

# Package ‘InterCellar’

October 14, 2021

**Title** InterCellar: an R-Shiny app for interactive analysis and exploration of cell-cell communication in single-cell transcriptomics

**Version** 1.0.0

**Description** InterCellar is implemented as an R/Bioconductor Package containing a Shiny app that allows users to interactively analyze cell-cell communication from scRNA-seq data. Starting from precomputed ligand-receptor interactions, InterCellar provides filtering options, annotations and multiple visualizations to explore clusters, genes and functions. Finally, the user can define interaction-pairs modules and link them to significant functional terms from Pathways or Gene Ontology.

**License** MIT + file LICENSE

**Imports** config, golem, shiny, DT, shinydashboard, shinyFiles, shinycssloaders, data.table, fs, dplyr, tidyr, circlize, colourpicker, dendextend, factoextra, ggplot2, plotly, plyr, shinyFeedback, shinyalert, tibble, umap, visNetwork, wordcloud2, readxl, htmlwidgets, colorspace, signal, scales, htmltools, ComplexHeatmap, grDevices, stats, tools, utils, biomaRt, rlang, fmsb

**Encoding** UTF-8

**RoxygenNote** 7.1.1

**Suggests** testthat (>= 3.0.0), knitr, rmarkdown, glue, graphite, processx, attempt, BiocStyle, igraph

**Config/testthat/edition** 3

**URL** <https://github.com/martaint/InterCellar>

**BugReports** <https://github.com/martaint/InterCellar/issues>

**VignetteBuilder** knitr

**biocViews** Software, SingleCell, Visualization, GO, Transcriptomics

**Depends** R (>= 4.1)

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

**git\_branch** RELEASE\_3\_13

**git\_last\_commit** 094f1da

**git\_last\_commit\_date** 2021-05-19

**Date/Publication** 2021-10-14

**Author** Marta Interlandi [cre, aut] (<<https://orcid.org/0000-0002-6863-2552>>)

**Maintainer** Marta Interlandi <marta.interlandi01@gmail.com>

## R topics documented:

annotateGO . . . . .	3
annotatePathways . . . . .	4
buildPairsbyFunctionMatrix . . . . .	4
checkLL_RR . . . . .	5
circlePlot . . . . .	5
combineAnnotations . . . . .	6
createBarPlot1_ggplot . . . . .	6
createBarPlot2_CV . . . . .	7
createBarPlot2_ggplot . . . . .	7
createBarPlot_CV . . . . .	8
createNetwork . . . . .	8
dendroIntPairModules . . . . .	9
elbowPoint . . . . .	9
ensemblLink . . . . .	10
getBack2BackBarplot . . . . .	10
getBarplotDF . . . . .	11
getBarplotDF2 . . . . .	11
getClusterNames . . . . .	12
getClusterNetwork . . . . .	12
getClusterSize . . . . .	13
getDotPlot_selInt . . . . .	13
getGeneTable . . . . .	14
getGObiomaRt . . . . .	14
getHitsf . . . . .	15
getIntFlow . . . . .	15
getNtermsBYdb . . . . .	16
getNumLR . . . . .	16
getRadarPlot . . . . .	17
getRankedTerms . . . . .	17
getSignificantFunctions . . . . .	18
getSunburst . . . . .	18
getUMAPipModules . . . . .	19
getUniqueDotplot . . . . .	20
goLink . . . . .	20
input.data . . . . .	21
read.CPDBv2 . . . . .	21
read.customInput . . . . .	22
read.SCsignalR . . . . .	22

<i>annotateGO</i>	3
run_app . . . . .	23
subsetFuncMatBYFlow . . . . .	24
swap.RLint . . . . .	24
uniprotLink . . . . .	25
updateInputLR . . . . .	25
<b>Index</b>	<b>26</b>

---

<i>annotateGO</i>	<i>Perform GO annotation of input data</i>
-------------------	--

---

**Description**

Perform GO annotation of input data

**Usage**

```

annotateGO(
  input_select_ensembl,
  input_go_evidence_exclude,
  input_go_sources_checkbox,
  input.data
)

```

**Arguments**

<code>input_select_ensembl</code>	ensembl version selected by user
<code>input_go_evidence_exclude</code>	evidence codes to exclude by user
<code>input_go_sources_checkbox</code>	GO sources to use by user
<code>input.data</code>	preprocessed input data

**Value**

GO\_annotation

annotatePathways      *Annotate pathways for input data*

---

**Description**

Annotate pathways for input data

**Usage**

```
annotatePathways(selected.db, input.data)
```

**Arguments**

selected.db	pathways sources to use
input.data	filtered input data

**Value**

pathways\_annotation

---

buildPairsbyFunctionMatrix  
*Build binary matrix with int-pairs in rows, functions in cols*

---

**Description**

Build binary matrix with int-pairs in rows, functions in cols

**Usage**

```
buildPairsbyFunctionMatrix(functions_df)
```

**Arguments**

functions_df	annotated df (GO/path/combined)
--------------	---------------------------------

**Value**

binary matrix

---

checkLL_RR	<i>Manually change the annotation of L-L and R-R pairs</i>
------------	--

---

**Description**

Manually change the annotation of L-L and R-R pairs

**Usage**

```
checkLL_RR(input.data)
```

**Arguments**

input.data      preprocessed table

**Value**

input.data

**Examples**

```
data(input.data)
checked.input.data <- checkLL_RR(input.data)
```

---

circlePlot	<i>Plot circle plot</i>
------------	-------------------------

---

**Description**

Plot circle plot

**Usage**

```
circlePlot(data, cluster_colors, ipm_color, int_flow, link.color)
```

**Arguments**

data              subset of input data by flow / intpair module  
cluster\_colors    global  
ipm\_color        single color for chosen int-pair module  
int\_flow         string specifying the flow  
link.color        string specifying variable by which to color links

**Value**

circle plot

---

combineAnnotations      *Combine GO annotation and pathways in a unique object*

---

**Description**

Combine GO annotation and pathways in a unique object

**Usage**

```
combineAnnotations(GO_annotation, pathways_annotation)
```

**Arguments**

```
GO_annotation    data  
pathways_annotation  
                 data
```

**Value**

combined annotation dataframe

---

createBarPlot1\_ggplot    *Create ggplot barplot to be saved in tiff*

---

**Description**

Create ggplot barplot to be saved in tiff

**Usage**

```
createBarPlot1_ggplot(barplotDF, input_cluster_selected_checkbox)
```

**Arguments**

```
barplotDF            dataframe with N interactions per cluster (auto/para)  
input_cluster_selected_checkbox  
                     checkbox input
```

**Value**

ggplot barplot

---

createBarPlot2\_CV      *Create barplot of number of interaction for selected cluster*

---

**Description**

Create barplot of number of interaction for selected cluster

**Usage**

```
createBarPlot2_CV(  
  barplotDF2,  
  input_cluster_selected_checkbox,  
  input_clust_barplot2  
)
```

**Arguments**

barplotDF2      dataframe with barplot data  
input\_cluster\_selected\_checkbox  
                  selected clusters to keep  
input\_clust\_barplot2  
                  selected cluster to plot

**Value**

plotly fig

---

createBarPlot2\_ggplot      *Create ggplot barplot of Nint per cluster selected*

---

**Description**

Create ggplot barplot of Nint per cluster selected

**Usage**

```
createBarPlot2_ggplot(  
  barplotDF2,  
  input_cluster_selected_checkbox,  
  input_clust_barplot2  
)
```

**Arguments**

barplotDF2      dataframe with barplot data  
input\_cluster\_selected\_checkbox  
                    selected clusters to keep  
input\_clust\_barplot2  
                    selected cluster to plot

**Value**

ggplot barplot

---

createBarPlot\_CV      *Create Barplot cluster-verse*

---

**Description**

Create Barplot cluster-verse

**Usage**

```
createBarPlot_CV(barplotDF, input_cluster_selected_checkbox)
```

**Arguments**

barplotDF      dataframe with N interactions per cluster (auto/para)  
input\_cluster\_selected\_checkbox  
                    checkbox input

**Value**

plotly barplot

---

createNetwork      *Create Network of clusters*

---

**Description**

Create Network of clusters

**Usage**

```
createNetwork(data.filt.cluster)
```



**Arguments**

`data.filt.cluster`  
 filtered input data (by clusters)

**Value**

list containing nodes and edges for network

---

`dendroIntPairModules` *Get dendrogram of int pair modules*

---

**Description**

Get dendrogram of int pair modules

**Usage**

`dendroIntPairModules(pairs_func_matrix)`

**Arguments**

`pairs_func_matrix`  
 binary matrix pairs x functions

**Value**

list with dendrogram, hclust and umap

---

`elbowPoint` *Determine the elbow point on a curve (from package akmedoids)*

---

**Description**

Given a list of x, y coordinates on a curve, function determines the elbow point of the curve.

**Usage**

`elbowPoint(x, y)`

**Arguments**

`x` vector of x coordinates of points on the curve  
`y` vector of y coordinates of points on the curve

**Details**

highlight the maximum curvature to identify the elbow point (credit: 'github.com/agentlans')

**Value**

an x, y coordinates of the elbow point.

---

ensemblLink	<i>Get html link to ensembl</i>
-------------	---------------------------------

---

**Description**

Get html link to ensembl

**Usage**

```
ensemblLink(ensembl)
```

**Arguments**

ensembl	symbol
---------	--------

**Value**

html link to website

---

getBack2BackBarplot	<i>Get back-to-back barplot for 2 conditions comparison</i>
---------------------	---

---

**Description**

Get back-to-back barplot for 2 conditions comparison

**Usage**

```
getBack2BackBarplot(tab_c1, tab_c2, lab_c1, lab_c2)
```

**Arguments**

tab_c1	table from csv file (barplot#1) containing data for condition 1
tab_c2	table from csv file (barplot#1)containing data for condition 2
lab_c1	label for condition 1
lab_c2	label for condition 2

**Value**

ggplot object

---

getBarplotDF	<i>Get dataframe for plotting barplot (all clusters)</i>
--------------	--

---

**Description**

Get dataframe for plotting barplot (all clusters)

**Usage**

```
getBarplotDF(data.filt.bar, input_cluster_selected_checkbox)
```

**Arguments**

data.filt.bar    filtered object (checkbox auto/para)  
input\_cluster\_selected\_checkbox  
checkbox input

**Value**

dataframe with number of interactions per cluster auto/para

---

getBarplotDF2	<i>Get dataframe for barplot (by cluster)</i>
---------------	---

---

**Description**

Get dataframe for barplot (by cluster)

**Usage**

```
getBarplotDF2(filt.data, input_cluster_selected_checkbox, input_clust_barplot2)
```

**Arguments**

filt.data        input data filtered in cluster-verse  
input\_cluster\_selected\_checkbox  
selected clusters to keep  
input\_clust\_barplot2  
selected cluster to plot

**Value**

dataframe with num int per cluster

---

getClusterNames      *Get clusters names from initial input data*

---

**Description**

Get clusters names from initial input data

**Usage**

```
getClusterNames(input.data)
```

**Arguments**

input.data      preprocessed input data

**Value**

named list of clusters

**Examples**

```
data(input.data)
cluster_list <- getClusterNames(input.data)
```

---

getClusterNetwork      *Creating edges dataframe for network of clusters*

---

**Description**

Creating edges dataframe for network of clusters

**Usage**

```
getClusterNetwork(input.data)
```

**Arguments**

input.data      preprocessed input data

**Value**

edges dataframe

---

getClusterSize	<i>Get Clusters size</i>
----------------	--------------------------

---

**Description**

Get Clusters size

**Usage**

```
getClusterSize(cl, edges.df)
```

**Arguments**

cl	cluster name
edges.df	dataframe with edges for network

**Value**

sum of interactions for that cluster

---

getDotPlot_selInt	<i>Functions to plot DotPlots</i>
-------------------	-----------------------------------

---

**Description**

Functions to plot DotPlots

**Usage**

```
getDotPlot_selInt(
  selected_tab,
  clust.order,
  low_color = "aquamarine",
  high_color = "#131780"
)
```

**Arguments**

selected_tab	table of selected rows from gene tableeeeeeee
clust.order	how to order clusters
low_color	of dotplot
high_color	of dotplot

**Value**

list with modified selected data and ggplot2 dotplot

getGeneTable                    *Get table for gene-verse*

---

**Description**

Get table for gene-verse

**Usage**

```
getGeneTable(input.data)
```

**Arguments**

input.data            preprocessed input data

**Value**

gene table with unique intpairs (no connection to clusters)

**Examples**

```
data(input.data)
gene_table <- getGeneTable(input.data)
```

---

getGObiomaRt                    *Connection to Ensembl via biomaRt to get GO terms*

---

**Description**

Connection to Ensembl via biomaRt to get GO terms

**Usage**

```
getGObiomaRt(input_select_ensembl, input.data)
```

**Arguments**

input\_select\_ensembl  
                          chosen version of Ensembl

input.data            filtered input data

**Value**

dataframe with GO annotation

---

getHitsf	<i>Subfunction to calculate significant functions by permutation test</i>
----------	---

---

**Description**

Subfunction to calculate significant functions by permutation test

**Usage**

```
getHitsf(mat, gpModules_assign)
```

**Arguments**

mat	binary matrix of functional terms by int-pairs
gpModules_assign	assignment of intpairs to modules

**Value**

matrix with hits

---

getIntFlow	<i>Get subset of interactions corresponding to a certain viewpoint and flow</i>
------------	---

---

**Description**

Get subset of interactions corresponding to a certain viewpoint and flow

**Usage**

```
getIntFlow(vp, input.data, flow)
```

**Arguments**

vp	viewpoint cluster
input.data	preprocessed/filtered input data
flow	one among directed_out, directed_in or undirected

**Value**

subset of data

**Examples**

```
data(input.data)
caf_out <- getIntFlow(vp = "CAF", input.data, flow = "directed_out")
```

---

getNtermsBYdb      *Calculate number of terms of a database*

---

**Description**

Calculate number of terms of a database

**Usage**

```
getNtermsBYdb(annotation)
```

**Arguments**

annotation      data from either pathways, GO or combined

**Value**

number of terms by dataset

---

getNumLR      *Get number of unique ligands and receptors*

---

**Description**

Get number of unique ligands and receptors

**Usage**

```
getNumLR(gene.table, type)
```

**Arguments**

gene.table      gene table of unique int-pairs  
type            either L or R

**Value**

number of L or R genes



---

getRadarPlot	<i>Get radar plot of relative numbers of interactions for a certain cell type</i>
--------------	---

---

**Description**

Get radar plot of relative numbers of interactions for a certain cell type

**Usage**

```
getRadarPlot(tab_c1, tab_c2, lab_c1, lab_c2, cell_name)
```

**Arguments**

tab_c1	table from csv file (barplot#2) containing data for condition 1
tab_c2	table from csv file (barplot#2) containing data for condition 2
lab_c1	label for condition 1
lab_c2	label for condition 2
cell_name	label of cell type of interest

**Value**

plot

---

getRankedTerms	<i>Get table with ranked functional terms</i>
----------------	---

---

**Description**

Get table with ranked functional terms

**Usage**

```
getRankedTerms(data.fun.annot, gene.table)
```

**Arguments**

data.fun.annot	annotated df (GO/path/combined)
gene.table	of unique intpairs

**Value**

table with ranking

getSignificantFunctions

*Calculate significant function per intpair module*

---

### **Description**

Calculate significant function per intpair module

### **Usage**

```
getSignificantFunctions(  
  subGenePairs_func_mat,  
  gpModules_assign,  
  rank.terms,  
  input_maxPval  
)
```

### **Arguments**

subGenePairs_func_mat	subset of binary mat
gpModules_assign	assignment of intpairs to modules
rank.terms	table of ranked functions
input_maxPval	threshold of significance

### **Value**

table with significant functions

---

getSunburst

*Get Sunburst plot of selected functional terms*

---

### **Description**

Get Sunburst plot of selected functional terms

### **Usage**

```
getSunburst(sel.data, func_selected, int_p_fun, cluster.colors)
```

**Arguments**

`sel.data`            dataframe of selected functions  
`func_selected`    the selected functional term  
`int_p_fun`            dataframe with int pairs annotated to this function  
`cluster.colors`    for plotting

**Value**

plotly figure

---

`getUMAPipModules`            *Get UMAP for IP modules*

---

**Description**

Get UMAP for IP modules

**Usage**

```

getUMAPipModules(
  intPairs.dendro,
  gpModules_assign,
  gene.table,
  ipm_colors,
  input_ipM_UMAPcolors
)

```

**Arguments**

`intPairs.dendro`            list output of dendrogram  
`gpModules_assign`        named vector of module assignment  
`gene.table`                unique intpairs table  
`ipm_colors`                for intpair modules  
`input_ipM_UMAPcolors`    user choice for coloring umap

**Value**

plotly umap

getUniqueDotplot      *Plot dotplot containing only unique int-pair/cluster pairs with many conditions*

---

**Description**

Plot dotplot containing only unique int-pair/cluster pairs with many conditions

**Usage**

```
getUniqueDotplot(data_dotplot)
```

**Arguments**

data\_dotplot      table with selected int\_pairs for multiple conditions

**Value**

ggplot object

---

goLink      *Get GO link*

---

**Description**

Get GO link

**Usage**

```
goLink(go_id)
```

**Arguments**

go\_id      string

**Value**

html link to website

---

input.data

*Input Data example*


---

**Description**

A dataset obtained from Tirosh et al melanoma dataset, running CellPhoneDBv2. This data is generated by InterCellar running read.CPDBv2()

**Usage**

```
input.data
```

**Format**

A data frame with 5638 rows and 11 variables:

**int\_pair** interaction pair name, geneA & geneB

**geneA** name, hgnc\_symbol

**geneB** name, hgnc\_symbol

**typeA** molecular type of geneA, either L (ligand) or R (receptor)

**typeB** molecular type of geneB, either L (ligand) or R (receptor)

**clustA** name of first cluster, either character or number

**clustB** name of second cluster, either character or number

**score** int-pair score as avg expression of geneA and geneB over clustA and clustB, decimal

**p\_value** int-pair pvalue, decimal

**annotation\_strategy** database from which the int-pair was retrieved

**int.type** either autocrine or paracrine

---

read.CPDBv2

*Read output from CellPhoneDB v2.*


---

**Description**

Output is a folder containing 4 .txt files - deconvoluted.txt: containing list of single genes and their mean expression in each cluster (not considered); - means.txt: containing list of interacting pairs with info regarding L/R, annotation strategy and mean value of all pairs over cluster couples. - pvalues.txt: same as means, but containing pvalue of each pair, for each cluster couple. - significant\_means.txt: only means of those pairs that have pvalue < 0.05. Has one more column:rank. If the statistical analysis is not run, the folder would contain only deconvoluted and means

**Usage**

```
read.CPDBv2(folder)
```

**Arguments**

folder                    folder containing output

**Value**

input.data which is the pre-processed object with annotated L-R pairs

---

read.customInput            *Read custom input file and re-structure it with InterCellar format*

---

**Description**

Read custom input file and re-structure it with InterCellar format

**Usage**

```
read.customInput(tab, separator)
```

**Arguments**

tab                        custom input table  
separator                character that separates two elements of an interaction pair

**Value**

preprocessed table

---

read.SCSignalR            *Read output from SingleCellSignalR*

---

**Description**

SCSR description: the output folder is a collection of txt files, one for each clusters pair considered. The "paracrine" option looks for ligands expressed in cluster A and their associated receptors according to LRdb that are expressed in any other cluster but A. These interactions are labelled "paracrine". The interactions that involve a ligand and a receptor, both differentially expressed in their respective cell clusters according to the **edgeR** analysis performed by the **cluster\_analysis()** function, are labelled "specific". The "autocrine" option searches for ligands expressed in cell cluster A and their associated receptors also expressed in A. These interactions are labelled "autocrine". Additionally, it searches for those associated receptors in the other cell clusters (not A) to cover the part of the signaling that is "autocrine" and "paracrine" simultaneously. These interactions are labelled "autocrine/paracrine". This file is a 4-column table: ligands, receptors, interaction types ("paracrine", "autocrine", "autocrine/paracrine" and "specific"), and the associated LRscore. InterCellar: rename autocrinelparacrine to paracrine

**Usage**

```
read.SCsignalR(folder)
```

**Arguments**

folder            containing output from SingleCellSignalR, named cell-signaling

**Value**

input.data: preprocessed object with annotated L-R pairs

---

run_app	<i>Run the Shiny Application</i>
---------	----------------------------------

---

**Description**

Run the Shiny Application

**Usage**

```
run_app(reproducible = TRUE)
```

**Arguments**

reproducible    boolean for setting a seed, making plots reproducible

**Value**

a running instance of InterCellar

**Examples**

```
## Not run:  
run_app()  
  
## End(Not run)
```

---

subsetFuncMatBYFlow     *Subset pairs-function matrix by selected flow*

---

**Description**

Subset pairs-function matrix by selected flow

**Usage**

```
subsetFuncMatBYFlow(pairs_func_matrix, flow_df)
```

**Arguments**

pairs\_func\_matrix     binary  
flow\_df                subset of input data by flow

**Value**

subset of binary mat

---

swap.RLint             *Swaps interaction pairs that are R-L to L-R*

---

**Description**

Swaps interaction pairs that are R-L to L-R

**Usage**

```
swap.RLint(RLint)
```

**Arguments**

RLint                 subset of R-L interactions

**Value**

input data with ordered L-R pairs and L-L/R-R



---

uniprotLink	<i>Get html link to uniprot</i>
-------------	---------------------------------

---

**Description**

Get html link to uniprot

**Usage**

```
uniprotLink(uniprot)
```

**Arguments**

uniprot            symbol

**Value**

html link to website

---

updateInputLR	<i>Function that orders all interaction pairs as L-R. Leaves unchanged the R-R and L-L</i>
---------------	--

---

**Description**

Function that orders all interaction pairs as L-R. Leaves unchanged the R-R and L-L

**Usage**

```
updateInputLR(input.data)
```

**Arguments**

input.data        uploaded data

**Value**

ordered input data

# Index

## \* datasets

input.data, 21

annotateGO, 3

annotatePathways, 4

buildPairsbyFunctionMatrix, 4

checkLL\_RR, 5

circlePlot, 5

combineAnnotations, 6

createBarPlot1\_ggplot, 6

createBarPlot2\_CV, 7

createBarPlot2\_ggplot, 7

createBarPlot\_CV, 8

createNetwork, 8

dendroIntPairModules, 9

elbowPoint, 9

ensemblLink, 10

getBack2BackBarplot, 10

getBarplotDF, 11

getBarplotDF2, 11

getClusterNames, 12

getClusterNetwork, 12

getClusterSize, 13

getDotPlot\_selInt, 13

getGeneTable, 14

getGObiomaRt, 14

getHitsf, 15

getIntFlow, 15

getNtermsBYdb, 16

getNumLR, 16

getRadarPlot, 17

getRankedTerms, 17

getSignificantFunctions, 18

getSunburst, 18

getUMAPipModules, 19

getUniqueDotplot, 20

goLink, 20

input.data, 21

read.CPDBv2, 21

read.customInput, 22

read.SCSignalR, 22

run\_app, 23

subsetFuncMatBYFlow, 24

swap.RLint, 24

uniprotLink, 25

updateInputLR, 25