

# Package ‘PanViz’

September 23, 2024

**Type** Package

**Title** Integrating Multi-Omic Network Data With Summary-Level GWAS Data

**Version** 1.6.0

## Description

This package integrates data from the Kyoto Encyclopedia of Genes and Genomes (KEGG) with summary-level genome-wide association (GWAS) data, such as that provided by the GWAS Catalog or GWAS Central databases, or a user's own study or dataset, in order to produce biological networks, termed IMONs (Integrated Multi-Omic Networks). IMONs can be used to analyse trait-specific polymorphic data within the context of biochemical and metabolic reaction networks, providing greater biological interpretability for GWAS data.

**License** Artistic-2.0

**Encoding** UTF-8

**LazyData** false

**RoxygenNote** 7.1.2

**BugReports** <https://github.com/LucaAnholt/PanViz/issues>

**URL** <https://github.com/LucaAnholt/PanViz>

**Imports** tidy, stringr, dplyr, tibble, magrittr, futile.logger, utils, easysv, rentrez, igraph, RColorBrewer, data.table, colorspace, grDevices, rlang, methods

**Depends** R (>= 4.2.0)

**Suggests** testthat (>= 3.0.0), BiocStyle, knitr, rmarkdown, networkD3,

**Config/testthat/edition** 3

**VignetteBuilder** knitr

**BiocType** Software

**biocViews** GenomeWideAssociation, Reactome, Metabolomics, SNP, GraphAndNetwork, Network, KEGG

**git\_url** <https://git.bioconductor.org/packages/PanViz>

**git\_branch** RELEASE\_3\_19

**git\_last\_commit** 27ec438  
**git\_last\_commit\_date** 2024-04-30  
**Repository** Bioconductor 3.19  
**Date/Publication** 2024-09-22  
**Author** Luca Anholt [cre, aut]  
**Maintainer** Luca Anholt <la1317@ic.ac.uk>

## Contents

adjl_to_G . . . . .	2
adjl_to_G_grouped . . . . .	3
adj_list_to_igraph . . . . .	4
colour_IMON . . . . .	4
dbSNP_query_check . . . . .	5
dbSNP_query_clean . . . . .	5
decompose_IMON . . . . .	6
ego_IMON . . . . .	6
er_snp_vector . . . . .	7
get_grouped_IMON . . . . .	7
get_IMON . . . . .	9
GWAS_data_reader . . . . .	10
multi_hex_col_mix . . . . .	11
NCBI_clean . . . . .	12
NCBI_clean_2 . . . . .	12
NCBI_dbSNP_query . . . . .	13
reaction_cleanup . . . . .	13
retry . . . . .	14
set_base_graph_attributes . . . . .	14
set_snp_grouping . . . . .	15
snp_gene_chr_match . . . . .	15
snp_gene_map . . . . .	16
<b>Index</b>	<b>17</b>

---

adjl\_to\_G

*adj\_to\_G*

---

## Description

Internal function that constructs an IMON (Integrated Multi-Omic Network) for an inputted adjacency list containing adjacency information between KEGG genes and queried SNPs.

## Usage

```
adjl_to_G(adjl_G_S)
```

**Arguments**

adjl\_G\_S - adjacency list containing relevant adjacencies between inputted SNPs and genes from KEGG

**Value**

igraph object representing total IMON for inputted SNPs

---

adjl\_to\_G\_grouped      *adjl\_to\_G\_grouped*

---

**Description**

Internal function that constructs either a variable-coloured or uncoloured IMON (Integrated Multi-Omic Network) for an inputted adjacency list containing adjacency information between KEGG genes and queried SNPs.

**Usage**

```
adjl_to_G_grouped(
  adjl_G_S,
  unique_group_names,
  unique_group_cols,
  group_snps,
  colour_groups,
  ego,
  progress_bar
)
```

**Arguments**

adjl\_G\_S - adjacency list containing relevant adjacencies between inputted SNPs and genes from KEGG

unique\_group\_names - a list of the unique group/variable names in the provided GWAS Catalog association file

unique\_group\_cols - a list of unique colours for each unique group/variable in the provided GWAS Catalog association file

group\_snps - a recursive list containing the lists of SNPs belonging to each unique group/variable in the provided GWAS Catalog association file

colour\_groups - boolean: whether or not user has chosen to colour the network by the unique group/variables in the provided GWAS Catalog association file

ego - the egocentric order (centred around the SNPs in the network) in which to build the network i.e. pathlength from SNPs downwards towards the metabolome

progress\_bar - boolean: whether or not user has decided to have a progress bar print to the console

**Value**

- an igraph object containing the IMON

---

adj\_list\_to\_igraph      *adj\_list\_to\_igraph*

---

**Description**

internal function that assembles all the KEGG data into a network/graph

**Usage**

```
adj_list_to_igraph(adjl_G_S)
```

**Arguments**

adjl\_G\_S                  adjacency list containing relevant adjacent SNPs/KEGG genes

**Value**

an igraph object, containing a network representing all the KEGG data

---

colour\_IMON              *colour network by categorical group levels*

---

**Description**

colour network by categorical group levels

**Usage**

```
colour_IMON(G, progress_bar)
```

**Arguments**

G                          - igraph object containing uncoloured IMON

progress\_bar              Boolean (default = TRUE) argument that controls whether or not a progress bar for calculations/KEGGREST API GET requests should be printed to the console

**Value**

- igraph object containing coloured IMON

---

dbSNP_query_check	<i>dbSNP_query_check</i>
-------------------	--------------------------

---

**Description**

dbSNP\_query\_check

**Usage**

dbSNP\_query\_check(query)

**Arguments**

query            - raw query data from NCBI dbSNP API

**Value**

- vector containing either 0 (denoting successful query) or NA (unsuccessful query)

---

dbSNP_query_clean	<i>dbSNP query clean up function</i>
-------------------	--------------------------------------

---

**Description**

Internal function clean up raw SNP data queried from NCBI dbSNP via Entrez API depending on whether or not it could be successfully queried

**Usage**

dbSNP\_query\_clean(query)

**Arguments**

query            - raw dbSNP query object

**Value**

- dataframe of separate chromosome number, position and ID

---

decompose_IMON	<i>decompose_IMON</i>
----------------	-----------------------

---

**Description**

This function returns a list of fully connected IMONs from a single parent unconnected IMON.

**Usage**

```
decompose_IMON(G)
```

**Arguments**

G - igraph object containing non-fully connected IMON

**Value**

- list of igraph objects, where each index contains a fully connected IMON

**Examples**

```
data("er_snp_vector")
G <- PanViz::get_IMON(snp_list = er_snp_vector, ego = 5, save_file = FALSE)
G_list <- decompose_IMON(G)
```

---

ego_IMON	<i>ego_IMON</i>
----------	-----------------

---

**Description**

Internal function for trimming IMON to ego-centred (centred around SNPs) to specified order (pathway length from SNPs)

**Usage**

```
ego_IMON(G, ego)
```

**Arguments**

G - igraph object representing IMON  
ego - the selected ego-centred path length

**Value**

- ego-centred IMON set at desired path length

---

er_snp_vector	<i>Summary-level GWAS data vector for estrogen-receptor positive breast cancer (EFO_1000649)</i>
---------------	--

---

**Description**

A dataset containing a vector of SNPs (summary-level GWAS data) associated with estrogen-receptor positive breast cancer (EFO\_1000649), collated by the GWAS Catalog.

**Usage**

```
data(er_snp_vector)
```

**Format**

A vector with 110 elements

---

get_grouped_IMON	<i>get IMON with SNP and or all network vertices coloured by group variables (either studies or phenotypes)</i>
------------------	---

---

**Description**

This function constructs an IMON (Integrated Multi-Omic Network) with SNPs/or whole network coloured by selected categorical levels (either studies or phenotypes)

**Usage**

```
get_grouped_IMON(
  dataframe,
  groupby = c("studies", "traits"),
  ego = 5,
  save_file = c(FALSE, TRUE),
  export_type = c("igraph", "edge_list", "graphml", "gml"),
  directory = c("wd", "choose"),
  colour_groups = c(FALSE, TRUE),
  progress_bar = c(TRUE, FALSE)
)
```

**Arguments**

dataframe	A dataframe including 3 columns in the following order and with the following names: snps, studies, traits (all character vectors)
groupby	Choose whether to group SNP and or network colouring by either studies or traits

ego	This dictates what length order ego-centred network should be constructed. If set to 5 (default and recommended), an IMON with the first layer of the connected metabolome will be returned. If set above 5, the corresponding extra layer of the metabolome will be returned. If set to 0 (not recommended) the fully connected metabolome will be returned. Note, this cannot be set between 0 and 5.
save_file	Boolean (default = FALSE) argument that indicates whether or not the user wants to save the graph as an exported file in their current working directory
export_type	This dictates the network data structure saved in your working directory. By default this outputs an igraph object, however, you can choose to export and save an edge list, graphml or GML file.
directory	If set to "choose" this argument allows the user to interactively select the directory of their choice in which they wish to save the constructed IMON, else the file will be saved to the working directory "wd" by default
colour_groups	Boolean (default = FALSE) chooses whether or not to colour the whole network by grouping variables
progress_bar	Boolean (default = TRUE) argument that controls whether or not a progress bar for calculations/KEGGREST API GET requests should be printed to the console

**Value**

An igraph object containing the constructed IMON with coloured SNPs/and or whole network by selected grouping variable

**Examples**

```
##getting GWAS Catalog association tsv file and cleaning up using
##GWAS_catalog_tsv_to_dataframe function:
path <- system.file("extdata",
  "gwas-association-downloaded_2021-09-13-EFO_1000649.tsv",
  package="PanViz")
df <- PanViz::GWAS_data_reader(file = path,
  snp_col = "SNPS",
  study_col = "STUDY",
  trait_col = "DISEASE/TRAIT")
##creating uncoloured IMON:
G <- PanViz::get_grouped_IMON(dataframe = df,
  groupby = "studies",
  ego = 5,
  save_file = FALSE,
  colour_groups = FALSE)
##creating IMON where vertices/edges are coloured by the variable study:
G <- PanViz::get_grouped_IMON(dataframe = df,
  groupby = "studies",
  ego = 5,
  save_file = FALSE,
  colour_groups = TRUE)
```



---

`get_IMON``get_IMON`

---

## Description

Internal function that constructs an IMON (Integrated Multi-Omic Network) for an inputted vector of SNPs and exports an igraph file.

## Usage

```
get_IMON(  
  snp_list,  
  ego = 5,  
  save_file = c(FALSE, TRUE),  
  export_type = c("igraph", "edge_list", "graphml", "gml"),  
  directory = c("wd", "choose"),  
  progress_bar = c(TRUE, FALSE)  
)
```

## Arguments

<code>snp_list</code>	A vector of SNPs (strings/characters) using standard NCBI dbSNP accession number naming convention (e.g. "rs185345278")
<code>ego</code>	This dictates what length order ego-centred network should be constructed. If set to 5 (default and recommended), an IMON with the first layer of the connected metabolome will be returned. If set above 5, the corresponding extra layer of the metabolome will be returned. If set to 0 (not recommended) the fully connected metabolome will be returned. Note, this cannot be set between 0 and 5.
<code>save_file</code>	Boolean (default = FALSE) argument that indicates whether or not the user wants to save the graph as an exported file in their current working directory
<code>export_type</code>	This dictates the network data structure saved in the chosen directory. By default this outputs an igraph object, however, you can choose to export and save an edge list, graphml or GML file.
<code>directory</code>	If set to "choose" this argument allows the user to interactively select the directory of their choice in which they wish to save the constructed IMON, else the file will be saved to the working directory "wd" by default
<code>progress_bar</code>	Boolean (default = TRUE) argument that controls whether or not a progress bar for calculations/KEGGREST API GET requests should be printed to the console

## Value

An igraph object containing the constructed IMON

**Examples**

```
##getting vector of SNPs to query:
data("er_snp_vector")
##build IMON using vector:
G <- PanViz::get_IMON(snp_list = er_snp_vector, ego = 5, save_file = FALSE)
```

---

GWAS\_data\_reader

*GWAS\_data\_reader*


---

**Description**

GWAS\_data\_reader

**Usage**

```
GWAS_data_reader(file, snp_col, study_col, trait_col)
```

**Arguments**

file	- Character (string) containing the directory path to a .tsv or .csv file containing summary level GWAS data, typically this can be sourced from major GWAS databases such as the GWAS Catalog or GWAS Central.
snp_col	- Character (string) reflecting the column name containing the SNP (standard dbSNP accession number, e.g. rs992531) data. In data sourced from the GWAS Catalog, this column will typically be named "SNPS" and in GWAS Central this will typically be "Source Marker Accession".
study_col	- Character (string) reflecting the column name containing the study names associated with each SNP. In data sourced from the GWAS Catalog, this column will typically be named "STUDY" and in GWAS Central this will typically be "Study Name".
trait_col	- Character (string) reflecting the column name containing the trait/phenotype names associated with each SNP. In data sourced from the GWAS Catalog, this column will typically be named "DISEASE/TRAIT" and in GWAS Central this will typically be "Annotation Name".

**Value**

A processed dataframe containing only the columns including GWAS studies, traits/phenotypes and relevant SNPs in NCBI standard accession number naming convention

## Examples

```
##getting directory path to GWAS Catalog association .tsv file:
path = system.file("extdata",
  "gwas-association-downloaded_2021-09-13-EFO_1000649.tsv",
  package="PanViz")
##opening/cleaning data:
df <- PanViz::GWAS_data_reader(file = path,
  snp_col = "SNPS",
  study_col = "STUDY",
  trait_col = "DISEASE/TRAIT")
##getting directory path to GWAS Central association .tsv file:
path = system.file("extdata", "GWASCentralMart_ERplusBC.tsv",
  package="PanViz")
##opening/cleaning data:
df <- PanViz::GWAS_data_reader(file = path,
  snp_col = "Source Marker Accession",
  study_col = "Study Name",
  trait_col = "Annotation Name")
```

---

multi\_hex\_col\_mix      *multi\_hex\_col\_mix*

---

## Description

This is a helper function that merges any vector of hex colours

## Usage

```
multi_hex_col_mix(col_vector)
```

## Arguments

col\_vector      - vector of hex colours

## Value

- a single mixed hex color from inputted hex codes

---

NCBI\_clean

*NCBI\_clean*

---

**Description**

NCBI\_clean

**Usage**

NCBI\_clean(queried\_data)

**Arguments**

queried\_data - input queried NCBI gene data

**Value**

remove genes with no genomic information from NCBI query

---

NCBI\_clean\_2

*NCBI\_clean\_2*

---

**Description**

NCBI\_clean\_2

**Usage**

NCBI\_clean\_2(queried\_data)

**Arguments**

queried\_data - rentrez object queried from NCBI

**Value**

return chromosome location, start and end position of gene from NCBI query

---

NCBI_dbSNP_query	<i>NCBI_dbSNP_query</i>
------------------	-------------------------

---

**Description**

NCBI\_dbSNP\_query

**Usage**

NCBI\_dbSNP\_query(snp\_list, progress\_bar)

**Arguments**

snp_list	- list of SNPs to be queried via NCBI dbSNP API
progress_bar	Boolean (default = TRUE) argument that controls whether or not a progress bar for calculations/KEGGREST API GET requests should be printed to the console

**Value**

- raw output from NCBI dbSNP API

---

reaction_cleanup	<i>reaction_cleanup</i>
------------------	-------------------------

---

**Description**

This function helps to clean up queried KEGG reaction recursive lists + separates compound/metabolite and reaction pair data into new sections

**Usage**

reaction\_cleanup(queried\_data)

**Arguments**

queried_data	- input queried KEGG reaction data
--------------	------------------------------------

**Value**

Trimmed recursive lists containing queried KEGG reaction data

---

retry	<i>Retry function</i>
-------	-----------------------

---

**Description**

Internal function for handling errors when accessing APIs

**Usage**

```
retry(
  expr,
  isError = function(x) "try-error" %in% class(x),
  maxErrors = 5,
  sleep = 0
)
```

**Arguments**

expr	This is the function you want to catch and handle errors from
isError	Function for evaluating if provided expression is throwing an error
maxErrors	The maximum number of errors it should handle from the function
sleep	The amount of sleep between a caught error and the next attempt

**Value**

The expression that has been either successfully ran or retried maximum number of times

---

set_base_graph_attributes	<i>set_base_graph_attributes</i>
---------------------------	----------------------------------

---

**Description**

set\_base\_graph\_attributes

**Usage**

```
set_base_graph_attributes(G, colour_groups)
```

**Arguments**

G	igraph object containing KEGG network
colour_groups	logical - whether or not user has indicated on colouring the network by categorical variable i.e. study or trait/phenotype (only available via PanViz::get_grouped_IMON())

**Value**

igraph object with node attributes set

---

set\_snp\_grouping      *snp grouping by chosen categorical variable*

---

**Description**

snp grouping by chosen categorical variable

**Usage**

```
set_snp_grouping(G, unique_group_names, unique_group_cols, group_snps)
```

**Arguments**

G                    - igraph object containing IMON  
unique\_group\_names                    - vector containing unique grouping variable names  
unique\_group\_cols                    - vector containing unique grouping colours for each variable  
group\_snps           - snps split by each variable/group

**Value**

- igraph object containing IMON with labelled and coloured snps by grouping variable

---

snp\_gene\_chr\_match      *snp\_gene\_chr\_match*

---

**Description**

snp\_gene\_chr\_match

**Usage**

```
snp_gene_chr_match(snp_loc, gene_loc)
```

**Arguments**

snp\_loc            - snp locations  
gene\_loc           - dataframe of genes and their chromosome numbers and start/stop positions

**Value**

- a recursive list of gene with their relative snps that have the same chromosome number

---

snp_gene_map	<i>Fast vectorised SNP to gene chromosome number and genomic location mapping</i>
--------------	---

---

**Description**

Fast vectorised SNP to gene chromosome number and genomic location mapping

**Usage**

```
snp_gene_map(gene_loc, snp_loc)
```

**Arguments**

gene_loc	dataframe containing KEGG genes and relevant chromosome number and positions
snp_loc	dataframe containing queried SNPs and relevant chromosome number and positions

**Value**

an adjacency list of SNPs with their relevant mapped genes to their genomic location



# Index

## \* datasets

- er\_snp\_vector, [7](#)
  
- adj\_list\_to\_igraph, [4](#)
- adjl\_to\_G, [2](#)
- adjl\_to\_G\_grouped, [3](#)
  
- colour\_IMON, [4](#)
  
- dbSNP\_query\_check, [5](#)
- dbSNP\_query\_clean, [5](#)
- decompose\_IMON, [6](#)
  
- ego\_IMON, [6](#)
- er\_snp\_vector, [7](#)
  
- get\_grouped\_IMON, [7](#)
- get\_IMON, [9](#)
- GWAS\_data\_reader, [10](#)
  
- multi\_hex\_col\_mix, [11](#)
  
- NCBI\_clean, [12](#)
- NCBI\_clean\_2, [12](#)
- NCBI\_dbSNP\_query, [13](#)
  
- reaction\_cleanup, [13](#)
- retry, [14](#)
  
- set\_base\_graph\_attributes, [14](#)
- set\_snp\_grouping, [15](#)
- snp\_gene\_chr\_match, [15](#)
- snp\_gene\_map, [16](#)