and testing high-dimensional mediation effects in epigenetic studies. *Bioinformatics*, 32(20):3150–3154, 2016. [p5]

Y. Zhang, M. Li, S. Wang, S. Dai, L. Luo, E. Zhu, H. Xu, X. Zhu, C. Yao, and H. Zhou. Gaussian mixture model clustering with incomplete data. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 17(1s):1–14, 2021. [p18]

H. Zhou, W. Pan, and X. Shen. Penalized model-based clustering with unconstrained covariance matrices. *Electronic Journal of Statistics*, 3:1473, 2009. [p14]

Yinqi Zhao
Division of Biostatistics, Department of Population and Public Health Sciences, University of Southern California
2001 N Soto St., Los Angeles, CA, 90032
United States
ORCiD: 0000-0003-2413-732X
yinqiz@usc.edu

Qiran Jia
Division of Biostatistics, Department of Population and Public Health Sciences, University of Southern California
2001 N Soto St., Los Angeles, CA, 90032
United States
ORCiD: 0000-0002-0790-5967
qiranjia@usc.edu

Jesse A. Goodrich
Division of Environmental Epidemiology, Department of Population and Public Health Sciences, University of Southern California
2001 N Soto St., Los Angeles, CA, 90032
United States
ORCid: 0000-0001-6615-0472
jagoodri@usc.edu

David V. Conti
Division of Biostatistics, Department of Population and Public Health Sciences, University of Southern California
2001 N Soto St., Los Angeles, CA, 90032
United States
ORCid: 0000-0002-2941-7833
dconti@usc.med.edu