Using the zinck package

Soham Ghosh

2024-04-10

About zinck

Microbiome data is high-dimensional, compositional, and zero-inflated which makes false discovery rate (FDR)-controlled taxonomic variable selection difficult. We propose **zinck**, a novel knockoff-based framework tailored for microbiome data exploiting a flexible generative model. It can properly capture the zero-inflation and complex dependence structure among microbes, enjoying the property of simultaneous variable selection and FDR control. Its potential efficacy is illustrated using various simulated and real data experiments.

The hierarchical structure of zinck

Before going into the structure of zinck, we define the following parameters:

1. Number of biological samples: D, number of microbial features: p, number of latent clusters K and the sequencing depth of the d^{th} sample is given by N_d, where $d = 1, \dots, D$.
2. $\theta_d = (\theta_{d1}, \dots, \theta_{dK})'$ is the vector of cluster mixing probabilities for the d^{th} biological sample, where $d \in 1, 2, \dots, D$
3. $\beta_k = (\beta_{k1}, \dots, \beta_{k(p-1)})'$ and $\beta_{kp} = 1 - \sum_{i=1}^{p-1} \beta_{ki}$ is the vector of feature proportions for each cluster $k = 1, 2, \dots, K$.
4. $\mathbf{z}_d = \left(z_{d1}, \dots, z_{dN_d}\right)'$ is the vector of cluster assignments for the d^{th} biological sample. For instance, $z_{dn} = k$ implies that the n^{th} sequencing read in the d^{th} sample belongs to the k^{th} cluster.
5. $\mathbf{w}_d = \left(w_{d1}, \dots, w_{dN_d}\right)'$ is the vector of the features drawn for each sequencing read for the d^{th} biological sample.
6. $\alpha = (\alpha, \dots, \alpha)^{K \times 1}$: symmetric hyperparameter of the Dirichlet prior of θ_d.
7. $\pi_k = (\pi_{k1}, \dots, \pi_{k(p-1)})'$: hyperparameter of the ZIGD distribution specifying the probability of being a structural zero for the j^{th} feature in the k^{th} subcommunity, where $j = 1, \dots, p$ and $k = 1, \dots, K$.
8. $\mathbf{a} = (a, \dots, a)^{(p-1) \times 1}$ and $\mathbf{b} = (b, \dots, b)^{(p-1) \times 1}$ are the symmetric hyperparameter vectors on the ZIGD of β_k.

It is to be noted that ZIGD here refers to the zero-inflated Generalized Dirichlet distribution.

zinck is a probabilistic hierarchical model with the following specification:

$$\begin{aligned} w_{dn} | z_{dn}, \beta_{z_{dn}} &\sim \text{Multinomial}(\beta_{z_{dn}}) \\ \beta_{z_{dn}} | \pi, \mathbf{a}, \mathbf{b} &\sim \text{ZIGD}\left(\pi_{z_{dn}}, a, b\right) \\ z_{dn} | \theta_d &\sim \text{Multinomial}(\theta_d) \\ \theta_d &\sim \text{Dirichlet}(\alpha) \end{aligned}$$

We denote the elements of the observed sample taxa matrix $\mathbf{X}^{D\times p}$ by (x_{dj}) as the observed read count of the j^{th} taxon for the d^{th} subject:

$$x_{dj} = \sum_{n=1}^{N_d} \mathbb{1}_{\{w_{dn}=j\}}$$

Generating knockoffs using zinck

We can exploit the knockoff generative model to learn the structure of the sample-feature count matrix $\mathbf{X}$ and generate a valid knockoff copy, which is further used for FDR-controlled feature selection. We fit the augmented LDA model to the microbiome count data matrix $\mathbf{X}$. The latent parameters of the model namely, θ_d and β_k are learnt by approximating their joint posterior distribution via the Automatic Differentiation Variational Inference (ADVI) algorithm or by drawing MCMC samples using a Collapsed Gibbs sampler. We use the learnt parameters $\tilde{\theta}_d, \tilde{\beta}_k$, for $d = 1, 2, \dots D$ and $k = 1, 2, \dots, K$ to generate a knockoff copy. For each sample d, we first sample a cluster allocation z_{dn} from Multinomial$(1, \tilde{\theta}_d)$ and then we sample a feature w_{dn} from the selected cluster z_{dn}, that is, $w_{dn} \sim$ Multinomial$(1, \tilde{\beta}_{z_{dn}})$. Finally, we form the knockoff matrix $\tilde{\mathbf{X}}^{D\times p} = \{\tilde{x}_{dj}\}$ by cumulating all the taxa read counts per subjects as illustrated previously.

We demonstrate the ability of zinck to capture the compositional and highly sparse nature of microbiome count data by comparing the heatmaps of the original sample taxa matrix $\mathbf{X}$ with its high quality knockoff copy, $\tilde{\mathbf{X}}$.

We consider a toy setting with 20 samples and 30 taxa with the library size between 400 and 500 for each sample, for a high zero-inflation level of 0.8. We use the `simulateZINLDA()` function from the `zinLDA` package to generate the original sample taxa matrix.

```r
set.seed(1) ##reproducibility

N.d=zinLDA::rdu(n=20,min=400,max=500) # Drawing random library sizes between 400, 500

sim_data = zinLDA::simulateZINLDA(D=20,V=30,N=N.d,K=5,Alpha=0.1,Pi=0.8,a=0.5,b=10)

X_original <- sim_data$sampleTaxaMatrix ## The original sample taxa count matrix
```

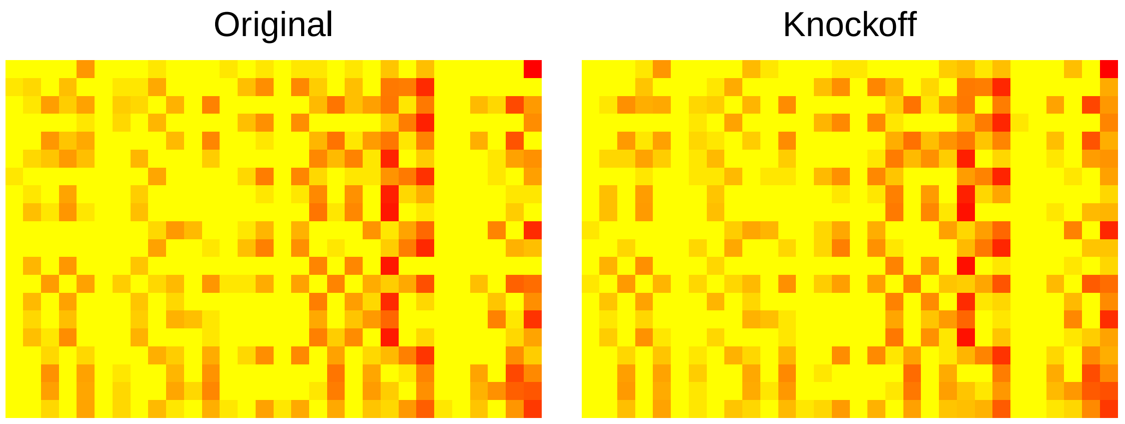
Now, we learn the compositional, zero-inflated structure of this generated matrix using the `zinck.fit()` function from the `zinck` package. Then, we will use the posterior estimates of the latent parameters to generate a knockoff copy for the original matrix. The knockoff generation is achieved by utilizing the function `generateKnockoff()`.

```r
model_zinck <- fit.zinck(X_original, num_clusters=5, method="Gibbs", seed=1)

Theta <- model_zinck$theta

Beta <- model_zinck$beta

X_zinck <- generateKnockoff(X_original,Theta,Beta,seed=1)

rownames(X_zinck) <- rownames(X_original)
```

We will now visualize the heatmaps of the original matrix and its corresponding knockoff copy. The function applies an arcsinh transformation to the data for normalization and better visualization of abundance patterns and zero inflation within the sample taxa matrix.

```r
heat1 <- draw_heatmap(X_original, "Original")

heat2 <- draw_heatmap(X_zinck, "Knockoff")
```

```r
plot_grid(heat1, heat2, ncol = 2, align="v")
```



It is evident from the above heatmaps that the knockoff copy is almost indistinguishable from the original matrix! This underscores the fact that the knockoff copy preserves the underlying structure of the observed sample taxa count matrix.

The complete zinck workflow

In this section, we will illustrate through an example the property of FDR controlled variable selection for `zinck`. Let us consider the species level colorectal cancer data-set (CRC) with the most abundant 300 species. There are a total of 574 subjects in the study out of which we randomly select 270 subjects. Thus, the truncated matrix $\mathbf{X}^{270\times 300}$ is taken to be our sample taxa matrix.

```r
load("count.Rdata")

dcount <- count[,order(decreasing=T,colSums(count,na.rm=T),
                       apply(count,2L,paste,collapse=''))][,1:300]
## ordering the columns w/ decreasing abundance

## Random Subject Selection
set.seed(1)

sel_index <- rbinom(nrow(dcount),size=1,prob=0.5)

selected_samples <- which(sel_index==1)

X <- dcount[selected_samples,]
```

Now, we inject 30 non-zero signals randomly among the top 200 most abundant species.

```r
Five1 <- c(-3,3,2.5,-1,-1.5)

Five2 <- c(3,3,-2,-2,-2)

Five3 <- c(1,-1,3,-2,-1)

Five4 <- c(-1,1,2,-1,-1)

Five5 <- c(3,3,-3,-2,-1)

Five6 <- c(-1,1,-2,1,1)
```

```r
Five_all <- c(Five1,Five2,Five3,Five4,Five5,Five6)

randBeta <- rep(0,300)

set.seed(1)

rand_indices <- sample(1:200,size=30,replace=FALSE)

set.seed(1)

randBeta[rand_indices] <- sample(Five_all, size=30, replace=FALSE)
```

We simulate a continuous response vector $\mathbf{Y}^{D\times 1}$ using the symmetric reformulation of the log-contrast model, which is the commonly-used regression model for microbiome covariate:

$$g(E(y_i)) = \sum_{j=1}^{p} \log(\bar{X}_{ij})\beta_j \quad \text{subject to} \quad \sum_{j=1}^{p} \beta_j = 0$$

where $g(\cdot)$ is the link function and $\bar{X}_{ij}$ is the observed proportions based on the count matrix with zero replaced by 0.5 to avoid taking log on zero values. See that the signal strengths are taken in such a way that the sum zero constraint is satisfied.

```r
n = nrow(X)

W <- log_normalize(X)

set.seed(1)

eps=rnorm(n,mean = 0, sd=1)

Y <- W %*% randBeta + eps
```

Now, we learn the latent zinck parameters by fitting the augmented LDA model on $\mathbf{X}$. We use the posterior estimates of θ and β.

```r
species_fit <- fit.zinck(X,num_clusters = 6,method = "ADVI", seed=123, alpha_param = 0.1)

theta <- species_fit$theta

beta <- species_fit$beta
```

We use the posterior estimates of these parameters to generate the knockoff copy.

```r
X_tilde <- generateKnockoff(X,theta,beta,seed=1)
```

Now that we have generated the knockoff copy, we will fit a model associating the response $\mathbf{Y}$ and the augmented set of covariates $[\mathbf{X}, \tilde{\mathbf{X}}]^{D\times 2p}$. Here, we assume that the data generating mechanism is known, that is the response is linearly related to the covariates. Thus, we fit a glmnet model and compute the importance statistics for each feature. Then, at a target FDR level of 0.1, we detect the non-zero features.

```r
W_tilde <- log_normalize(X_tilde)

selected_species = zinck.filter(W,W_tilde,Y,model="glmnet",fdr=0.1,offset=1)
##
## Selected variables:
```

```
##  [1]   7  14  21  33  34  37  43  51  68  70  73  74  79  84  85  89 105 106 110
## [20] 126 129 162 163 165 167 172 182 187 188 190 199 255
```

We can evaluate the performance of our filter by comparing with the true non-zero signals.

```
index <- rand_indices

index_est <- selected_species

neg_index <- (1:300)[-c(index)]

if(length(index_est)==0){
  neg_index_est <- 1:300
} else{
  neg_index_est <- (1:300)[-c(index_est)]
}

# ### Evaluation metrics ###
TP <- sum(index_est %in% index==TRUE) ## True Positives
FP <- sum(index_est %in% index==FALSE) ## False Positives
FN <- sum(index %in% index_est == FALSE) ## False Negatives
TN <- sum(neg_index_est %in% neg_index == TRUE) ## True Negatives

# ## False Discovery Proportion ##
estimated_FDR <- FP/(FP+TP)
print(estimated_FDR)
## [1] 0.0625

## Power ##
estimated_power <- TP/(TP+FN)
print(estimated_power)
## [1] 1
```

Real Data Analysis

Now, let us assume that we do not know the data generating mechanism. Also, to demonstrate the robustness of our model towards the type of response variable, let us take binary outcomes. For this analysis, we consider the genus level data from the same CRC study. Again, we have $n = 574$ patients and $p = 133$ distinct genera. The response variable $\mathbf{Y}$ is binary, which is taken to be the CRC case(1)-control(0) status.

```
load("count.genus.RData")
load("meta.RData")

X <- count.genus[,order(decreasing = T,colSums(count.genus,na.rm=T),apply(count.genus,2L,paste,collapse

# ## Case-control status (Response) ##

Y_cat <- as.factor(meta$Group)
lookup <- c("CTR"=0,"CRC"=1)
Y <- lookup[Y_cat]
```

Following the same workflow, let us first generate the knockoff copy for $\mathbf{X}$.

```
fit_CRC <- fit.zinck(X,num_clusters = 8, method="ADVI",tuned=FALSE,seed=11, alpha_param = 0.1)
```

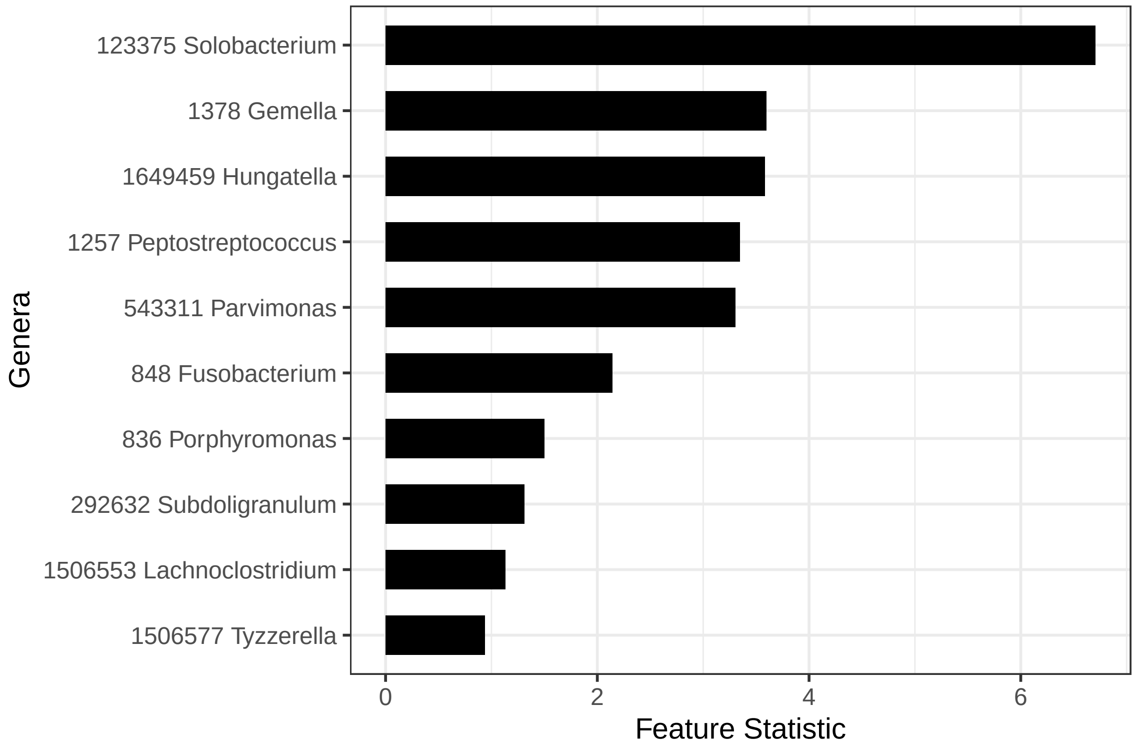
```r
theta_CRC <- fit_CRC$theta

beta_CRC <- fit_CRC$beta

X_tilde_CRC <- generateKnockoff(X,theta_CRC,beta_CRC,seed=1)
```

Since the data generating mechanism is unknown, we consider fitting a tree-based model like Random Forest since, it can naturally include interactions between the microbial taxa and also allows the filter to adapt to non-linear relationships between **Y** and **X**. The selected genera at a target FDR level of 0.1 are displayed below.

```r
selected_genera = zinck.filter(X,X_tilde_CRC,Y,model="Random Forest",fdr=0.1,offset=1,seed=11)
##
## Selected variables:
##  [1]   3   9  19  38  41  43  44  51  52  66  84  87  94 106
```

We can visualize the importance of these selected genera using the Feature Statistics obtained by contrasting the original and the knockoff features.

```r
set.seed(1)
W <- stat.random_forest(X,X_tilde_CRC,Y)
T <- knockoff.threshold(W,fdr=0.1,offset=1)

names <- colnames(X[,which(W>=T)])
data.genus <- data.frame(impscores=sort(W[which(W>=T)],decreasing = FALSE),
                         name=factor(names, levels=names), y=seq(length(names))*0.9)

plot.genus <- ggplot(data.genus)+geom_col(aes(impscores,name),
                                          fill="black",width=0.6)+theme_bw()+
  ylab("Genera")+xlab("Feature Statistic")

plot.genus
```

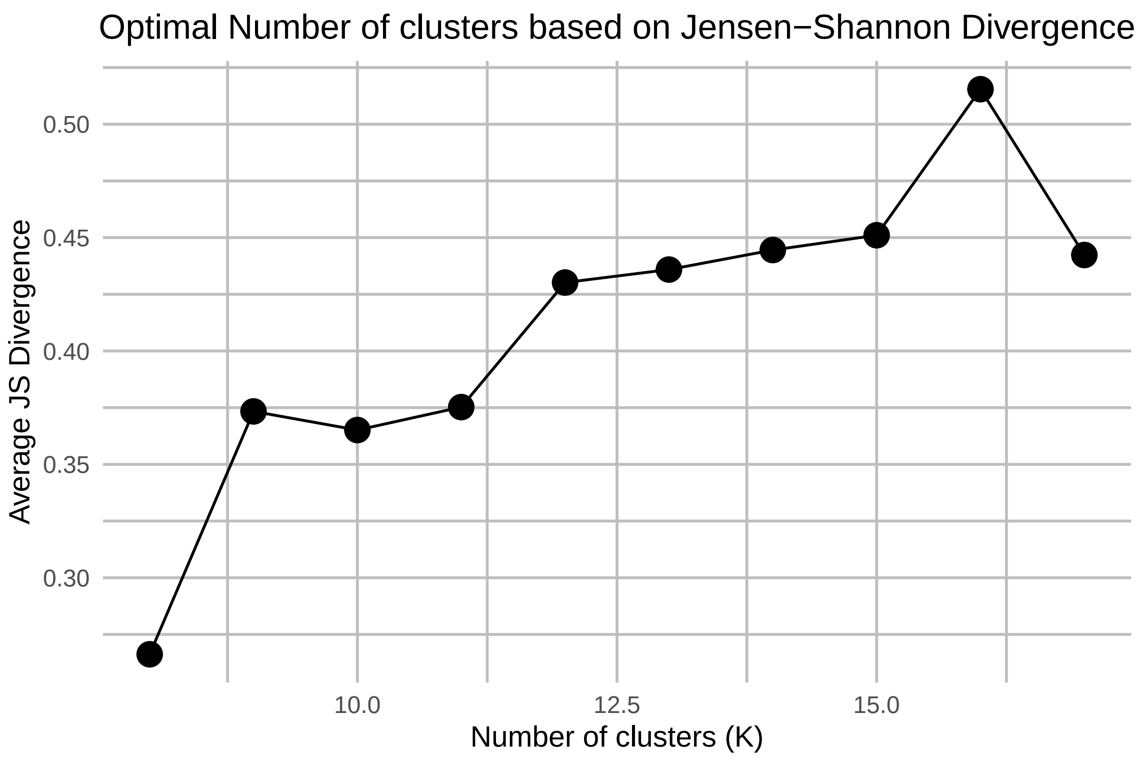


Choice of k?

An integral part of the zinck workflow is to choose an optimal value for k (number of clusters). We use the Jensen-Shannon (JS) divergence criterion for choosing the optimal number of clusters k for zinck. We can acheive this by fitting the zinck model for a range of k values. Finally, we choose the k that maximizes the average JS divergence. This signifies that for that model, the clusters are, on average, the most distinct from each other.

Here is an illustration for the CRC genus level data-set with $n = 574$ patients and $p = 133$ genera. We vary k from 8 to 17 and obtain the optimal k which maximizes the JS divergence.

```
load("count.genus.RData")
dcount <- count.genus[,order(decreasing=T,colSums(count.genus,na.rm=T), apply(count.genus,2L,paste,colla
## ordering the columns w/ decreasing abundance
X <- dcount
## Run this command to find the optimal K value ##
optimal_k(X,kmin=8,kmax=17,seed_list=list(1,11,1,1,1,1,1,1,1,1))
## Running /Library/Frameworks/R.framework/Resources/bin/R CMD SHLIB foo.c
## clang -mmacosx-version-min=10.13 -I"/Library/Frameworks/R.framework/Resources/include" -DNDEBUG   -I
## In file included from <built-in>:1:
## In file included from /Library/Frameworks/R.framework/Versions/4.1/Resources/library/StanHeaders/inc
## In file included from /Library/Frameworks/R.framework/Versions/4.1/Resources/library/RcppEigen/inclu
## In file included from /Library/Frameworks/R.framework/Versions/4.1/Resources/library/RcppEigen/inclu
## /Library/Frameworks/R.framework/Versions/4.1/Resources/library/RcppEigen/include/Eigen/src/Core/util
## namespace Eigen {
## ^
## /Library/Frameworks/R.framework/Versions/4.1/Resources/library/RcppEigen/include/Eigen/src/Core/util
## namespace Eigen {
##                ^
##                ;
```



```
## [1] 16
```

It is clear from the Divergence plot that k=16 is optimal for fitting the zinck model to the CRC genus level data.

References

1. Latent dirichlet allocation Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003).
2. Zero-inflated generalized Dirichlet multinomial regression model for microbiome compositional data analysis Tang, Z. Z., & Chen, G. (2019).
3. Automatic Differentiation Variational Inference Kucukelbir, A., Tran, D., Ranganath, R., Gelman, A., & Blei, D. (2017).]
4. A Zero-Inflated Latent Dirichlet Allocation Model for Microbiome Studies Deek, R., and Li, H. (2021).