

# Package ‘MoonlightR’

September 28, 2021

**Type** Package

**Title** Identify oncogenes and tumor suppressor genes from omics data

**Version** 1.18.0

**Date** 07-08-2020

**Author** Antonio Colaprico\*, Catharina Olsen\*, Claudia Cava, Thilde Terkelsen, Laura Cantini, Andre Olsen, Gloria Bertoli, Andrei Zinovyev, Emmanuel Barillot, Isabella Castiglioni, Elena Papaleo, Gianluca Bontempi

**Maintainer** Antonio Colaprico <axc1833@med.miami.edu>, Catharina Olsen <colsen@ulb.ac.be>

**Depends** R (>= 3.5), doParallel, foreach

**Imports** parmigene, randomForest, SummarizedExperiment, gplots, circlize, RColorBrewer, HiveR, clusterProfiler, DOSE, Biobase, limma, grDevices, graphics, TCGAbiolinks, GEOquery, stats, RISmed, grid, utils

**Description** Motivation: The understanding of cancer mechanism requires the identification of genes playing a role in the development of the pathology and the characterization of their role (notably oncogenes and tumor suppressors). Results: We present an R/bioconductor package called MoonlightR which returns a list of candidate driver genes for specific cancer types on the basis of TCGA expression data. The method first infers gene regulatory networks and then carries out a functional enrichment analysis (FEA) (implementing an upstream regulator analysis, URA) to score the importance of well-known biological processes with respect to the studied cancer type. Eventually, by means of random forests, MoonlightR predicts two specific roles for the candidate driver genes: i) tumor suppressor genes (TSGs) and ii) oncogenes (OCGs). As a consequence, this methodology does not only identify genes playing a dual role (e.g. TSG in one cancer type and OCG in another) but also helps in elucidating the biological processes underlying their specific roles. In particular, MoonlightR can be used to discover OCGs and TSGs in the same cancer type. This may help

in answering the question whether some genes change role between early stages (I, II) and late stages (III, IV) in breast cancer. In the future, this analysis could be useful to determine the causes of different resistances to chemotherapeutic treatments.

**License** GPL (>= 3)

**biocViews** DNAMethylation, DifferentialMethylation, GeneRegulation, GeneExpression, MethylationArray, DifferentialExpression, Pathways, Network, Survival, GeneSetEnrichment, NetworkEnrichment

**Suggests** BiocStyle, knitr, rmarkdown, testthat, devtools, roxygen2, png

**VignetteBuilder** knitr

**LazyData** true

**URL** <https://github.com/ibsquare/MoonlightR>

**BugReports** <https://github.com/ibsquare/MoonlightR/issues>

**RoxygenNote** 6.1.1

**Encoding** UTF-8

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

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

**git\_last\_commit** d01eece

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

**Date/Publication** 2021-09-28

## R topics documented:

dataFilt	3
dataGRN	4
dataURA	4
DEGsmatrix	5
DiseaseList	5
DPA	6
EAGenes	7
FEA	7
GDCprojects	8
geneInfo	8
GEO_TCGAtab	9
getDataGEO	9
getDataTCGA	10
GRN	11
GSEA	12
knownDriverGenes	12
listMoonlight	13

<i>dataFilt</i>	3
LPA . . . . .	13
moonlight . . . . .	14
MoonlightR . . . . .	15
plotCircos . . . . .	15
plotFEA . . . . .	16
plotNetworkHive . . . . .	17
plotURA . . . . .	18
PRA . . . . .	18
tabGrowBlock . . . . .	19
URA . . . . .	20
<b>Index</b>	<b>21</b>

---

<code>dataFilt</code>	<i>Gene Expression (Rnaseqv2) data from TCGA LUAD</i>
-----------------------	---

---

**Description**

A data set containing the following data:

**Usage**

```
data(dataFilt)
```

**Format**

A 13742x20 matrix

**Details**

- `dataFilt` matrix with 13742 rows (genes) and 20 columns samples with TCGA's barcodes (10TP, 10NT)

**Value**

a 13742x20 matrix

---

dataGRN	<i>GRN gene regulatory network output</i>
---------	---

---

**Description**

output from GRN function

**Usage**

```
data(dataGRN)
```

**Format**

A large list of 2 elements

**Details**

- dataGRN list of 2 elements miTFGenes, maxmi from GRN function

**Value**

a large list of 2 elements

---

dataURA	<i>Output example from function Upstram Regulator Analysis</i>
---------	--

---

**Description**

A data set containing the following data:

**Usage**

```
data(dataURA)
```

**Format**

A data frame with 100 rows and 2 variables

**Details**

- dataURA matrix with 100 rows (genes) and 2 columns "apoptosis" "proliferation of cells"

**Value**

a 100x2 matrix

---

DEGsmatrix

*DEG Differentially expressed genes*

---

**Description**

A data set containing the following data:

**Usage**

```
data(DEGsmatrix)
```

**Format**

A 3502x5 matrix

**Details**

- DEGsmatrix matrix with 3502 rows (genes) and five columns "logFC" "logCPM" "LR" "PValue" "FDR"

**Value**

the 3502x5 matrix

---

DiseaseList

*Information on 101 biological processes*

---

**Description**

A data set containing the following data:

**Usage**

```
data(DiseaseList)
```

**Format**

A list of 101 matrices

**Details**

- DiseaseList list for 101 biological processes, each containing a matrix with five columns: ID, Genes.in.dataset, Prediction based on expression direction, Log ratio, Findings

**Value**

list of 101 matrices

---

DPA

*DPA*

---

### Description

This function carries out the differential phenotypes analysis

### Usage

```
DPA(dataType, dataFilt, dataConsortium = "TCGA", fdr.cut = 0.01,  
    logFC.cut = 1, diffmean.cut = 0.25, samplesType, colDescription,  
    gset, gsetFile = "gsetFile.RData")
```

### Arguments

dataType	selected
dataFilt	obtained from getDataTCGA
dataConsortium	is TCGA or GEO, default TCGA
fdr.cut	is a threshold to filter DEGs according their p-value corrected
logFC.cut	is a threshold to filter DEGs according their logFC
diffmean.cut	diffmean.cut for DMR
samplesType	samplesType
colDescription	colDescription
gset	gset
gsetFile	gsetFile

### Value

result matrix from differential phenotype analysis

### Examples

```
dataDEGs <- DPA(dataFilt = dataFilt, dataType = "Gene expression")
```

---

EAGenes

*Information about genes*

---

**Description**

A data set containing the following data:

**Usage**

```
data(EAGenes)
```

**Format**

A 20038x5 matrix

**Details**

- EAGenes matrix with 20038 rows (genes) and five columns "ID" "Gene" "Description" "Location" "Family"

**Value**

a 20038x5 matrix

---

FEA

*FEA*

---

**Description**

This function carries out the functional enrichment analysis (FEA)

**Usage**

```
FEA(BPname = NULL, DEGsmatrix)
```

**Arguments**

BPname	BPname biological process such as "proliferation of cells", "ALL" (default) if FEA should be carried out for all 101 biological processes
DEGsmatrix	DEGsmatrix output from DEA such as dataDEGs"

**Value**

matrix from FEA

**Examples**

```
dataDEGs <- DPA(dataFilt = dataFilt,  
dataType = "Gene expression")  
dataFEA <- FEA(DEGsmatrix = dataDEGs)
```

---

**GDCprojects***Information on GDC projects*

---

**Description**

A character vector of GDC projects:

**Usage**

```
data(GDCprojects)
```

**Format**

A character vector of 39 elements

**Details**

- character vector for GDC projects.

**Value**

character vector of 39 elements

---

**geneInfo***Information about genes for normalization*

---

**Description**

A data set containing the following data:

**Usage**

```
data(geneInfo)
```

**Format**

A data frame with 20531 rows and 3 variables

**Details**

- geneInfo matrix with 20531 rows (genes) and 3 columns "geneLength" "gcContent" "chr"



**Value**

a 20531x3 matrix

---

GEO_TCGAtab	<i>Information on GEO data (and overlap with TCGA)#' A data set containing the following data:</i>
-------------	--

---

**Description**

- GEO\_TCGAtab a 18x12 matrix that provides the GEO data set we matched to one of the 18 given TCGA cancer types

**Usage**

```
data(GEO_TCGAtab)
```

**Format**

A 101x3 matrix

**Value**

a 101x3 matrix

---

getDataGEO	<i>getDataGEO</i>
------------	-------------------

---

**Description**

This function retrieves and prepares GEO data

**Usage**

```
getDataGEO(GEOobject = "GSE39004", platform = "GPL6244",
            TCGAtumor = NULL)
```

**Arguments**

GEOobject	GEOobject
platform	platform
TCGAtumor	tumor name

**Value**

return GEO gset

**Examples**

```
## Not run:
dataGEO <- getDataGEO(GEOobject = "GSE20347",platform = "GPL571")

## End(Not run)
```

---

getDataTCGA

*getDataTCGA*

---

**Description**

This function retrieves and prepares TCGA data

**Usage**

```
getDataTCGA(cancerType, dataType, directory, cor.cut = 0.6,
  qnt.cut = 0.25, nSample, stage = "ALL", subtype = 0,
  samples = NULL)
```

**Arguments**

cancerType	select cancer type for which analysis should be run. panCancer for all available cancer types in TCGA. Defaults to panCancer
dataType	is dataType such as gene expression, cnv, methylation etc.
directory	Directory/Folder where the data was downloaded. Default: GDCdata
cor.cut	cor.cut
qnt.cut	qnt.cut
nSample	nSample
stage	stage
subtype	subtype
samples	samples

**Value**

returns filtered TCGA data

**Examples**

```
## Not run:
dataFilt <- getDataTCGA(cancerType = "LUAD",
  dataType = "Gene expression", directory = "data", nSample = 4)

## End(Not run)
```

---

GRN	<i>Generate network</i>
-----	-------------------------

---

### Description

This function carries out the gene regulatory network inference using parmigene

### Usage

```
GRN(TFs, DEGsmatrix, DiffGenes = FALSE, normCounts, kNearest = 3,  
    nGenesPerm = 10, nBoot = 10)
```

### Arguments

TFs	a vector of genes.
DEGsmatrix	DEGsmatrix output from DEA such as dataDEGs
DiffGenes	if TRUE consider only diff.expr genes in GRN
normCounts	is a matrix of gene expression with genes in rows and samples in columns.
kNearest	the number of nearest neighbors to consider to estimate the mutual information. Must be less than the number of columns of normCounts.
nGenesPerm	nGenesPerm
nBoot	nBoot

### Value

an adjacent matrix

### Examples

```
dataDEGs <- DEGsmatrix  
dataGRN <- GRN(TFs = rownames(dataDEGs)[1:100],  
              DEGsmatrix = dataDEGs,  
              DiffGenes = TRUE,  
              normCounts = dataFilt)
```

GSEA

*GSEA*

---

**Description**

This function carries out the GSEA enrichment analysis.

**Usage**

```
GSEA(DEGsmatrix, top, plot = FALSE)
```

**Arguments**

DEGsmatrix	DEGsmatrix output from DEA such as dataDEGs
top	is the number of top BP to plot
plot	if TRUE return a GSEA's plot

**Value**

return GSEA result

**Examples**

```
dataDEGs <- DEGsmatrix  
# dataFEA <- GSEA(DEGsmatrix = dataDEGs)
```

---

knownDriverGenes*Information on known cancer driver gene from COSMIC*

---

**Description**

A data set containing the following data:

**Usage**

```
data(knownDriverGenes)
```

**Format**

A 101x3 matrix

**Details**

- TSG known tumor suppressor genes
- OCG known oncogenes

**Value**

a 101x3 matrix

---

listMoonlight	<i>Output list from Moonlight</i>
---------------	-----------------------------------

---

**Description**

A list containing the following data:

**Usage**

```
data(listMoonlight)
```

**Format**

A Large list with 5 elements

**Details**

- listMoonlight output from moonlight's pipeline containing dataDEGs, dataURA, listCandidates

**Value**

output from moonlight pipeline

---

LPA	<i>LPA</i>
-----	------------

---

**Description**

This function carries out the literature phenotype analysis (LPA)

**Usage**

```
LPA(dataDEGs, BP, BPlist)
```

**Arguments**

dataDEGs	is output from DEA
BP	is biological process
BPlist	is list of genes annotated in BP

**Value**

table with number of pubmed that affects, increase or decrease genes annotated in BP

**Examples**

```
data(DEGsmatrix)
BPselected <- c("apoptosis")
BPannotations <- DiseaseList[[match(BPselected, names(DiseaseList))]]$ID
```

---

moonlight

*moonlight pipeline*

---

**Description**

moonlight is a tool for identification of cancer driver genes. This function wraps the different steps of the complete analysis workflow. Providing different solutions:

1. MoonlighR::FEA
2. MoonlighR::URA
3. MoonlighR::PIA

**Usage**

```
moonlight(cancerType = "panCancer", dataType = "Gene expression",
  directory = "GDCdata", BPname = NULL, cor.cut = 0.6,
  qnt.cut = 0.25, Genelist = NULL, fdr.cut = 0.01, logFC.cut = 1,
  corThreshold = 0.6, kNearest = 3, nGenesPerm = 10,
  DiffGenes = FALSE, nBoot = 100, nTF = NULL, nSample = NULL,
  thres.role = 0, stage = NULL, subtype = 0, samples = NULL)
```

**Arguments**

cancerType	select cancer type for which analysis should be run. panCancer for all available cancer types in TCGA. Defaults to panCancer
dataType	dataType
directory	directory
BPname	biological processes to use, if NULL: all processes will be used in analysis, RF for candidate; if not NULL the candidates for these processes will be determined (no learning)
cor.cut	cor.cut Threshold
qnt.cut	qnt.cut Threshold
Genelist	Genelist
fdr.cut	fdr.cut Threshold
logFC.cut	logFC.cut Threshold

corThreshold	corThreshold
kNearest	kNearest
nGenesPerm	nGenesPerm
DiffGenes	DiffGenes
nBoot	nBoot
nTF	nTF
nSample	nSample
thres.role	thres.role
stage	stage
subtype	subtype
samples	samples

**Value**

table with cancer driver genes TSG and OCG.

**Examples**

```
dataDEGs <- DPA(dataFile = dataFile, dataType = "Gene expression")
# to change with moonlight
```

---

MoonlightR

*MoonlightR*

---

**Description**

MoonlightR

---

plotCircos

*plotCircos*

---

**Description**

This function visualize the plotCircos

**Usage**

```
plotCircos(listMoonlight, listMutation = NULL,
  additionalFilename = NULL, intensityColOCG = 0.5,
  intensityColTSG = 0.5, intensityColDual = 0.5, fontSize = 1)
```

**Arguments**

listMoonlight	output Moonlight function
listMutation	listMutation
additionalFilename	additionalFilename
intensityColOCG	intensityColOCG
intensityColTSG	intensityColTSG
intensityColDual	intensityColDual
fontSize	fontSize

**Value**

no return value, plot is saved

**Examples**

```
plotCircos(listMoonlight = listMoonlight, additionalFilename = "_ncancer5")
```

---

plotFEA

*plotFEA*

---

**Description**

This function visualize the functional enrichment analysis (FEA)'s barplot

**Usage**

```
plotFEA(dataFEA, topBP = 10, additionalFilename = NULL, height, width,
  offsetValue = 5, angle = 90, xleg = 35, yleg = 5, titleMain,
  minY = -5, maxY = 10, mycols = c("#8DD3C7", "#FFFB3", "#BEBADA"))
```

**Arguments**

dataFEA	dataFEA
topBP	topBP
additionalFilename	additionalFilename
height	Figure height
width	Figure width
offsetValue	offsetValue
angle	angle



xleg	xleg
yleg	yleg
titleMain	title of the plot
minY	minY
maxY	maxY
mycols	colors to use for the plot

**Value**

no return value, FEA result is plotted

**Examples**

```
dataFEA <- FEA(DEGsmatrix = DEGsmatrix)
plotFEA(dataFEA = dataFEA, additionalFilename = "_example",height = 20,width = 10)
```

---

plotNetworkHive	<i>plotNetworkHive: Hive network plot</i>
-----------------	---

---

**Description**

This function visualizes the GRN as a hive plot

**Usage**

```
plotNetworkHive(dataGRN, namesGenes, thres, additionalFilename = NULL)
```

**Arguments**

dataGRN	output GRN function
namesGenes	list TSG and OCG to define axes
thres	threshold of edges to be included
additionalFilename	additionalFilename

**Value**

no results Hive plot is executed

**Examples**

```
data(knownDriverGenes)
data(dataGRN)
plotNetworkHive(dataGRN = dataGRN, namesGenes = knownDriverGenes, thres = 0.55)
```

---

plotURA *plotURA: Upstream regulatory analysis heatmap plot*

---

**Description**

This function visualizes the URA in a heatmap

**Usage**

```
plotURA(dataURA, additionalFilename = "URApot")
```

**Arguments**

dataURA	output URA function
additionalFilename	figure name

**Value**

heatmap

**Examples**

```
data(dataURA)
dataDual <- PRA(dataURA = dataURA,
BPname = c("apoptosis", "proliferation of cells"),
thres.role = 0)
TSGs_genes <- names(dataDual$TSG)
OCGs_genes <- names(dataDual$OCG)
plotURA(dataURA = dataURA[c(TSGs_genes, OCGs_genes),], additionalFilename = "_example")
```

---

PRA *Pattern Recognition Analysis (PRA)*

---

**Description**

This function carries out the pattern recognition analysis

**Usage**

```
PRA(dataURA, BPname, thres.role = 0)
```

**Arguments**

dataURA	output URA function
BPname	BPname
thres.role	thres.role

**Value**

returns list of TSGs and OCGs when biological processes are provided, otherwise a randomForest based classifier that can be used on new data

**Examples**

```
data(dataURA)
dataDual <- PRA(dataURA = dataURA,
BPname = c("apoptosis", "proliferation of cells"),
thres.role = 0)
```

---

tabGrowBlock	<i>Information growing/blocking characteristics for 101 selected biological processes</i>
--------------	---

---

**Description**

A data set containing the following data:

**Usage**

```
data(tabGrowBlock)
```

**Format**

A 101x3 matrix

**Details**

- tabGrowBlock matrix that defines if a process is growing or blocking cancer development, for each 101 biological processing

**Value**

a 101x3 matrix

**Description**

This function carries out the upstream regulator analysis

**Usage**

```
URA(dataGRN, DEGsmatrix, BPname, nCores = 1)
```

**Arguments**

dataGRN	output GNR function
DEGsmatrix	output DPA function
BPname	biological processes
nCores	number of cores to use

**Value**

an adjacent matrix

**Examples**

```
dataDEGs <- DEGsmatrix
dataGRN <- GRN(TFs = rownames(dataDEGs)[1:100],
  DEGsmatrix = dataDEGs,
  DiffGenes = TRUE,
  normCounts = dataFilt)
dataURA <- URA(dataGRN = dataGRN,
  DEGsmatrix = dataDEGs,
  BPname = c("apoptosis",
    "proliferation of cells"))
```

# Index

## \* datasets

- dataFilt, [3](#)
  - dataGRN, [4](#)
  - dataURA, [4](#)
  - DEGsmatrix, [5](#)
  - DiseaseList, [5](#)
  - EAGenes, [7](#)
  - GDCprojects, [8](#)
  - geneInfo, [8](#)
  - GEO\_TCGAtab, [9](#)
  - knownDriverGenes, [12](#)
  - listMoonlight, [13](#)
  - tabGrowBlock, [19](#)
- 
- dataFilt, [3](#)
  - dataGRN, [4](#)
  - dataURA, [4](#)
  - DEGsmatrix, [5](#)
  - DiseaseList, [5](#)
  - DPA, [6](#)
- 
- EAGenes, [7](#)
- 
- FEA, [7](#)
- 
- GDCprojects, [8](#)
  - geneInfo, [8](#)
  - GEO\_TCGAtab, [9](#)
  - getDataGEO, [9](#)
  - getDataTCGA, [10](#)
  - GRN, [11](#)
  - GSEA, [12](#)
- 
- knownDriverGenes, [12](#)
- 
- listMoonlight, [13](#)
  - LPA, [13](#)
- 
- moonlight, [14](#)
  - MoonlightR, [15](#)
  - MoonlightR-package (MoonlightR), [15](#)
- 
- plotCircos, [15](#)
  - plotFEA, [16](#)
  - plotNetworkHive, [17](#)
  - plotURA, [18](#)
  - PRA, [18](#)
- 
- tabGrowBlock, [19](#)
- 
- URA, [20](#)