A User Manual for PRANA

Seungjun Ahn

Oct 16th, 2022

We introduce a Pseudo-value Regression Approach for Network Analysis (PRANA) (Ahn *et al.*, 2022). To our knowledge, this is the first attempt of utilizing a regression modeling for the differential network (DN) analysis by collective gene expression levels under two experimental conditions (*e.g.* 'current' vs. 'non-current smokers' or 'high-risk' vs. 'low-risk'). We start from the mutual information (MI) criteria, followed by pseudo-value calculations, which are then entered into a robust regrssion model.

Requirements

Please download the following R code from my GitHub repository (https://github.com/sjahnn/PRANA) to employ our method.

- `TotalConnectivity.R` is to calculate the total connectivity (a continuous version of degree centrality of a gene) of estimated association matrix.
- `EmpiricalBayes_Datta_2005.R` is to compute the adjusted p-values via the empirical Bayes approach (Datta and Datta, 2005), an extension of Westfall-Young step-down procedure.
- `PRANA_main.R` is the primary code for the analysis. This will load the two aforementioned R codes.

Example

A real-data analysis was performed to showcase the utility of PRANA. First and formost, please download `combinedCOPD_RelatedGenesOnly.RDS` from my GitHub respository. This contains clinical and expression data for 406 samples and 28 COPD-related genes that were highlighted in a recent genome-wide association study (Sakornsakolpat *et al.*, 2019). The full data is available from the Gene Expression Database with accession number GSE158699 (Wang *et al.*, 2021).

Preparation

Below are the R packages that you will need to install.

```
# library(dnapath) # To obtain mutual information (MI) estimate via ARACNE.
# library(dplyr)   # To use bind_rows() later as part of converting results into data.frame.
# library(parallel) # To use mclapply() when re-estimating the association matrix.
# library(robustbase) # To fit a robust regression.
```

Please provide the directory information where your three R codes and RDS data file downloaded from the GitHub repository.

```
#dir_main = "your file directory where you saved three R codes."
#dir_COPD_RGO_data = "your file directory where you saved RDS file from the repository."
```

Load the two R codes and dataset.

```
# This will load the code calculating adjusted p-values for each genes.
source(file.path(dir_main, "EmpiricalBayes_Datta_2005.R"))
# This is to calculate the total connectivity (thetahats).
```

```
source(file.path(dir_main, "TotalConnectivity.R"))
# Combined phenotype and expression data.
combinedCOPDdat_RGO = readRDS(file.path(dir_COPD_RGO_data, "combinedCOPD_RelatedGenesOnly.rds"))
```

Of note, `est_method` is to specify the method to estimate the association matrix based on your expression data.

```
est_method = run_aracne # ARACNE
# gene expression data part of the downloaded data.
rnaseqdat = combinedCOPDdat_RGO[ , 8:ncol(combinedCOPDdat_RGO)]
rnaseqdat = as.data.frame(apply(rnaseqdat, 2, as.numeric))

## Additional covariates (phenotypic data) sorted by current smoking groups:
# FYI, the first column is ID, so not using it.
phenodat = combinedCOPDdat_RGO[order(combinedCOPDdat_RGO$currentsmoking), 2:7]
```

If you are using your own data, please make sure the rows and columns of the dataset are structured with sample size n and p genes, respectively.

```
head(rnaseqdat)
```

```
##      10370    10420        1306    155185   158158     1653     1762    23389
## 1 4.917861 3.734176 -1.31140328 0.5647421 6.385813 5.097941 4.128757 9.002604
## 2 4.910440 3.635769  0.22251390 0.5235894 5.855833 5.363952 2.528875 9.091389
## 3 4.780466 3.972245 -0.43239870 1.2447386 6.808926 5.094280 2.828394 9.505245
## 4 5.174201 3.766206  0.39812841 0.8498109 6.385062 4.880685 3.125678 8.997497
## 5 5.005041 3.748032  0.06569693 1.5858134 6.544912 5.456545 2.921149 9.342738
## 6 4.932705 3.760431 -1.31140330 1.5994793 6.894563 5.377052 2.416610 9.379518
##     253461    26112    27436     3308     3696      374739     3842      406
## 1 7.942981 7.359096 8.186901 6.677820 3.974534  0.01606445 7.485297 6.407828
## 2 7.645493 6.867246 8.349231 6.850738 2.442948  1.12199290 7.957759 6.469674
## 3 7.286699 7.220765 7.976890 6.692954 3.555573  0.41212940 8.382312 7.013921
## 4 7.084021 7.133832 7.862947 6.876650 2.248458 -0.04328272 8.262895 6.547131
## 5 7.259189 6.632053 8.169747 6.651758 2.569326  0.59573306 8.236147 6.827497
## 6 7.084326 6.889783 7.889045 6.558667 2.551350 -0.62334670 8.187849 6.763165
##      56986    57188     6239     7067     7871    79961    79991       8224
## 1 4.569967 6.489402 6.598030 3.670566 6.934592 8.254305 4.400996 -0.5576978
## 2 4.304797 6.619162 6.017377 3.310633 6.588355 7.263601 5.696222 -0.2464944
## 3 4.708971 7.338757 5.447805 3.491882 6.951830 7.601995 4.768126 -0.2062002
## 4 4.688851 7.425033 5.671156 2.981478 6.783600 6.726483 5.838237  0.6333935
## 5 4.954366 7.243932 6.084683 2.806273 7.085988 7.254145 5.430470 -0.4349200
## 6 4.295874 7.374586 6.215701 3.221056 6.992236 6.964783 5.568724 -0.6725092
##       8853      8870     9258     9686
## 1 3.154888 -1.311403 4.773540 4.970460
## 2 2.826145 -1.311403 5.986476 4.557343
## 3 1.269788  2.489060 5.301694 5.048249
## 4 2.739033 -1.311403 5.541685 4.123486
## 5 1.818676 -1.311403 5.127274 5.622021
## 6 2.126144  2.444345 5.272971 4.725351
```

```
head(phenodat)
```

```
##   currentsmoking packyrs  age gender race FEV1perc
## 2              0    72.0 59.8      2    2     61.8
## 3              0    24.0 75.5      2    1     89.0
## 5              0    35.8 62.5      1    1     98.8
```

```
## 6              0    35.0 78.5      1    1     98.9
## 8              0    30.0 54.1      1    1     89.6
## 9              0    46.0 58.1      1    1     99.2
```

Ok! we are done with the preparation. Let's apply the PRANA to the COPDGene study data.

Apply the PRANA

The main variable of our interest in this analysis is the current smoking status. We obtain the indices of subjects who are 'current' vs. 'non-current smokers.' These indices are used to dichotomize expression dataset into 'current (Group B)' and 'non-current smokers (Group A).' This is important as the we estimate the group-specific $p \times p$ association matrices. This remarks the Step 1 of the main code provided.

```
##############################################################################################
#  STEP 1. Estimate an association matrix via ARACNE from the RNA-seq expression data.
##############################################################################################
# Indices for non-current smoker (namely Group A)
newindex_A = which(combinedCOPDdat_RGO$currentsmoking == 0)
# Indices for current smoker (namely Group B)
newindex_B = which(combinedCOPDdat_RGO$currentsmoking == 1)
# Expression data for Group A using indices above.
rnaseqdatA = rnaseqdat[newindex_A, ]
# Expression data for Group B using indices above.
rnaseqdatB = rnaseqdat[newindex_B, ]

# Estimate an association matrix for Group A
nw_est_grpA = est_method(rnaseqdatA, verbose = F)
# Estimate an association matrix for Group B
nw_est_grpB = est_method(rnaseqdatB, verbose = F)

n_A <- length(newindex_A) # Sample size for Group A
n_B <- length(newindex_B) # Sample size for Group B
```

Next, the Step 2 is to calculate the column sum of the estimated association matrix to obtain the total connectivity for each gene. Notationally, it correponds to $\hat{\theta}_k$ shown in our paper.

Do you notice `thetahats` function? This is a function, called from `TotalConnectivity.R`. Please do not forget to download from the repository and load it appropriately!

```
##############################################################################################
#  STEP 2. Calculate total connectivity by taking the column sum of the association matrix
# to obtain the total connectivity for each gene.
##############################################################################################
thetahat_grpA = thetahats(nw_est_grpA)
thetahat_grpB = thetahats(nw_est_grpB)
```

The Step 3 is the re-estimation part of our method. Please be aware that this may take some time. For each gene, the re-estimation process requires n such calculations with the data size of $n - 1$.

`mclapply` is a parallelized version of `lapply`. The number of cores can be adjusted by specifying `mc.cores` option in the `mclapply` function.

```
##############################################################################################
#  STEP 3. Re-estimate association matrix using the expression data without i-th subject.
#          Then, calculate total connectivity from the reestimated association matrix.
##############################################################################################
# Re-estimation part
```

```
nw_est_drop_grpA <- mclapply(newindex_A, function(j) est_method(rnaseqdatA[-j, ], verbose = F))
nw_est_drop_grpB <- mclapply(newindex_B, function(j) est_method(rnaseqdatB[-j, ], verbose = F))
```

Below is to calculate $\hat{\theta}_{k(i)}$, the column sum of a gene calculated from the re-estimated association matrix using the expression data without the ith subject.

```
# Group-specific total connectivity for each gene.
thetahat_drop_grpA <- sapply(nw_est_drop_grpA, thetahats)
thetahat_drop_grpB <- sapply(nw_est_drop_grpB, thetahats)
```

Upon the completion of $\hat{\theta}_k$ and $\hat{\theta}_{k(i)}$ from Step 2 and 3, we move onto the Step 4 to calculate jackknife pseudo-values, denoted as $\tilde{\theta}_{ik}$.

The input for `thetatilde` function requires $\hat{\theta}_k$, $\hat{\theta}_k$, and the sample size for each groups.

```
#############################################################################################
#  STEP 4. Calculate the jackknife pseudo-values.
#############################################################################################
thetatildefun <- function(thetahatinput, thetahatdropinput, sizegroup) {
        thetatildeout = matrix(NA, ncol=length(thetahatinput), nrow=sizegroup)
        thetatildeout = sapply(1:nrow(thetahatdropinput), function(k) {
                sizegroup * thetahatinput[k] - (sizegroup - 1) * thetahatdropinput[k, ]
        })
        return(thetatildeout)
}

# Use thetatilde function to calculate pseudo-values for each groups.
thetatilde_grpA = thetatildefun(thetahat_grpA, thetahat_drop_grpA, n_A)
thetatilde_grpB = thetatildefun(thetahat_grpB, thetahat_drop_grpB, n_B)
thetatilde = rbind(thetatilde_grpA, thetatilde_grpB)
colnames(thetatilde) = colnames(rnaseqdat) # Map the column names (gene names)
```

As this time, we have all the ingredients for the main dish. That is, a regression is fitted to regress the pseudo-values on a set of covariates. In this example, we have used smoking pack years, age, gender, race, and FEV1 as additional covariates. As a reminder, the binary current smoking status variable is used as the main grouping variable.

```
#############################################################################################
#  STEP 5. Fit a robust regression model
#############################################################################################
pseudo.beta_list <- lapply(1:ncol(thetatilde), function(i) {
        m <- thetatilde[, i]
        df <- data.frame(phenodat,
                         m = m)
        fit <- ltsReg(m ~ currentsmoking + packyrs + age + gender + race + FEV1perc,
                      data = df, mcd=FALSE) # include a set of covariates in this model
        return(fit)
})
```

`pseudo.beta_list` is a function that stores the results for each gene. The for-loop below is to extract the p-values (and coefficient estimates $\hat{\beta}$ in case of reporting) from each fitted model.

```
### Obtain p-values (and beta coefficients) from model:
beta_hat = vector(mode = "list", ncol(thetatilde))
p_values = vector(mode = "list", ncol(thetatilde))
k = NULL
for(k in 1:ncol(thetatilde)) {
```

```
        # The beta coefficients for each model:
        beta_hat[[k]] <- summary(pseudo.beta_list[[k]])$coef[-1, "Estimate"]
        # P-values for each model:
        p_values[[k]] <- summary(pseudo.beta_list[[k]])$coef[-1, "Pr(>|t|)"]

}

# Convert list into data.frame
beta_hat = as.data.frame(bind_rows(beta_hat))
# Map the gene names to the data.frame for betahats
rownames(beta_hat) <- colnames(rnaseqdat)

# Convert list into data.frame
p_values = as.data.frame(bind_rows(p_values))
# Map the gene names to the data.frame for p-values
rownames(p_values) <- colnames(rnaseqdat)
```

Again, our main interest is in the current smoking status to declare whether a gene is differentially connected (DC) between current and non-current smokers at the user-specified significance level. Thus, we subset the vector of p-values for current smoking status from the `p_values` data.frame.

Then, the adjusted p-values are computed via the empirical Bayes approach (`EBS()` function below). Please make sure you loaded `EmpiricalBayes_Datta_2005.R` in the Preparation stage earlier, if you face any error message.

```
# p-values for current smoking status (binary group variable)
current_smoke_pval = p_values[, 1]

# Compute the adjusted p-values via empirical Bayes approach.
# NOTE: EBS() is code from Datta S and Datta S (2005).
adjp_values = EBS(pvo = current_smoke_pval, alpha = 0.05, B = 500, h = 1)
# Map the gene IDs/names to the data.frame for adj p-values
names(adjp_values) <- colnames(rnaseqdat)
```

Lastly, return the gene IDs (`sigDCpseudo`) of the significantly DC genes from PRANA at the 0.05 significance level.

```
sigDCpseudo = adjp_values[which(adjp_values < 0.05)]
names(sigDCpseudo)
```

```
##  [1] "10370"  "10420"  "155185" "1653"   "1762"   "23389"  "253461" "27436"
##  [9] "3308"   "3696"   "374739" "3842"   "406"    "56986"  "57188"  "7067"
## [17] "7871"   "79961"  "79991"  "8224"   "8853"   "8870"   "9258"
```

References

[1] Ahn, S., Grimes, T., Datta, S. (2022). A pseudo-value regression approach for differential network analysis of co-expression data. Under review at *BMC Bioinformatics*.

[2] Datta, S. and Datta, S. (2005). Empirical Bayes screening of many p-values with applications to microarray studies. *Bioinformatics*, 21(9), 1987–1994.

[3] Sakornsakolpat, P., Prokopenko, D., Lamontagne, M., and et al. (2019). Genetic landscape of chronic obstructive pulmonary disease identifies heterogeneous cell-type and phenotype associations. *Nature Genetics*, 51(3), 494–505.

[4] Wang, Z., Masoomi, A., Xu, Z., Boueiz, A., Lee, S., Zhao, T., Bowler, R., Cho, M., Silverman, E., Hersh, C., Dy, J., and Castaldi, P. (2021). Improved prediction of smoking status via isoform-aware RNA-seq deep learning models. *PLoS Computational Biology*, 17(10).