Message ID: **190** Entry time: **Mon Apr 5 15:47:19 2021**

Author:	Saptarshi
Type:	Other
Category:	General
Subject:	Aspergillus fumigatus: The construction of the prior network

This elog summarizes the steps taken to construct the prior network of Aspergillus fumigatus (hereafter, Aspergillus).

The prior-network construction is performed through motif scanning.
The motifs are acquired from the following sources:

- the CIS-BP database (http://cisbp.ccbr.utoronto.ca/),
- the in-house motifs provided by the Keller lab.

We also investigated the JASPAR database (http://jaspar.genereg.net/). However, no Aspergillus fumigatus data is found on Jaspar as of Feb 24, 2020.
Note: If someone is interested in working with the JASPAR motifs for some other species, he/she may contact Erika. Erika has used human and mouse motifs from the JASPAR 2020 r
their motifs compatible with hg19 and mm10. See the directories in step 1 of Method: https://elog.discovery.wisc.edu/ErikaActionItems/329. There are READMEs in those directories

In the following sections, we describe how the motifs are processed from each sources.

The CIS-BP Motifs

Download the CIS-BP Motifs
Go to http://cisbp.ccbr.utoronto.ca/ -> menu "Bulk downloads" -> Select species "Aspergillus fumigatus" -> Check all boxes ->
Download species archive -> a download link appears, e.g., http://cisbp.ccbr.utoronto.ca/tmp/Aspergillus_fumigatus_2021_02_24_7:38_pm.zip .
Use the download link to download the motifs in the desired directory.

```
## pwd=/mnt/dv/wid/projects5/Roy-Aspergillus/Data/Motif/Motif_Afum_CisBp/
curl -L http://cisbp.ccbr.utoronto.ca/tmp/Aspergillus_fumigatus_2021_02_24_7:38_pm.zip -o Aspergillus_fumigatus_2021_02_24_7:38_pm.zip
```

Unzip

```
## pwd=/mnt/dv/wid/projects5/Roy-Aspergillus/Data/Motif/Motif_Afum_CisBp/
unzip Aspergillus_fumigatus_2021_02_24_7:38_pm.zip
```

Once unzipped, there will be a README.txt. Go through it to learn which file contains what information.

Cite CIS-BP while publishing:
Determination and inference of eukaryotic transcription factor sequence specificity.
Weirauch MT, Yang A, Albu M, Cote AG, Montenegro-Montero A, Drewe P, Najafabadi HS, Lambert SA, Mann I, Cook K, Zheng H, Goity A, van Bakel H, Lozano JC, Galli M, Lewsey MG, Govindarajan S, Shaulsky G, Walhout AJ, Bouget FY, Ratsch G, Larrondo LF, Ecker JR, Hughes TR.
Cell. 2014 Sep 11;158(6):1431-43. doi: 10.1016/j.cell.2014.08.009.
PMID: 25215497

Processing steps 1-2 in this elog closely follow steps 1-4 in Sara's guideline for motif scanning: https://elog.discovery.wisc.edu/Software/166.

Step 1: Prepare motifs in the PFM format

Use the following command to prepare some prerequisite files, e.g.,
prepare Motif_ID (e.g., M00003_2.00) to TF_Name (e.g., AFUA_3G13920) mapping and save it in "info.txt". It's a many-to-one mapping.
Some other files that are prepared through this command are: names.txt, motifIDs.txt, motifPWMs.txt, meme_files/Cis_BP_afum_0.meme.
There might be more files that are prepared through this command.

```
## pwd=/mnt/dv/wid/projects5/Roy-Aspergillus/Data/Motif/
bash PrepInfo_Afum.sh
```

Generate "afumPFM.txt" that represents each motif in five lines, e.g.,
>M01649_2.00;AFUA_2G07900
A [159319 259571 61610 155602 4192 810655 35237 343722 293030 327019]
C [403796 353906 194576 15916 882855 30810 68175 179439 206495 212729]
G [210800 194228 720 740485 3755 87168 299490 326959 131667 249640]
5 T [226085 192295 743094 87997 109198 71367 597098 149880 368808 210612]
The first line is of the format "><Motif_ID>;<TF_Name>". If there are more than one TF, then they are separated by "::", e.g.,
>M01106_2.00;AFUA_4G08590::AFUA_5G05990.
The remaining four lines are the position frequency matrix of the motif.

```
## pwd=/mnt/dv/wid/projects5/Roy-Aspergillus/Data/Motif/
module load anaconda3-2020.02 ## Load python version 3.7.6
python makePFM.py --motifdir Motif_Afum_CisBp/pwms_all_motifs --tfinfo Motif_Afum_CisBp/TF_Information_all_motifs.txt --motifnames Motif_Afum_CisBp/info.txt --outname a
```

Step 2: Identify or prepare BSGenome genome assembly package
Convert genome assembly's .fasta file to a .2bit file using tool "faToTwoBit": https://genome.ucsc.edu/goldenpath/help/twoBit.html

```
## pwd=/mnt/dv/wid/projects7/Roy-Aspergillus/Data/Motif/
##
./faToTwoBit /mnt/dv/wid/projects5/Roy-Aspergillus/Data/GenomeAssembly/Aspergillus_fumigatus.ASM265v1.dna.toplevel.fa Afum.ASM265v1.2bit
```

Then, create file Afum.ASM265v1.seed as instructed in Sara's elog.

Subsequently, use the following commands:

```
## Load R-3.5.3
module load anaconda3-2020.02

## Enter an R session. ">" represents the R command prompt.
R

## Set path to a local R library.
> if (!requireNamespace("BiocManager", quietly = TRUE))
    install.packages("BiocManager")
It would prompt you to create a local R library for the installation.
Saptarshi used "/mnt/ws/home/spyne/R/x86_64-conda_cos6-linux-gnu-library/3.5".

## Install the BSgenome package.
> BiocManager::install("BSgenome")

## Prepare the genome package of the given genome assembly
> library("BSgenome", lib.loc = "/mnt/ws/home/spyne/R/x86_64-conda_cos6-linux-gnu-library/3.5")
> BSgenome::forgeBSgenomeDataPkg("Afum.ASM265v1.seed") ## Creates sub-dir "BSgenome.Afum.ASM265v1"

## Quit the R session and return to bash
> q()
```

Next, test installing the created package with the following steps:

```
R CMD build BSgenome.Afum.ASM265v1 ## whcih will prepare a tar file of the package, BSgenome.Afum.ASM265v1_1.0.0.tar.gz
#> Warning: invalid uid value replaced by that for user 'nobody'
#> Warning: invalid gid value replaced by that for user 'nobody'
```

Please ignore the warnings.

```
R CMD check -l /mnt/ws/home/spyne/R/x86_64-conda_cos6-linux-gnu-library/3.5 BSgenome.Afum.ASM265v1_1.0.0.tar.gz ## check package
## You may check the log file at /mnt/dv/wid/projects5/Roy-Aspergillus/Data/Motif/BSgenome.Afum.ASM265v1.Rcheck/00check.log

R CMD INSTALL -l /mnt/ws/home/spyne/R/x86_64-conda_cos6-linux-gnu-library/3.5 BSgenome.Afum.ASM265v1_1.0.0.tar.gz ## install package
Check and installation are OK.
```

Then, count the number of motifs in afumPFM.txt.
To do that, open the file in vim and count the occurrences of pattern ">M" (every motif starts with this pattern) using vim command ":%s/>M//gn".
Returns "628 matches on 628 lines". Therefore, there are 628 motifs.

Set the number of motifs in run_motif_scan.sh and run scans for each individual motif.

```
## pwd=/mnt/dv/wid/projects5/Roy-Aspergillus/Data/Motif/
## Takes almost 2 hours for 628 motifs.
bash run_motif_scan.sh > motif_scan.out 2>&1
```

In the output directory (i.e. MotifScanResults/CisBp/), we'll find paired files {1.pwmout.RData, 1.pwmout.rc.RData} for motif 1, ..., paired files {628.pwmout.RData, 628.pwmout.rc.R
(628 * 2) = 1,256 files. Here, <motif_sl_no>.pwmout.RData and <motif_sl_no>.pwmout.rc.RData represent the instances on + and - strand sequences of the genome, respectively.
strand, hence, the term "rc" in the filename.

Then for each motif you'll use the writeMot2.R script to prepare a .bed format file.

```
## Write each motif in a .bed format file.
## Takes around 22 mins for 628 motifs.
## pwd=/mnt/dv/wid/projects5/Roy-Aspergillus/Data/Motif/
bash write_motifs.sh > write_motifs.out 2>&1
```

It outputs <motif_sl_no>.all.motifs.txt in directory MotifScanResults/CisBp/. The content of <motif_sl_no>.all.motifs.txt is in the .bed format as shown below.

```
head 1.all.motifs.txt
#> 1       1469    1478    +       5.0666739044367
#> 1       2236    2245    +       5.40667448758231
## Column 1 is Chromosome ID (there are 8 chromosomes in total for Aspergillus), column 2 is the starting coordinate, column 3 is the end coordinate,
## column 4 is either + strand or - strand, and column 5 is the log-likelihood score.
## For more details on the .bed format, please see https://en.wikipedia.org/wiki/BED_(file_format) .
```

Step 3: Generate Gene TSS Information
TSS = Transcriptional start site.
Generate gene TSS information of genes and other elements, such as pseudogenes.
For genes, the third column of "Aspergillus_fumigatus.ASM265v1.49.gff3" would say "gene".
For other elements, it would say the type of the element.
We kept all the elements. Otherwise, the total does not match the number of so-called genes in our TPM count files.

```
## pwd=/mnt/dv/wid/projects5/Roy-Aspergillus/Data/Motif/
## Outputs geneTSS_all.txt
bash getGeneTSS_all.sh
```

Step 4: Map Motifs to Genes
Map each pair of (start, end) coordinates to a gene based on the following algorithm.
If it is a + strand, then map when (coordinate start >= (gene start - upstream window)) && (coordinate end <= (gene start + downstream window)).
Else if it is a - strand, then map when (coordinate start >= (gene end - downstream window)) && (coordinate end <= (gene end + upstream window)).
Note: To produce the formula for - strand, replace {gene start, upstream window, downstream window} with {gene end, downstream window, upstream window}, respectively, in the
For this project, upstream window = 10 Kbp and downstream window = 1Kbp are used. The 1Kbp downstream window is pretty standard. For the upstream window, 2Kbp or 5Kbp ca

```
## pwd=## pwd=/mnt/dv/wid/projects5/Roy-Aspergillus/Data/Motif/
## Requires: mapMotifsToGenes.py
##
## For each input <motif_sl_no>.all.motifs.txt, outputs two files:
## (1) <motif_sl_no>.mapped.genes.txt
## (2) <motif_sl_no>.mapped.genes.summary.txt
bash map_motifs_to_genes.sh
```

Step 5: Map Motif Serial Numbers to TF Names

```
## pwd=/mnt/dv/wid/projects5/Roy-Aspergillus/Data/Motif/
## Requires: mapMotifSlNoToTFs.py
## Output: motif_to_TF_map.txt
bash map_MotifSlNo_to_TFs.sh
```

Step 6: Map TF Names to Target Gene Names with Confidence Scores

```
## Load Python 2.
module load anaconda2-2018.12

## pwd=/mnt/dv/wid/projects5/Roy-Aspergillus/Data/Motif/
python mapTFtoGene.py motif_to_TF_map.txt MotifScanResults/CisBp/U10_D1 .mapped.genes.summary.txt tf_to_gene_map_CisBp_U10_D1.txt
## Output: MotifScanResults/CisBp/U10_D1/tf_to_gene_map_CisBp_U10_D1.txt
## U10 = 10Kbp upstream window
## D1 = 1Kbp downstream window
```

The Keller Motifs

The Keller motifs are saved in Google drive: https://drive.google.com/drive/u/1/folders/1oEnp4BZexD_XguPaIP_JhkFBqNasAi_T. See the README for details.

We need to prepare similar files for the Keller motifs as provided by CIS-BP. That way we can process the Keller motif through the same pipeline as used for the CIS-BP motifs.
Therefore, the first step was to figure out which files we need to prepare for the Keller motifs. It turned out that they are: (a) "TF_Information_all_motifs.txt", and (b) a directory nan
representing the position weight matrix (pwm) of a distinct motif. Please note that although the directory name says "pwm", actually they are position probability matrices (ppms) sin
whenever the term "pwm" is mentioned, actually it means the ppms.

"TF_Information_all_motifs.txt" is manually created at /mnt/dv/wid/projects5/Roy-Aspergillus/Data/Motif/Motif_Afum_Keller/ mimicking that of the CIS-BP motifs.

For generating the pwm files, we employ the following procedure.
First, a sub-directory named "seqs_all_motifs" is created under /mnt/dv/wid/projects5/Roy-Aspergillus/Data/Motif/Motif_Afum_Keller/ . Inside this sub-directory, one text file is create
"M00002_1.00.txt" for motif 2. Inside each file, the corresponding motif sequence is written in the IUPAC format (https://www.bioinformatics.org/sms/iupac.html).
Then the following script is used to generate a pwm from each sequence.

```
## pwd=/mnt/dv/wid/projects5/Roy-Aspergillus/Data/Motif/
## Requires: iupac_seq_to_ppm.r
bash gen_pwm_Keller.sh
```

The aforementioned script uses 0.9 as the "high probability". Let us illustrate with an example. Suppose, a particular position in a given sequence has a high probability for one partic
remaining letters. Then G will be assigned a probability value of 0.9. On the other hand, 0.1 will be equally distributed between the remaining letters.
Similarly, if more than one letter have high probabilities for a particular position, then 0.9 will be equally distributed among them. On the other hand, 0.1 will be equally distributed be
Following this probability distribution, the aforementioned script outputs the pwm files in /mnt/dv/wid/projects5/Roy-Aspergillus/Data/Motif/Motif_Afum_Keller/pwms_all_motifs/P90/
directory, "M00001_1.00.txt" contains the pwm for motif 1, "M00002_1.00.txt" for motif 2, and so on.

Now, we have the equivalent files for the Keller motifs as that of the CIS_BP motifs. Therefore, we now process them following the same steps.
Processing steps 1-4 closely follow Sara's guideline for motif scanning: https://elog.discovery.wisc.edu/Software/166.

Step 1: Prepare motifs in the PFM format

Use the following command to prepare some prerequisite files, e.g.,
prepare Motif_ID (e.g., M00003_2.00) to TF_Name (e.g., AFUA_3G13920) mapping and save it in "info.txt". It's a many-to-one mapping.
Some other files that are prepared through this command are: names.txt, motifIDs.txt, motifPWMs.txt.
There might be more files that are prepared through this command.

```
## pwd=/mnt/dv/wid/projects5/Roy-Aspergillus/Data/Motif/
bash PrepInfo_Afum_Keller.sh
```

Generate "afumPFM.txt" that represents each motif in five lines, e.g.,
>M01649_2.00;AFUA_2G07900
A [159319 259571 61610 155602 4192 810655 35237 343722 293030 327019]
C [403796 353906 194576 15916 882855 30810 68175 179439 206495 212729]
G [210800 194228 720 740485 3755 87168 299490 326959 131667 249640]
5 T [226085 192295 743094 87997 109198 71367 597098 149880 368808 210612]
The first line is of the format "><Motif_ID>;<TF_Name>". If there are more than one TF, then they are separated by "::", e.g.,
>M01106_2.00;AFUA_4G08590::AFUA_5G05990.
The remaining four lines are the position frequency matrix of the motif.

```
## pwd=/mnt/dv/wid/projects5/Roy-Aspergillus/Data/Motif/
module load anaconda3-2020.02 ## Load python version 3.7.6
python makePFM.py --motifdir Motif_Afum_Keller/pwms_all_motifs/p90 --tfinfo Motif_Afum_Keller/TF_Information_all_motifs.txt --motifnames Motif_Afum_Keller/info.txt --ou
```

Step 2: Identify or prepare BSGenome genome assembly package
The BSGenome genome assembly package installation has already been done during the CIS-BP motif processing. Therefore, we can skip those sub-steps and move on to Keller-spec

Count the number of motifs in afumPFM_Keller.txt.
To do that, open the file in vim and count the occurrences of pattern ">M" (every motif starts with this pattern) using vim command ":%s/>M//gn".
Returns "4 matches on 4 lines". Therefore, there are 4 motifs.

Set the number of motifs in run_motif_scan.sh and run scans for each individual motif.

```
## pwd=/mnt/dv/wid/projects5/Roy-Aspergillus/Data/Motif/
## Takes 1 min.
bash run_motif_scan_Keller.sh > motif_scan_Keller.out 2>&1
```

In the output directory (i.e. MotifScanResults/Keller/), we'll find paired files {1.pwmout.RData, 1.pwmout.rc.RData} for motif 1, ..., paired files {4.pwmout.RData, 4.pwmout.rc.RData
8 such files. Here, <motif_sl_no>.pwmout.RData and <motif_sl_no>.pwmout.rc.RData represent the instances on + and - strand sequences of the genome, respectively. - strand is r
the term "rc" in the filename.

Then for each motif you'll use the writeMot2.R script to prepare a .bed format file.

```
## Write each motif in a .bed format file.
## pwd=/mnt/dv/wid/projects5/Roy-Aspergillus/Data/Motif/
bash write_motifs_Keller.sh > write_motifs_Keller.out 2>&1
```

It outputs <motif_sl_no>.all.motifs.txt in directory MotifScanResults/Keller/. The content of <motif_sl_no>.all.motifs.txt is in the .bed format as shown below.

```
head 1.all.motifs.txt
#> 1       2184    2189    +       6.59167373200866
#> 1       3477    3482    +       6.59167373200866
## Column 1 is Chromosome ID (there are 8 chromosomes in total for Aspergillus), column 2 is the starting coordinate, column 3 is the end coordinate,
```

```
## column 4 is either + strand or - strand, and column 5 is the log-likelihood score.
## For more details on the .bed format, please see https://en.wikipedia.org/wiki/BED_(file_format) .
```

Step 3: Generate Gene TSS Information
TSS = Transcriptional start site.
Generate gene TSS information of genes and other elements, such as pseudogenes.
For genes, the third column of "Aspergillus_fumigatus.ASM265v1.49.gff3" would say "gene".
For other elements, it would say the type of the element.
We kept all the elements. Otherwise, the total does not match the number of so-called genes in our TPM count files.

This step is independent of whether the motifs are from CIS_BP or the Keller lab. Since we have already produced the gene TSS information (/mnt/dv/wid/projects5/Roy-Aspergillus/D processing, no more action is required.

Step 4: Map Motifs to Genes
Map each pair of (start, end) coordinates to a gene based on the following algorithm.
If it is a + strand, then map when (coordinate start >= (gene start - upstream window)) && (coordinate end <= (gene start + downstream window)).
Else if it is a - strand, then map when (coordinate start >= (gene end - downstream window)) && (coordinate end <= (gene end + upstream window)).
Note: To produce the formula for - strand, replace {gene start, upstream window, downstream window} with {gene end, downstream window, upstream window}, respectively, in the
For this project, upstream window = 10 Kbp and downstream window = 1Kbp are used. The 1Kbp downstream window is pretty standard. For the upstream window, 2Kbp or 5Kbp ca

```
## pwd=## pwd=/mnt/dv/wid/projects5/Roy-Aspergillus/Data/Motif/
## Requires: mapMotifsToGenes.py
##
## For each input <motif_sl_no>.all.motifs.txt, outputs two files:
## (1) <motif_sl_no>.mapped.genes.txt
## (2) <motif_sl_no>.mapped.genes.summary.txt
bash map_motifs_to_genes_Keller.sh MotifScanResults/Keller 10000 1000 MotifScanResults/Keller/P90_U10_D1
## P90 = High probability value of 0.90
## U10 = 10Kbp or 10,000bp upstream window
## D1 = 1Kbp or 1,000bp downstream window
```

Step 5: Map Motif Serial Numbers to TF Names
Again, this step is independent of CIS-BP and Keller. During the CIS-BP motif processing, we have already completed this step and produced /mnt/dv/wid/projects5/Roy-Aspergillus/D

Step 6: Map TF Names to Target Gene Names with Confidence Scores

```
## Load Python 2.
module load anaconda2-2018.12
```

```
## pwd=/mnt/dv/wid/projects5/Roy-Aspergillus/Data/Motif/
python mapTFtoGene.py motif_to_TF_map.txt MotifScanResults/Keller/P90_U10_D1 .mapped.genes.summary.txt tf_to_gene_map_Keller_P90_U10_D1.txt
## Output: MotifScanResults/Keller/P90_U10_D1/tf_to_gene_map_Keller_P90_U10_D1.txt
## P90 = High probability value of 0.90
## U10 = 10Kbp upstream window
## D1 = 1Kbp downstream window
```

Merge CIS-BP and Keller TF-to-Gene Map Files
At this point, we have two TF-to-Gene map or "prior edge" files

- From CIS-BP: /mnt/dv/wid/projects5/Roy-Aspergillus/Data/Motif/MotifScanResults/CisBp/U10_D1/tf_to_gene_map_CisBp_U10_D1.txt; contains 88,769,214 edges.
- From Keller: /mnt/dv/wid/projects5/Roy-Aspergillus/Data/Motif/MotifScanResults/Keller/P90_U10_D1/tf_to_gene_map_Keller_P90_U10_D1.txt; contains 891,342 edges.

Between and within the files, there could be a large number of duplicate edges for a number of reasons, such as one particular TF can map to one specific gene via multiple motifs.

Combine the aforementioned files into a single file.

```
## pwd=/mnt/dv/wid/projects5/Roy-Aspergillus/Data/Motif/
bash merge_tf_to_gene_maps.sh MotifScanResults/CisBp/U10_D1/tf_to_gene_map_CisBp_U10_D1.txt MotifScanResults/Keller/P90_U10_D1/tf_to_gene_map_Keller_P90_U10_D1.txt Moti
## Output: MotifScanResults/Merged/tf_to_gene_map_Merged_P90_U10_D1.txt
```

Aggregate edge scores by taking the max scores for each (TF, target) gene pair.

```
## pwd=/mnt/dv/wid/projects5/Roy-Aspergillus/Data/Motif/
## Requires:
## aggreEdgeScores.py
bash aggre_edge_scores.sh MotifScanResults/Merged/tf_to_gene_map_Merged_P90_U10_D1.txt MotifScanResults/Merged/tf_to_gene_map_Merged_P90_U10_D1_aggregated.txt max
## Output: MotifScanResults/Merged/tf_to_gene_map_Merged_P90_U10_D1_aggregated.txt . It contains 936,557 edges.
```

Subset the Prior Network for the Common Genes
Spencer has made a list of genes that are present in both RNA-Seq and microarray data. It contains 9,252 genes: /mnt/dv/wid/projects5/Roy-Aspergillus/Data/common_genes.txt.
Therefore, only those genes should be present in the prior network. For that purpose, we apply the following procedure.

```
## Load Python 3.7.6
module load anaconda3-2020.11
```

Subset the prior network retaining only the common genes.

```
Subset the prior network for the common genes
##pwd=/mnt/dv/wid/projects5/Roy-Aspergillus/Data/Motif/
python subset_priorNet_for_common_genes.py \
MotifScanResults/Merged/tf_to_gene_map_Merged_P90_U10_D1_aggregated.txt \
/mnt/dv/wid/projects5/Roy-Aspergillus/Data/common_genes.txt \
MotifScanResults/Merged/tf_to_gene_map_Merged_P90_U10_D1_aggregated_common_genes.txt
## Output: MotifScanResults/Merged/tf_to_gene_map_Merged_P90_U10_D1_aggregated_common_genes.txt
## It contains 860,437 edges.
```

Make the Prior Network Sparse
The derived prior network is still dense, as expected.
Ali suggested around 20% density for the prior network. Therefore, once the TF list is prepared, calculate

E = #(TFs in the prior network) * #(all genes).
Take 20% of E = P.
Sort the prior network in descending order of confidence scores. Then select the top P number of edges and remove the remaining edges.

Count #(TFs in the prior network)

```
## pwd=/mnt/dv/wid/projects5/Roy-Aspergillus/Data/Motif/MotifScanResults/Merged/
cut -f1 tf_to_gene_map_Merged_P90_U10_D1_aggregated_common_genes.txt | sort | uniq | wc -l
# Output: 94
```

#(all genes) = #(common genes between RNA-Seq and microarray) = 9,252 (/mnt/dv/wid/projects5/Roy-Aspergillus/Data/common_genes.txt)

Therefore,
E = (94 * 9,252) = 869,688.

P = floor(0.2 * E) = floor(0.2 * 869,688) = floor(173,937.6) = 173,937 = Maximum allowable number of edges in the prior network.

Currently, the prior network (MotifScanResults/Merged/tf_to_gene_map_Merged_P90_U10_D1_aggregated_common_genes.txt) contains 860,437 edges.
Therefore, we have to sort the edges in the descending order of confidence scores and then select the top 173,937 edges.

Sort the edges in the descending order of confidence scores.

```
## pwd=/mnt/dv/wid/projects5/Roy-Aspergillus/Data/Motif/MotifScanResults/Merged/
## -n = sort in the numeric order; not in the alphabetical order.
## -r = sort in the descending order.
## -k3 = sort based on column 3 i.e. the confidence scores.
sort -nrk3 tf_to_gene_map_Merged_P90_U10_D1_aggregated_common_genes.txt > tf_to_gene_map_Merged_P90_U10_D1_aggregated_common_genes_sorted.txt
## Output: tf_to_gene_map_Merged_P90_U10_D1_aggregated_common_genes_sorted.txt.
## It contains 860,437 edges. They are sorted in the descending order of their confidence scores.
## Max edge confidence score = 20.00080600, min = 5.00302300. Multiple edges can have the same score.
```

Select the top 173,937 edges and remove the remaining edges.

```
## pwd=/mnt/dv/wid/projects5/Roy-Aspergillus/Data/Motif/MotifScanResults/Merged/
head -n 173937 tf_to_gene_map_Merged_P90_U10_D1_aggregated_common_genes_sorted.txt > tf_to_gene_map_Merged_P90_U10_D1_aggregated_common_genes_sorted_sparse.txt
## Output: tf_to_gene_map_Merged_P90_U10_D1_aggregated_common_genes_sorted_sparse.txt
## It contains 173,937 edges. They are sorted in the descending order of their confidence scores.
## Max edge confidence score = 20.00080600, min = 9.07347100. Multiple edges can have the same score.
```

Change the Range of the Confidence Scores

EstimateNCA requires that the confidence scores are in the range [0, 1].
To change the range of the confidence scores to [0, 1], perform **percentile ranking**.

```
## pwd=/mnt/dv/wid/projects5/Roy-Aspergillus/Data/Motif/MotifScanResults/Merged/
## "incr" implies that the higher the score the better. "decr" implies that the lower the score the better.
/mnt/dv/wid/projects2/Roy-common/programs/programs/rankedges/rankEdges \
tf_to_gene_map_Merged_P90_U10_D1_aggregated_common_genes_sorted_sparse.txt \
tf_to_gene_map_Merged_P90_U10_D1_aggregated_common_genes_sorted_sparse_pRanked.txt \
incr
## Output: tf_to_gene_map_Merged_P90_U10_D1_aggregated_common_genes_sorted_sparse_pRanked.txt
## It contains 173,937 edges. However, the percentile ranking program produces an unsorted output.
## Therefore, we have to sort the edges once again.
```

Once again, sort the edges in the descending order of confidence scores.

```
## pwd=/mnt/dv/wid/projects5/Roy-Aspergillus/Data/Motif/MotifScanResults/Merged/
## -n = sort in the numeric order; not in the alphabetical order.
## -r = sort in the descending order.
## -k3 = sort based on column 3 i.e. the confidence scores.
sort -nrk3 \
tf_to_gene_map_Merged_P90_U10_D1_aggregated_common_genes_sorted_sparse_pRanked.txt \
> tf_to_gene_map_Merged_P90_U10_D1_aggregated_common_genes_sorted_sparse_pRanked_sorted.txt
## Output: tf_to_gene_map_Merged_P90_U10_D1_aggregated_common_genes_sorted_sparse_pRanked_sorted.txt.
## It contains 173,937 edges. They are sorted in the descending order of their confidence scores.
## Max edge confidence score = 1, min = 0.000108378. Multiple edges can have the same score.
```

Thus, the final prior network is /mnt/dv/wid/projects5/Roy-Aspergillus/Data/Motif/MotifScanResults/Merged/tf_to_gene_map_Merged_P90_U10_D1_aggregated_common_genes_sort
It contains 173,937 edges. Max edge confidence score = 1, min = 0.000108378. Multiple edges can have the same score.