Genes that are best friends: rank-backwards-rank analysis of correlation matrix

A.V. Favorov

Vavilov Institute of General Genetics, Russian Academy of Sciences, 119333, Moscow, Gubkina str., Moscow, Russia; 3Department of Oncology, Division of Biostatistics and Bioinformatics, Johns Hopkins University School of Medicine , 550 North Broadway, Baltimore, Maryland, United States of America, favorov@sensi.org

V.E. Ramensky

Center for Neurobehavioral Genetics, University of California, Los Angeles, Gonda Building, Room 4357B, 695 Charles E. Young Drive South, Los Angeles, California 90095, United States of America, ramensky@gmail.com

D.O. Bredikhin

Bioinformatics Group of Faculty of Bioengineering and Bioinformatics, Lomonosov Moscow State University, 119991, Moscow, GSP-1, Leninskiye Gory, MSU, 1-73, Moscow, Russia, bredikhin@fbb.msu.ru

A.A. Mironov

Bioinformatics Group of Faculty of Bioengineering and Bioinformatics, Lomonosov Moscow State University, 119991, Moscow, GSP-1, Leninskiye Gory, MSU, 1-73, Moscow, Russia, mironov@bioinf.fbb.msu.ru

Suppose we have a kind of gene-to-gene correlation matrix, e.g. correlation of expression. The more the genes are alike, the lager the value is. Actually, the following holds for any kind of comparable value, we will refer to it as a correlation just to be more definite. We want to find the list of best 'friends' for each (or for some particular) gene (our gene). Friends are the genes (they) that are the closest to our gene. The naïve approach would be to rank all the genes by our-to-him correlation. Its drawback is that we cannot tell the genes that are correlated with everything from genes that are correlated specifically with our gene. A kind of normalization could be applied to the correlation vectors of ‘they’ genes, but it is a way to prefer the genes that have a few friend for those that have a lot instead of to prefer specific friends of our gene for those that are friends of everybody, and it is not exactly the same thing. In addition, the result of ranking of normalized data can depend on the normalization procedure more than on the initial data.

The rank-backwards-rank (RBR) approach we propose here is based on a simple proposition that for a gene that is a good friend of our gene (to be more formal, it is specifically correlated with our gene), the rank of our gene in the tested gene's correlations list is to be rather high. Otherwise, if the tested gene is a friend-of-everybody, our gene's rank in his list can be quite low, while the tester's gene rank in our list is still high. Thus, we gather the the 'where we are in his list' ranks and then we sort this ranks to create the best-friends list.

Here is a formal description of the procedure.

Suppose we have similarity (e.g., correlation) matrix C_{ij} for time- or sample- series of a set of k genes $G = \{g_i : i \in 1..k\}$. The requirements for the matrix itself are very relaxed - it is to represent similarity of the series corresponding to genes and $\mathrm{C}_{ij} > \mathrm{C}_{il}$ means that g_i data are more similar to that of g_j than to that of g_l.

Now, we want to rank all genes $\{G \setminus g_i\}$ by their specific concordance (friendship) with gene g_i. The naïve ranking by C_{ij} will rank genes g_j by their similarity with g_i rather than by their specific similarity. Instead of this, let's define the backwards-rank (friendship) function $F(j \mid i)$ that is the rank of C_{ij} in the list of C_{*j}. In other words, the function shows how specifically similar is g_i for g_j. Now, we can define genes that are specifically concordant (or discordant) with g_i by ranking the backwards ranks. The best friends of g_i is the top of the ordered $F(j \mid i)$ list. Similarly, the main antagonists are in the tail of the list.

Currently, we see two ways of application of the RBR method: it could be used to investigate the transcriptional neighborhood of a gene of interest, e.g. of a gene that was picked up by GWAS, or it could be used to picture the transcriptional network structure by converting the RBR's to a similarity measure, e.g. $|g_i g_k| = \max\left(rank_k\{F(* \mid i)\}, rank_i\{F(* \mid \mathrm{k})\}\right)$ for top ranks and $|g_i g_k| = \min\left(rank_k\{F(* \mid i)\}, rank_i\{F(* \mid \mathrm{k})\}\right)$ for tail ranks, and then by applying some clustering of clique-identifying methods on the similarities graph.