The Innovation

Article

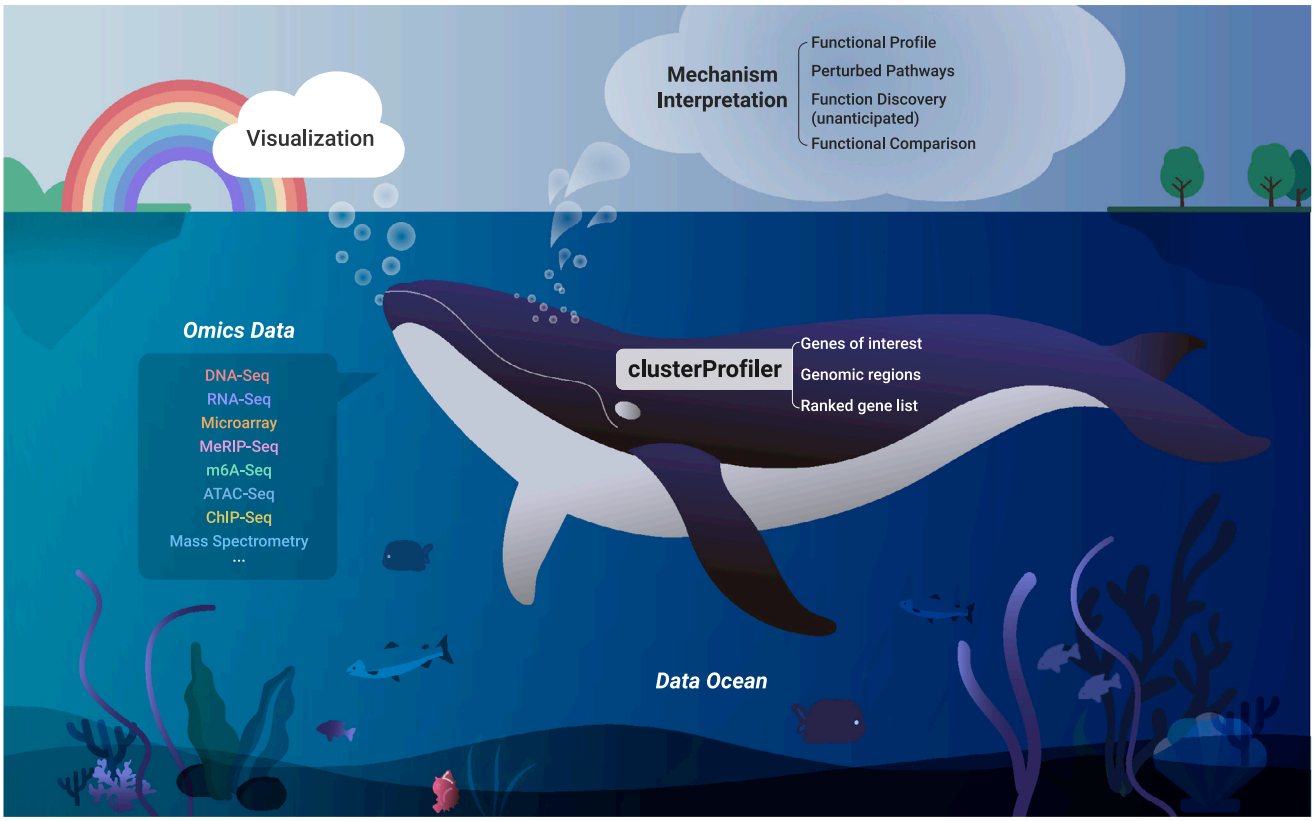
clusterProfiler 4.0: A universal enrichment tool for interpreting omics data

Tianzhi Wu,[1,5] Erqiang Hu,[1,5] Shuangbin Xu,[1] Meijun Chen,[1] Pingfan Guo,[1] Zehan Dai,[1] Tingze Feng,[1] Lang Zhou,[1] Wenli Tang,[1] Li Zhan,[1] Xiaocong Fu,[1] Shanshan Liu,[1] Xiaochen Bo,[2,*] and Guangchuang Yu[1,3,4,*]

*Correspondence: boxc@bmi.ac.cn (X.B.); gcyu1@smu.edu.cn (G.Y.)

Received: May 8, 2021; Accepted: June 29, 2021; Published Online: July 1, 2021; https://doi.org/10.1016/j.xinn.2021.100141

Graphical abstract



Public summary

- clusterProfiler supports exploring functional characteristics of both coding and non-coding genomics data for thousands of species with up-to-date gene annotation
- It provides a universal interface for gene functional annotation from a variety of sources and thus can be applied in diverse scenarios
- It provides a tidy interface to access, manipulate, and visualize enrichment results to help users achieve efficient data interpretation
- Datasets obtained from multiple treatments and time points can be analyzed and compared in a single run, easily revealing functional consensus and differences among distinct conditions

Article

clusterProfiler 4.0: A universal enrichment tool for interpreting omics data

Tianzhi Wu,[1,5] Erqiang Hu,[1,5] Shuangbin Xu,[1] Meijun Chen,[1] Pingfan Guo,[1] Zehan Dai,[1] Tingze Feng,[1] Lang Zhou,[1] Wenli Tang,[1] Li Zhan,[1] Xiaocong Fu,[1] Shanshan Liu,[1] Xiaochen Bo,[2,*] and Guangchuang Yu[1,3,4,*]

[1]Department of Bioinformatics, School of Basic Medical Sciences, Southern Medical University, Guangzhou 510515, China

[2]Department of Biotechnology, Beijing Institute of Radiation Medicine, Beijing 100850, China

[3]Guangdong Provincial Key Laboratory of Proteomics, School of Basic Medical Sciences, Southern Medical University, Guangzhou 510515, China

[4]Microbiome Medicine Center, Department of Laboratory Medicine, Zhujiang Hospital, Southern Medical University, Guangzhou 510515, China

[5]These authors contributed equally

*Correspondence: boxc@bmi.ac.cn (X.B.); gcyu1@smu.edu.cn (G.Y.)

Received: May 8, 2021; Accepted: June 29, 2021; Published Online: July 1, 2021; https://doi.org/10.1016/j.xinn.2021.100141

Citation: Wu T., Hu E., Xu S., et al., (2021). clusterProfiler 4.0: A universal enrichment tool for interpreting omics data. The Innovation **2(3)**, 100141.

Functional enrichment analysis is pivotal for interpreting high-throughput omics data in life science. It is crucial for this type of tool to use the latest annotation databases for as many organisms as possible. To meet these requirements, we present here an updated version of our popular Bioconductor package, clusterProfiler 4.0. This package has been enhanced considerably compared with its original version published 9 years ago. The new version provides a universal interface for functional enrichment analysis in thousands of organisms based on internally supported ontologies and pathways as well as annotation data provided by users or derived from online databases. It also extends the *dplyr* and *ggplot2* packages to offer tidy interfaces for data operation and visualization. Other new features include gene set enrichment analysis and comparison of enrichment results from multiple gene lists. We anticipate that clusterProfiler 4.0 will be applied to a wide range of scenarios across diverse organisms.

Keywords: clusterProfiler; biological knowledge mining; functional analysis; enrichment analysis; visualization

INTRODUCTION

Functional enrichment analysis is one of the most widely used techniques for interpreting gene lists or genome-wide regions of interest (ROIs)[1] derived from various high-throughput studies. Although many tools have been developed for gene-centric or epigenomic enrichment analysis, most are designed for model organisms or specific domains (e.g., fungi,[2] plants[3]) embedded with particular annotations such as Gene Ontology (GO) and the Kyoto Encyclopedia of Genes and Genomes (KEGG).[4] Non-model organisms and functional annotations other than GO and KEGG are poorly supported. Moreover, the increasing concerns for the quality of gene annotation have raised an alarm in biomedical research. Because annotation databases have diverse or irregular update periods, many tools may fail to update the corresponding information in time. A previous study[5] reported that about 42% of the tools were outdated by more than 5 years and that functional significance was severely underestimated, with only 26% of biological processes or pathways captured in comparison with those employing up-to-date annotation. Such negative impacts of outdated annotation can be propagated for years and can hinder follow-up studies. Reanalyzing the GTEx dataset[6] published by the ENCODE consortium using clusterProfiler uncovered a large number of new pathways, which were missed in the analysis using out-of-date annotation (https://github.com/GuangchuangYu/enrichment4GTEx_clusterProfiler), and new hypotheses were generated based on these new pathways.

The clusterProfiler library was first published in 2012[7] and designed to perform over-representation analysis (ORA)[8] using GO and KEGG for several model organisms and to compare functional profiles of various conditions on one level (e.g., different treatment groups). Since then, clusterProfiler has matured substantially and currently supports several ontology and pathway annotations, thousands of species with up-to-date gene annotation, users' annotation data for novel species, and emerging new annotations. Both ORA and gene set enrichment analysis (GSEA)[9] are supported. The comparison utility is extended to support a complex experimental design that allows comparison of functional profiles of various conditions on different levels. The clusterProfiler library has many unique features, including a tidy interface that can manipulate the enrichment result and directly support the visualization of the enrichment result using *ggplot2* (Tables 1 and S2). Moreover, we have developed several packages to complement its functionalities, including ChIPseeker to connect functional analysis with genomic ROIs,[10] GOSemSim[11] to remove redundant GO terms, and enrichplot to visualize the enrichment results. These complementary packages enable clusterProfiler to stand out among other tools. The clusterProfiler library is one of the most popular Bioconductor packages. It has been incorporated in more than 30 CRAN and Bioconductor packages (Table S1), several pipelines (e.g., The Cancer Genome Atlas [TCGA] Workflow[12] and ViralLink[13]), and online platforms (e.g., NASQAR[14] and ABioTrans[15]).

RESULTS

Gene ontology

The clusterProfiler package provides the *enrichGO* and *gseGO* functions for ORA and GSEA using GO.[16] Instead of providing species-specific GO annotation, clusterProfiler relies on genome-wide annotation packages (OrgDb) released by the Bioconductor project. There are 20 OrgDb packages available in Bioconductor for different species, such as human, mouse, fly, yeast, and worm. These packages are updated biannually. GO annotation for non-model organisms can be queried online via the AnnotationHub package, which provides web services for accessing genome-wide annotations from various data providers (e.g., UCSC, Ensembl, NCBI, STRING, and GENCODE). With the efforts from the Bioconductor community to maintain up-to-date GO annotation for model and non-model organisms, clusterProfiler supports GO analysis on more species compared with other tools. Moreover, a data frame of GO annotation (e.g., retrieve data from the BiomaRt or UniProt database using taxonomic ID) can be used to construct an OrgDb using the AnnotationForge package or directly through the universal interface for enrichment analysis.

GO terms are organized as a directed acyclic graph, in which a directed edge denotes a parent-child semantic relationship. A parent term might be significantly enriched only because it contains all the genes of a significantly over-represented child term. Consequently, the list of enriched GO terms is often too long and contains redundant terms, which hinders effective interpretation. Therefore, clusterProfiler integrates a *simplify* function to eliminate such redundant GO terms. This function employs the GOSemSim[17] package to calculate semantic similarities among enriched GO terms using multiple methods based on information content or graph structure.

Highly similar GO terms (e.g., >0.7) will be removed by applying the *simplify* function to retain a representative term (e.g., the most significant term). The following example shows an ORA on Biological Process (BP) to identify significant BP terms associated with the differentially expressed genes (DEGs). The geneList dataset, which contains fold change of gene expression levels between breast tumor and normal samples and is provided by the DOSE package, was used in this example. The DEGs were identified by a criterion of fold change >2. As demonstrated in Figure 1A, the top 30 enriched terms are highly connected, and it seems that the DEGs are associated with a single functional module. Visualizing top enriched terms is a common approach to present and interpret the enrichment result. However, the top results are dominated by a large number of highly similar terms. After removing redundant terms, the result reveals a more global view with several different functional modules (Figure 1B). This feature simplifies the enrichment results, assists in interpretation, and avoids the annotation/interpretation bias.[18]

Table 1. Major clusterProfiler functions

Function	Description
enrichGO	ORA using GO
enrichKEGG	ORA using KEGG pathway
enrichMKEGG	ORA using KEGG module
enrichWP	ORA using WikiPathways
enricher	general interface for ORA
gseGO	GSEA using GO
gseKEGG	GSEA using KEGG pathway
gseMKEGG	GSEA using KEGG module
gseWP	GSEA using WikiPathways
GSEA	general interface for GSEA
compareCluster	compare functional profiles for genes obtained from different conditions
merge_result	merge enrichment results for comparison
read.gmt	parse gene set file in GMT format
read.gmt.wp	parse WikiPathways GMT file
download_KEGG	download the latest version of the KEGG pathway/module
get_wp_organism	list supported organisms of WikiPathways
bitr	biological ID translator using OrgDb
bitr_kegg	biological ID translator using the KEGG database
setReadable	convert IDs in enrichment result to human-readable gene symbols using OrgDb
go2ont	convert GO ID to corresponding ontology (BP, CC, MF)
go2term	convert GO ID to a descriptive term
ko2name	convert KO ID to a descriptive name
buildGOmap	infer GO indirect annotation from direct annotation
browseKEGG	open specific KEGG pathway in a web browser with genes highlighted
dropGO	drop GO terms of specific level or a specific terms (mostly too general) from enrichment result
gofilter	restrict enrichment result at a specific GO level
geneInCategory	extract input genes (for ORA) or core enriched genes (for GSEA) that belong to a specific functional category
simplify	remove redundant GO terms from enrichment result
arrange	order enrichment result by the values of selected variables
filter	subset enrichment result that satisfies user conditions
group_by	group enrichment results by selected variable
mutate	add new variable to enrichment result
select	select variables in enrichment result
summarise	create summary statistics from enrichment result

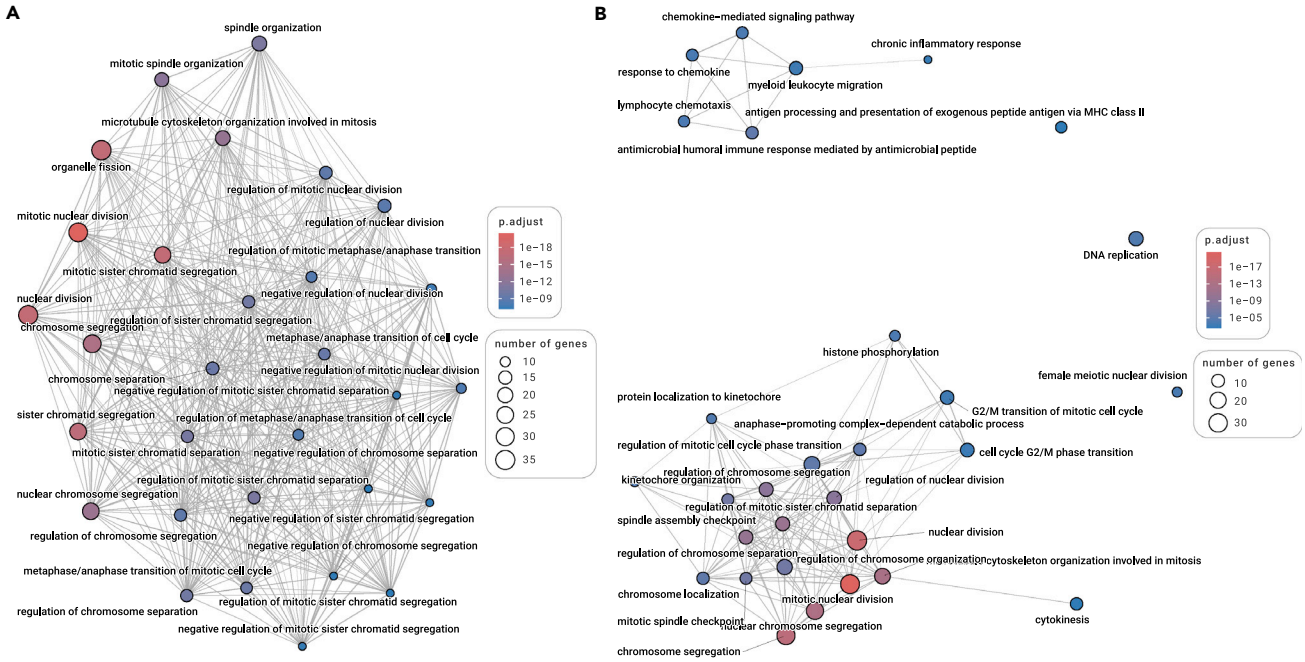


Figure 1. Gene ontology enrichment analysis The original result (A) and a simplified version (B) were visualized as enrichment map networks. Each node represents a gene set (i.e., a GO term) and each edge represents the overlap between two gene sets.

```
library(clusterProfiler)

data(geneList, package="DOSE")
## fold change > 2 as DE genes
de <- names(geneList)[abs(geneList) > 2]

ego <- enrichGO(de, OrgDb = "org.Hs.eg.db", ont="BP",
  readable=TRUE)

## use simplify to remove redundant terms
ego2 <- simplify(ego, cutoff=0.7, by="p.adjust",
  select_fun=min)
```

Kyoto encyclopedia of genes and genomes

KEGG is an encyclopedia of genes and genomes.[19] Molecular functions are represented by networks of interactions and reactions mainly in the form of KEGG pathways and modules. A KEGG module is a collection of manually defined function units. In some situations, KEGG modules have a more straightforward interpretation. Both KEGG pathways and KEGG modules are supported by clusterProfiler. Many software tools that support KEGG analysis have stopped updating since July 2011 when KEGG initiated an academic subscription model for FTP downloading. These tools use relatively old KEGG data, and the result might be inaccurate and misleading. Fortunately, the KEGG web resource is freely available. The clusterProfiler package does not pack any KEGG data. Instead, it queries the latest online KEGG database through web API to perform functional analysis. The advantage of this feature is obvious: it allows clusterProfiler to use up-to-date data and support all the species that have KEGG annotation (more than 6,000 species are listed in http://www.genome.jp/kegg/catalog/org_list.html). Moreover, clusterProfiler supports the KEGG Orthology database and can be used to perform functional characterization of the microbiomes.[20]

In the following example, GSEA was performed with KEGG pathway. Figure 2A shows the plotting of GSEA enrichment results to visualize the top five perturbed pathways, i.e., the top five highest absolute values of the normalized enrichment score (NES).[9] The NES indicates the shift of genes belonging to a certain pathway toward either end of the ranked list and represents pathway activation or suppression. To further explore the pathway crosstalk effects, we visualized gene expression distribution of core enrichment genes using an UpSet plot (Figure 2B). The result shows that the expression values of genes in the intersection of cell-cycle and DNA-replication pathways are higher than those uniquely belonging to either of the two pathways. These overlapping genes are mainly minichromosome maintenance (MCM) genes, which can potentially serve as biomarkers for tumor diagnosis.[21] The intersection of the interleukin-17 (IL-17) signaling pathway and the proteasome pathway is only associated with one gene, interferon-γ (IFN-γ). The IL-17 signaling pathway induces an inflammatory response,[22] while IFN-γ regulates proteasome formation.[23] These effects ultimately reshape the tumor microenvironment.

```
kk <- gseKEGG(geneList, organism = "hsa")
```

Universal interface for biomedical gene sets

With the advancement of the sequencing technology, the investigation into functions for transcriptomes from non-model organisms is increasingly demanded. However, most tools in this field are designed for GO and KEGG analyses with support limited to one or several model organisms. Besides, there are increasingly more biological knowledge databases available for exploring functional characteristics from different perspectives, such as Disease Ontology,[24] Reactome Pathway,[25] Medical Subject Headings,[26], and WikiPathway.[27] There is an urgent need for integration and support of these databases. To address these issues, clusterProfiler provides two general functions, *enricher* and *GSEA* for ORA and GSEA, with user-provided gene annotations. These two functions allow the application of all ontologies or pathways curated in diverse databases as the background in customized analyses. Therefore, users could easily import external annotations (e.g., electronic annotations using Blast2GO[28] and KAAS[29] for GO and KEGG annotations, respectively) for newly sequenced species. Moreover, it is convenient

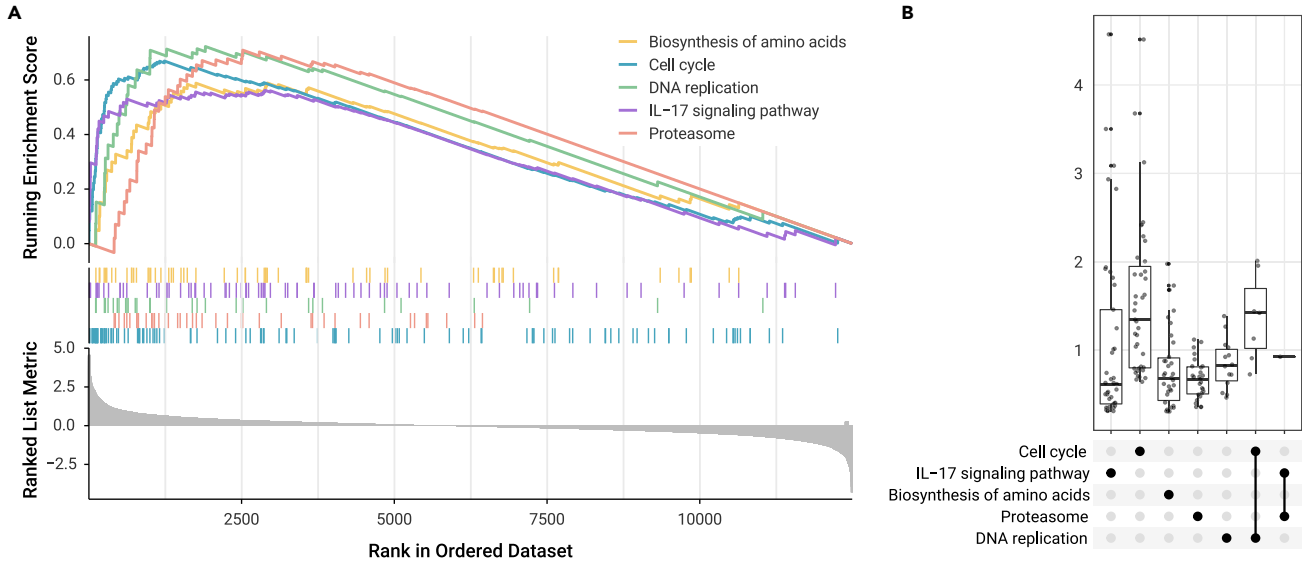


Figure 2. KEGG pathway enrichment analysis In GSEA enrichment plot (A), the curves represent the running sum of the enrichment scores, the middle part of the graph shows the position of genes that are related to certain pathways, and the bottom part of the graph displays how the metric (e.g., fold change) is distributed along with the list. The UpSet plot (B) visualizes the metric distribution of core enrichment genes. It differentiates genes that uniquely belong to a pathway or are associated with two or more pathways.

to perform functional analysis using up-to-date annotations from all popular databases, such as InterPro, Clusters of Orthologous Groups, and Mouse Phenotype Ontology, to name a few, without waiting for the updates of other tools. It would be suitable for the timely analysis of gene sets with emerging interests, such as human cell markers[30] and COVID-19-related gene sets.

The gene set annotation required by *enricher* and *GSEA* is a two-column data frame with one column representing gene set names (ID or descriptive name) and the other showing the corresponding genes. The gene matrix transposed (GMT) format is widely used to distribute gene set annotations. There are many gene set libraries available online (e.g., https://maayanlab.cloud/Enrichr/#stats), including MSigDB (Molecular Signatures Database), Disease Signatures, and CCLE (Cancer Cell Line Encyclopedia). To enable the utilization of these gene sets in clusterProfiler as the background annotation to explore the underlying biological mechanisms, clusterProfiler provides a parser function, *read.gmt*, to import GMT files that can be directly passed to the *enricher* and *GSEA* functions. In the following example, we used the *GSEA* function to perform gene set enrichment analysis using WikiPathways (Figure 5B). The annotation data were parsed by using *read.gmt.wp*, which is a customized version of *read.gmt* for importing GMT files from WikiPathways.

```
## downloaded from https://wikipathways-data.wmcloud.org/
   current/gmt/
gmt <- 'wikipathways-20210310-gmt-Homo_sapiens.gmt'
wp <- read.gmt.wp(gmt)
ewp <- GSEA(geneList, TERM2GENE=wp[,c("wpid",
   "gene")], TERM2NAME=wp[,c("wpid", "name")])
```

Functional interpretation of genomic ROIs

With the increasing availability of genomic sequences, non-coding genomic regions (e.g., *cis*-regulatory elements, non-coding RNAs, and transposons) have posed a demanding challenge to exploration of their roles in various biological processes.[1] Unlike coding genes, non-coding genomic regions are typically not well functionally annotated. Analyzing biological functions of the proximal genes is a common strategy in research on the biological meaning of a set of non-coding genomic regions. Software tools, such as the Genomic Regions Enrichment of Annotations Tool (GREAT),[31] are implemented to follow this strategy. However, these tools only support a limited number of species. For example, GREAT is designed for human and mouse only. In addition, many tools only take the host or nearest genes into consideration but ignore long-distance regulations. Our in-house developed package, ChIPseeker,[10] is originally designed for chromatin immunoprecipitation (ChIP) peak annotation, comparison, and visualization and has been employed to analyze genome-wide ROIs, such as open chromatin regions obtained by DNase-seq[32] and ATAC-seq.[33] To facilitate biological interpretation of genome-wide regions, we implemented a function, *seq2gene*, in ChIPseeker to associate genomic regions with coding genes through many-to-many mapping. It automatically maps genomic regions to host genes (either located in exon or intron), proximal genes (located in the promoter region), and flanking genes (located upstream and downstream within user-specified distance). The *seq2gene* function supports a wide variety of species if a genomic annotation, such as the TxDb (UCSC-based) or EnsDb (Ensembl-based) object, is available. After mapping genomic regions to coding genes, clusterProfiler can be employed to perform functional enrichment analysis of the coding genes to assign biological meanings to the set of genomic regions. The combination of ChIPseeker and clusterProfiler allows more biological ontology or pathway databases to be utilized to explore functions of genomic regions for a wide variety of species.

```
library(ChIPseeker)
## the file can be downloaded using 'downloadGSMbedFi-
   les("GSM1295076")'
file <- "GSM1295076_CBX6_BF_ChipSeq_mergedReps_peaks.
   bed.gz"

gr <- readPeakFile(file)
library(TxDb.Hsapiens.UCSC.hg19.knownGene)
TxDb <- TxDb.Hsapiens.UCSC.hg19.knownGene
genes <- seq2gene(gr, tssRegion=c(-1000, 1000),
   flankDistance = 3000, TxDb)
```

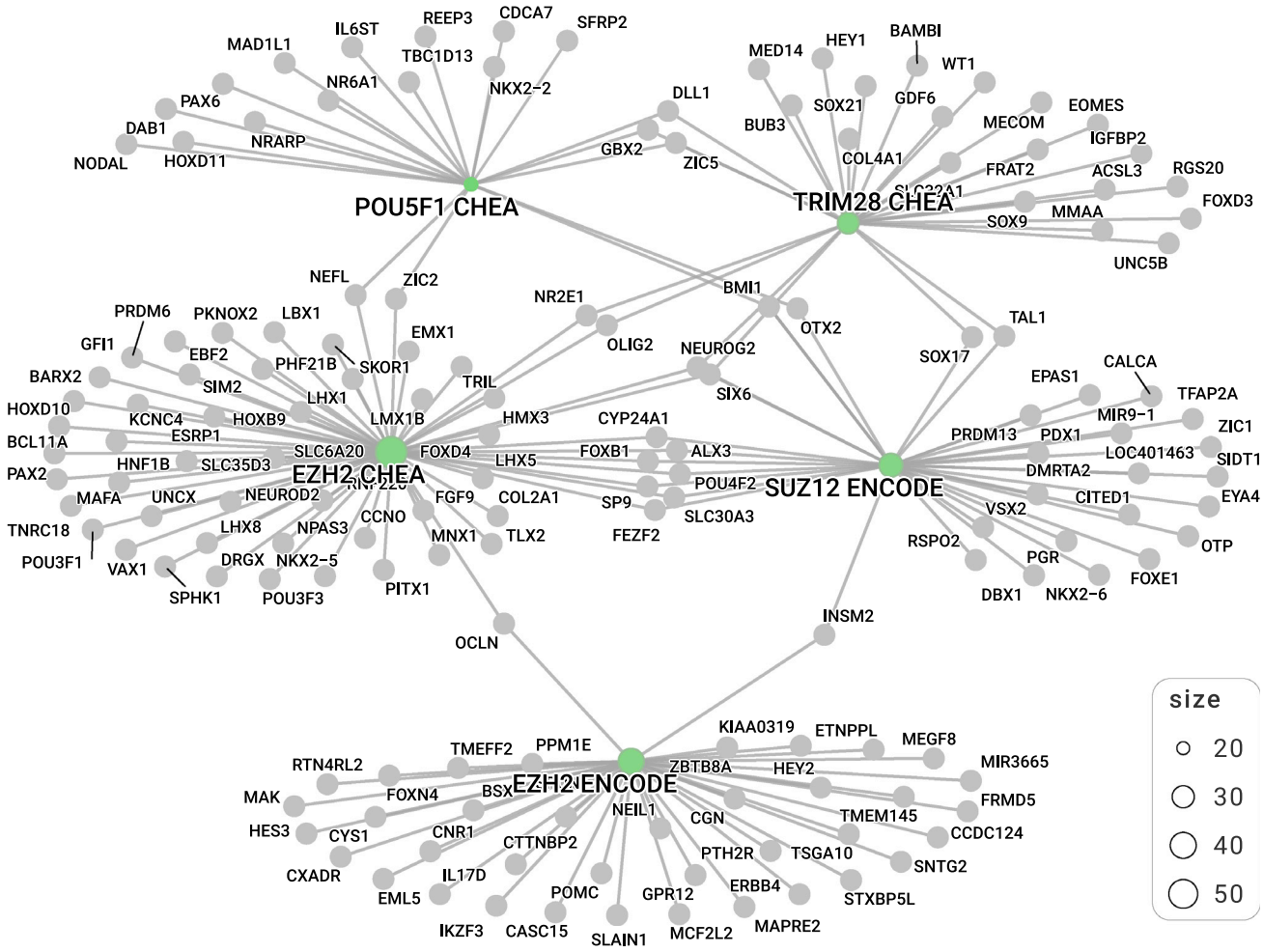


Figure 3. Functional enrichment analysis of genomic regions of interest Genomic regions are linked to coding genes, which are then used to identify transcript cofactors by testing significant overlap of target genes.

```
library(clusterProfiler)
g <- bitr(genes, 'ENTREZID', 'SYMBOL', 'org.Hs.eg.db')

## downloaded from
https://maayanlab.cloud/Enrichr/geneSetLibrary?mode=text&
libraryName=ENCODE_and_ChEA_Consensus_TFs_from_ChIP-X
encode <- read.gmt("ENCODE_and_ChEA_Consensus_
  TFs_from_ChIP-X.txt")

enricher(g$SYMBOL, TERM2GENE=encode)
```

A dataset of ChIP-seq with antibody against CBX6 (GEO: GSM1295076) was used in the above example. The genomic binding regions were mapped to coding genes using the *seq2gene* function with UCSC genomic annotation. The Entrez gene IDs were converted into gene symbols using the *bitr* function implemented in clusterProfiler. To identify and characterize transcript cofactors, we performed functional enrichment analysis using the ENCODE and ChEA transcript factor gene sets. The result was visualized as a category-gene network (Figure 3), which showed that genes associated with CBX6 (obtained by the *seq2gene* function) significantly overlap with genes regulated by POU5F1, TRIM28, SUZ12, and EZH2. OCT4 (POU5F1)[34] and KAP1 (TRIM28)[35] have been reported to interact with polycomb repressive complex 1 (PRC1), and CBX6 is a known subunit of PRC1.[36] SUZ12 and EZH2 are core components of PRC2 and negatively regulate CBX6.[37] These pieces of evidence support the effectiveness of these analyses including the mapping of genomic ROIs to coding genes and functional enrichment, which suggest that this method can be used to identify unknown cofactors (Figure 3) and characterize functions of genomic regions.

Comparison among different conditions

The clusterProfiler library is designed to allow the comparison of functional enrichment results from multiple experimental conditions or multiple time points. With an input of a collection of gene lists, the *compareCluster* function applies a function (e.g., *enricher*) with user settings to perform functional enrichment analysis for each of the gene lists and aggregates the results into a single object. Thus, enrichment results of multiple groups are easily explored and plotted together for comparison with a user-friendly interface. Comparing functional profiles can reveal functional consensus and differences among different experiments and helps in identifying differential functional modules in omics datasets. In the updated version, *compareCluster* provides a new interface supporting a formula that is widely used in R for specifying statistical models; this allows more complicated experimental designs to be supported (e.g., time-course experiment with different treatments). With the infrastructure of clusterProfiler to support a wide range of ontology and pathway annotations and multiple organisms, the comparison can be applied to many circumstances.

The dataset, DE_GSE8057, was derived from the GEO: GSE8057 dataset in the GEO database. The GSE8057 dataset contains expression data from

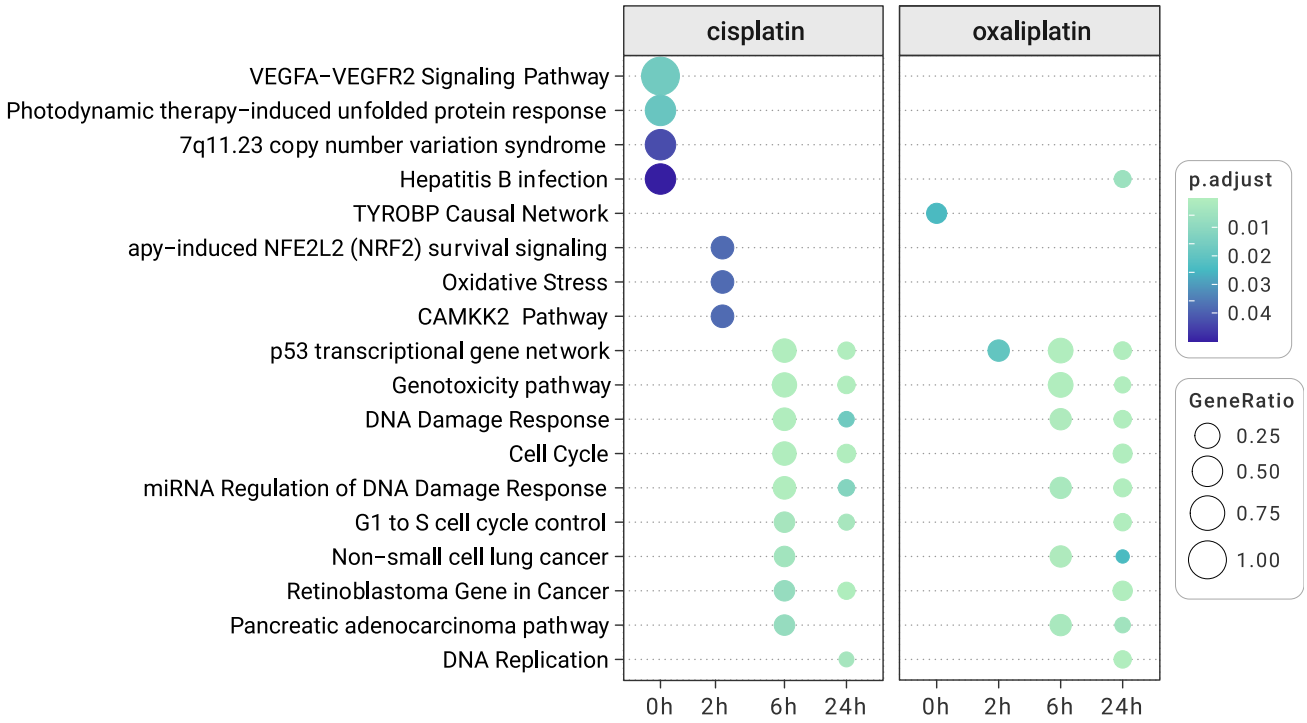


Figure 4. Comparing functional profiles among different levels of conditions The *compareCluster* function performed enrichment analysis simultaneously for eight lists of DEGs. The results were visualized as a dot plot with an x axis representing one level of conditions (time course) and a facet panel indicating another level of conditions (drug treatments).

ovarian cancer cells at multiple time points (0, 2, 6, and 24 h) and under two treatment conditions (cisplatin and oxaliplatin).[38] The DE_GSE8057 dataset contains DEGs obtained from different treatments and time points versus control samples. Eight groups of DEG lists (specified by the formula *Gene ~ Time + Treatment*) were analyzed simultaneously using *compareCluster* with WikiPathways. The result (Figure 4) indicates that the two drugs have distinct effects at the beginning but consistent effects in the later stages. Several pathways including DNA damage and cell-cycle progression were perturbed by either cisplatin or oxaliplatin drug exposure. The finding is consistent with the discovery obtained by data-driven modeling.[38]

```
data(DE_GSE8057)

xx <- compareCluster(Gene~time+treatment,
  data=DE_GSE8057, fun = enricher,
  TERM2GENE=wp[,c("wpid", "gene")],
  TERM2NAME=wp[,c("wpid", "name")])
```

Data frame interface for accessing enriched results

The outputs of ORA and GSEA are *enrichResult* and *gseaResult* objects, respectively, while the output of *compareCluster* is a *compareClusterResult* object. These S4 objects contain input data, analysis settings, and enriched results, which allow more informative data to be available for downstream interpretation and visualization. To enable easy access to the enriched result, clusterProfiler implements *as.data.frame* methods to convert the S4 objects to data frames that can be easily exported as CSV files. In addition, clusterProfiler provides a data frame interface that mimics data frame operations to access rows, columns, and subsets of rows and columns from the S4 objects of the enriched result. Users can use *head* and *tail* to print part of the result. The *nrow*, *ncol*, and *dim* methods are also supported to access basic information such as how many pathways are enriched.

```
## head or tail to print first or last n rows
head(ego, 2)
## ID Description GeneRatio BgRatio pvalue
## GO:0140014 GO:0140014 mitotic nuclear division 34/
   194 286/18866 2.171838e-26
## GO:0000280 GO:0000280 nuclear division 36/194 428/
   18866 1.099719e-22
## p.adjust qvalue
## GO:0140014 6.700119e-23 5.710790e-23
## GO:0000280 1.696316e-19 1.445841e-19
## geneID
##  GO:0140014  CDCA8/CDC20/KIF23/CENPE/MYBL2/CCNB2/
   NDC80/NCAPH/DLGAP5/UBE2C/NUSAP1/TPX2/TACC3/NEK2/
   UBE2S/CDK1/MAD2L1/KIF18A/CDT1/BIRC5/KIF11/TTK/
   NCAPG/AURKB/CHEK1/TRIP13/PRC1/KIFC1/KIF18B/
   AURKA/CCNB1/KIF4A/PTTG1/BMP4
##  GO:0000280  CDCA8/CDC20/KIF23/CENPE/MYBL2/CCNB2/
   NDC80/TOP2A/NCAPH/ASPM/DLGAP5/UBE2C/NUSAP1/TPX2/
   TACC3/NEK2/UBE2S/CDK1/MAD2L1/KIF18A/CDT1/BIRC5/
   KIF11/TTK/NCAPG/AURKB/CHEK1/TRIP13/PRC1/KIFC1/
   KIF18B/AURKA/CCNB1/KIF4A/PTTG1/BMP4
## Count
## GO:0140014 34
## GO:0000280 36
```

The [and $ operators for subsetting are also supported. We redefined the [[operator to help users access which genes are annotated by a selected

ontology or pathway. For GSEA output, the [[operator will return core enriched genes (i.e., genes in the leading edge) of the selected gene set.

```
## subset result using '[' and '$'
ego[1:2, c("ID", "Description", "pvalue", "p.adjust")]

## ID Description pvalue p.adjust
## GO:0140014 GO:0140014 mitotic nuclear division
   2.171838e-26 6.700119e-23
## GO:0000280 GO:0000280 nuclear division 1.099719e-
   22 1.696316e-19

head(ego$Description)

## [1] "mitotic nuclear division"
## [2] "nuclear division"
## [3] "organelle fission"
## [4] "mitotic sister chromatid segregation"
## [5] "sister chromatid segregation"
## [6] "chromosome segregation"

## genes annotated by specific term
ego[["GO:0140014"]]

## [1] "CDCA8" "CDC20" "KIF23" "CENPE" "MYBL2" "CCNB2"
   "NDC80" "NCAPH"
## [9] "DLGAP5" "UBE2C" "NUSAP1" "TPX2" "TACC3" "NEK2"
   "UBE2S" "CDK1"
## [17] "MAD2L1" "KIF18A" "CDT1" "BIRC5" "KIF11" "TTK"
   "NCAPG" "AURKB"
## [25] "CHEK1" "TRIP13" "PRC1" "KIFC1" "KIF18B"
   "AURKA" "CCNB1" "KIF4A"
## [33] "PTTG1" "BMP4"
```

Tidy interface for data operation

To facilitate data manipulation and exploration of the enrichment result, clusterProfiler extends the *dplyr* verbs to support *enrichResult*, *gseaResult*, and *compareClusterResult* objects. Following the concept of tidiness, these verbs provide robust and standardized operations for data transformation and can be assembled into a workflow using the pipe operator (%>%). This allows users to explore the results effectively and develop reproducible and human-readable pipelines. For example, it allows the filtering of enriched results using different criteria (e.g., adjusted p values less than 0.001, and the number of input genes annotated to the enriched term should be greater than 10).

```
dim(ego)

## [1] 197 9

ego2 <- filter(ego, p.adjust < 0.001, Count > 10)
dim(ego2)

## [1] 44 9
```

For ORA results, clusterProfiler provides *geneRatio* (ratio of input genes that are annotated in a term) and *BgRatio* (ratio of all genes that are annotated in this term). However, other concepts are widely used to help in interpreting enrichment results, such as the rich factor and fold enrichment. A rich factor is defined as the ratio of input genes (e.g., DEGs) that are annotated in a term to all genes that are annotated in this term. The fold enrichment is defined as the ratio of the frequency of input genes annotated in a term to the frequency of all genes annotated to that term, and it is easy to calculate by dividing *geneRatio* by *BgRatio*. Here, as an example, we used the *mutate* verb to create a new column of *richFactor* based on information available in the clusterProfiler output.

```
ego3 <- mutate(ego, richFactor = Count / as.numeric
   (sub("/\\d+", "", BgRatio)))
```

The following example uses the GSEA enrichment result generated in the previous session. The result was sorted by absolute values of NESs using the *arrange* verb. NES is an indicator to interpret the degree of enrichment. A positive NES indicates that members of the gene set tend to appear at the top of the rank (pathway activation), and a negative NES indicates the opposite circumstance (pathway suppression). We used the *group_by* verb to group the result based on the sign of NES, and the *slice* verb was used to extract the first five enriched pathways for each group (i.e., five activated pathways that have the largest NES values and five suppressed pathways that have the smallest NES values). These verbs return the same object type as their input and do not affect downstream analysis and visualization.

```
ewp2 <- arrange(ewp, desc(abs(NES))) %>%
        group_by(sign(NES)) %>%
        slice(1:5)
```

Visualization using Ggplot2

The *enrichplot* package is originally derived from DOSE and clusterProfiler packages and serves as a *de facto* visualization tool for visualizing enrichment results for outputs from clusterProfiler as well as DOSE, ReactomePA, and meshes. These methods allow users without programming skills to generate effective visualization to explore and interpret results. All the visualization methods implemented are based on *ggplot2*, which allows customization using the grammar of graphics. Moreover, we also extend *ggplot2* to support enrichment results so that users can use the *ggplot2* syntax directly to visualize enrichment results. The following example demonstrates the application of *ggplot2* grammar of graphics to visualize the GO enrichment result (ORA) as a lollipop chart using the rich factor that was generated in the previous session using the *dplyr* verbs (Figure 5A).

```
library(ggplot2)
library(forcats)

ggplot(ego3, showCategory = 10,
        aes(richFactor,
        fct_reorder(Description, richFactor))) +
  geom_segment(aes(xend=0, yend = Description)) +
  geom_point(aes(color=p.adjust, size = Count)) +
  scale_color_gradientn
                (colours=c("#f7ca64", "#46bac2",
                "#7e62a3"),
                trans = "log10",
                guide=guide_colorbar(reverse=TRUE,
                order=1)) +
  scale_size_continuous(range=c(2, 10)) +
```

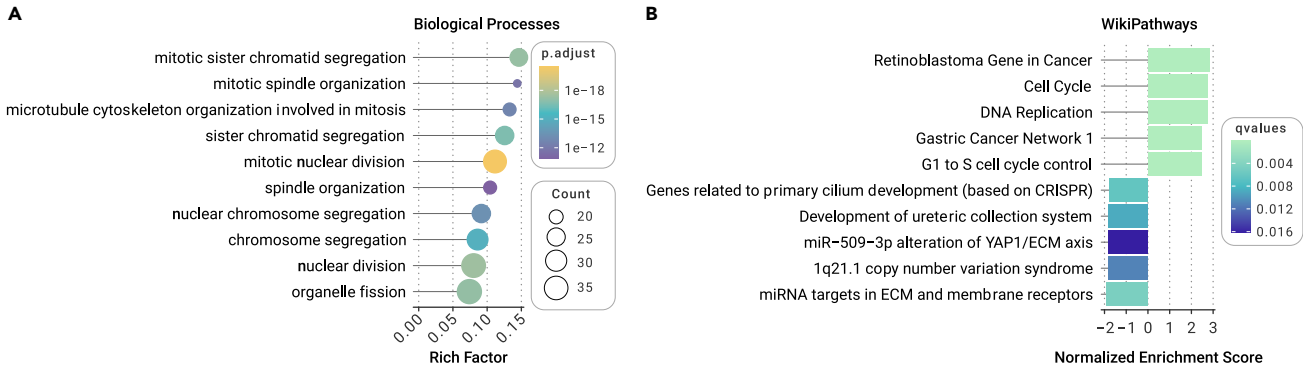


Figure 5. Visualizing enrichment results using *ggplot2* A lollipop chart to visualize the rich factors from ORA (A) and a bar chart to visualize normalized enrichment scores from GSEA (B).

```
  theme_dose(12) +
  xlab("Rich Factor") +
  lab(NULL) +
  ggtitle("Biological Processes")
```

In Figure 5B, the most significant activated and suppressed pathways (GSEA) selected by a series of *dplyr* verb operations are visualized as a bar chart using the *ggplot2* syntax. Although visualization methods used to generate Figure 5 are not provided in clusterProfiler, it is easy to generate such graphs using the tidy interface and *ggplot2*. The combination of the tidy interface for data wrangling and the support of *ggplot2* for visualization creates many possibilities for users to explore and visualize enrichment results using consistent grammar. It allows novel exploratory approaches to reveal unanticipated mechanisms as well as prototyping new visualization methods.

```
ggplot(ewp2, showCategory=10,
       aes(NES, fct_reorder(Description, NES),
         fill=qvalues)) +
  geom_col() +
  scale_fill_gradientn(colours=c("#b3eebe",
     "#46bac2", "#371ea3"),
    guide=guide_colorbar(reverse=TRUE)) +
  theme_dose(12) +
  xlab("Normalized Enrichment Score") +
  ylab(NULL) +
  ggtitle("WikiPathways")
```

Package interoperability

The clusterProfiler package is a versatile tool for enrichment analysis. It is developed within the Bioconductor ecosystem and has become an essential part of this ecosystem. Currently there are more than 30 R packages that rely on clusterProfiler to perform functional analysis for different topics, especially for cancer research. GO analysis relies on GO annotation maintained by the community, and the enrichment analysis for genomic regions relies on genomic annotation maintained by UCSC and Ensembl. There are R packages that contain gene set annotation (e.g., msigdbr) and R client libraries for accessing pathway data (e.g., rWikiPathways). These data can be used directly as background annotation in clusterProfiler through the universal interface to characterize the functional profile of omics data. The ORA algorithm is implemented in the DOSE package[21] developed in-house, and the GSEA algorithm is implemented in DOSE and fgsea[39] packages.

Our team has developed several packages to complement the functionality of clusterProfiler. ChIPseeker[10] bridges the genomic region with functional enrichment by annotating the genomic region to associated genes. GOSemSim[11] provides more than five methods for measuring semantic similarity. It allows removal of redundant terms using semantic similarities among GO terms and allows enrichment results to be visualized in semantic space so that similar terms cluster together. The DOSE[24] package supports functional enrichment from the disease perspective, including disease ontology, the network of cancer genes, and disease gene network. The ReactomePA[25] and meshes[26] packages support functional analysis using Reactome Pathways and Medical Subject Headings, respectively. DOSE, ReactomePA, and meshes are developed within the framework of clusterProfiler, and the enrichment analysis functions provided in these packages can be used in *compareCluster* for the comparison of functional profiles under various conditions and at different time points. The enrichplot package provides several visualization methods to generate publication-quality figures to help users interpret the results (Figures 1, 2, 3, and 4; supplemental information). This package suite provides a comprehensive set of tools for mining biological knowledge to elucidate and interpret molecular mechanisms (Figure 6).

DISCUSSION AND CONCLUSIONS

Pathway enrichment analysis is an essential step toward identifying biological themes that are most characteristic of high-throughput sequencing data. The clusterProfiler library provides a set of functions to unveil biological functions and pathways. Compared with many other tools that do not update background annotation databases in timely fashion and only support a limited number of organisms, clusterProfiler uses up-to-date biological knowledge of genes and biological processes (GO and KEGG) and supports thousands of organisms. In addition, clusterProfiler provides a universal interface for functional analysis with user-provided annotations. This creates the possibility to apply clusterProfiler on functional characterization of different types of data with different biological knowledge. The tidy interface provided in clusterProfiler harmonizes data structures and workflows and makes it easier for the community to develop modular manipulation, visualization, and analysis methods to supplement the existing ecosystem. clusterProfiler has already been integrated into more than 30 packages to perform functional analysis on data obtained using different techniques, including ATAC-seq, multi-region sequencing (MRS), CRISPR/Cas9 screens, and mass spectrometry (Table S1). The clusterProfiler package can be easily integrated into analysis pipelines. For example, the Gene Ontology Meta Annotator for Plants (GOMAP) is optimized for GO annotation of large, repetitive plant genomes.[40] Users can develop a pipeline to combine GOMAP with

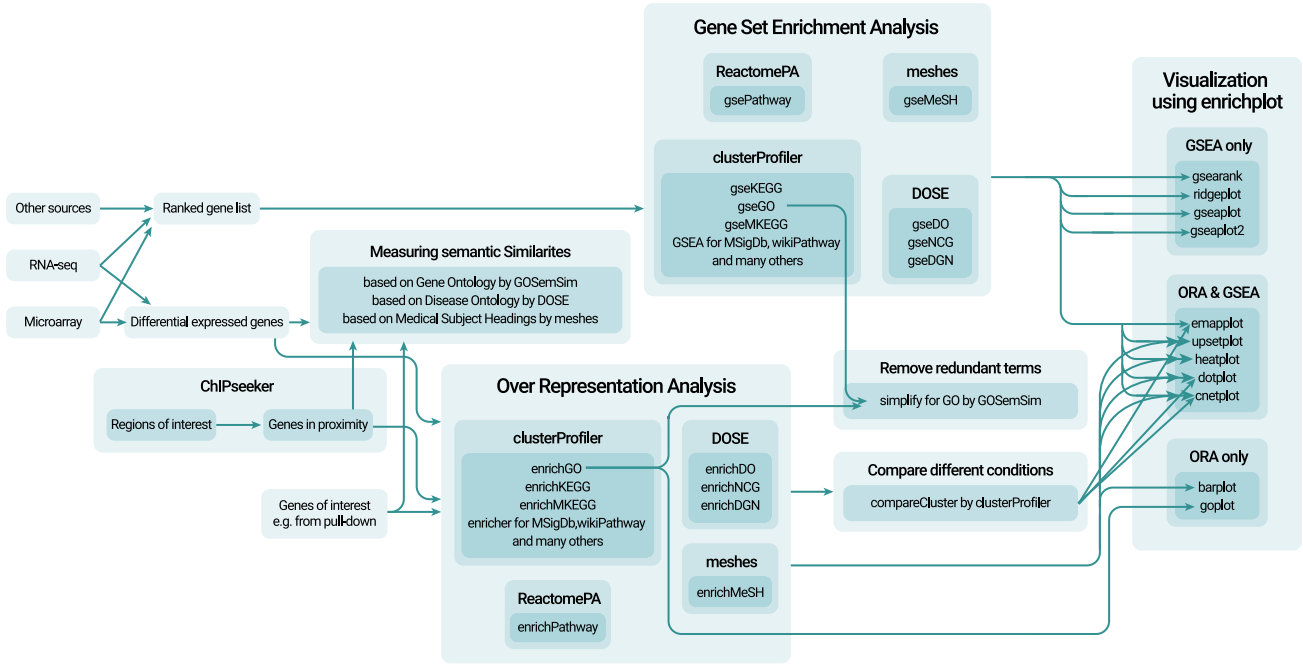


Figure 6. A package suite for mining biological knowledge clusterProfiler is an essential core for functional analysis, the functionalities of which are enhanced by several companion packages.

clusterProfiler to characterize the functionality of sequencing data from plant species, including wheat, maize, soy, and rice. The clusterProfiler library has been incorporated into different pipelines, such as TCGA Workflow,[12] recount workflow,[41] RNASeqR,[42] and MAGeCKFlute.[43]

clusterProfiler 4.0 contains several new features, including the tidy interface and the compatibility of using *ggplot2* for visualization. There is no API change for functional enrichment analyses, and this version is fully compatible with downstream packages (Table S1). After long-term maintenance, clusterProfiler is mature and unlikely to introduce significant API changes in future development. In the event of an API change, we will maintain backward compatibility for at least 1 year and provide a warning message about the change. The clusterProfiler library is freely available at https://www.bioconductor.org/packages/clusterProfiler. The development version of clusterProfiler is hosted on GitHub (https://github.com/YuLab-SMU/clusterProfiler), with many active users. A complete reference of the package suite (Figure 6) is available in the online book, https://yulab-smu.top/biomedical-knowledge-mining-book/, with many examples and detailed explanations on biological knowledge mining. Source codes to reproduce Figures 1, 2, 3, 4, and 5, as well as detailed information about the datasets used in the examples, are available in supplemental information. The clusterProfiler library is one of the popular tools used in functional enrichment analysis (more than 2,500 citations in 2020 according to Google Scholar), and we anticipate that clusterProfiler will continue to be a valuable resource to support the discovery of mechanistic insights and improve our understanding of health and disease.

REFERENCES

1. Dozmorov, M.G. (2017). Epigenomic annotation-based interpretation of genomic data: from enrichment analysis to machine learning. Bioinformatics ***33***, 3323–3330.
2. Priebe, S., Kreisel, C., Horn, F., et al. (2015). FungiFun2: a comprehensive online resource for systematic analysis of gene lists from fungal species. Bioinformatics ***31***, 445–446.
3. Yi, X., Du, Z., and Su, Z. (2013). PlantGSEA: a gene set enrichment analysis toolkit for plant community. Nucleic Acids Res. ***41***, W98–W103.
4. Nam, D., and Kim, S.-Y. (2008). Gene-set approach for expression pattern analysis. Brief. Bioinform. ***9***, 189–197.
5. Wadi, L., Meyer, M., Weiser, J., et al. (2016). Impact of outdated gene annotations on pathway enrichment analysis. Nat. Methods ***13***, 705–706.
6. Melé, M., Ferreira, P.G., Reverter, F., et al. (2015). The human transcriptome across tissues and individuals. Science ***348***, 660–665.
7. Yu, G., Wang, L.-G., Han, Y., et al. (2012). clusterProfiler: an R Package for comparing biological themes among gene clusters. OMICS J. Integr. Biol. ***16***, 284–287.
8. Boyle, E.I., Weng, S., Gollub, J., et al. (2004). GO::TermFinder—open source software for accessing Gene Ontology information and finding significantly enriched Gene Ontology terms associated with a list of genes. Bioinformatics ***20***, 3710–3715.
9. Subramanian, A., Tamayo, P., Mootha, V.K., et al. (2005). Gene set enrichment analysis: a knowledge-based approach for interpreting genome-wide expression profiles. Proc. Natl. Acad. Sci. U S A ***102***, 15545–15550.
10. Yu, G., Wang, L.-G., and He, Q.-Y. (2015). ChIPseeker: an R/Bioconductor package for ChIP peak annotation, comparison and visualization. Bioinformatics ***31***, 2382–2383.
11. Yu, G. (2020). Gene ontology semantic similarity analysis using GOSemSim. Methods Mol. Biol. Clifton NJ ***2117***, 207–215.
12. Silva, T.C., Colaprico, A., Olsen, C., et al. (2016). TCGA Workflow: analyze cancer genomics and epigenomics data using Bioconductor packages. F1000Research. ***5***, 1542.
13. Treveil, A., Bohar, B., Sudhakar, P., et al. (2021). ViralLink: an integrated workflow to investigate the effect of SARS-CoV-2 on intracellular signalling and regulatory pathways. PLOS Comput. Biol. ***17***, e1008685.
14. Yousif, A., Drou, N., Rowe, J., et al. (2020). NASQAR: a web-based platform for high-throughput sequencing data analysis and visualization. BMC Bioinformatics ***21***, 267.
15. Zou, Y., Bui, T.T., and Selvarajoo, K. (2019). ABioTrans: a biostatistical tool for transcriptomics analysis. Front. Genet. ***10***, 499.
16. The Gene Ontology Consortium (2019). The gene ontology resource: 20 years and still GOing strong. Nucleic Acids Res. ***47***, D330–D338.
17. Yu, G., Li, F., Qin, Y., et al. (2010). GOSemSim: an R package for measuring semantic similarity among GO terms and gene products. Bioinformatics ***26***, 976–978.
18. Haynes, W.A., Tomczak, A., and Khatri, P. (2018). Gene annotation bias impedes biomedical research. Sci. Rep. ***8***, 1362.
19. Kanehisa, M., Furumichi, M., Tanabe, M., et al. (2017). KEGG: new perspectives on genomes, pathways, diseases and drugs. Nucleic Acids Res. ***45***, D353–D361.
20. Zolti, A., Green, S.J., Sela, N., et al. (2020). The microbiome as a biosensor: functional profiles elucidate hidden stress in hosts. Microbiome ***8***, 71.
21. Alison, M.R., Hunt, T., and Forbes, S.J. (2002). Minichromosome maintenance (MCM) proteins may be pre-cancer markers. Gut ***50***, 290–291.
22. Miossec, P. (2021). Local and systemic effects of IL-17 in joint inflammation: a historical perspective from discovery to targeting. Cell. Mol. Immunol. ***18***, 860–865.
23. Heink, S., Ludwig, D., Kloetzel, P.-M., et al. (2005). IFN-γ-induced immune adaptation of the proteasome system is an accelerated and transient response. Proc. Natl. Acad. Sci. U S A ***102***, 9241–9246.
24. Yu, G., Wang, L.-G., Yan, G.-R., et al. (2015). DOSE: an R/Bioconductor package for disease ontology semantic and enrichment analysis. Bioinformatics ***31***, 608–609.

25. Yu, G., and He, Q.-Y. (2016). ReactomePA: an R/Bioconductor package for reactome pathway analysis and visualization. Mol. Biosyst. ***12***, 477–479.
26. Yu, G. (2018). Using meshes for MeSH term enrichment and semantic analyses. Bioinformatics ***34***, 3766–3767.
27. Martens, M., Ammar, A., Riutta, A., et al. (2021). WikiPathways: connecting communities. Nucleic Acids Res. ***49***, D613–D621.
28. Conesa, A., Götz, S., García-Gómez, J.M., et al. (2005). Blast2GO: a universal tool for annotation, visualization and analysis in functional genomics research. Bioinformatics ***21***, 3674–3676.
29. Moriya, Y., Itoh, M., Okuda, S., et al. (2007). KAAS: an automatic genome annotation and pathway reconstruction server. Nucleic Acids Res. ***35***, W182–W185.
30. Zhang, X., Lan, Y., Xu, J., et al. (2019). CellMarker: a manually curated resource of cell markers in human and mouse. Nucleic Acids Res. ***47***, D721–D728.
31. McLean, C.Y., Bristor, D., Hiller, M., et al. (2010). GREAT improves functional interpretation of cis-regulatory regions. Nat. Biotechnol. ***28***, 495–501.
32. Liu, Y., Fu, L., Kaufmann, K., et al. (2019). A practical guide for DNase-seq data analysis: from data management to common applications. Brief. Bioinform ***20***, 1865–1877.
33. Yin, S., Lu, K., Tan, T., et al. (2020). Transcriptomic and open chromatin atlas of high-resolution anatomical regions in the rhesus macaque brain. Nat. Commun. ***11***, 474.
34. Oliviero, G., Munawar, N., Watson, A., et al. (2015). The variant Polycomb Repressor Complex 1 component PCGF1 interacts with a pluripotency sub-network that includes DPPA4, a regulator of embryogenesis. Sci. Rep. ***5***, 18388.
35. Cheng, B., Ren, X., and Kerppola, T.K. (2014). KAP1 represses differentiation-inducible genes in embryonic stem cells through cooperative binding with PRC1 and derepresses pluripotency-associated genes. Mol. Cell. Biol. ***34***, 2075–2091.
36. Santanach, A., Blanco, E., Jiang, H., et al. (2017). The Polycomb group protein CBX6 is an essential regulator of embryonic stem cell identity. Nat. Commun. ***8***, 1235.
37. Deng, H., Guan, X., Gong, L., et al. (2019). CBX6 is negatively regulated by EZH2 and plays a potential tumor suppressor role in breast cancer. Sci. Rep. ***9***, 197.
38. Brun, Y.F., Varma, R., Hector, S.M., et al. (2008). Simultaneous modeling of concentration-effect and time-course patterns in gene expression data from microarrays. Cancer Genomics Proteomics ***5***, 43–53.
39. Korotkevich, G., Sukhov, V., Budin, N., et al. (2021). Fast gene set enrichment analysis. bioRxiv. https://doi.org/10.1101/060012.
40. Wimalanathan, K., and Lawrence-Dill, C.J. (2021). Gene ontology Meta annotator for plants (GOMAP). bioRxiv. https://doi.org/10.1101/809988.
41. Collado-Torres, L., Nellore, A., and Jaffe, A.E. (2017). Recount workflow: accessing over 70,000 human RNA-seq samples with Bioconductor. F1000Research. ***6***, 1558.
42. Chao, K.-H., Hsiao, Y.-W., Lee, Y.-F., et al. (2019). RNASeqR: an R package for automated two-group RNA-Seq analysis workflow. IEEE/ACM Trans. Comput. Biol. Bioinform. https://doi.org/10.1109/TCBB.2019.2956708.
43. Wang, B., Wang, M., Zhang, W., et al. (2019). Integrative analysis of pooled CRISPR genetic screens using MAGeCKFlute. Nat. Protoc. ***14***, 756–780.

ACKNOWLEDGMENTS

This work was supported by a startup fund from Southern Medical University.

AUTHOR CONTRIBUTIONS

G.Y. and X.B. planned the study, analyzed and interpreted the data, and drafted the manuscript. T.W. and E.H. analyzed and interpreted the data, and revised the manuscript. S.X., M.C., and P.G. were responsible for data collection and data analysis, and revised the manuscript. Z.D., T.F., and L.Z. contributed to data analysis and interpretation. W.T., L.Z., X.F., and S.L. participated in data analysis and manuscript revision. All authors have given final approval for the manuscript to be published and have agreed to be responsible for all aspects of the manuscript.

DECLARATION OF INTERESTS

The authors declare no competing interests.

SUPPLEMENTAL INFORMATION

Supplemental information can be found online at https://doi.org/10.1016/j.xinn.2021.100141.

LEAD CONTACT WEBSITE

https://yulab-smu.top.

The Innovation, Volume *2*

Supplemental Information

clusterProfiler 4.0: A universal enrichment tool for interpreting omics data

Tianzhi Wu, Erqiang Hu, Shuangbin Xu, Meijun Chen, Pingfan Guo, Zehan Dai, Tingze Feng, Lang Zhou, Wenli Tang, Li Zhan, Xiaocong Fu, Shanshan Liu, Xiaochen Bo, and Guangchuang Yu

Supplemental File of

clusterProfiler 4.0: A universal enrichment tool for interpreting omics data

Tianzhi Wu[§], Erqiang Hu[§], Shuangbin Xu, Meijun Chen, Pingfan Guo, Zehan Dai, Tingze Feng, Lang Zhou, Wenli Tang, Li Zhan, Xiaocong Fu, Shanshan Liu, Xiaochen Bo[*] and Guangchuang Yu[*]

[*]correspondence: Guangchuang Yu <gcyu1@smu.edu.cn> and Xiaochen Bo <boxc@bmi.ac.cn>

1 Installation

To install `clusterProfiler` package, please enter the following command in R:

```
if (!requireNamespace("BiocManager", quietly = TRUE))
    install.packages("BiocManager")
BiocManager::install("clusterProfiler")
```

To reproduce examples in this document, you need to install several extra packages:

```
install.packages(c("forcats", "ggplot2", "ggnewscale", "ggupset"))
BiocManager::install(c("org.Hs.eg.db", "enrichplot",
            "ChIPseeker", "TxDb.Hsapiens.UCSC.hg19.knownGene"))
```

2 Docker image

To help users to build the computing environment, we also provided a docker image[1]. Users can pull and run it according to the following commands. They don't need to install the dependency packages.

1. Install Docker (https://www.docker.com/). For example:

```
# Terminal of Ubuntu
sudo apt-get install docker.io
```

2. Pull the Docker image from Docker Hub:

```
# Terminal of Ubuntu
sudo docker pull xushuangbin/clusterprofilerdocker:latest
```

3. Run the image:

```
# Terminal of Ubuntu
sudo docker run -e PASSWORD=yourpassword -p 8787:8787 xushuangbin/clusterprofilerdocker
```

4. Log in to RStudio at `http://localhost:8787` using username `rstudio` and password `yourpassword`. For Windows users, you also need to provide your IP address, you can find it using `docker-machine ip default`. Inside the RStudio, you can run the examples provided in this document.

Besides, the clusterProfiler package can be installed in virtual environment using `conda`, see also https://anaconda.org/bioconda/bioconductor-clusterprofiler.

3 Bioinformatics tools that depends on clusterProfiler

The `clusterProfiler` library is one of the fundamental packages and it had been incorporated in more than thirty R packages (in CRAN or Bioconductor) to perform functional enrichment analysis for different topics, especially for cancer research (Table S1).

```
db <- utils::available.packages(repo=BiocManager::repositories())
pkgs <- tools::package_dependencies('clusterProfiler', db=db,
              which = c("Depends", "Imports"), reverse=TRUE)[[1]]
sort(pkgs)
```

```
##  [1] "AutoPipe"        "bioCancer"       "CEMiTool"        "CeTF"
##  [5] "conclus"         "DAPAR"           "debrowser"       "eegc"
##  [9] "enrichTF"        "esATAC"          "ExpHunterSuite"  "famat"
```

[1]https://hub.docker.com/r/xushuangbin/clusterprofilerdocker

```
## [13] "fcoex"           "GDCRNATools"     "immcp"           "IRISFGM"
## [17] "maEndToEnd"      "MAGeCKFlute"     "methylGSA"       "miRspongeR"
## [21] "MoonlightR"      "multiSight"      "netboxr"         "PFP"
## [25] "recountWorkflow" "RNASeqR"         "RVA"             "signatureSearch"
## [29] "TCGAbiolinksGUI" "TCGAWorkflow"    "TimiRGeN"
```

Table S1: R packages that rely on clusterProfiler to perform functional analysis.

Package	Description
AutoPipe	Automated Transcriptome Classifier Pipeline: Comprehensive Transcriptome Analysis
bioCancer	Interactive Multi-Omics Cancers Data Visualization and Analysis
CEMiTool	Co-expression Modules identification Tool
CeTF	Coexpression for Transcription Factors using Regulatory Impact Factors and Partial Correlation and Information Theory analysis
conclus	ScRNA-seq Workflow CONCLUS - From CONsensus CLUSters To A Meaningful CONCLUSion
DAPAR	Tools for the Differential Analysis of Proteins Abundance with R
debrowser	Interactive Differential Expresion Analysis Browser
eegc	Engineering Evaluation by Gene Categorization (eegc)
enrichTF	Transcription Factors Enrichment Analysis
esATAC	An Easy-to-use Systematic pipeline for ATACseq data analysis
ExpHunterSuite	Package For The Comprehensive Analysis Of Transcriptomic Data
famat	Functional analysis of metabolic and transcriptomic data
fcoex	FCBF-based Co-Expression Networks for Single Cells
GDCRNATools	an R/Bioconductor package for integrative analysis of lncRNA, mRNA, and miRNA data in GDC
immcp	Candidate Prescriptions Discovery Based on Pathway Fingerprint
IRISFGM	Comprehensive Analysis of Gene Interactivity Networks Based on Single-Cell RNA-Seq
maEndToEnd	An end to end workflow for differential gene expression using Affymetrix microarrays
MAGeCKFlute	Integrative Analysis Pipeline for Pooled CRISPR Functional Genetic Screens
methylGSA	Gene Set Analysis Using the Outcome of Differential Methylation
miRspongeR	Identification and analysis of miRNA sponge interaction networks and modules
MoonlightR	Identify oncogenes and tumor suppressor genes from omics data
multiSight	Multi-omics Classification, Functional Enrichment and Network Inference analysis
netboxr	netboxr
PFP	Pathway Fingerprint Framework in R
recountWorkflow	recount workflow: accessing over 70,000 human RNA-seq samples with Bioconductor
RNASeqR	an R package for automated two-group RNA-Seq analysis workflow
RVA	RNAseq Visualization Automation
signatureSearch	Environment for Gene Expression Searching Combined with Functional Enrichment Analysis
TCGAbiolinksGUI	TCGAbiolinksGUI: A Graphical User Interface to analyze cancer molecular and clinical data
TCGAWorkflow	TCGA Workflow Analyze cancer genomics and epigenomics data using Bioconductor packages
TimiRGeN	Time sensitive microRNA-mRNA integration, analysis and network generation tool

Moreover, clusterProfiler has been incorporated into different workflows and analysis websites (including shiny apps).

Workflows that incorporates `clusterProfiler`:

- TCGA Workflow: Analyze cancer genomics and epigenomics data using Bioconductor packages[2]
- Microbe-Flow: a comprehensive workflow for bacterial genomics, pathogenomics and genomic epidemiology[3]
- ViralLink: An integrated workflow to investigate the effect of SARS-CoV-2 on intracellular signalling and regulatory pathways[4]
- Integrative analysis of pooled CRISPR genetic screens using MAGeCKFlute[5]
- MUSIC: Model-based Understanding of SIngle-cell CRISPR screening[6]
- An end to end workflow for differential gene expression using Affymetrix microarrays[7]
- recount workflow: Accessing over 70,000 human RNA-seq samples with Bioconductor[8]
- RNAseq workflow[9]
- RNAseq Analysis[10]

[2]https://f1000research.com/articles/5-1542
[3]https://neatseq-flow.readthedocs.io/projects/neatseq-flow-modules/en/latest/Workflow_docs/Microbe-Flow.html
[4]https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1008685
[5]https://www.nature.com/articles/s41596-018-0113-7
[6]https://github.com/bm2-lab/MUSIC
[7]https://f1000research.com/articles/5-1384/v2
[8]https://f1000research.com/articles/6-1558
[9]https://github.com/twbattaglia/RNAseq-workflow
[10]https://learn.gencore.bio.nyu.edu/rna-seq-analysis/gene-set-enrichment-analysis/

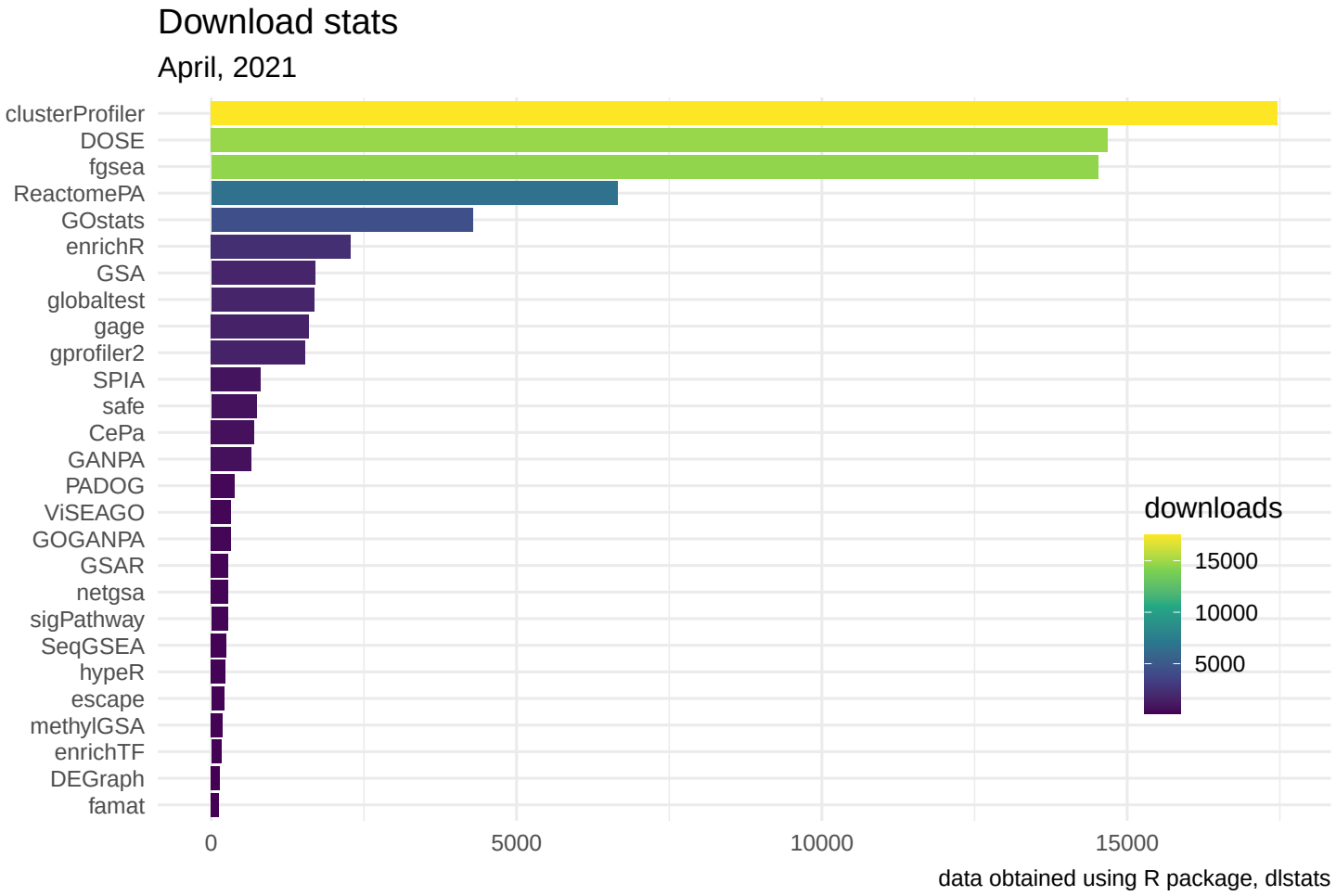
- Automated transcriptomics data analysis workflow using pathway and network analysis approaches[11]

Analysis websites (or shiny apps) that incorporates `clusterProfiler`:

- NASQAR: a web-based platform for high-throughput sequencing data analysis and visualization[12]
- Shiny-Seq: advanced guided transcriptome analysis[13]
- ProteoRE: A biologist-oriented Galaxy platform for proteomics data exploration[14]
- Netpredictor: R and Shiny package to perform drug-target network analysis and prediction of missing links[15]
- ABioTrans: A Biostatistical Tool for Transcriptomics Analysis[16]
- SigBio-Shiny: A standalone interactive application for detecting biological significance on a set of genes[17]

4 Comparing clusterProfiler with other tools

Here, we compare `clusterProfiler` with other R packages that also can perform functional enrichment analysis (Table S2). The packages in Table S2 were ordered by monthly download stats (April 2021).



Focus on the R ecosystem, `clusterProfiler` is the most popular package for functional enrichment analysis. Compare to other tools, `clusterProfiler` has many good features. It internally supports GO and KEGG for thousands of species, allows users to specify background gene set, provides general interface for external annotation data, works with GMT files, and supports comparing functional profiles among different conditions.

Several R packages ouput tabular result (e.g., data frame). Data frame is simple and easy to process and visualize using tidy tools (e.g., dplyr) and ggplot2. However, many useful information including input data, parameter setting and gene set, are missing. These information maybe useful for further interpretation and visualization. Instead, most of the R packages

[11] https://fairdomhub.org/studies/837
[12] https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-020-03577-4
[13] https://bmcresnotes.biomedcentral.com/articles/10.1186/s13104-019-4471-1
[14] https://github.com/vloux/ProteoRE
[15] https://bmcbioinformatics.biomedcentral.com/articles/10.1186/s12859-018-2254-7
[16] https://www.frontiersin.org/articles/10.3389/fgene.2019.00499/full
[17] https://github.com/sk-sahu/sig-bio-shiny

encapsulate enrichment result into more complicated R object (S3, S4 or R6) to include enrichment result with associated data. This will prevent users to explore the result using tidy tools and ggplot2. The `clusterProfiler` and its sub-packages (including `DOSE` and `ReactomePA`) provide tidy interface to process enrichment result and directly supports of visualizing enrichment result in ggplot2. To our knowledge, this feature cannot be found in other R packages that also output enrichment result as complicated R object.

Table S2: Comparing clusterProfiler with other tools

		Input and annotation						Method			Interpretation				
Software	Repo	Annotation	Supported organisms	ID conversion	Updated KEGG	External annotation data	Support GMT file	Algorithm	Selection of background set	Profile comparison	Output	Tidy interface	Support ggplot2	Visualization methods	Remove redundant terms
clusterProfiler	2	GO, KEGG, WikiPathways	plenty	Y	Y	Y	Y	ORA, GSEA	Y	Y	enrichResult, gseaResult, compareClusterResult (S4)	Y	Y	11	Y
DOSE	2	DisGeNE, DO, NCG	1	N	NA	N	N	ORA, GSEA	Y	N	enrichResult, gseaResult (S4)	Y	Y	11	Y
fgsea	2	NA	NA	Y	NA	Y	Y	ORA, GSEA	Y	N	data.table	Y	Y	2	N
ReactomePA	2	Reactome	7	N	NA	N	N	ORA, GSEA	Y	N	enrichResult, gseaResult (S4)	Y	Y	11	N
GOstats	2	NA	NA	N	NA	Y	N	ORA	Y	N	GOHyperGResult (S4)	N	N	1	N
enrichR	1	GO, KEGG, WikiPathways, BioCarta, Reactome, GEO, GeneSigDB, HPO, KEA, MSigDB, COVID-19 Related Gene Sets	5	Y	N	N	N	ORA	N	N	list	N	N	3	N
GSA	1	NA	NA	N	NA	Y	Y	Gene set analysis	N	N	GSA (S3)	N	N	1	N
globaltest	2	GO, KEGG, MSigDB, Anni	21	N	N	N	N	regression analysis	N	N	gt (S4)	N	N	2	N
gage	2	GO, KEGG	plenty	Y	Y	Y	N	GSEA	N	N	list	N	N	0	Y
gprofiler2	1	GO, KEGG, Reactome,WikiPathways, miRTarBase, TRANSFAC, Human Protein Atlas, protein complexes from CORUM, HPO	plenty	Y	N	Y	Y	ORA	N	Y	list	N	N	1	N
SPIA	2	KEGG	plenty	N	Y	N	N	Signaling Pathway Impact Analysis	Y	N	data.frame	Y	Y	1	N
safe	2	GO, KEGG, PFAM, Reactome	20	N	N	Y	N	ORA, Wilcoxon rank sum, Pearson's chi-squared type statistic, t-statistic	N	N	SAFE (S4)	N	N	2	N
CePa	1	NCI_Nature, KEGG, BioCarta, Reactome	1	N	N	N	N	CePa	Y	N	cepa (S3)	N	N	3	N
GANPA	1	NA	NA	N	NA	Y	N	GANPA	N	N	.csv files	N	N	0	N
PADOG	2	KEGG	1	N	Y	Y	N	PADOG	N	Y	data.frame	Y	Y	0	N
ViSEAGO	2	GO	21	N	N	N	N	ORA, GSEA	N	Y	fgsea, enrich_GO_terms (S4)	N	N	2	N
GOGANPA	1	NA	NA	N	NA	Y	N	GO-Functional-Network-based Gene-Set-Analysis	N	N	.csv file	N	N	0	N
GSAR	2	NA	NA	N	NA	Y	N	two-sample Nnparametric multivariate test	N	N	list	N	N	0	N
netgsa	1	NA	NA	N	NA	Y	N	netgsa	N	N	list	N	N	3	N
sigPathway	2	NA	NA	N	NA	Y	N	GSEA, sigPathway	N	N	list	N	N	0	N
SeqGSEA	2	NA	NA	Y	NA	Y	Y	GSEA	N	N	SeqGeneSet (S4)	N	N	0	N
hypeR	2	MSigDB, KEGG, Reactome, MetaboAnalyst	11	N	N	Y	N	ORA, GSEA	Y	N	hyp (R6)	N	N	3	N
escape	2	MSigDB	11	N	N	Y	N	GSEA	N	N	data.frame	N	N	6	N
methylGSA	2	GO, KEGG, Reactome	1	Y	N	N	N	ORA, GSEA	N	N	data.frame	Y	Y	0	N
enrichTF	2	Transcription factor information	2	N	NA	N	N	t-tests, ORA	N	N	list	N	N	0	N
DEGraph	2	KEGG	plenty	N	Y	Y	N	t-tests	N	N	list	N	N	1	N
famat	2	GO, KEGG, Wikipathways, Reactome	1	N	Y	N	N	ORA	N	N	list	N	N	0	N

[1] Repo: 1 for CRAN and 2 for Bioconductor

[2] Supported organisms: 'NA' for not applicable as there is no species annotation data internally supported by the package; 'plenty' for hundreds or thousands species supported (mostly for KEGG and/or GO)

[3] Tidy interface: whether the output object can be processed directly using tidy tools such as dplyr

[4] Support ggplot2: whether the output object can be visualized directly using ggplot2 command

[5] Y for supported, N for not supported and NA for not applicable

5 Data sets

Three data sets were used in this document, including:

- `geneList` provided by the DOSE package
- `DE_GSE8057` provided by the clusterProfiler package
- `GSM1295076_CBX6_BF_ChipSeq_mergedReps_peaks.bed.gz` provided by the ChIPseeker package

The `geneList` was derived from the R package breastCancerMAINZ that contains 200 breast cancer samples, including 29 samples in grade I, 136 samples in grade II and 35 samples in grade III. The ratio of geometric mean of grade III samples versue geometric mean of grade I samples for each gene was computed. The `geneList` data set contains logarithm of these ratios (base 2).

The `DE_GSE8057` data set was derived from the `GSE8057` data set which can be downloaded in GEO and the expreimental design was documented in https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE8057. All the treated samples were compared with control samples by different conditions using the limma package. The `DE_GSE8057` data set contains differential expressed genes (DEGs) for each condition and the DEGs were selected in case of expression values with fold change > 1 or adjusted p value < 0.05.

The GSM1295076_CBX6_BF_ChipSeq_mergedReps_peaks.bed.gz file can be accessed via `ChIPseeker::getSampleFiles()[[4]]` or downloaded using the command `ChIPseeker::downloadGSMbedFiles("GSM1295076")`. The experimental design was documented in https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSM1295076.

In addition to GO and KEGG, two additional gene sets were used in the manuscript, including:

- ENCODE_and_ChEA_Consensus_TFs_from_ChIP-X
- WikiPathways

The `ENCODE_and_ChEA_Consensus_TFs_from_ChIP-X` was downloaded from https://maayanlab.cloud/Enrichr/geneSetLibrary?mode=text&libraryName=ENCODE_and_ChEA_Consensus_TFs_from_ChIP-X. This gene set was used to identify transcriptional factors associated with genomic regions obtained from a ChIPseq experiment.

The WikiPathways, `wikipathways-20210310-gmt-Homo_sapiens.gmt` was downloaded from https://wikipathways-data.wmcloud.org/current/gmt/. This gene set was used to identify biological pathways using community curated knowledge.

6 Examples of using clusterProfiler

This session provides source codes to reproduce the figures presented in the manuscript.

6.1 GO enrichment analysis

```
library(clusterProfiler)
library(enrichplot)

## geneList for GSEA examples
data(geneList, package="DOSE")

## fold change > 2 as DE genes, for ORA examples
de <- names(geneList)[abs(geneList) > 2]


ego <- enrichGO(de, OrgDb = "org.Hs.eg.db", ont="BP", readable=TRUE)

## use simplify to remove redundant terms
ego2 <- simplify(ego, cutoff=0.7, by="p.adjust", select_fun=min)


## visualization
ego <- pairwise_termsim(ego)
ego2 <- pairwise_termsim(ego2)

p1 <- emapplot(ego, cex_label_category=.8, cex_line=.5) + coord_cartesian()
p2 <- emapplot(ego2, cex_label_category=.8, cex_line=.5) + coord_cartesian()

p1 <- p1 + scale_fill_continuous(low = "#e06663", high = "#327eba", name = "p.adjust",
                                 guide = guide_colorbar(reverse = TRUE, order=1), trans='log10')
p2 <- p2 + scale_fill_continuous(low = "#e06663", high = "#327eba", name = "p.adjust",
```

```
                    guide = guide_colorbar(reverse = TRUE, order=1), trans='log10')

cowplot::plot_grid(p1, p2, labels=c("A", "B"), rel_widths=c(1, 1.2))
```

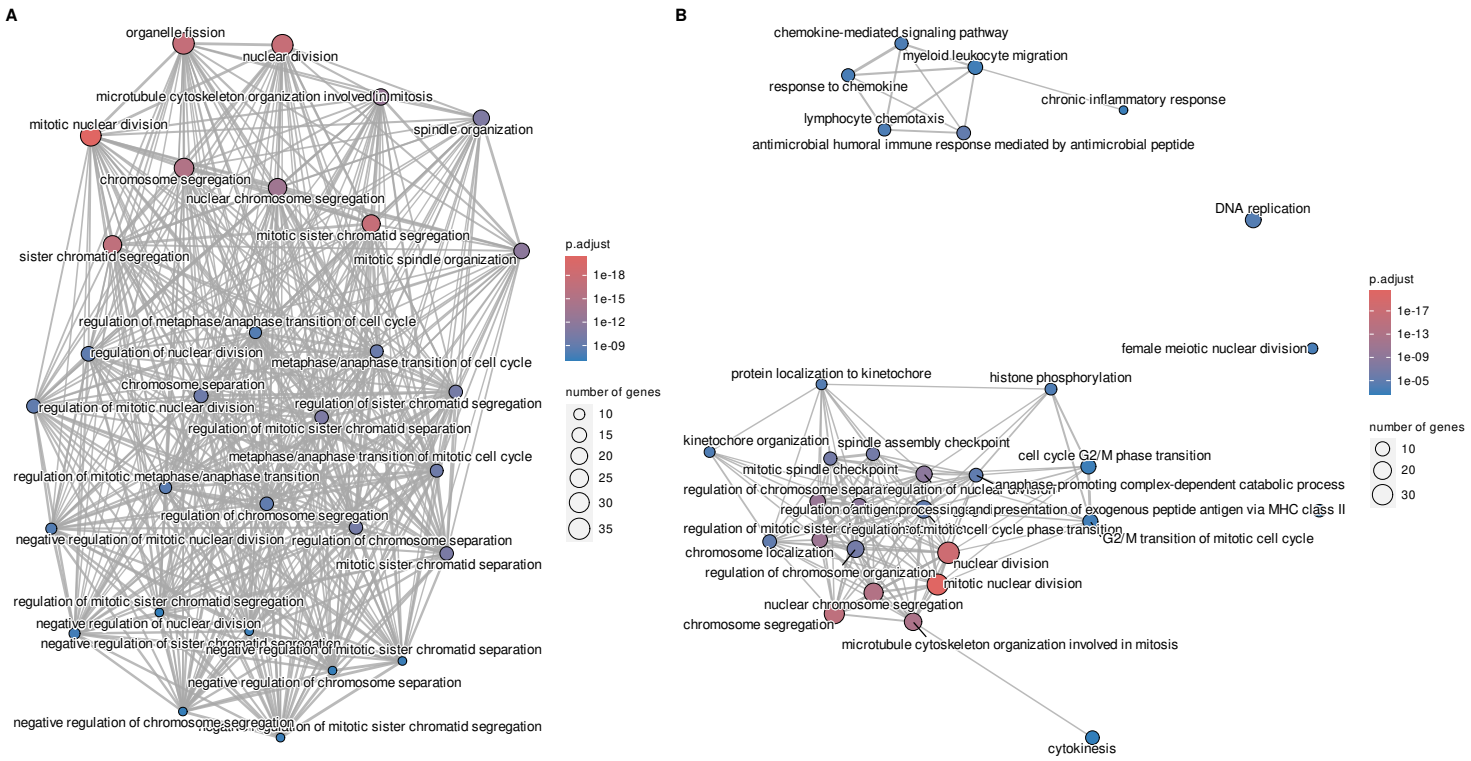
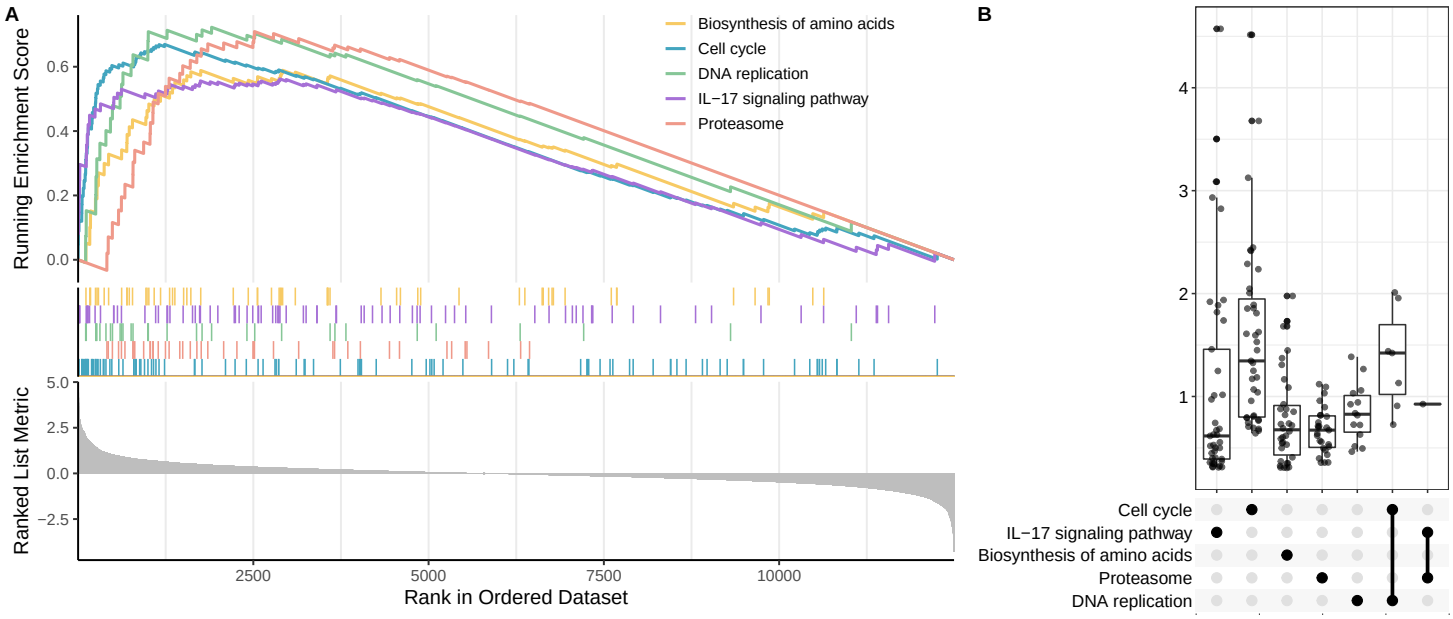


Fig. 1: Gene ontology enrichment analysis.

6.2 KEGG enrichment analysis

```
data(geneList, package="DOSE")
kk <- gseKEGG(geneList, organism = "hsa", eps=0)

## sorted by absolute values of NES
kk2 <- arrange(kk, desc(abs(NES)))

## visualization
color <- c("#f7ca64", "#43a5bf", "#86c697", "#a670d6", "#ef998a")
kp1 <- gseaplot2(kk2, 1:5, color = color, pvalue_table=F, base_size=14)
kp2 <- upsetplot(kk2, n=5)
cowplot::plot_grid(kp1, kp2, rel_widths=c(1, .5), labels=c("A", "B"))
```



6.3 Functional interpretation of genomic regions of interest

```
library(ChIPseeker)
## the file can be downloaded using `downloadGSMbedFiles("GSM1295076")`
file <- "GSM1295076_CBX6_BF_ChipSeq_mergedReps_peaks.bed.gz"
gr <- readPeakFile(file)

library(TxDb.Hsapiens.UCSC.hg19.knownGene)
TxDb <- TxDb.Hsapiens.UCSC.hg19.knownGene
genes <- seq2gene(gr, tssRegion=c(-1000, 1000), flankDistance = 3000, TxDb)

library(clusterProfiler)
## downloaded from 'https://maayanlab.cloud/Enrichr/geneSetLibrary?mode=
## text&libraryName=ENCODE_and_ChEA_Consensus_TFs_from_ChIP-X'
encode <- read.gmt("ENCODE_and_ChEA_Consensus_TFs_from_ChIP-X.txt")
g <- bitr(genes, 'ENTREZID', 'SYMBOL', 'org.Hs.eg.db')
```

```
## Warning in bitr(genes, "ENTREZID", "SYMBOL", "org.Hs.eg.db"): 5.32% of input
## gene IDs are fail to map...
```

```
x <- enricher(g$SYMBOL, TERM2GENE=encode)
cnetplot(x, cex_label_gene=0.6,
       color_category = "#97c497", color_gene = "#c4c4c4") +
       guides(size = guide_legend(override.aes=list(shape=1)))
```

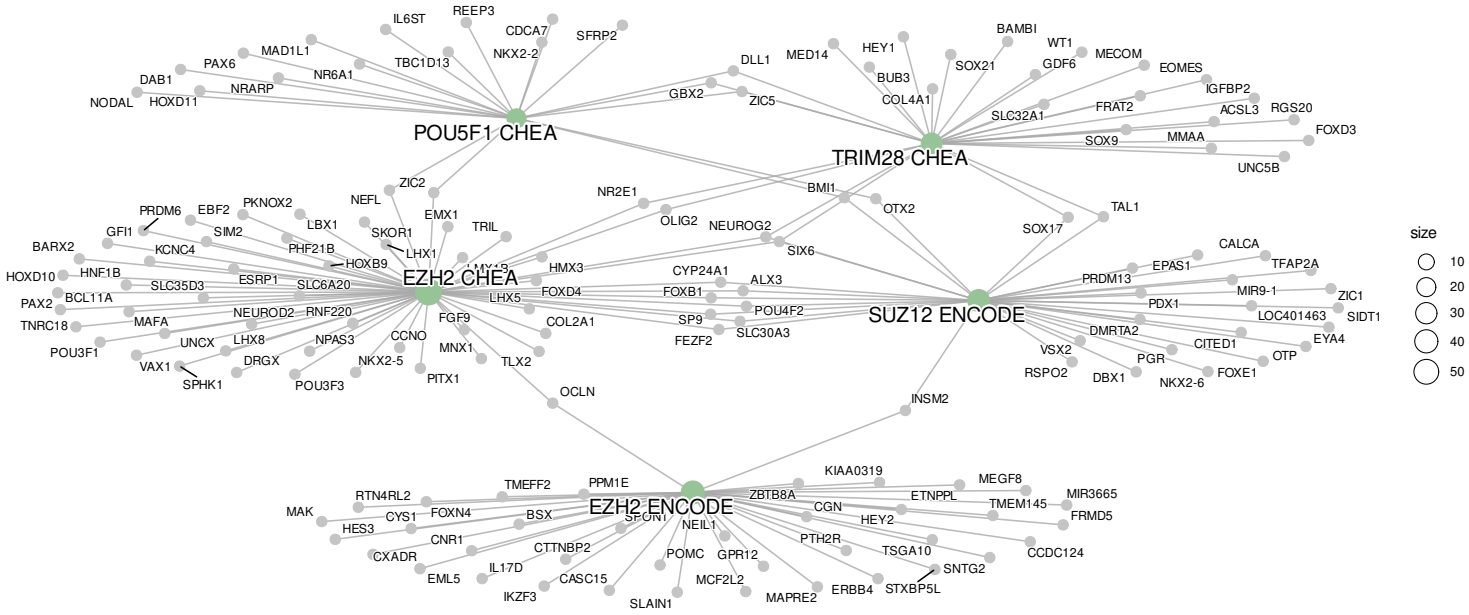


Fig. 2: Functional enrichment analysis of genomic regions of interest.

6.4 Comparison for different conditions

```
## downloaded from https://wikipathways-data.wmcloud.org/current/gmt/
gmt <- 'wikipathways-20210310-gmt-Homo_sapiens.gmt'
wp <- read.gmt.wp(gmt)

data(DE_GSE8057)

xx <- compareCluster(Gene~time+treatment, data=DE_GSE8057, fun = enricher,
              TERM2GENE=wp[,c("wpid", "gene")], TERM2NAME=wp[,c("wpid", "name")])

pp <- dotplot(xx, x="time") + facet_grid(~treatment) +
    aes(x=fct_relevel(time, c('0h', '2h', '6h', '24h'))) + xlab(NULL) +
    scale_color_gradientn(colours=c("#b3eebe", "#46bac2", "#371ea3"),
                    guide=guide_colorbar(reverse=TRUE, order=1)) +
    guides(size = guide_legend(override.aes=list(shape=1))) +
    theme(panel.grid.major.y = element_line(linetype='dotted', color='#808080'),
          panel.grid.major.x = element_blank())
```

```
print(pp)
```

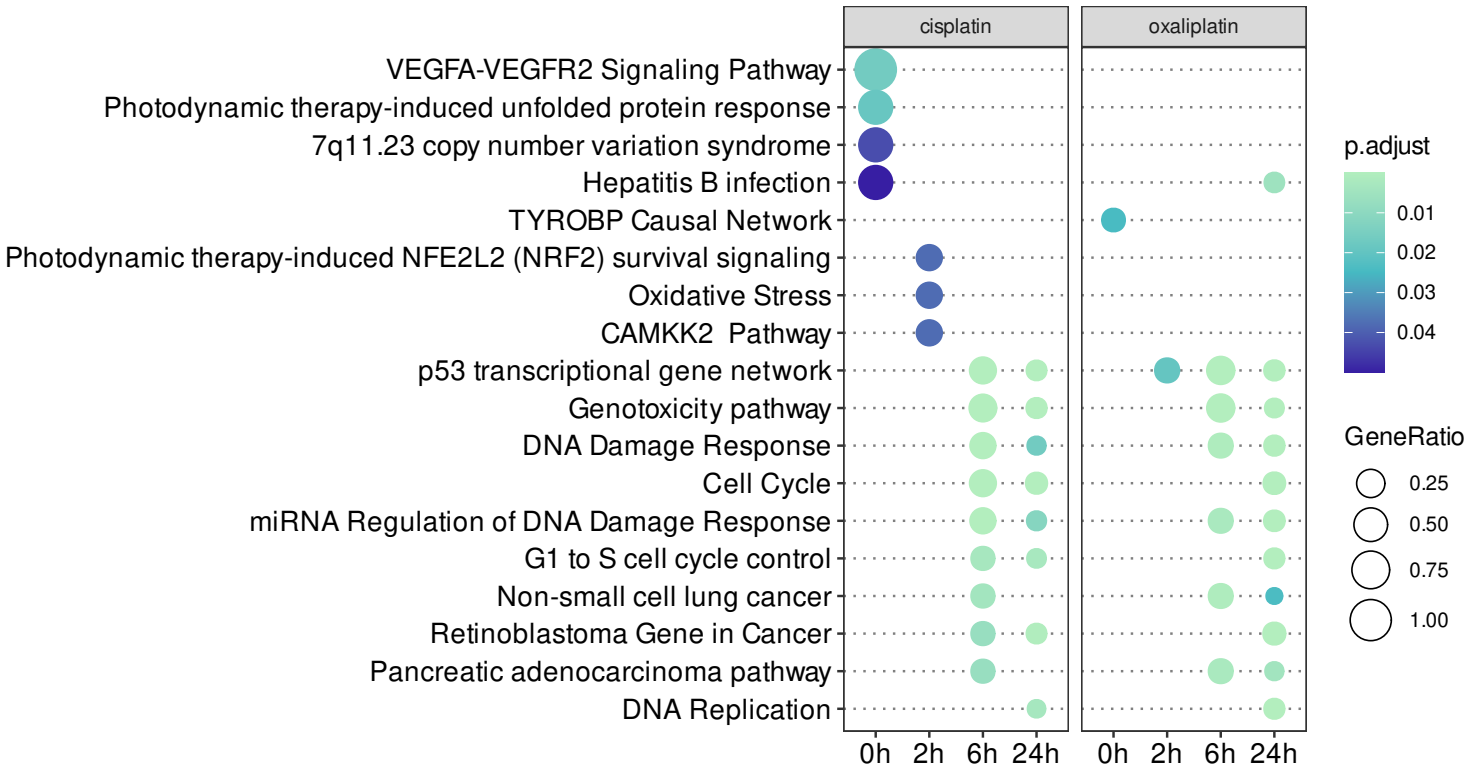
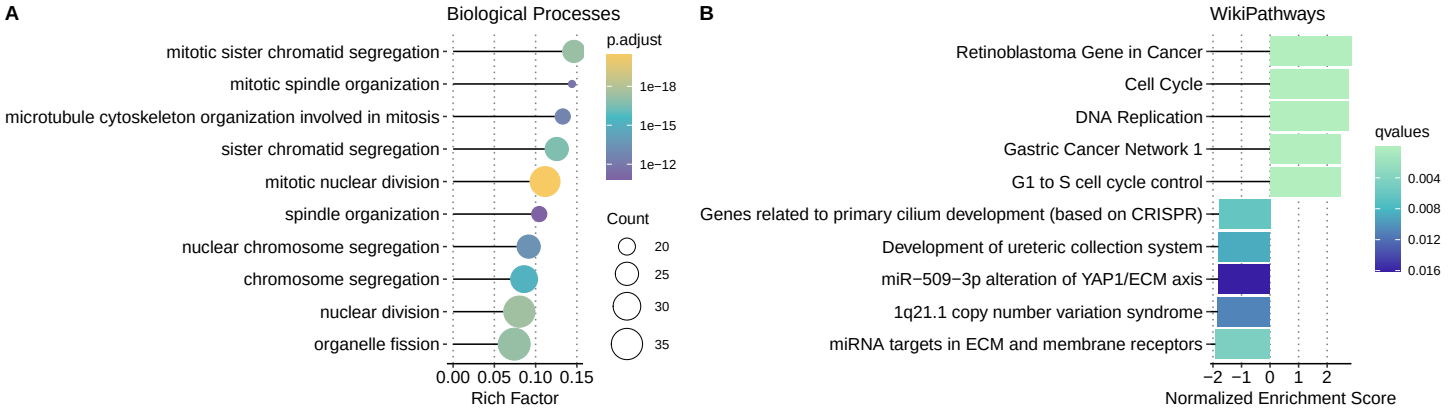


Fig. 3: Comparing functional profiles among different levels of conditions.

6.5 Visualization using ggplot2

```
library(forcats)
library(ggplot2)

ewp <- GSEA(geneList, TERM2GENE=wp[,c("wpid", "gene")], TERM2NAME=wp[,c("wpid", "name")])

ewp2 <- arrange(ewp, desc(abs(NES))) %>%
        group_by(sign(NES)) %>%
        slice(1:5)
ego3 <- mutate(ego, richFactor = Count / as.numeric(sub("/\\d+", "", BgRatio)))

mytheme <- theme(panel.border=element_blank(),
        panel.grid.major=element_line(linetype='dotted', colour='#808080'),
        panel.grid.major.y=element_blank(),
        panel.grid.minor=element_blank(),
        axis.line.x = element_line())

g1 <- ggplot(ego3, showCategory = 10,
  aes(richFactor, fct_reorder(Description, richFactor))) +
  geom_segment(aes(xend=0, yend = Description)) +
  geom_point(aes(color=p.adjust, size = Count)) +
  scale_color_gradientn(colours=c("#f7ca64", "#46bac2", "#7e62a3"),
                        trans = "log10",
                        guide=guide_colorbar(reverse=TRUE, order=1)) +
  scale_size_continuous(range=c(2, 10)) +
  scale_x_continuous(expand=c(0,0)) +
  theme_dose(12) +
  mytheme +
  xlim(NA, 0.15) +
  guides(size = guide_legend(override.aes=list(shape=1))) +
  xlab("Rich Factor") +
  ylab(NULL) +
  ggtitle("Biological Processes")

g2 <- ggplot(ewp2, showCategory=10,
        aes(NES, fct_reorder(Description, NES), fill=qvalues)) +
    geom_col() +
    geom_segment(mapping=aes(x=-2.1,
                             xend=ifelse(sign(NES)>0, 0, NES),
                             yend=Description)) +
    scale_x_continuous(expand=c(0,0)) +
    scale_fill_gradientn(colours=c('#b3eebe', "#46bac2", '#371ea3'),
                         guide=guide_colorbar(reverse=TRUE)) +
    theme_dose(12) +
    mytheme +
    xlab("Normalized Enrichment Score") +
    ylab(NULL) +
    ggtitle("WikiPathways")

cowplot::plot_grid(g1, g2, labels=c("A", "B"))
```



NOTE:

1. source codes and datasets to produce this file can be obtained online[18].
2. setting font and adding rounded rectangular as background for each of the legends, as presented in the manuscript, can be done by the `ggfun`[19] package.

[18] https://github.com/YuLab-SMU/supplemental-clusterProfiler-v4
[19] https://cran.r-project.org/package=ggfun

7 Session information

Here is the output of `sessionInfo()` of the system on which the Supplemental file was compiled:

```
## - Session info ---------------------------------------------------------------
##  setting  value
##  version  R version 4.1.0 (2021-05-18)
##  os       Arch Linux
##  system   x86_64, linux-gnu
##  ui       X11
##  language (EN)
##  collate  en_US.UTF-8
##  ctype    en_US.UTF-8
##  tz       Asia/Chongqing
##  date     2021-07-05
##
## - Packages -------------------------------------------------------------------
##  package                       * version    date       lib source
##  AnnotationDbi                 * 1.54.0     2021-05-19 [1] Bioconductor
##  ape                             5.5        2021-04-25 [1] CRAN (R 4.1.0)
##  aplot                           0.0.6      2020-09-03 [1] CRAN (R 4.1.0)
##  assertthat                      0.2.1      2019-03-21 [1] CRAN (R 4.1.0)
##  Biobase                       * 2.52.0     2021-05-19 [1] Bioconductor
##  BiocFileCache                   2.0.0      2021-05-19 [1] Bioconductor
##  BiocGenerics                  * 0.38.0     2021-05-19 [1] Bioconductor
##  BiocIO                          1.2.0      2021-05-19 [1] Bioconductor
##  BiocManager                     1.30.15    2021-05-11 [1] CRAN (R 4.1.0)
##  BiocParallel                    1.26.0     2021-05-19 [1] Bioconductor
##  biomaRt                         2.48.0     2021-05-19 [1] Bioconductor
##  Biostrings                      2.60.0     2021-05-19 [1] Bioconductor
##  bit                             4.0.4      2020-08-04 [1] CRAN (R 4.1.0)
##  bit64                           4.0.5      2020-08-30 [1] CRAN (R 4.1.0)
##  bitops                          1.0-7      2021-04-24 [1] CRAN (R 4.1.0)
##  blob                            1.2.1      2020-01-20 [1] CRAN (R 4.1.0)
##  bookdown                        0.22       2021-04-22 [1] CRAN (R 4.1.0)
##  boot                            1.3-28     2021-05-03 [1] CRAN (R 4.1.0)
##  cachem                          1.0.5      2021-05-15 [1] CRAN (R 4.1.0)
##  caTools                         1.18.2     2021-03-28 [1] CRAN (R 4.1.0)
##  ChIPseeker                    * 1.28.3     2021-05-21 [1] Bioconductor
##  cli                             2.5.0      2021-04-26 [1] CRAN (R 4.1.0)
##  clusterProfiler               * 4.1.1      2021-07-05 [1] Bioconductor
##  colorspace                      2.0-1      2021-05-04 [1] CRAN (R 4.1.0)
##  conflicted                    * 1.0.4      2019-06-21 [1] CRAN (R 4.1.0)
##  cowplot                         1.1.1      2020-12-30 [1] CRAN (R 4.1.0)
##  crayon                          1.4.1      2021-02-08 [1] CRAN (R 4.1.0)
##  curl                            4.3.1      2021-04-30 [1] CRAN (R 4.1.0)
##  data.table                      1.14.0     2021-02-21 [1] CRAN (R 4.1.0)
##  DBI                             1.1.1      2021-01-15 [1] CRAN (R 4.1.0)
##  dbplyr                          2.1.1      2021-04-06 [1] CRAN (R 4.1.0)
##  DelayedArray                    0.18.0     2021-05-19 [1] Bioconductor
##  digest                          0.6.27     2020-10-24 [1] CRAN (R 4.1.0)
##  DO.db                           2.9        2021-05-22 [1] Bioconductor
##  DOSE                          * 3.19.1     2021-06-13 [1] Bioconductor
##  downloader                      0.4        2015-07-09 [1] CRAN (R 4.1.0)
##  dplyr                           1.0.6      2021-05-05 [1] CRAN (R 4.1.0)
##  ellipsis                        0.3.2      2021-04-29 [1] CRAN (R 4.1.0)
##  enrichplot                    * 1.13.1     2021-06-30 [1] Bioconductor
##  evaluate                        0.14       2019-05-28 [1] CRAN (R 4.1.0)
##  fansi                           0.5.0      2021-05-25 [1] CRAN (R 4.1.0)
##  farver                          2.1.0      2021-02-28 [1] CRAN (R 4.1.0)
##  fastmap                         1.1.0      2021-01-25 [1] CRAN (R 4.1.0)
##  fastmatch                       1.1-0      2017-01-28 [1] CRAN (R 4.1.0)
##  fgsea                           1.18.0     2021-05-19 [1] Bioconductor
##  filelock                        1.0.2      2018-10-05 [1] CRAN (R 4.1.0)
##  forcats                       * 0.5.1      2021-01-27 [1] CRAN (R 4.1.0)
##  generics                        0.1.0      2020-10-31 [1] CRAN (R 4.1.0)
##  GenomeInfoDb                  * 1.28.0     2021-05-19 [1] Bioconductor
##  GenomeInfoDbData                1.2.6      2021-05-21 [1] Bioconductor
##  GenomicAlignments               1.28.0     2021-05-19 [1] Bioconductor
##  GenomicFeatures               * 1.44.0     2021-05-19 [1] Bioconductor
##  GenomicRanges                 * 1.44.0     2021-05-19 [1] Bioconductor
##  ggforce                         0.3.3      2021-03-05 [1] CRAN (R 4.1.0)
##  ggnewscale                      0.4.5      2021-01-11 [1] CRAN (R 4.1.0)
##  ggplot2                       * 3.3.4      2021-06-16 [1] CRAN (R 4.1.0)
##  ggraph                          2.0.5      2021-02-23 [1] CRAN (R 4.1.0)
##  ggrepel                         0.9.1      2021-01-15 [1] CRAN (R 4.1.0)
```

```
##  ggtree                        3.1.1.992  2021-06-13 [1] Bioconductor
##  glue                          1.4.2      2020-08-27 [1] CRAN (R 4.1.0)
##  GO.db                         3.13.0     2021-05-22 [1] Bioconductor
##  GOSemSim                      2.18.0     2021-05-19 [1] Bioconductor
##  gplots                        3.1.1      2020-11-28 [1] CRAN (R 4.1.0)
##  graphlayouts                  0.7.1      2020-10-26 [1] CRAN (R 4.1.0)
##  gridExtra                     2.3        2017-09-09 [1] CRAN (R 4.1.0)
##  gtable                        0.3.0      2019-03-25 [1] CRAN (R 4.1.0)
##  gtools                        3.8.2      2020-03-31 [1] CRAN (R 4.1.0)
##  hms                           1.1.0      2021-05-17 [1] CRAN (R 4.1.0)
##  htmltools                     0.5.1.1    2021-01-22 [1] CRAN (R 4.1.0)
##  httr                          1.4.2      2020-07-20 [1] CRAN (R 4.1.0)
##  igraph                        1.2.6      2020-10-06 [1] CRAN (R 4.1.0)
##  IRanges                     * 2.26.0     2021-05-19 [1] Bioconductor
##  jsonlite                      1.7.2      2020-12-09 [1] CRAN (R 4.1.0)
##  kableExtra                  * 1.3.4      2021-02-20 [1] CRAN (R 4.1.0)
##  KEGGREST                      1.32.0     2021-05-19 [1] Bioconductor
##  KernSmooth                    2.23-20    2021-05-03 [1] CRAN (R 4.1.0)
##  knitr                         1.33       2021-04-24 [1] CRAN (R 4.1.0)
##  labeling                      0.4.2      2020-10-20 [1] CRAN (R 4.1.0)
##  lattice                       0.20-44    2021-05-02 [1] CRAN (R 4.1.0)
##  lazyeval                      0.2.2      2019-03-15 [1] CRAN (R 4.1.0)
##  lifecycle                     1.0.0      2021-02-15 [1] CRAN (R 4.1.0)
##  magrittr                    * 2.0.1      2020-11-17 [1] CRAN (R 4.1.0)
##  MASS                          7.3-54     2021-05-03 [1] CRAN (R 4.1.0)
##  Matrix                        1.3-3      2021-05-04 [1] CRAN (R 4.1.0)
##  MatrixGenerics                1.4.0      2021-05-19 [1] Bioconductor
##  matrixStats                   0.58.0     2021-01-29 [1] CRAN (R 4.1.0)
##  memoise                       2.0.0      2021-01-26 [1] CRAN (R 4.1.0)
##  munsell                       0.5.0      2018-06-12 [1] CRAN (R 4.1.0)
##  nlme                          3.1-152    2021-02-04 [1] CRAN (R 4.1.0)
##  org.Hs.eg.db                * 3.13.0     2021-05-22 [1] Bioconductor
##  patchwork                     1.1.1      2020-12-17 [1] CRAN (R 4.1.0)
##  pillar                        1.6.1      2021-05-16 [1] CRAN (R 4.1.0)
##  pkgconfig                     2.0.3      2019-09-22 [1] CRAN (R 4.1.0)
##  plotrix                       3.8-1      2021-01-21 [1] CRAN (R 4.1.0)
##  plyr                          1.8.6      2020-03-03 [1] CRAN (R 4.1.0)
##  png                           0.1-7      2013-12-03 [1] CRAN (R 4.1.0)
##  polyclip                      1.10-0     2019-03-14 [1] CRAN (R 4.1.0)
##  prettyunits                   1.1.1      2020-01-24 [1] CRAN (R 4.1.0)
##  progress                      1.2.2      2019-05-16 [1] CRAN (R 4.1.0)
##  purrr                         0.3.4      2020-04-17 [1] CRAN (R 4.1.0)
##  qvalue                        2.24.0     2021-05-19 [1] Bioconductor
##  R6                            2.5.0      2020-10-28 [1] CRAN (R 4.1.0)
##  rappdirs                      0.3.3      2021-01-31 [1] CRAN (R 4.1.0)
##  RColorBrewer                  1.1-2      2014-12-07 [1] CRAN (R 4.1.0)
##  Rcpp                          1.0.6      2021-01-15 [1] CRAN (R 4.1.0)
##  RCurl                         1.98-1.3   2021-03-16 [1] CRAN (R 4.1.0)
##  reshape2                      1.4.4      2020-04-09 [1] CRAN (R 4.1.0)
##  restfulr                      0.0.13     2017-08-06 [1] CRAN (R 4.1.0)
##  rjson                         0.2.20     2018-06-08 [1] CRAN (R 4.1.0)
##  rlang                         0.4.11     2021-04-30 [1] CRAN (R 4.1.0)
##  rmarkdown                   * 2.8        2021-05-07 [1] CRAN (R 4.1.0)
##  Rsamtools                     2.8.0      2021-05-19 [1] Bioconductor
##  RSQLite                       2.2.7      2021-04-22 [1] CRAN (R 4.1.0)
##  rstudioapi                    0.13       2020-11-12 [1] CRAN (R 4.1.0)
##  rtracklayer                   1.52.0     2021-05-19 [1] Bioconductor
##  rvcheck                     * 0.1.8      2020-03-01 [1] CRAN (R 4.1.0)
##  rvest                         1.0.0      2021-03-09 [1] CRAN (R 4.1.0)
##  S4Vectors                   * 0.30.0     2021-05-19 [1] Bioconductor
##  scales                        1.1.1      2020-05-11 [1] CRAN (R 4.1.0)
##  scatterpie                    0.1.6      2021-04-23 [1] CRAN (R 4.1.0)
##  sessioninfo                   1.1.1      2018-11-05 [1] CRAN (R 4.1.0)
##  shadowtext                    0.0.8      2021-04-23 [1] CRAN (R 4.1.0)
##  stringi                       1.6.2      2021-05-17 [1] CRAN (R 4.1.0)
##  stringr                       1.4.0      2019-02-10 [1] CRAN (R 4.1.0)
##  SummarizedExperiment          1.22.0     2021-05-19 [1] Bioconductor
##  svglite                       2.0.0      2021-02-20 [1] CRAN (R 4.1.0)
##  systemfonts                   1.0.2      2021-05-11 [1] CRAN (R 4.1.0)
##  tibble                        3.1.2      2021-05-16 [1] CRAN (R 4.1.0)
##  tidygraph                     1.2.0      2020-05-12 [1] CRAN (R 4.1.0)
##  tidyr                         1.1.3      2021-03-03 [1] CRAN (R 4.1.0)
##  tidyselect                    1.1.1      2021-04-30 [1] CRAN (R 4.1.0)
##  tidytree                      0.3.4      2021-05-22 [1] CRAN (R 4.1.0)
##  treeio                        1.17.1.992 2021-06-13 [1] Bioconductor
##  tweenr                        1.0.2      2021-03-23 [1] CRAN (R 4.1.0)
```

```
##  TxDb.Hsapiens.UCSC.hg19.knownGene * 3.2.2      2021-05-22 [1] Bioconductor
##  utf8                                1.2.1      2021-03-12 [1] CRAN (R 4.1.0)
##  vctrs                               0.3.8      2021-04-29 [1] CRAN (R 4.1.0)
##  viridis                             0.6.1      2021-05-11 [1] CRAN (R 4.1.0)
##  viridisLite                         0.4.0      2021-04-13 [1] CRAN (R 4.1.0)
##  webshot                             0.5.2      2019-11-22 [1] CRAN (R 4.1.0)
##  wget                              * 0.0.1      2020-04-27 [1] local
##  withr                               2.4.2      2021-04-18 [1] CRAN (R 4.1.0)
##  xfun                                0.23       2021-05-15 [1] CRAN (R 4.1.0)
##  XML                                 3.99-0.6   2021-03-16 [1] CRAN (R 4.1.0)
##  xml2                                1.3.2      2020-04-23 [1] CRAN (R 4.1.0)
##  XVector                             0.32.0     2021-05-19 [1] Bioconductor
##  yaml                                2.2.1      2020-02-01 [1] CRAN (R 4.1.0)
##  zlibbioc                            1.38.0     2021-05-19 [1] Bioconductor
##
## [1] /home/ygc/R/library
## [2] /usr/lib/R/library
```