COSMOS-tutorial

A. Dugourd, A. Gabor and K. Zirngibl

11/10/2020

Introduction

COSMOS (Causal Oriented Search of Multi-Omic Space) is a method that integrates phosphoproteomics, transcriptomics, and metabolomics data sets. COSMOS leverages extensive prior knowledge of signaling pathways, metabolic networks, and gene regulation with computational methods to estimate activities of transcription factors and kinases as well as network-level causal reasoning. This pipeline can provide mechanistic explanations for experimental observations across multiple omic data sets.

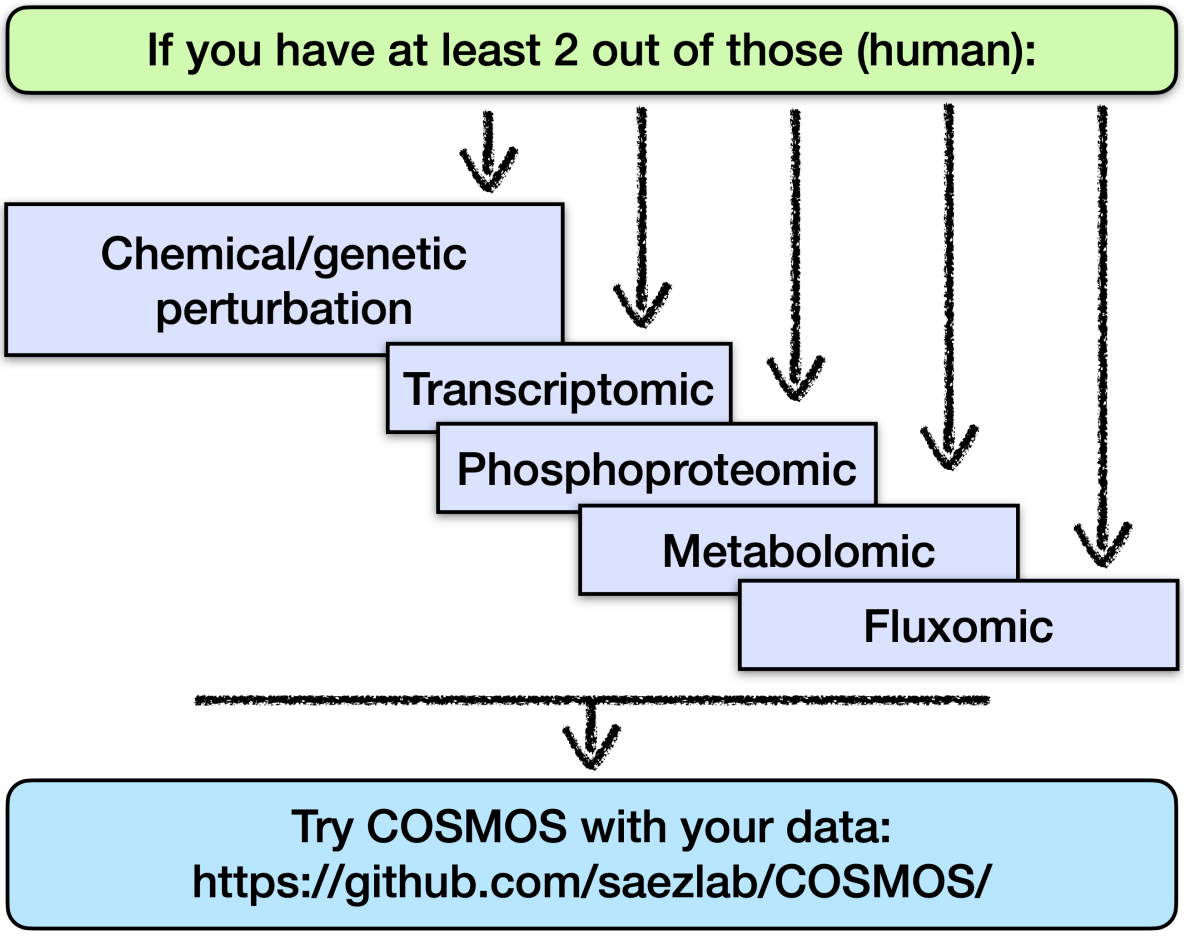

Figure 1: data_intro_figure

First, we load the package

```
library(cosmos)
```

Tutorial section: signaling to metabolism

In this part, we can set up the options for the CARNIVAL run, such as timelimit and min gap tolerance.

The user should provide a path to its CPLEX/cbc executable

You can check the CARNIVAL_options variable to see all possible options that can be adjusted

```
CARNIVAL_options <- cosmos::default_CARNIVAL_options()
CARNIVAL_options$solverPath <- "~/Documents/cplex"
CARNIVAL_options$solver <- "cplex" #or cbc
CARNIVAL_options$timelimit <- 3600
CARNIVAL_options$mipGAP <- 0.05
CARNIVAL_options$threads <- 2
```

In the next section, we prepare the input to run cosmos The signaling inputs are the result of footprint based TF and kinase activity estiamtion For more info on TF activity estiamtion from transcriptomic data, see:https://github.com/saezlab/transcriptutorial (Especially chapter 4)

Here we use of toy PKN, to see the full meta PKN, you can load it with load_meta_pkn()

The metabolites in the prior knowledge network are identified as XMetab__*PUBCHEMid***compartment** or XMetab__*BIGGid***compartment** or example "XMetab__*6804***m**". The compartment code is the BIGG model standard (r, c, e, x, m, l, n, g). Thus we will first need to map whatever identifer for metabolite the data has to the one of the network. Genes are identified as XENTREZid (in the signaling part of network) or XGene####___ENTREZid (in the reaction network part of network)

The maximum network depth will define the maximum number of step downstream of kinase/TF COSMOS will look for deregulated metabolites. Good first guess for max depth could be around 6 (here is 15 for the toy dataset)

The differential experession data is used to filter out wrong TF-target interactions in this context after a pre-optimisation.

The list of genes in the differential expression data will also be used as a reference to define which genes are expressed or not (all genes in the diff_expression_data are considered expressed, and genes that are no in diff_expression_data are removed from the network)

```
test_for <- preprocess_COSMOS_signaling_to_metabolism(meta_network = toy_sif,
                                        signaling_data = toy_signaling_input_carnival_vec,
                                        metabolic_data = toy_metab_input_carnival_vec,
                                                    diff_expression_data = toy_RNA,
                                                    maximum_network_depth = 15,
                                                    remove_unexpressed_nodes = T,
                                                    CARNIVAL_options = CARNIVAL_options
                                                    )
```

```
## [1] "COSMOS: all 15 signaling nodes from data were found in the meta PKN"
## [1] "COSMOS: all 10 metabolic nodes from data were found in the meta PKN"
## [1] "COSMOS: 259 of the 385 genes in expression data were found as transcription factor target"
## [1] "COSMOS: 259 of the 5318 transcription factor targets were found in expression data"
## [1] "COSMOS: removing unexpressed nodes from PKN..."
## [1] "COSMOS: 24 interactions removed"
## [1] "COSMOS: 4 input/measured nodes are not in PKN any more: XMetab__124886___c____, XMetab__107738_
```

```
## [1] "COSMOS: removing nodes that are not reachable from inputs within 15 steps"
## [1] "COSMOS: 287 from  671 interactions are removed from the PKN"
## [1] "COSMOS: removing nodes that are not observable by measurements within 15 steps"
## [1] "COSMOS: 282 from  384 interactions are removed from the PKN"
## [1] "COSMOS: 11 input/measured nodes are not in PKN any more: X4150, X6688, X3164, X2305, X10379, X3
## [1] "COSMOS:  3 interactions are removed from the PKN based on consistency check between TF activity
## [1] "COSMOS:  2 interactions are removed from the PKN based on consistency check between TF activity
## [1] "COSMOS: all 4 signaling nodes from data were found in the meta PKN"
## [1] "COSMOS: all 6 metabolic nodes from data were found in the meta PKN"
## [1] "COSMOS: 259 of the 385 genes in expression data were found as transcription factor target"
## [1] "COSMOS: 259 of the 5318 transcription factor targets were found in expression data"
```

In this part, we can set up the options for the actual run, such as timelimit and min gap tolerance.

The running time should be much higher here than in pre-optimisation. You cna increase the number of threads to use if you have many available CPUs.

```
CARNIVAL_options$timelimit <- 14400
CARNIVAL_options$mipGAP <- 0.05
CARNIVAL_options$threads <- 2
```

This is where cosmos run.

```
test_result_for <- run_COSMOS_signaling_to_metabolism(data = test_for,
                                                      CARNIVAL_options = CARNIVAL_options)
```

Finally, we process the results of the first cosmos run, to translate gene names and metabolites name.

```
metab_to_pubchem_vec <- metab_to_pubchem$name
names(metab_to_pubchem_vec) <- metab_to_pubchem$pubchem

test_result_for <- format_COSMOS_res(test_result_for,
                                     metab_mapping = metab_to_pubchem_vec,
                       measured_nodes = unique(c(names(toy_metab_input_carnival_vec),
                                                 names(toy_signaling_input_carnival_vec))),
                                     omnipath_ptm = omnipath_ptm)
```

Tutorial section: metabolism to signaling

Before we run the metabolism to signaling part, we need to prepare again the inputs.

```
CARNIVAL_options$timelimit <- 3600
CARNIVAL_options$mipGAP <- 0.05
CARNIVAL_options$threads <- 2
```

Now that the correct time is set up for the pre-optimisation run, we can prepare the inputs.

```
test_back <- preprocess_COSMOS_metabolism_to_signaling(meta_network = toy_sif,
                                       signaling_data = toy_signaling_input_carnival_vec,
                                       metabolic_data = toy_metab_input_carnival_vec,
                                                    diff_expression_data = toy_RNA,
                                                    maximum_network_depth = 15,
                                                    remove_unexpressed_nodes = F,
                                                    CARNIVAL_options = CARNIVAL_options

)
```

```
## [1] "COSMOS: all 15 signaling nodes from data were found in the meta PKN"
## [1] "COSMOS: all 10 metabolic nodes from data were found in the meta PKN"
## [1] "COSMOS: 259 of the 385 genes in expression data were found as transcription factor target"
## [1] "COSMOS: 259 of the 5318 transcription factor targets were found in expression data"
## [1] "COSMOS: removing nodes that are not reachable from inputs within 15 steps"
## [1] "COSMOS: 184 from  720 interactions are removed from the PKN"
## [1] "COSMOS: removing nodes that are not observable by measurements within 15 steps"
## [1] "COSMOS: 343 from  536 interactions are removed from the PKN"
## [1] "COSMOS: 8 input/measured nodes are not in PKN any more: XMetab__6029___l____, XMetab__124886___
## [1] "COSMOS:  3 interactions are removed from the PKN based on consistency check between TF activity
## [1] "COSMOS:  0 interactions are removed from the PKN based on consistency check between TF activity
## [1] "COSMOS: all 15 signaling nodes from data were found in the meta PKN"
## [1] "COSMOS: all 2 metabolic nodes from data were found in the meta PKN"
## [1] "COSMOS: 259 of the 385 genes in expression data were found as transcription factor target"
## [1] "COSMOS: 259 of the 5318 transcription factor targets were found in expression data"
```

Then we can run cosmos to connect metabolism to signaling. The running time here usually needs to be longer, as this problem seems to be harder to solve for CPLEX.

Finally we can format the result of the backward run as well (same as for forward run)

```
test_result_back <- format_COSMOS_res(test_result_back,
                                      metab_mapping = metab_to_pubchem_vec,
                        measured_nodes = unique(c(names(toy_metab_input_carnival_vec),
                                                  names(toy_signaling_input_carnival_vec))),
                                      omnipath_ptm = omnipath_ptm)
```

Tutorial section: Merge forward and backward networks and visualise network

Here we simply take the union of forward and backward runs to create a full network solution lopping between signaling, gene-regulation and metabolism. Since there is an overlapp between the result network of forward and backward run, you may optionally want to check if there are any node sign that are incoherent in the overlapp between the two solutions.

```
full_sif <- as.data.frame(rbind(test_result_for[[1]], test_result_back[[1]]))
full_attributes <- as.data.frame(rbind(test_result_for[[2]], test_result_back[[2]]))

full_sif <- unique(full_sif)
full_attributes <- unique(full_attributes)
```

This function will generate a dynamic network plot centered on a given node of the network solution, and connecting it to measured nodes in the given range (here 5 steps).

```
network_plot <- display_node_neighboorhood(central_node = 'PRKACA',
                                           sif = full_sif,
                                           att = full_attributes,
                                           n = 5)

# network_plot
```

```
sessionInfo()
```

```
## R version 4.0.2 (2020-06-22)
## Platform: x86_64-apple-darwin17.0 (64-bit)
## Running under: macOS  10.16
##
```

```
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.0/Resources/lib/libRblas.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.0/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] parallel  stats4    stats     graphics  grDevices utils     datasets
## [8] methods   base
##
## other attached packages:
## [1] org.Hs.eg.db_3.11.4  AnnotationDbi_1.52.0 IRanges_2.24.1
## [4] S4Vectors_0.28.1     Biobase_2.50.0       BiocGenerics_0.36.0
## [7] cosmos_0.1.5
##
## loaded via a namespace (and not attached):
##  [1] mixtools_1.2.0       httr_1.4.2           UniProt.ws_2.30.0
##  [4] jsonlite_1.7.2       bit64_4.0.5          splines_4.0.2
##  [7] foreach_1.5.1        assertthat_0.2.1     BiocFileCache_1.14.0
## [10] RBGL_1.66.0          blob_1.2.1           yaml_2.2.1
## [13] Category_2.56.0      viper_1.24.0         pillar_1.5.1
## [16] RSQLite_2.2.5        lattice_0.20-41      glue_1.4.2
## [19] digest_0.6.27        colorspace_2.0-0     htmltools_0.5.1.1
## [22] Matrix_1.2-18        GSEABase_1.52.1      lpSolve_5.6.15
## [25] XML_3.99-0.5         pkgconfig_2.0.3      genefilter_1.72.1
## [28] CARNIVAL_1.1.0       purrr_0.3.4          xtable_1.8-4
## [31] scales_1.1.1         tibble_3.1.0         proxy_0.4-25
## [34] annotate_1.68.0      generics_0.1.0       ggplot2_3.3.2
## [37] ellipsis_0.3.1       cachem_1.0.4         survival_3.2-3
## [40] magrittr_2.0.1       crayon_1.4.1         memoise_2.0.0
## [43] evaluate_0.14        fansi_0.4.2          doParallel_1.0.16
## [46] MASS_7.3-51.6        segmented_1.3-3      class_7.3-17
## [49] graph_1.68.0         tools_4.0.2          hms_1.0.0
## [52] lifecycle_1.0.0      stringr_1.4.0        bcellViper_1.26.0
## [55] dorothea_1.3.0       kernlab_0.9-29       munsell_0.5.0
## [58] compiler_4.0.2       e1071_1.7-6          rlang_0.4.10
## [61] RCurl_1.98-1.2       grid_4.0.2           iterators_1.0.13
## [64] htmlwidgets_1.5.1    visNetwork_2.0.9     rappdirs_0.3.3
## [67] igraph_1.2.6         bitops_1.0-6         rmarkdown_2.7
## [70] gtable_0.3.0         codetools_0.2-16     curl_4.3
## [73] DBI_1.1.0            R6_2.5.0             knitr_1.31
## [76] dplyr_1.0.5          fastmap_1.1.0        bit_4.0.4
## [79] utf8_1.2.1           KernSmooth_2.23-17   readr_1.4.0
## [82] stringi_1.5.3        Rcpp_1.0.6           vctrs_0.3.7
## [85] dbplyr_2.1.1         tidyselect_1.1.0     xfun_0.22
```