*Ontoscope:* Determining “identity-defining” transcription factors for various cell types

Vignette authors: Kartikay Chadha, Shivani Kamdar, Burton Mendonca, Anam Qudrat

University of Toronto

Department of Biochemistry

Compiled: April 27, 2016

**Contents**

**1 Introduction 2**

**2 Processing overview 2**

**3 Submodules and workflow**  **z**

3.1 Normalization z

3.2 FANTOM expression data import z

3.3 Background derivation from cell ontology z

3.4 Protein and transcription factor network analysis z

3.4.1 STRING

3.4.2 TRRUST

3.4.3 Regnet

3.5 Differential gene expression analysis z

3.6 Integration of network- and expression-based influence scores z

3.7 Selection of transcription factors needed for conversion z

**4 Post-processing and analysis z**

4.1 Visualization z

4.2 Literature-based confidence score z

4.3 Validation from published experimental data z

**5 Acknowledgments z**

**6 Session info z**

**7 References z**

**1 Introduction**

This document offers an overview of the R package *Ontoscope*, which uses publicly available expression and protein-protein interaction data to computationally determine a list of transcription factors able to facilitate conversion from one cell type to another. Following determination of this transcription factor list, it is also designed to assign confidence scores to each factor based on literature search and validation against published experimental data and/or other computational prediction software.

The *Ontoscope* package is designed to independently validate the findings of Owen Rackham and colleagues’ *Mogrify* ([www.mogrify.net](http://www.mogrify.net))1. Its workflow is based on the published *Mogrify* protocol, with modifications including the usage of updated transcription factor definitions and the inclusion of different regulatory networks (specifically, the TRRUST2 and Regnet3 protein-protein interaction datasets).

This vignette contains an overview of the package workflow, defining initialization parameters and walking through each submodule with examples and usage instructions provided.

**2 Processing overview**

*Ontoscope* determines transcription factors required for conversion based on calculation of network- and expression-based influence scores for differentially expressed factors in the target cell line as compared to the source. Target and source cells are identified by FANTOM consortium cell line IDs, and are restricted to those cell types for which FANTOM gene expression data from CAGE-seq is available.

Calculation involves six primary phases:

1. *Import of expression data:* The first step is to define source and target cell lines, reading in raw FANTOM expression count data for each cell line. Transcription factors are defined from published literature and databases, and all gene names are normalized to HGNC IDs in order to ensure accuracy of comparison between cell lines and datasets. Download of FANTOM count data is automatic upon input of cell line IDs, and does not require further input from the user.
2. *Calculation of background:* Something
3. *Differential expression-based influence score assignment:* Something
4. *Interaction-based influence score assignment:* In order to determine the regulatory influence of a transcription factor it is necessary to build a regulatory network. Ontoscope builds regulatory networks based on three publicly available data sources: STRING, TRRUST and RegNetwork. Based on this, the gene network-based influence on each TF is calculated for the conversion between cell states.
5. *Integration and binning:* Something
6. *Visualization and validation:* Something

**3 Submodules and workflow**

This section outlines examples and functions underlying the usage of *Ontoscope* to generate lists of transcription factors for conversion.

Users should begin by defining cells of origin and desired target cell type as follows:

> sourcecell <- “eye”

> target <- “fibroblast”

Here, a retinal to fibroblast cell conversion has been used as an example. It is important to define both keywords and FANTOM IDs for desired cell types, as while FANTOM IDs will be used for the actual Ontoscope workflow, keyword definitions are important for visualization and literature validation.

Next, users should select their desired FANTOM IDs for source and target cells by sourcing and running the **fantom\_import** submodule for both source and target cell types. Output for the source cell search is as follows:

> source(“./fantom\_import/fantom\_main.R”)

> fantomSearch(sourcecell)

V1 FANTOM.5.Ontology.ID FANTOM.5.Access.Number

1375 eye - muscle inferior rectus, donor1 FF:10272-104E2 1381

1376 eye - muscle lateral, donor2 FF:10298-104H1 1382

1377 eye - muscle medial, donor2 FF:10299-104H2 1383

1378 eye - muscle superior, donor2 FF:10297-104G9 1384

1379 eye - vitreous humor, donor1 FF:10268-104D7 1385

1380 eye, fetal, donor1 FF:10054-101G9 1386

From the list, the desired specific subtype (ie: lateral eye muscle, vitreous humor, fetal eye) of cell can be selected, and the FANTOM Ontology ID (FF:ID) noted. Here, we have chosen to convert inferior rectus eye muscle to cardiac fibroblast. FF:IDs may then be defined as follows:

> sourceFF <- “FF:10272-104E2”

> targetFF <- “FF:11268-116G8”

Following this, simply running runOntoscope.R will process the entire workflow for you, outputting a list of top transcription factors required for conversion with confidence rankings based on literature as well as heatmap and interaction network visualizations. The runOntoscope.R package also contains built-in functionality to cross-validate the transcription factor lists generated by Ontoscope against published conversion lists from either literature or the MOGRIFY package.

**3.1 Normalization**

Prior to beginning the workflow proper, it is important to first normalize all gene identifiers in order to allow comparison between gene lists and data from different sources. Ontoscope takes as input four main sources of data: gene expression data from FANTOM5, and protein-protein interaction and regulatory network data from STRING4, TRRUST, and REGNET. While FANTOM5 and TRRUST datasets contain HGNC identifiers for genes, STRING primarily identifies genes based on Ensembl IDs, and must be normalized to HGNC symbols for comparison purposes. Furthermore, while REGNET uses HGNC identifiers for its gene interactions, these IDs are four years old at the time of publication of this vignette, and may be outdated.

In order to normalize STRING interactions, the **normalizeWeave.R** submodule is sourced, using the R Bioconductor package biomaRt to create a new datafile from the base STRING data table with approximately 8.5 million protein-protein interactions. In order to avoid extremely lengthy processing times, STRING Ensembl protein IDs are placed in a new data frame as row names, following which a new vector of HGNC symbols is created with biomaRt. The updated file, **curatedOutput.Rdata**, is then available for usage in downstream applications.

Should Regnet files require updated HGNC symbol assignments, Entrez IDs can be retrieved from Regnet files and transcribed into HGNC symbols as well by altering the following lines in **normalizeRegnet.R** :

> IDmap <- data.frame(entrezgene=IDmap, HGNC=””, stringsAsFactors=FALSE)

> BMmap <- getBM(filters = “entrezgene”,

attributes = c(“entrezgene”, output),

values = IDmap$entrezgene,

mart = ensembl)

> colnames(BMmap) <- c(“entrezgene”, output)

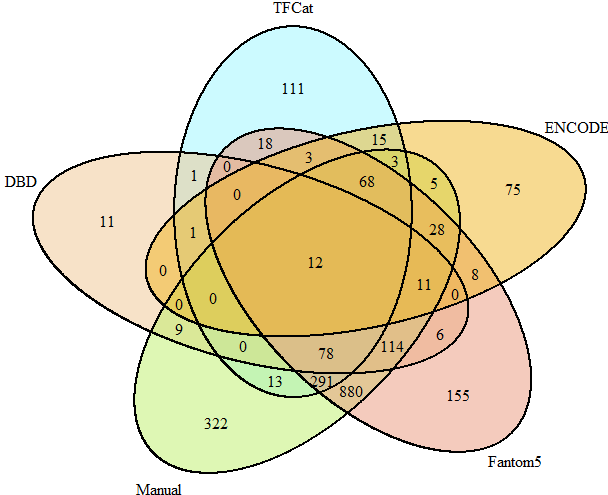
All input datafiles will now have been normalized to identify genes by HGNC symbols.

Finally, transcription factors must be defined prior to performing analysis. Sourcing the **normalizeTF.R** subscript will automatically compile a list of transcription factors common among at least two out of the five following datasets:

1. The TFCat database
2. The DBD (DNA-binding domain) database
3. A manually assembled list of transcription factors with annotations5
4. The FANTOM5 transcription factor dataset
5. ENCODE transcription factor data derived from their ChIP-seq antibody list

The choice of how many lists to use as a cutoff was based on variations in overlap between the lists **(Figure 1)**, and may be easily modified through subsetting of the MergedList table generated by this submodule, which lists all transcription factors and whether or not they are present in each dataset, via modification of the following line:

> TFList <- unique(MergedList[!MergedList$Count==”1”,])$Gene.Symbol



**Figure 1: Varying degrees of overlap between human transcription factor datasets.** Venn diagram depicting large overlap between the manually curated and Fantom5 lists, while other datasets possess less unique and overlapping transcription factors. Generated by the VennDiagrams package: draw.quintuple.venn()

Note that all transcription factors have been normalized to HGNC symbols where appropriate. The normalize submodule allows for easy replacement or update of lists through the replacement of their .csv datafiles with updated data.

**3.2 FANTOM expression data import**

Expression profiles for genes across available cell lines have been derived from cap analysis of gene expression (CAGE) sequencing by the FANTOM5 consortium. The FANTOM import submodule allows for the import and processing of raw read counts from this sequencing data, accepting several different possible search terms as inputs and allowing for import of normalized read counts as an alternative. For the purposes of the Ontoscope workflow, the **fantom\_main.R** submodule has already been sourced. Furthermore, as we perform our own normalization and background derivations further downstream, we must focus on importing raw read counts rather than normalized expression data.

The **runOntoscope.R** script first searches the FANTOM5 database for matching FF IDs and downloads the raw expression counts for each as follows:

> FFVect <- c(sourceFF, targetFF)

> fantomOntology(FFVect)

[1] "Sample\_DB Loaded!"

Returning RAW COUNTS

MATCHED: 2 of 2

2 Search Result(s) Were Found. Loading...

Loading Results from Fantom Access Number 1381 ( 1 / 2 ) ...

Results from Fantom Access Number 1381 Loaded!

Loading Results from Fantom Access Number 377 ( 2 / 2 ) ...

Results from Fantom Access Number 377 Loaded!

All results have been loaded into fantomResults

fantomResults is a large, subsettable list of dataframes containing genetic annotations, peak numbers, gene names, and HGNC, Uniprot, and Entrez Gene IDs for every sample, with one dataframe per sample. These results can be summarized in one dataframe, with normalized HGNC symbol gene names and raw expression counts for each gene, through the fantomSummarize command:

> fantomSummarize(5)

Preparing the Genes

Summarizing:eye - muscle inferior rectus, donor1.CNhs13444.10272-104E2

Summarizing:Fibroblast - Cardiac, donor1.CNhs12498.11268-116G8

Filtering Relevant Results. This step takes awhile ...

Preparing Normalized Gene Names ...

All Genes Normalized!

Fixing Duplicates ...

Applying Threshold ...

Your results have been summarized in: fantomCounts!

The bracketed number indicates the minimum threshold for read counts in each gene. Genes with read counts less than the threshold number (here, 5) will not be included in the summarized fantomCounts

file, which is now ready for downstream processing.

**3.3 Background derivation from cell ontology**

**3.4 Protein and transcription factor network analysis**

Ontoscope measures the importance of a transcription factor for a particular cell conversion by calculating its regulatory influence on its local network. In order to calculate “network influence” scores, Ontoscope reconstructs gene regulatory networks from three publicly available databases: STRING, TRRUST and RegNetwork.

3.4.1 STRING

STRING is a database of protein-protein interactions, either known or predicted based on various criteria from gene co-expression to abstract text mining [6].

Raw data downloaded from STRING comes as a delimited text file, with protein interaction records. Each interaction record includes the following data: Protein Ensembl IDs; Neighborhood score; Fusion score; Co-occurrence score; Co-expression score; Experimental score; Database score; Textmining score; Combined score.

The **normalizeWeave.R** script converts this raw text file into an R dataframe with HGNC symbols to identify the proteins.

The input to the script is the file path to the raw STRING text file, set with the following command:

STRINGsource <- file.path(“<file path>”)

When run, the script first converts the text file into an R dataframe and cleans up the protein Ensembl IDs by removing “9606.” prefixes.

Next, the Ensembl IDs of the proteins are converted to HGNC symbols, with the following general procedure:

1. Create a data frame of unique Ensembl IDs from the STRING dataframe
2. Collect HGNC symbols for each Ensembl ID, with the useMart() and getBM() functions from the biomaRt package
3. Add each HGNC symbol to the Ensembl IDs data frame from step 1
4. Replace the protein IDs in the STRING dataframe with the HGNC symbol
5. Rearrange the columns to move the HGNC symbols into the first two columns

The output of the **normalizeWeave.R** script is an R dataframe, saved as “curatedOutput.Rdata”, of the protein-protein interactions with each row representing a STRING protein-protein interaction, with proteins identified by HGNC symbols.

The **WEAVE-ALL.R** script contains code (below) to convert the “curatedOutput.Rdata” STRING protein‑protein interaction dataframe into a network i.e. an igraph network. The igraph network nodes are genes and the undirected edges are protein-protein interactions.

load("curatedOutput.Rdata")

high\_conf\_interactions = src[src$combined\_score > 700,]

STRGRAPH <- graph\_from\_data\_frame(high\_conf\_interactions, directed = TRUE)

Note that only “high-confidence” interactions, with a STRING combined score of are included in the network.

3.4.2 TRRUST

TRRUST is a manually curated transcriptional regulatory network produced using a sentence‑based text‑mining approach [7].

TRRUST network data is downloaded as a delimited text file, with each line representing a TF-gene relationship with the following data: transcription factor; regulated gene; type of regulation (Activation, Repression or Unknown); PMID reference for the journal article. An example is shown below:

AATF BAK1 Unknown 22983126

AATF BAX Repression 22909821

AATF BBC3 Unknown 22983126

AATF CDKN1A Unknown 17157788

AATF MYC Activation 20549547

AATF TP53 Unknown 17157788

The **TRRUST\_network.R** script contains code to create an igraph network from the TRRUST raw text file. The nodes in the igraph network are genes or TFs and the edges are regulatory relationships, directed from TFs to genes.

First, the TRRUST network text file must be converted to a dataframe using the loadTRRUST function. This function requires the raw text file "trrust\_rawdata.txt" to be in the working directory.

x <- loadTRRUST()

Next, the fixColumns function can be used to assign appropriate columns names to the dataframe.

x <- fixColumns(x)

The loaded dataframe can then be converted to an igraph object.

igraph\_object <- graph.data.frame(x)

The **TRRUST\_network.R** script also contains various filtering, network analysis and visualization functions that can act on a TRRUST network, either in dataframe or igraph network format.

**Filtering functions**

| **Function** | **Description** |
| --- | --- |
| typeSelect <-function(x, type,invert\_mode) | Returns interactions of a certain type in TRRUST data frame x. A boolean, invert\_mode, determines whether the type is included (TRUE) or excluded (FALSE) from the results. |
| genGeneChar<-function() | Generates a vector of gene IDs to use in filtering. |
| randomGenes <-function(x,number) | Selects a number of random genes from TRRUST data frame x. |
| filterGenes <- function(x,gene\_char) | Filters TRRUST data frame x using a vector of gene IDs gene\_char. Needs global variable trrust\_network\_mode to be set by user (see setMode below) |
| setMode <- function(mode\_num) | Sets global variable trrust\_network\_mode for filtering based on mode\_num.  mode\_num == 1 means the user has a list of TFs and wants to find out what genes they affect.  mode\_num == 2 means the user has a list of genes and wants to find out what TFs regulate those genes. |

**Network analysis functions**

| **Function** | **Description** |
| --- | --- |
| getEdges <-function(x) | Returns edges between TFs and genes based on TRRUST data frame x. |
| getNodes <- function(x) | Returns nodes between TFs and genes based on TRRUST data frame x. |
| getWeights <- function(nodes,edges) | Assigns weight to nodes of a network based on the betweenness centrality calculated for each node based on the edges of the network. |
| getClusters <- function(nodes,edges) | Clusters nodes of a network based on the betweenness centrality calculated for each node based on the edges of the network. |
| getPMIDs <- function(edges,trrust) | Assigns PMID values to titles of the edges of a network. PMID values are from a TRRUST data frame trrust. |
| getAction <- function(edges,trrust) | Assigns regulatory action type labels to the edges of an igraph network. Labels are from a TRRUST data frame trrust. |

**Visualization functions**

| **Function** | **Description** |
| --- | --- |
| visGraph <-function(nodes,edges) | Visualization of igraph network via visNetwork function |
| visGraph\_f <-function(nodes,edges) | Visualization of igraph network via visNetwork function |
| exportVis <- function(name) | Exports visGraph results |
| exportVis\_f <- function(name) | Exports visGraph\_f results |

3.4.3 RegNetwork

RegNetwork is a database containing TF-TF, TF-gene, TF-miRNA, miRNA-TF, and miRNA-gene interactions for human and mouse. It was constructed based on 25 selected databases, as well as inferred regulatory relationships based on TF-binding site motifs [8].

The **REGNET.R** script converts .csv files of regulatory relationships to an igraph network. The nodes of the network are TFs or genes, and the edges are regulatory relationships, directed from regulators to targets.

First, the .csv tables must be exported from the RegNetwork database search page (<http://www.regnetworkweb.org/search.jsp>). For the initial Ontoscope build, three .csv files were downloaded by selecting “Human” as the organism and “High”, “Medium” and “Low” confidence levels. Below is an example of the .csv data:

"regulator\_symbol","regulator\_id","target\_symbol","target\_id","database","evidence","confidence"

"ABL1","25","SHC3","53358","kegg","Experimental","High"

"ABL1","25","STAT5B","6777","kegg","Experimental","High"

"ABL1","25","CBLB","868","kegg","Experimental","High"

"ABL1","25","CBLC","23624","kegg","Experimental","High"

"ABL1","25","CD55","1604","kegg","Experimental","High"

Next, the high, medium and low confidence .csv data is combined into an R dataframe, with HGNC symbols of regulators and targets in the first 2 columns.

HIGHCONF <- read.csv("REGNET\_HIGH\_CONF.csv", header=TRUE, sep= ",")

MEDIUMCONF <- read.csv("REGNET\_MEDIUM\_CONF.csv", header=TRUE, sep= ",")

LOWCONF <- read.csv("REGNET\_LOW\_CONF.csv", header=TRUE, sep= ",")

REGNETDB <- rbind(HIGHCONF, MEDIUMCONF, LOWCONF)

REGNETDB <- REGNETDB[, c(1,3,2,4,5,6,7)]

The REGNETDB dataframe is then filtered to remove miRNA interactions, which are undesirable for calculating network influence scores in Ontoscope.

The lines of code below are taken from different parts of **REGNET.R**. The miRNADB array shows the miRNA databases (<http://www.regnetworkweb.org/source.jsp>), whose interactions are to be filtered out.

miRNADB <- c("microT", "miRanda", "miRBase", "miRecords", "miRTarBase", "PicTar", "Tarbase", "TargetScan", "transmir")

As shown below, the database filteredDB is created by removing interactions whose database belongs to miRNADB.

for (db in miRNADB) {

filteredDB<-filteredDB[ grep(db, filteredDB$database, invert=TRUE), ]

…

Finally, the combined, filtered R dataframe of regulatory interactions is converted to a directed graph.

REGNETGRAPH <- graph\_from\_data\_frame(filteredDB, directed = TRUE)

3.4.4 WEAVE-ALL.R

The WEAVE-ALL.R script contains code to combine the on STRING, TRRUST and RegNetwork generated igraph networks into one file. The graphs still remain separate, allowing the user to select which network to use in the downstream modules. The following inputs are required:

1. "curatedOutput.Rdata" generated by normalizeWeave.R script, located in the **WEAVE‑ALL.R** working directory
2. TRRUST raw data stored in the **TRRUST\_network.R** working directory
3. High, medium and low confidence CSV data tables from RegNetwork website, located in **REGNET.R** working directory

When run, the script generates and saves an .RData file that contains all three regulatory networks (based on STRING, TRRUST and RegNetwork) as igraph objects.

save(STRGRAPH, REGNETGRAPH, TRRUSTGRAPH, getTFSubgraph, file="WEAVE.RData")

It also contain s function getTFSubgraph that generates a subgraph centered around a particular TF or gene in a regulatory network. For example, the code below generates the network neighborhood of transcription factor MYC of order 2 based on the REGNETGRAPH gene regulatory network.

SubgraphList<-getTFSubgraph("MYC", 2, REGNETGRAPH)

**3.5 Differential gene expression analysis**

**3.6 Integration of network- and expression-based influence scores**

The network-based influence scores are calculated based on the STRING database. To calculate the score of each TF, there are three scores used. Firstly the **Gsx** or the differential gene expression score is used which determines the gene expression in the source and target cell types. Secondly, the **Lrn** is used which defines the distance between the gene and its regulatory TF. The farther away a particular gene is from a TF, the smaller the influence of that TF. Thirdly, the **Orn** or the out degree of the parent of the gene in the network is identified. The larger the number of interactions the parent of the gene has, the less significant is its interaction with the gene or interest.

The **Gsx** score has been previously calculated and hence is acquired. Next, the getTFSubgraph is loaded from the WEAVE.RData as an igraph object. After a subgraph is plotted, the script goes through every gene in the TF’s sphere of influence and acquires the gene from the Gsx score table. It then finds the distance away from the TF or **Lrn**.

> Lrn <- shortest.paths(graph <- sub, v <- current\_gene, to <TF) [1]

The parent of the gene and its interaction partners or **Orn** is then determined which is used to calculate the edges of the nodes in the network.

The final TF score is calculated using the following formula:

> Gene\_Score <- Gsx \* (1/(Lrn\*Lrn)) \* (1/Orn)

The output generates a TF score that is based on its network of influence.

**3.7 Selection of transcription factors needed for conversion**

**4 Post-processing and analysis**

**4.1 Visualization**

**4.2 Literature-based confidence score**

**4.3 Validation from published experimental data**

**5 Acknowledgments**

This package was developed as part of the BCB420/JTB2020 bioinformatics class as part of the Department of Biochemistry at the University of Toronto, Canada, with the support of Dr. Boris Steipe. We wish to acknowledge all co-authors of *Ontoscope* package modules, who have equally contributed to the development of this project: Julian Mazzitelli, Dmitry Horodetsky, Christopher Li, Kartikay Chadha, Fupan Yao, Shivani Kamdar, Dan Litovitz, Howard Wu, Anam Qudrat, Jialun Tom Chen, Ryoga Li, Eugenia Barkova, Burton Mendonca, Phil Fradkin, and Boris Steipe.

**6 Session Info**

> sessionInfo(package=NULL)

(The output of the above command needs to be filled in by Dr. Steipe after integrating all the code and running it).

**7 References**

[1] Rackham OJ, Firas J, Fang H, Oates ME, Holmes ML, Knaupp AS, FANTOM Consortium, Suzuki H, Nefzeger CM, Daub CO, Shin JW, Petretto E, Forrest AR, Hayashizaki Y, Polo JM, Gough J. A predictive computational framework for direct reprogramming between human cell types. *Nat Genet,* 48(3): 331-335, 2016.

[2] Han H, Shim H, Shin D, Shim JE, Ko Y, Shin J, Kim H, Cho A, Kim E, Lee T, Kim H, Kim K, Yang S, Bae D, Yun A, Kim S, Kim CY, Cho HJ, Kang B, Shin S, Lee I. TRRUST: a reference database of human transcriptional regulatory interactions. *Sci Rep,* 5:11432, 2015.

[3] Chi SM, Seo YK, Park YK, Yoon S, Park CY, Kim YS, Kim SY, Nam D. REGNET: mining context-specific human transcription networks using composite genomic information. *BMC Genomics,* 15:450, 2014.

[4] Szklarczyk D, Franceschini A, Wyder S, Forslund K, Heller D, Huerta-Cepas J, Simonovic M, Roth A, Santos A, Tsafou KP, Kuhn M, Bork P, Jensen LJ, von Mering C. STRING v10: protein-protein interaction networks, integrated over the tree of life. *Nucleic Acids Res*, 43:D447-52, 2015.

[5] Vaquerizas JM, Kummerfield SK, Teichmann SA, Luscombe NM. A census of human transcription factors: function, expression and evolution. *Nat Rev Genet*, 10(4):252-263, 2009.

[6] STRING database, <http://www.string-db.org/>

[7] Han, H., Shim, H., Shin, D., Shim, J. E., Ko, Y., Shin, J., . . . Lee, I. (2015). TRRUST: A reference database of human transcriptional regulatory interactions. Sci. Rep. Scientific Reports, 5, 11432. doi:10.1038/srep11432

[8] Liu, Z., Wu, C., Miao, H., & Wu, H. (2015). RegNetwork: An integrated database of transcriptional and post-transcriptional regulatory networks in human and mouse. Database, 2015. doi:10.1093/database/bav095