PPIpriority

Daniele Pittari*[1]

[1]Polytechnic University of Milan, University of Milan

*danielepittari@gmail.com

11 gennaio 2023

Package

PPIpriority 0.1.0

Contents

1 Introduction

In the last twenty years, the technological advancement of high throughput approaches has revolutionized the field of bio medical research. With the greater availability of complex data information, the development of tools able to model data and extrapolate information has become of great interest.

In this context, graph modelling and network analysis have become popular approaches for modelling molecular interactions. Gene co-expression networks and protein-protein interaction (PPI) networks represent valuable models for performing target prioritization of genes/proteins of interest.

In this package, we present a tool which utilizes the STRINGdb PPI data sets and allows for target prioritization based on a random walk in the PPI from genes of interest (seeds), known for being associated with the phenotype/biological process of interest (seeds).

2 Methods

Methodological studies benchmarking the performance of different strategies for disease gene prioritization in PPI networks have demonstrated that approaches utilizing random walking with restart showed good performance in the different conditions tested (Köhler et al. 2008).

Our tool allows to conveniently combine the retrieval of PPI data from the STRINGdb (via the homonym package) and conveniently perfom target prioritization using a random walking (RW) with restart from the RANKS package.

The output of our package consists of a data.frame containing a ranking of the genes based on the RW.

3 Workflow

The workflow of PPIpriority is a two-step process

1. Retrieve the PPI network of interest from the STRINGdb database, utilizing the function "build_ppi_network" and creating a "PPIpriority_obj" which will store your network as well as the STRINGdb object. This allows the user to continue data analysis on the downloaded PPI via the STRINGdb package.
2. Perform target prioritization on the network stored in the PPIpriority_obj via using the function "prioritize_targets" with at least gene as seed for the random walk.

4 Showcase

4.0.1 Prioritizing putative targets for a protein of interest: the case of ERp44

The first example shows a case of target prioritization on a human PPI.

Given our protein of interest ERp44, we want to prioritize putative functional targets based on PPI.

ERP44 is a protein chaperone that cycles from the Endoplasmic Reticulum (ER) to the cis-Golgi and vice-versa. The role of this protein is to selectively retrieve to the ER those proteins that have erroneously left the it through vesicle transport. Among the ERP44 binding partners we have ER-localized enzymes (e.g. PRDX4, ERAP) or subunits of oligomeric protein complexes (e.g. IgMs).

First step: we retrieve the version "11.5" of the STRINGdb PPI for the human species (9606) and set a threshold of interaction confidence of 600.

To do this, simply use "build_ppi_network" with appropriate inputs for STRINGdb:

```
library(PPIpriority)
#Set  extended timeout, not necessary while using the package
options(timeout=700)

network_ppi_obj<- build_ppi_network(version ="11.5", species = 9606,
score_threshold=700, input_directory=getwd())
#> ***********  STRING - https://string-db.org   ***********
#> (Search Tool for the Retrieval of Interacting Genes/Proteins)
#> version: 11.5
#> species: 9606
#> ............please wait............
#> proteins: 19566
#> interactions: 505968
```

Now we run target prioritization: to do this, use "prioritize_targets" on your PPIpriority object. The function will require one or more seed genes that will be used to initialize the random walk. In our case, ERp44 will be our seed. In addition, if you want to analyze a list of candidate genes to prioritize, you can insert them as "candidates". The function will pre-filter the results just based on the genes of interests.

In this scenario, since we are interested in all of the possible results, we will not provide candidate genes among "prioritize_targets" inputs. We are just passing the HUGO gene name of ERP44, which is ERP44 indeed.

```
output_df<-prioritize_targets(network_ppi_obj, seeds = "ERP44")

head(output_df,15) # show the df
#>               STRING_id preferred_id rank     prob_RW
#> 1  9606.ENSP00000379042        ERO1L  1.0 0.028324318
#> 2  9606.ENSP00000338927      ZNF385A  2.0 0.027325057
#> 3  9606.ENSP00000314407          CA8  3.0 0.024110345
#> 4  9606.ENSP00000389814       ADIPOQ  4.0 0.024052645
#> 5  9606.ENSP00000368646        PRDX4  5.0 0.024021980
#> 6  9606.ENSP00000346635       ERO1LB  6.0 0.024000013
#> 7  9606.ENSP00000272902        SUMF1  8.5 0.024000000
#> 8  9606.ENSP00000296754        ERAP1  8.5 0.024000000
#> 9  9606.ENSP00000301522        PRDX2  8.5 0.024000000
#> 10 9606.ENSP00000306253        ITPR1  8.5 0.024000000
#> 11 9606.ENSP00000385385        PDIA6 11.0 0.006606143
#> 12 9606.ENSP00000340989          SFN 12.0 0.005465833
#> 13 9606.ENSP00000417196        PCBP4 13.0 0.005465070
#> 14 9606.ENSP00000380432          INS 14.0 0.005370837
#> 15 9606.ENSP00000369081       TXNDC5 15.0 0.004412693
```

ERO1L/ERO1LB, PRDX4, SUMF1, ERAP1 are among the top candidates. All these proteins have been experimentally demonstrated to physically interact with ERp44 (Anelli et al. 2002, Fraldi et al. 2008, Hisatsune et al. 2015, Yang et al. 2016).

As for what concerns the output data.frame: the tool returns the preferred id for your input genes according to STRINGdb.

In these examples we are using HUGO symbols as inputs for seeds (or candidates, if any). We suggest to use recommended gene symbol formats. You can also utilize other formats (e.g. NCBI Entrez Gene), PPIpriority will convert them into STRINGdb IDs. To achieve better conversion of genes into STRINGdb IDs, the mapping step is performed by calling the map method of STRINGdb. Check the STRINGdb package for more information.

4.0.2 Performing the same analysis NCBI Entrez Gene for ERP44

```
output_df_erp44_entrez<-prioritize_targets(network_ppi_obj, seeds = "23071")

# is the output equal to out previous dataset obtained with HUGO symbols?
all.equal(output_df_erp44_entrez,output_df)
#> [1] TRUE
```

4.0.3 Prioritizing a list of multiple candidates, given multiple known genes: the case of the functional partners ERp44, LMAN1 and MCFD2

This case concerns the prioritization of candidate targets given multiple seeds.

In plasma cells, ERP44, LMAN1 and MCFD2 have been proposed to create functional structures in the ER that may facilitate the assembly of complex oligomeric proteins.

Given that we know that coagulation factor V (F5) is a binder of this complex, we run the analysis with other secreted proteins to prioritize to evalutate if the algorithm scores F5 among the top ranked gene.

```
seeds_genes<-c("ERP44","LMAN1", "MCFD2")
candidate_genes<-c("AREG", "IL1A", "IL1B", "MMP3", "PRL", "F5")

output_df_msmc<-prioritize_targets(network_ppi_obj,
                                   seeds = seeds_genes,
                                   candidates=candidate_genes)

# show the df
output_df_msmc
#>              STRING_id preferred_id  rank      prob_RW
#> 1 9606.ENSP00000356771           F5    75 1.387952e-03
#> 2 9606.ENSP00000379097         AREG  1061 8.063097e-06
#> 3 9606.ENSP00000302150          PRL  2585 7.579086e-07
#> 4 9606.ENSP00000299855         MMP3  7884 8.444671e-11
#> 5 9606.ENSP00000263341         IL1B  9988 7.839316e-16
#> 6 9606.ENSP00000263339         IL1A 10376 7.458867e-19
```

The analysis correctly ranks coagulation factor V (F5) at the top in the ranks.

5 Beyond PPIpriority: continuing your analysis with STRINGdb or other tools for network analysis

The PPIpriority package allows for extrapolating the network modeled during the "build_ppi_network" step as well as the STRINGdb object. This can be performed by using the methods "get_network" and "get_STRINGdb_connection" methods respectively. These methods are accessors to the PPIpriority obj S4 class slots "graph" and "STRINGdb_connection".

5.0.1 Retrieve the network from PPIpriority obj:

```
get_network(network_ppi_obj)
#> IGRAPH 4a84646 UNW- 16814 252984 --
#> + attr: name (v/c), symbol (v/c), weight (e/n)
#> + edges from 4a84646 (vertex names):
#>  [1] 9606.ENSP00000000233--9606.ENSP00000324287
#>  [2] 9606.ENSP00000000233--9606.ENSP00000387286
#>  [3] 9606.ENSP00000000233--9606.ENSP00000262812
#>  [4] 9606.ENSP00000000233--9606.ENSP00000158762
#>  [5] 9606.ENSP00000000233--9606.ENSP00000449270
#>  [6] 9606.ENSP00000000233--9606.ENSP00000480707
#>  [7] 9606.ENSP00000000233--9606.ENSP00000263245
#>  [8] 9606.ENSP00000000233--9606.ENSP00000440005
#> + ... omitted several edges
```

5.0.2 Retrieve the STRINGdb obj from PPIpriority obj:

```
get_STRINGdb_connection(network_ppi_obj)
#> ***********  STRING - https://string-db.org   ***********
#> (Search Tool for the Retrieval of Interacting Genes/Proteins)
#> version: 11.5
#> species: 9606
#> ............please wait............
#> proteins: 19566
#> interactions: 505968
```

6 References

Köhler S, Bauer S, Horn D, Robinson PN. Walking the interactome for prioritization of candidate disease genes. Am J Hum Genet. 2008 Apr;82(4):949-58. doi: 10.1016/j.ajhg.2008.02.013. Epub 2008 Mar 27. PMID: 18371930; PMCID: PMC2427257.

Anelli T, Alessio M, Mezghrani A, Simmen T, Talamo F, Bachi A, Sitia R. ERp44, a novel endoplasmic reticulum folding assistant of the thioredoxin family. EMBO J. 2002 Feb 15;21(4):835-44. doi: 10.1093/emboj/21.4.835. PMID: 11847130; PMCID: PMC125352.

Fraldi A, Zito E, Annunziata F, Lombardi A, Cozzolino M, Monti M, Spampanato C, Ballabio A, Pucci P, Sitia R, Cosma MP. Multistep, sequential control of the trafficking and function of the multiple sulfatase deficiency gene product, SUMF1 by PDI, ERGIC-53 and ERp44. Hum Mol Genet. 2008 Sep 1;17(17):2610-21. doi: 10.1093/hmg/ddn161. Epub 2008 May 28. PMID: 18508857.

Hisatsune C, Ebisui E, Usui M, Ogawa N, Suzuki A, Mataga N, Takahashi-Iwanaga H, Mikoshiba K. ERp44 Exerts Redox-Dependent Control of Blood Pressure at the ER. Mol Cell. 2015 Jun 18;58(6):1015-27. doi: 10.1016/j.molcel.2015.04.008. Epub 2015 May 7. PMID: 25959394.

Yang K, Li DF, Wang X, Liang J, Sitia R, Wang CC, Wang X. Crystal Structure of the ERp44-Peroxiredoxin 4 Complex Reveals the Molecular Mechanisms of Thiol-Mediated Protein Retention. Structure. 2016 Oct 4;24(10):1755-1765. doi: 10.1016/j.str.2016.08.002. Epub 2016 Sep 15. PMID: 27642162.

7 Session information

```
#> R version 4.2.2 Patched (2022-11-10 r83330)
#> Platform: x86_64-pc-linux-gnu (64-bit)
#> Running under: Ubuntu 22.04.1 LTS
#>
#> Matrix products: default
#> BLAS:   /usr/lib/x86_64-linux-gnu/openblas-pthread/libblas.so.3
#> LAPACK: /usr/lib/x86_64-linux-gnu/openblas-pthread/libopenblasp-r0.3.20.so
#>
#> locale:
#>  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
#>  [3] LC_TIME=it_IT.UTF-8        LC_COLLATE=C
#>  [5] LC_MONETARY=it_IT.UTF-8    LC_MESSAGES=en_US.UTF-8
#>  [7] LC_PAPER=it_IT.UTF-8       LC_NAME=C
#>  [9] LC_ADDRESS=C               LC_TELEPHONE=C
#> [11] LC_MEASUREMENT=it_IT.UTF-8 LC_IDENTIFICATION=C
#>
#> attached base packages:
#> [1] stats     graphics  grDevices utils     datasets  methods   base
#>
#> other attached packages:
#> [1] PPIpriority_0.1.0 knitr_1.41        BiocStyle_2.24.0
#>
#> loaded via a namespace (and not attached):
#>  [1] gtools_3.9.4        tidyselect_1.2.0    xfun_0.35
#>  [4] generics_0.1.3      vctrs_0.5.1         htmltools_0.5.4
#>  [7] stats4_4.2.2        hash_2.2.6.2        yaml_2.3.6
#> [10] utf8_1.2.2          chron_2.3-58        blob_1.2.3
#> [13] RBGL_1.72.0         rlang_1.0.6         pillar_1.8.1
#> [16] NetPreProc_1.2      withr_2.5.0         glue_1.6.2
#> [19] DBI_1.1.3           BiocGenerics_0.42.0 bit64_4.0.5
#> [22] RColorBrewer_1.1-3  gsubfn_0.7          lifecycle_1.0.3
#> [25] plyr_1.8.8          stringr_1.5.0       STRINGdb_2.8.4
#> [28] caTools_1.18.2      evaluate_0.19       memoise_2.0.1
```

```
#> [31] fastmap_1.1.0       fansi_1.0.3         proto_1.0.0
#> [34] Rcpp_1.0.9          KernSmooth_2.23-20  RANKS_1.1
#> [37] BiocManager_1.30.19 cachem_1.0.6        limma_3.52.4
#> [40] plotrix_3.8-2       graph_1.74.0        bit_4.0.5
#> [43] gplots_3.1.3        png_0.1-8           digest_0.6.30
#> [46] stringi_1.7.8       bookdown_0.31       dplyr_1.0.10
#> [49] PerfMeas_1.2.5      cli_3.4.1           tools_4.2.2
#> [52] bitops_1.0-7        magrittr_2.0.3      sqldf_0.4-11
#> [55] tibble_3.1.8        RCurl_1.98-1.9      RSQLite_2.2.19
#> [58] pkgconfig_2.0.3     assertthat_0.2.1    rmarkdown_2.19
#> [61] rstudioapi_0.14     R6_2.5.1            igraph_1.3.5
#> [64] compiler_4.2.2
```