Stoichiometric correlation analysis (SCA)
User's manual

Kevin Schwahn

25.04.2017

This manuscript summarizes the analysis performed in the publication "Stoichiometric correlation analysis: principles of metabolic functionality from metabolomics data." The pipeline has been primarily designed for metabolite data.

The steps to perform SCA are summarized as follows. After loading all data and functions, all possible triplets and quadruples of the metabolites in the data set as well as their respective stoichiometric correlations can be calculated using the function *ks_stoichiometric_correlation*. The resulting stoichiometric correlation coefficients can be used to find the maximal correlation for each triplet and quadruple using the functions *ks_find_max_cor_tr* and *ks_find_max_cor_qu*. Additionally, the stoichiometric correlation of all pairs can be estimated using the functions *ks_pairwise_cor* and *ks_find_max_cor*.

Clearly, the number of triplets and quadruples are tremendously growing by increasing the number of metabolites. Therefore, using large number of metabolite in SCA (more than 30) cause memory shortage in R. To cope with the memory limitation in R, the functions *ks_stoichiometric_correlation*, *ks_find_max_cor_tr* and *ks_find_max_cor_qu* were implemented. These functions create temporary files in the directory in which they are called. Please keep this in mind that during the analysis large number of files can be created; however, the advantage is that these files can be used to continue the analysis in case of any interruptions such as unexpected errors. In this example, we provided an automated way to create a temporary folder, in which all temporary files are stored.

Finally, the functions *ks_make_table*, *ks_make_bipartite_graph* and *ks_shared_metabolites* allow investigating and producing appropriate output.

All functions and scripts were tested on Linux (Ubuntu - 14.04.5 LTS) and Windows 10 operating system. Additionally, Python needs to be installed on your system. The Python scripts were tested with Python 2.7 and Python 3.6. Please keep in mind that parallelization is not supported on Windows operating systems.

In the following sections, the procedure to perform SCA is described in more details with an example. In addition, the folder **Example_Data+Script** includes the R code in which SCA is performed on *A. thanliana* and *E. coli* data sets which were presented in the research paper.

Preparation

Before starting the analysis, a few preparation steps are needed. These contain:

- Loading functions
- Creating temporary folder
- Loading data

Additionally, ensure that Python is installed on your system and within your systems PATH-variable.

Loading functions

The folder "Functions" contains all R-functions needed for the SC analysis. The following code snippet loads all functions into the workspace:

```
file.sources = list.files(path = "Functions",pattern="/*.R",full.names = T)
sapply(file.sources,source,.GlobalEnv)
```

The following packages are needed to be installed in R before starting the analysis:

- Hmisc
- igraph
- parallel

Having the following packages installed on your system, the output can be written into the Excel formatted files. In case these packages are not available on your system, .csv files are generated.

- XLConnect
- rJava

Creating temporary folder

As mentioned, performing all calculations within a single R-workspace can lead to memory shortage. To address this issue, a temporary folder should be created to store the temporary files created within the analysis. The temporary folder and files can be deleted when all calculations are done.

The following code snippet produces the temporary folder.

```
subDir = "data_run/"

if (file.exists(subDir)){
  setwd(subDir)
} else {
  dir.create(subDir)
  setwd(subDir)
}
```

Loading data

The metabolomics data should be organized in the following way:

- Metabolites in rows
- Conditions/time points in columns

```
data = read.table("Example_Data+Script/Ara_data.txt",sep="\t",header = T)
dim(data)
```

```
## [1]  19 913
```

```
data[1:5,1:5]
```

```
##                X0.1 X0.2 X0.3 X0.4 X0.5
## Succinic acid 1.01 0.79 1.11 0.93 1.23
## Fumaric acid  0.86 1.03 0.77 0.99 0.74
## Malic acid    0.65 1.19 0.69 0.95 0.88
## Alanine       0.87 0.92 0.95 0.74 0.76
## Valine        0.85 0.89 0.84 0.77 0.87
```

Stoichiometric Correlation Analysis

After the preparation step, the stoichiometric correlation coefficients can be calculated. This can be done using the function *ks_stoichiometric_correlation*. This function creates temporary files in the temporary folder. The argument *indicies* defines the coefficients that the metabolite data are going to be multiplied by. *nblocks* should be set to the number of metabolites. To decrease the running time, the stoichiometric correlation analysis can be done in a parallel manner. To this end, *NRcluster* should be set to a value higher than one, which shows the number of cores to be used.

```
data_Cor = ks_stoichiometric_correlation(
              Data=data,indicies=c(1:4),
              nblocks=nrow(data),NRcluster=1,names=rownames(data))
```

After calculating stoichiometric correlation coefficients, the maximal stoichiometric correlation for triplets and quadruples have to be estimated. The input arguments for functions *ks_find_max_cor_tr* and *ks_find_max_cor_qu* are similar to the ones for the function *ks_stoichiometric_correlation*. The input argument *triplets* (*quadruples*) of the function *ks_find_max_cor_tr* (*ks_find_max_cor_qu*) is set to *data_Cor$triplets* (*data_Cor$quadruples*) resulting from the function *ks_stoichiometric_correlation*. The argument *indicies* has to be set to the same value (coefficients set) as in the function *ks_stoichiometric_correlation*. Finding maximal stoichiometric correlation can also be done in parallel manner. The number of cores needed for this analysis is obtained by the product of *NRcluster1* and *NRcluster2* values. Please have in mind that the parallelization is not supported on Windows operating systems. In the function *ks_find_max_cor_qu* the argument *tr* defines the threshold for stoichiometric correlation coefficients. If this value is set above 0, fewer number of quadruples will be tested for the maximal correlation. The default value of *tr* is 0.8.

```
ks_find_max_cor_tr(triplets=data_Cor$triplets,indicies=c(1:4),NRcluster1=1,NRcluster2=1)

ks_find_max_cor_qu(quadruples=data_Cor$quadruples,indicies=c(1:4),NRcluster1=1,tr=0.8)
```

After calculating the maximal stoichiometric correlations, the Python scripts have to be called. The Python scripts efficiently create all maximal triplets and quadruples in separate files. This is especially needed if more than 30 metabolites are investigated.

The following code chunk summarizes all temporary files into two files: **data_triplets.txt** and **data_quadruples.txt**.

```
command = "python"
path2trip='"../../Functions/File_read_triples.py"'
path2quad='"../../Functions/File_read_quadruples.py"'

# Add path to script as first arg
allArgs = c(path2trip,'"data_triplets.txt"')
output = system2(command, args=allArgs, stdout=TRUE)
allArgs = c(path2quad,'"data_quadruples.txt"')
output = system2(command, args=allArgs, stdout=TRUE)
```

The files with the maximal correlation have to be loaded into an R session.

```
data_tr = read.table("data_triplets.txt",header=T,sep = "\t")
head(data_tr)
```

```
##                                  names correlations      p_value
## 1       1*Alanine_2*Arginine->Leucine    0.5173432 0.000000e+00
## 2 1*Alanine_2*Arginine->Phenylalanine    0.7008961 0.000000e+00
## 3        1*Alanine_2*Arginine->Valine    0.7021881 0.000000e+00
## 4 1*Alanine_2*Asparagine ->Isoleucine    0.6889541 0.000000e+00
## 5   1*Alanine_2*Asparagine ->Tyrosine    0.6302674 0.000000e+00
## 6    1*Alanine_2*Glutamine->Aspartate    0.1943290 3.213449e-09
##   adjust_p_value
## 1   0.000000e+00
## 2   0.000000e+00
## 3   0.000000e+00
## 4   0.000000e+00
## 5   0.000000e+00
## 6   4.620428e-09
```

```
data_qu = read.table("data_quadruples.txt",header=T,sep="\t")
head(data_qu)
```

```
##                                                names correlations p_value
## 1           1*Alanine_3*Lysine->1*Valine_2*Tyrosine    0.9654425       0
## 2  1*Alanine_3*Phenylalanine->1*Malic acid_3*Valine    0.8579858       0
## 3   1*Alanine_4*Isoleucine->1*beta-alanine_3*Lysine    0.9449176       0
## 4  1*Alanine_4*Isoleucine->1*Fumaric acid_4*Leucine    0.9681512       0
## 5   1*Alanine_4*Isoleucine->1*Fumaric acid_4*Lysine    0.9367551       0
## 6 1*Alanine_4*Isoleucine->1*Fumaric acid_4*Tyrosine    0.9437932       0
##   adjust_p_value
## 1              0
## 2              0
## 3              0
## 4              0
## 5              0
## 6              0
```

The data frames *data_tr* and *data_qu* should then be combined into a list as follows:

```
data_max_cor = list(triplets=data_tr,quadruples=data_qu)
```

The pairwise correlation coefficients between metabolites can be calculated using the function *ks_pairwise_cor*. The argument *log* should be set to *TRUE* for calculating the pairwise stoichiometric correlation coefficients. If the function is called with *log=F*, the standard Pearson correlation coefficients are calculated. The function *ks_find_max_cor* uses the output from the function *ks_pairwise_cor* to find the maximal stoichiometric correlations for pairwise correlations.

```
data_pair_log=ks_pairwise_cor(Data=data,log=T)
data_pair_log_max=ks_find_max_cor(Data=data_pair_log)
```

Output

The function *ks_make_table* creates the output in a data frame structure. The arguments of this function are *pair*, the pairwise correlations (output of the function *ks_find_max_cor*), as well as *Corr*, the list including triplets and quadruples (output of the Python script). Additionally, *Names*, the name of the metabolites and *tr*, the threshold for the correlation coefficients are used as an input arguments of this function. Only pairs, triplets and quadruples with correlation coefficients above the threshold, *tr*, will be considered. The output of this function is a *data.frame*, which can be directly written into a file. The *data.frame* contains the following information per metabolite:

- *Total_number_of_correlations*: number of total stoichiometric correlations
- *Triplet_number_correlation*: number of stoichiometric correlations due to triplets
- *Quadruple_number_correlation*: number of stoichiometric correlations due to quadruples

- *Pairs_number_correlation*: number of stoichiometric correlations due to pairs
- *Triplet_mean_correlation*: mean of stoichiometric correlations due to triplets
- *Quadruple_mean_correlation*: mean of stoichiometric correlations due to quadruples
- *Pairs_mean_correlation*: mean of stoichiometric correlations due to pairs
- *Triplet_max_correlation*: maximum stoichiometric correlation due to triplets
- *Quadruple_max_correlation*: maximum stoichiometric correlation due to quadruples
- *Pairs_max_correlation*: maximum stoichiometric correlation due to pairs
- *Triplet_min_correlation*: minimum stoichiometric correlation due to triplets
- *Quadruple_min_correlation*: minimum stoichiometric correlation due to quadruples
- *Pairs_min_correlation*: minimum stoichiometric correlation due to pairs
- *Stoichiometric_mean*: mean indices of the maximal correlations
- *Stoichiometric_max*: maximum indices of the maximal correlations

```
write.table(
  ks_make_table(pair=data_pair_log_max,Corr=data_max_cor,Names=rownames(data),tr=0.8),
  file = "data_complete_table_08.tab",row.names=F,col.names=T,quote = F,sep="\t")

write.table(
  ks_make_table(pair=data_pair_log_max,Corr=data_max_cor,Names=rownames(data),tr=0.85),
  file = "data_complete_table_085.tab",row.names=F,col.names=T,quote = F,sep="\t")

write.table(
  ks_make_table(pair=data_pair_log_max,Corr=data_max_cor,Names=rownames(data),tr=0.9),
  file = "data_complete_table_09.tab",row.names=F,col.names=T,quote = F,sep="\t")
```

```
data_complete_table_08 =
  ks_make_table(pair=data_pair_log_max,Corr=data_max_cor,Names=rownames(data),tr=0.8)
```

```
head(data_complete_table_08)
```

```
##           Names Total_number_of_correlations Triplet_number_correlation
## 1 Succinic acid                          444                         31
## 2  Fumaric acid                          376                         22
## 3    Malic acid                          367                         22
## 4       Alanine                          406                         26
## 5        Valine                         1203                        140
## 6       Leucine                         1266                        142
##   Quadruple_number_correlation Pairs_number_correlation
## 1                          413                        0
## 2                          354                        0
## 3                          345                        0
## 4                          380                        0
## 5                         1058                        5
## 6                         1120                        4
##   Triplet_mean_correlation Quadruple_mean_correlation
## 1                0.8926006                  0.8997701
## 2                0.9025779                  0.8993464
## 3                0.9003569                  0.9000514
## 4                0.8972733                  0.8978808
## 5                0.8572479                  0.8696380
## 6                0.9275254                  0.9245783
##   Pairs_mean_correlation Triplet_max_correlation Quadruple_max_correlation
## 1                     NA               0.9744128                 0.9831480
## 2                     NA               0.9707399                 0.9766461
## 3                     NA              0.9716654                0.9777344
## 4                     NA              0.9719510                0.9832491
## 5              0.8359387              0.9741138                0.9835517
## 6              0.9361395              0.9799599                0.9851220
##   Pairs_max_correlation Triplet_min_correlation Quadruple_min_correlation
## 1                    NA              0.8043702                0.8001331
## 2                    NA              0.8173469                0.8000703
## 3                    NA              0.8012424                0.8001421
## 4                    NA              0.8100721                0.8007352
## 5             0.8601702              0.8012424                0.8002574
## 6             0.9724945              0.8050826                0.8000703
##   Pairs_min_correlation Stoichiometric_mean Stoichiometric_max
## 1                    NA            2.505834                  4
## 2                    NA            2.489726                  4
## 3                    NA            2.499298                  4
## 4                    NA            2.488550                  4
## 5             0.8060250            2.426640                  4
## 6             0.8601702            2.473132                  4
```

Comparative analysis

Coupling degree for metabolite m indicates the number of stoichiometric correlations above a given threshold τ in which the metabolite m participated. Therefore, to compare two species based on the constrained maximal correlation approach, the coupling degree of metabolites can be used.

Function *ks_make_bipartite_graph* constructes a bipartite graph using the pairs, triplets, and quadruples with maximal correlations. A bipartite graph is composed of two disjoint sets of nodes: U and V, where U contains the metabolite pairs, triplets, and quadruples with maximal stoichiometric correlation coefficients and V includes all single metabolites (*i.e.*, *rownames(data)*). An edge between the nodes of the two sets is drawn, if a metabolite m participates in a pair, triplet or quadruple. The coupling degree of a metabolite m is then the degree of the node m in the constructed bipartite graph.

The input arguments of the function *ks_make_bipartite_graph* are: *pairs*, the pairwise maximal correlations (output of the function *ks_find_max_cor*), as well as *triplets* and *quadruples*, the maximal correlation due to triplets and quadruples (the output of the Python script, *data_max_cor$triplets* and *data_max_cor$quadruples*), respectively. To reconstruct the bipartite graph, the pairs, triplets, and quadruples with stoichiometric correlation coefficients above the threshold, *tr*, will be considered. As a hint, *data2* corresponds to the second metabolomics data set which includes metabolite profiles of the second species. The same procedure as done for the first data set (*i.e.*, *data* in this example) can be done for the second data set *data2* to detect maximal stoichiometric correlations. Therefore, the data variables that their name is started with "*data2_*" correspond to the result of SCA on the second data set *data2*.

```
data1_bipartite_graph_08 = ks_make_bipartite_graph(
                             pairs=data_pair_log_max,triplets=data_max_cor$triplets,
                             quadruples=data_max_cor$quadruples,tr=0.8)

data1_bipartite_graph_09 = ks_make_bipartite_graph(
                             pairs=data_pair_log_max,triplets=data_max_cor$triplets,
                             quadruples=data_max_cor$quadruples,tr=0.9)

data2_bipartite_graph_08 = ks_make_bipartite_graph(
                             pairs=data2_pair_log_max,triplets=data2_max_cor$triplets,
                             quadruples=data2_max_cor$quadruples,0.8)

data2_bipartite_graph_09 = ks_make_bipartite_graph(
                             pairs=data2_pair_log_max,triplets=data2_max_cor$triplets,
                             quadruples=data2_max_cor$quadruples,tr=0.9)
```

```
data1_bipartite_graph_08
```

```
## IGRAPH DNWB 3119 12063 --
## + attr: name (v/c), type (v/l), weight (e/n)
## + edges (vertex names):
##  [1] Isoleucine->Threonine    ->Isoleucine
##  [2] Isoleucine->Threonine    ->Threonine
##  [3] Isoleucine->Phenylalanine->Isoleucine
##  [4] Isoleucine->Phenylalanine->Phenylalanine
##  [5] Proline->Threonine       ->Proline
##  [6] Proline->Threonine       ->Threonine
##  [7] Proline->Phenylalanine   ->Proline
##  [8] Proline->Phenylalanine   ->Phenylalanine
## + ... omitted several edges
```

The function *ks_graph_to_dataframe* converts the resulting bipartite graph from function *ks_make_bipartite_graph* into a *data.frame* structure. The input arguments are two bipartite graphs, *graph1* and *graph2*, generated by the function *ks_make_bipartite_graph* for the two species under comparison (e.g., *A. thaliana* and *E. coli*), as well as the names of metabolites in both metabolimics data sets: *name1* and *name2*. *column_name1* and *column_name2* define the column names of the *data.frame* returned as an output of this function. The output is a single *data.frame* which includes the metabolite names and their corresponding coupling degrees (*i.e.* node degrees) for both species under comparison.

```
degree_08 = ks_graph_to_dataframe(
                graph1=data1_bipartite_graph_08,graph2=data2_bipartite_graph_08,
                name1=rownames(Ecoli),name2=rownames(Ara),
                column_name1="Data1 metabolite coupling degree",
                column_name2="Data2 metabolite coupling degree")

degree_09 = ks_graph_to_dataframe(
                graph1=data1_bipartite_graph_09,graph2=data2_bipartite_graph_09,
                name1=rownames(data),name2=rownames(data2),
                column_name1="Data1 metabolite coupling degree",
                column_name2="Data2 metabolite coupling degree")
```

```
head(degree_08)
```

```
##    Metabolites Ecoli metabolite coupling degree  Metabolites
## 1      Leucine                              321      Glycine
## 2     Tyrosine                              323    Glutamine
## 3 beta-alanine                              372   Malic acid
## 4   Methionine                              383      Proline
## 5      Glycine                              411 Fumaric acid
## 6    Glutamine                              412    Aspartate
##   Ara metabolite coupling degree
## 1                            335
## 2                            357
## 3                            367
## 4                            370
## 5                            376
## 6                            384
```

The function *write_list* allows writing the list structured input into an Excel file. Each entry of the list will be written into a separated worksheet of the Excel file. The function takes two arguments as an input: *my_list*, which is the list to be written into an Excel file and *wb_name*, the name of the Excel file. The function *write_list* will test whether the package XLConnect is available or not. In case this package is not available, each entry of the input list will be separately written into a .csv file format. The .csv file names are automatically set.

```
out_list = list(degree_08,degree_09)
names(out_list) <- c("degree_08","degree_09")
write_list(my_list=out_list,wb_name = "metabolite_coupling_degree.xlsx")
```

The function *ks_shared_metabolites* returns the overlap of the maximal correlations due to the pairs, triplets, and quadruples between the two species under comparison at a desired threshold. The same set of metabolites should be used for both species for a meaningful comparison. The input arguments are: *data_pair* and *data2_pair*, the maximal correlations due to pairs, *data_all* and *data2_all*, the maximal correlation due to triplets and quadruples for the two species under comparison. Additionally, *tr*, the threshold for the correlation coefficients is set to filter the maximal correlations. The input arguments *name1* and *name2* are used to set the names of the two data sets within the returned list. The output is a list with four data frames, a *data.frame* summarizing the number of pairs, triplets, and quadruples shared between the two data sets, the overlap of pairs, the overlap of triplets, and the overlap of quadruples between the two data sets.

```
Shared_metabolites_08 = ks_shared_metabolites(
                          data_pair=data_pair_log_max,data_all=data_max_cor,
                          data2_pair=data2_pair_log_max,data2_all=data2_max_cor,
                          0.8,name1="Data_set_1",name2="Data_set_2")

Shared_metabolites_085 = ks_shared_metabolites(
                          data_pair=data_pair_log_max,data_all=data_max_cor,
                          data2_pair=data2_pair_log_max,data2_all=data2_max_cor,
                          0.85,name1="Data_set_1",name2="Data_set_2")

Shared_metabolites_09 = ks_shared_metabolites(
                           data_pair=data_pair_log_max,data_all=data_max_cor,
                           data2_pair=data2_pair_log_max,data2_all=data2_max_cor,
                           0.9,name1="Data_set_1",name2="Data_set_2")

write_list(Shared_metabolites_08,wb_name ="Shared_metabolites_08.xls")
write_list(Shared_metabolites_085,wb_name ="Shared_metabolites_085.xls")
write_list(Shared_metabolites_09,wb_name ="Shared_metabolites_09.xls")
```

```
Shared_metabolites_08[[1]]
```

```
##                               Compare Pairs Triplets Quadruples
## 1                             E .coli     9      319       2772
## 2                         A. thaliana    13      384       2825
## 3 Overlap of E. coli and A. thaliana     1       65        457
```

This document was created with the following R-version

```
 sessionInfo()
```

```
## R version 3.3.2 (2016-10-31)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 14.04.5 LTS
##
## locale:
```

```
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
##  [3] LC_TIME=de_DE.UTF-8        LC_COLLATE=en_US.UTF-8
##  [5] LC_MONETARY=de_DE.UTF-8    LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=de_DE.UTF-8       LC_NAME=C
##  [9] LC_ADDRESS=C               LC_TELEPHONE=C
## [11] LC_MEASUREMENT=de_DE.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
## [1] igraph_1.0.1
##
## loaded via a namespace (and not attached):
##  [1] backports_1.0.5 magrittr_1.5    rprojroot_1.2   tools_3.3.2
##  [5] htmltools_0.3.5 yaml_2.1.14     Rcpp_0.12.10    stringi_1.1.2
##  [9] rmarkdown_1.4   knitr_1.15.1    stringr_1.2.0   digest_0.6.12
## [13] evaluate_0.10
```