Vignette for *PANR* : Posterior association network and enriched functional gene modules inferred from rich phenotypes of gene perturbations

Xin Wang, Roland F. Schwarz, Mauro Castro
Klaas W. Mulder and Florian Markowetz

May 3, 2016

Contents

1 Introduction

An important goal of systems biology is to understand how genes act in concert with each other as functional modules to control a biological function or get involved in a biological process. Large-scale gene silencing coupled with rich phenotypic screening paves the road towards a systematic understanding of gene functions as well as their interactions. Notably, regardless of particular screening technology and phenotypes selected (e.g. biochemical marker [**?**], cell morphologies [**?**], tissue architectures [**?**]), associations such as correlation coefficients between phenotypic profiles of genes are widely used to investigate their functional interactions [**?**, **?**, **?**].

However, predicting an association network has two big challenges: (a) how to make a statistically meaningful cutoff to select significant interactions from the densely connected network; (b) how to incorporate prior knowledge about functional interactions. In this paper, we propose an efficient Bayesian mixture modelling approach to address these two challenges. We quantify the statistical significance of functional connections between genes

by performing a beta-mixture modelling of gene associations. We employ a stratification strategy to take into consideration potential prior knowledge for the functional network such as protein-protein interactions.

To fit the beta-mixture model to screening data, we perform MAP (maximum *a posteriori*) inference based on the EM algorithm [**?**]. The algorithm alternates between computing the expectation of the log-posterior probability based on the current estimates for the latent variables and maximizing the expected log-posterior. Having estimated the parameters in the beta-mixture model, we compute posterior probabilities for each gene pair belonging to the positive, negative or lack of association component. To perform a model selection for each edge, a signal-to-noise ratio (SNR), the posterior odds in favor of signal (association) to noise (lack of association), is computed.

All bioinformatic steps to infer a PAN are realised in the *PANR* package, which includes essentially two S4 classes for beta-mixture modelling and network inference, respectively. The rest of the viegnette is arranged in the following order:

- To begin, we take a brief overview of the package. We introduce the workflow to infer a posterior association network and gene modules starting from rich phenotyping screens.
- Second, we introduce the methods and algorithm for *PANR* .
- Third, we conduct a case study using the data set from Bakal et al. 2007 [**?**] to demonstrate how to use the package step by step.
- Finally, we demonstrate another application on the data set from Mulder 2011 [**?**].

2 An overview of PANR

The workflow of *PANR* is illustrated in Figure **??**.

PANR takes as input various types of rich phenotyping screens such as gene expressions, cell morphologies, etc. following gene silencing experiments [**?**, **?**, **?**]. The input data is supposed to be preprocessed and summarized (e.g. by using *cellHTS2* or other user-defined methods such as [**?**]). The preprocessed data should be a numeric matrix with rows and columns corresponding to genes and phenotypes (or conditions).

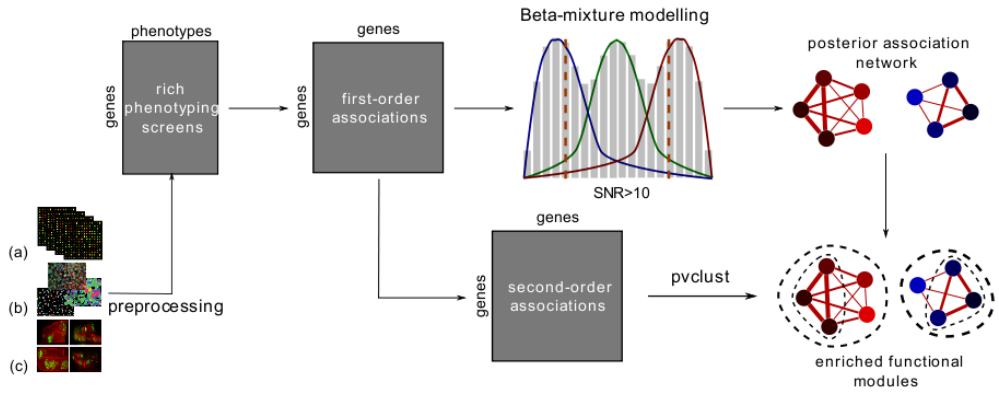

Figure 1: **The workflow of PANR .** *PANR* can be applied to a variety of phenotyping screens such as gene expression(a[**?**]), cell morphologies(b[**?**]), tissue architectures(c[**?**]), etc.

Quantifying gene associations A conventional way to quantify the functional association between two genes is to compute the similarity between their phenotypes based on correlation coefficients (e.g. [**?**]). *PANR* prefers the uncentered correlation coefficient (also known as cosine similarity), which was described as a function that considers both magnitude and direction [**?**]. It has been proven to be a highly desirable metric for exploring gene expression patterns [**?**, **?**, **?**]. Thus, we will focus on cosine similarities throughout this manual.

Beta-mixture modelling *PANR* conducts beta-mixture modelling on association densities to quantify the significances of gene interactions. Specifically, for each pair of genes we compute a posterior probabily for it belonging to the positive ('+'), negative ('-') association or lack of association ('$\times$') component of a beta-mixture distribution. We also extend the original model to allow incorporation of possible prior knowledge of functional associations such as protein-protein interactions.

Inferring a posterior association network Following the beta-mixture modelling, we formalize a novel gene network–posterior association network (PAN) to represent co-functions between genes inferred from perturbations with respect to studied biological function or process. As the name implies,

a PAN encodes *a posteriori* beliefs of functional association types on edges between genes. To infer a PAN, we compute posterior odds for each gene pair in favor of signal (the '+' and '-' components) to noise (the '×' component). These posterior odds can be used to weigh edges of a PAN, and a cutoff score (e.g. 10) can be selected to exclude non-significant edges.

Searching for significantly enriched functional modules *PANR* performs hierarchical clustering on second-order similarities, a popular measure of gene modularity, between genes to search for enriched modules. To assess the uncertainty of the clustering analysis, *PANR* computes a p-value for each cluster using multiscale bootstrap resampling powered by *pvclust* [**?**].

As a demonstration, in this vignette, we introduce how to perform these analyses on two types of RNA interference screens [**?**, **?**]. For the other types of phenotyping screens, the users can design their own classes, methods and pipelines very easily based on this package to infer PANs.

3 Methods

3.1 Quantifying the significance of gene associations

3.1.1 Measuring gene associations

We will focus on cosine similarities throughout this vignette, although other centered correlation coefficients can also be used potentially.

Let $\mathbf{X} = [x_{ik}]_{i=1,2,\cdots,n;k=1,2,\cdots,r}$ be a matrix of measured phenotypes, in which n and r denote the number of genes and replicates in the experiment, respectively. The cosine similarity here between gene i and j is their normalized dot product, namely:

$$c_{ij} = \frac{\mathbf{x}_i \cdot \mathbf{x}_j}{\|\mathbf{x}_i\|\|\mathbf{x}_j\|} \tag{1}$$

A cosine similarity ranges from -1 (exactly opposite) to 1 (exactly the same) with 0 indicating independence. The biological meaning for a positive or negative cosine similarity is that two genes are positively or negatively regulated, affected or functionally related, depending on the type of phenotype measured.

Having quantified the associations between genes, a natural and critical question is how to tell the significance. The most conventional solution is to simulate a null distribution of associations using permutations, which is used to perform hypothesis tests with the alternative hypothesis that the studied gene association is significantly different from the null. A p-value can be computed to assess the statistical significance for each pair of genes being functionally associated. Alternatively, we address this challenge by a Bayesian mixture modelling approach. An advantage over permutation tests, as we will demonstrate later in this paper, is that it allows incorporation of *a priori* beliefs on edges.

3.1.2 Bayesian mixture modelling

Finite mixture models have been used to identify co-expressed genes from gene expression data [**?**]. An efficient methodology was proposed by Ji et al., which models densities of correlation coefficients of gene expression levels by a mixture of a finite number of beta distributions [**?**]. Here, we propose a beta-mixture distribution to model associations of phenotypic readouts of loss-of-function genetic screens to predict functional connections between genes.

Our motivation for the model came from the observation that the distribution of cosine similarities computed from RNAi screens exhibits three peaks: one at the central and the other two at both tails (Figure **??**). A natural and simple way to model the distribution is to employ a mixture of three beta distributions. The biological interpretations for the three mixture components are that they represent positive association (+), negative association (-) and lack of association (x), respectively.

In the following paragraphs, we describe in details the global beta-mixture model, the stratified model for prior incorporation, Bayesian regularization, posterior probabilities and model inference by maximum *a posteriori* estimation.

Likelihood function For simplicity, we denote the set of association scores (e.g. cosine similarities) as $\mathbf{a}^{'} = \{a_u^{'} : u = 1, 2, \cdots, \binom{n}{2}\}$. To fit the range of beta-distributions, we use linearly transformed scores $\mathbf{a} = \{a_u : u = 1, 2, \cdots, \binom{n}{2}\}$, in which $a_u = (a_u^{'} + 1)/2$.

We assume that a_u follow a mixture of three beta distributions, namely:

$$a_u \sim \sum_{m \in \mathbf{M}} \pi_m f_m(a_u|\alpha_m, \beta_m), \quad \mathbf{M} = \{`+', `-', `\times'\} , \tag{2}$$

where $f(a_u|\alpha_m, \beta_m)$ is a beta density function with α_m and β_m as shape parameters.

Let $\mathbf{Z} = [\mathbf{z}_u]_{u=1,2,\cdots,n}$ be a matrix of hidden data, where $\mathbf{z}_u = [z_{um}]_{m \in \mathbf{M}}$ is a vector of latent indicator variables for gene u, in which:

$$z_{um} = \begin{cases} 1 & \text{if } a_u \text{ comes from component } m \\ 0 & \text{otherwise} \end{cases} . \tag{3}$$

$\mathbf{z}_u$ is independent and identically distributed according to an three-category multinomial distribution with probabilities $\boldsymbol{\pi} = [\pi_m]_{m \in \mathbf{M}}$.

The likelihood of the sets of parameter $\boldsymbol{\theta}$ and $\boldsymbol{\pi}$ given the complete data $\mathbf{a}$ and $\mathbf{Z}$ is:

$$\begin{aligned} L(\boldsymbol{\pi}, \boldsymbol{\theta}; \mathbf{a}, \mathbf{Z}) &= P(\mathbf{a}, \mathbf{Z}|\boldsymbol{\pi}, \boldsymbol{\theta}) \\ &= \prod_{u=1}^{n} P(a_u, \mathbf{z}_u|\boldsymbol{\pi}, \boldsymbol{\theta}) \\ &= \prod_{u=1}^{n} \prod_{m \in \mathbf{M}} [\pi_m f_m(a_u|\alpha_m, \beta_m)]^{z_{um}} \end{aligned} \tag{4}$$

The logarithm of the above likelihood is:

$$l(\boldsymbol{\pi}, \boldsymbol{\theta}; \mathbf{a}, \mathbf{Z}) = \sum_{u=1}^{n} \sum_{m \in \mathbf{M}} z_{um} [\log \pi_m + \log f_m(a_u|\alpha_m, \beta_m)] \quad . \tag{5}$$

Based on the log-likelihood function, Ji et al. proposed an Expectation-Maximization (EM) algorithm [**?**] to estimate parameters [**?**].

Incorporating prior knowledge We demonstrated in our manuscript [**?**] that gene pairs with evidences of protein-protein interactions in the nucleus tend to have higher functional associations. However, such prior information is ignored in the above global mixture model, which treats every association equally multinomially distributed with the same parameters. Inspired by the

stratified Gaussian mixture model proposed by Pan et al. for clustering of microarray data [**?**], we extend to a stratified beta mixture model, in which the full set of associations $\mathbf{a}$ is partitioned to disjoint subsets $\{\mathbf{a}_k\}_{k=1,2,\cdots,d}$ (e.g. subsets of associations with and without PPIs). Consequently, the stratified probability density function becomes:

$$f_{(k)}(a_u; \mathbf{\Pi}, \boldsymbol{\theta}) = \sum_{m \in \mathbf{M}} \pi_{km} f_m(a_u | \alpha_m, \beta_m) \ , \tag{6}$$

in which $m \in \mathbf{M}$ specifies the mixture component and $\mathbf{\Pi} = [\pi_{km}]_{k=1,2,\cdots,d, m \in \mathbf{M}}$ denotes the set of mixture coefficients affiliated with different partition sets.

Correspondingly, we derive the extended log-likelihood:

$$l(\mathbf{\Pi}, \boldsymbol{\theta} | \mathbf{a}, \mathbf{Z}) = \sum_{u=1}^{n} \sum_{m \in \mathbf{M}} z_{um} (\log \pi_{km} + \log f_m(a_u | \alpha_m, \beta_m)) \ . \tag{7}$$

Bayesian regularization To obtain smoother estimates of the parameters and guide the selection of model structures, we perform Bayesian regularization for the mixture model by introducing dirichlet priors for the likelihood:

$$\begin{aligned} P(\mathbf{\Pi} | \mathbf{\Gamma}^*) &= \prod_{k=1}^{d} Dir(\boldsymbol{\pi}_k | \boldsymbol{\gamma}_k^*) \\ &\propto \prod_{k=1}^{d} \prod_{m \in \mathbf{M}} {\pi_{km}^*}^{\gamma_{km}^* - 1} \ , \end{aligned} \tag{8}$$

where $\mathbf{\Gamma}^* = [\gamma_{km}^*]_{k=1,2,\cdots,d, m \in \mathbf{M}}$ is a matrix of hyperparameters for the dirichlet prior with each row corresponding to a stratum and each column to a mixture component. The posterior probability can be written as:

$$
\begin{aligned}
P(\mathbf{\Pi}, \boldsymbol{\theta}, \mathbf{\Gamma}^* | \mathbf{a}, \mathbf{Z}) &\propto P(\mathbf{a}, \mathbf{Z} | \mathbf{\Pi}, \boldsymbol{\theta}, \mathbf{\Gamma}^*) P(\mathbf{\Pi} | \mathbf{\Gamma}^*) \\
&= \prod_{k=1}^{d} P(\mathbf{a}_k | \mathbf{Z}_k, \boldsymbol{\theta}) P(\mathbf{Z}_k | \boldsymbol{\pi}_k) P(\boldsymbol{\pi}_k | \mathbf{\Gamma}^*) \\
&\propto \prod_{k=1}^{d} \{ \prod_{v \in \mathbf{a}_k} \prod_{m \in \mathbf{M}} f_m^{z_{vm}}(a_v | \alpha_m, \beta_m) \cdot \\
&\quad \prod_{m \in \mathbf{M}} \pi_{km}^{\sum_{v \in \mathbf{a}_k} z_{vm} + \gamma^*_{km} - 1} \} \ . \qquad (9)
\end{aligned}
$$

The corresponding log-posterior probability is:

$$
\begin{aligned}
&\log P(\mathbf{\Pi}, \boldsymbol{\theta}, \mathbf{\Gamma}^* | \mathbf{a}, \mathbf{Z}) = \\
&\quad \sum_{k=1}^{d} \sum_{m \in \mathbf{M}} \{ \sum_{v \in \mathbf{a}_k} z_{vm} \log f_m(a_v | \alpha_m, \beta_m) + (\sum_{v \in \mathbf{a}_k} z_{vm} + \gamma^*_{km} - 1) \log \pi_{km} \} \qquad (10)
\end{aligned}
$$

For a dirichlet prior distribution $Dir(\boldsymbol{\gamma})$, to specify the hyperparameters we adopt the following decomposition:

$$
\boldsymbol{\gamma} = \gamma_0 \cdot \boldsymbol{p}, \qquad (11)
$$

where $\boldsymbol{p}$ is a prior distribution normalized to 1 specifying the prior beliefs towards different mixture components and γ_0 is a scale parameter specifying the strength of prior beliefs.

Posterior probability Having estimated the paramters in the beta-mixture model, the posterior probability for association $a_v \in \mathbf{a}_k, k \in \{0, 1\}$ belonging to the '+', '-' or '×' mixture component can be computed by:

$$
P(z_{vm} = 1 | a_v, \mathbf{\Pi}, \boldsymbol{\theta}, \mathbf{\Gamma}^*) \propto \pi_{km} f_m(a_v | \alpha_m, \beta_m) \ . \qquad (12)
$$

Maximum *a posteriori* (MAP) inference We propose to perform MAP estimation using a similar EM algirithm as Ji et al., which alternates between computing the expectation of the log-posterior probability based on the current estimates for the latent variables and maximizing the expected log-posterior:

- **E**xpectation**-step**: Given currently estimated parameters and latent variables, the expected value of the log-posterior probability is:

$$\begin{aligned} Q(\mathbf{\Pi}, \boldsymbol{\theta}|\mathbf{\Pi}^{(t)}, \boldsymbol{\theta}^{(t)}) &= E_{\mathbf{Z}|\mathbf{a},\mathbf{\Pi}^{(t)},\boldsymbol{\theta}^{(t)}} \log P(\mathbf{\Pi}, \boldsymbol{\theta}, \mathbf{\Gamma}^*|\mathbf{a}, \mathbf{Z}) \\ &= \sum_{k=1}^{d} \sum_{m \in \mathbf{M}} \{ \sum_{v \in \mathbf{a}_k} z_{vm}^{(t)} \log f_m(a_v|\alpha_m, \beta_m) + \\ &\quad (\sum_{v \in \mathbf{a}_k} z_{vm}^{(t)} + \gamma_{km}^* - 1) \log \pi_{km} \} \quad , \end{aligned} \tag{13}$$

  where for association $v \in \mathbf{a}_k$:

$$z_{vm}^{(t)} = \frac{\pi_{km}^{(t)} f_m(a_v|\alpha_m^{(t)}, \beta_m^{(t)})}{\sum_{m \in \mathbf{M}} \pi_{km}^{(t)} f_m(a_v|\alpha_m^{(t)}, \beta_m^{(t)})} \quad . \tag{14}$$

- **M**aximization**-step**: Update the estimates for parameter $\mathbf{\Pi}$ and $\boldsymbol{\theta}$ to optimize the expected value in Eq. (**??**). Derived from the partial derivatives of the Q function with respect to the mixture coefficients, the updating function is obtained as follows:

$$\pi_{km}^{(t+1)} = \frac{\sum_{v \in \mathbf{a}_k} z_{vm}^{(t)} + \gamma_{km}^* - 1}{|\mathbf{a}_k| + \sum_{m' \in \mathbf{M}} (\gamma_{km'}^* - 1)} \quad , \tag{15}$$

  where $|\mathbf{a}_k|$ is the length of $\mathbf{a}_k$. When $\boldsymbol{\gamma}_k^*$ is uniformly distributed for $k = 1, 2, \cdots, d$, the MAP estimation degenerates to ML estimation.

  Due to the difficulty to derive a closed-form expression to estimate the parameters of beta distributions, similar to Ji et al. [**?**] we use the 'nlm' function in R [**?**] to fit these parameters numerically.

In practice, our method differs from the global beta-mixture model proposed by Ji et al. in the following aspects:

- The global beta-mixture model proposed by Ji et al. has a challenge to determine the number of beta distributions using a model selection criterion (e.g. AIC, BIC or ICL-BIC). We deliberately apply a three-component beta-mixture model to fit association densities of perturbation screens under a very reasonable biological assumption as we discussed before.

- We fit a beta distribution to association scores computed from permuted screening data to fix the mixture component representing lack of association. This strategy can help avoid potential overfitting in the global model.
- Our extended stratified mixture model allows integration of prior knowledge such as protein-protein interactions.

3.2 Posterior association networks

3.2.1 Representation

Let $\mathcal{V} = \{g_i\}$, $i = 1, 2, \cdots, n$ be a set of genes being perturbed in an experiment, where n stands for the total number of genes. A posterior association network (PAN) $\mathcal{G}_{PAN} = (\mathcal{V}, \mathcal{E})$ is a type of gene network encoding gene functions on vertices ($\mathcal{V}$) and functional connections between genes on edges ($\mathcal{E} = \{e_{ij} : i, j \in \mathcal{V}\}$). Importantly, in PANs vertices and edges carry statistical information as well. For example, in a PAN for genetic screens, each vertex (gene perturbed) has a corresponding z-score representing its loss of function, whereas each edge encodes *a posteriori* belief on the functional association between two genes. These statistical information allows further filtering out non-significant interactions to obtain a sparse network.

3.2.2 Inference

Normally, the PAN inferred from rich phenotyping screens is very dense in edges. A further filtering step can be performed on edges to obtain a sparse network with only those significant functional associations. To perform a model selection for each edge, a posterior odds for edge a_u between gene i and j in favor of association to lack of association is computed as follows:

$$K_{ij} = K_u = \frac{P(z_{u\text{`}\times\text{'}} = 0 | a_u, \mathbf{\Pi}, \boldsymbol{\theta}, \mathbf{\Gamma}^*)}{P(z_{u\text{`}\times\text{'}} = 1 | a_u, \mathbf{\Pi}, \boldsymbol{\theta}, \mathbf{\Gamma}^*)} \quad . \tag{16}$$

A cutoff score K_0 (e.g. 3, 5 or 10) can be set to filter out non-significant edges, guided by the interpretation of Bayes factors by Harold Jeffreys [**?**]. The parse PAN obtained can be denoted as $\mathcal{G}'_{PAN} = (\mathcal{V}', \mathcal{E}')$, where $\mathcal{E}' = \{e_{ij} : K_{ij} > K_0\}$ and $\mathcal{V}' = \{g_i : g_i \in \mathcal{V}, \exists j \in V \backslash \{i\} : K_{ij} > K_0\}$

Starting from genetic perturbation screens, the following steps are involved to infer a PAN of significant functional interactions:

- Quantify the statistical significance of gene functions using conventional measures such as z-scores.
- Fit a beta-mixture model to association densities and compute posterior probabilities.
- Weigh edges by posterior odds in favor of association to lack of association, and exclude non-significant edges by selecting a cutoff.
- Determine the sign of each edge by comparing the posterior probabilities for it belonging to the mixture component representing positive and negative associations.

4 Case study I–inferring a functional association network regulating epidermal stem cell fate

We develop this model for Mulder et al. to infer an association network of functional interactions between chromatin factors [**?**]. The details about how to reproduce data and figures will be added later.

5 Case study II–inferring a functional association network regulating cell morphology

We demonstrate another application of *PANR* to predicting a functional association network regulating cell morphology in *Drosophila*. The data set we use here comes from quantitative cell morphological screening for 249 gene-overexpression or RNAi knock-down treatment conditions by Bakal et al. [**?**]. For each indivisual cell, 145 different geometric features were computed by imaging analysis, and are subsequently scored with neural networks (NNs) trained to discriminate seven reference TCs with distinctive morphologies. For each TC, NN z-scores were computed from all scored cells in this

TC (more details in [**?**]). The data we obtained after all the preprocessing steps is a matrix of NN z-scores with rows and columns corresponding to 249 TCs and seven reference TCs (the data was downloaded from http://arep.med.harvard.edu/QMS in July 2011).

First of all, we load the package and the preprocessed rich phenotypes:

```
> library(PANR)
> data(Bakal2007)

> dim(Bakal2007)

[1] 273   7
```

5.1 Beta-mixture modelling of gene association densities

As shown in Figure **??**, our first step is to quantify functional gene associations and their statistical significances. We first create an object of S4 class *BetaMixture*:

```
> bm1<-new("BetaMixture", pheno=Bakal2007, metric="cosine",
+ model="global", order=1)

> bm1

A three-component beta-mixture model:

-input
phenotype partition     model    metric
  273 x 7      None    global    cosine
```

Gene association scores are computed and stored at slot 'association' when initiating the object `bm1`. An object can also be created directly from associations when screening data is not available.

Having obtained an object, we next fit a beta distribution to permuted phenotypes to estimate shape parameters for the 'x' (lack of association) component of the mixture model:

```
> bm1<-fitNULL(bm1, nPerm=10, thetaNULL=c(alphaNULL=4, betaNULL=4),
+ sumMethod="median", permMethod="all", verbose=TRUE)
```

Please note that the S4 method *fitNULL* summarizes estimated parameters from multiple fitting results by taking the median or average values. There are two choices for the method to do permutation: to permute all or keep replicate positions. Selecting the latter option will result in a flatter NULL distribution. This option is recommended when inputted data consists of multiple replicates for each phenotype to make more conservative model selctions for association types.

The summarized shape parameters in the last step are used to fix the shape of the beta distribution modelling the 'x' component. To fit the three-beta mixture model, the user needs to input the following arguments:

- `para`: initial values for parameter estimation by the EM algorithm (see [**?**] for details) including (a) `zInit`: a [No. of gene associations] x 3 [mixture components] matrix of initial posterior probabilities; (b) `thetaInit`: a named vector of six numeric values standing for shape parameters; (c) `gamma`: a [No. of partitions] x 3 [mixture components] matrix of hyperparamers for the dirichlet priors.

  When `zInit` is set to 'NULL', initial values will be generated by assigning $\{a_u : a_u < 1/3\}$, $\{a_u : 1/3 < a_u < 2/3\}$, and $\{a_u : a_u > 2/3\}$ to the '-', 'x' and '+' component, respectively. When `gamma` is set to 'NULL', an uninformative (uniform) dirichlet prior will be automatically applied. However, when the `gamma` is not 'NULL', for each association partition hyperparameters for the corresponding dirichlet prior should be supplied. Please note that when `ctrl$fitNULL` is set to 'TRUE', then `para$alphaNULL` and `para$betaNULL` are supposed to be filled in the estimated parameters from the fitting of the 'x' component.

- `ctrl`: control arguments for the EM algorithm including (a) `fitNULL`: specifying whether or not the 'x' component needs to be fitted; (b) `tol`: the tolerence of the EM algorithm; (c) `maxIter`: the maximal number of iterations.

- since *PANR* uses function *nlm* to estimate shape parameters of beta distributions, the other arguments of *nlm* are also allowed to be inputted to control the mixture model fitting.

For the case study, we fit the mixture model using the following code:

```
> bm1<-fitBM(bm1, para=list(zInit=NULL, thetaInit=c(alphaNeg=2,
+ betaNeg=4, alphaNULL=bm1@result$fitNULL$thetaNULL[["alphaNULL"]],
```

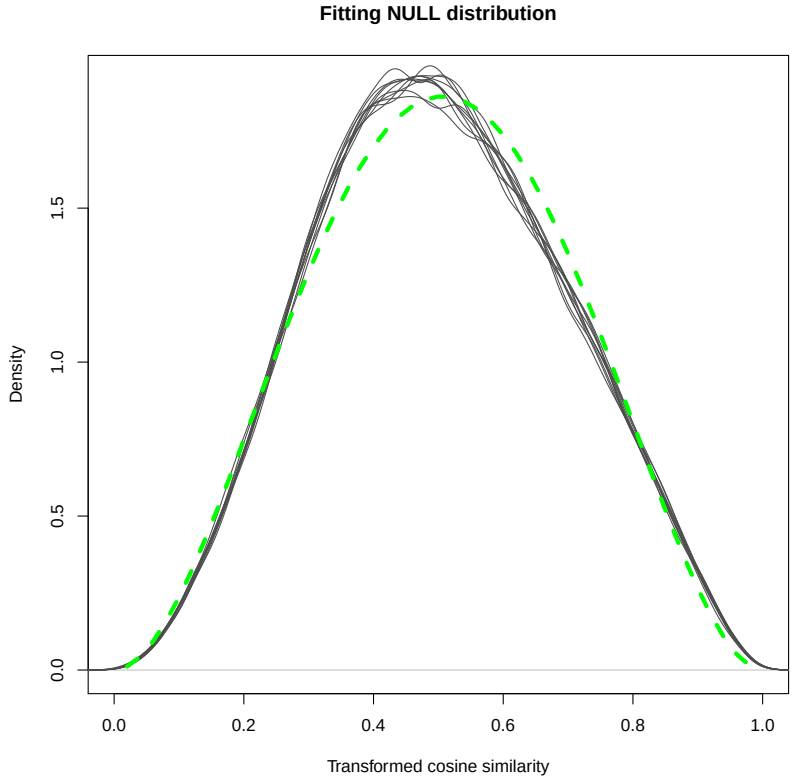

Figure 2: Fitting of the distribution representing lack of association to permuted data.

```
+ betaNULL=bm1@result$fitNULL$thetaNULL[["betaNULL"]],
+ alphaPos=4, betaPos=2), gamma=NULL),
+ ctrl=list(fitNULL=FALSE, tol=1e-1), verbose=TRUE, gradtol=1e-3)
```

PANR provides a function to visualize the fitting results so that the user can investigate if or not there is a good fitting of the model to their data.

To view the fitting of the lack of association component:

```
> view(bm1, "fitNULL")
```

As shown in Fig. **??**, we obtain a good fitting of a beta distribution (green dashed curve of beta probability density) to association densities of the permuted data (gray density curves).

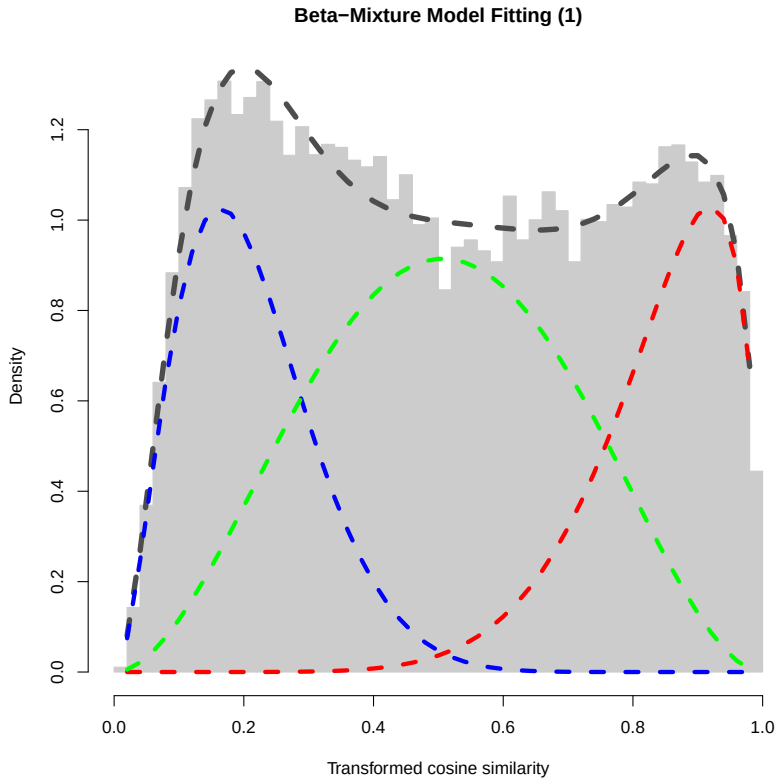

Figure 3: Fitting of the beta-mixture model.

Similarly, the same function can be used to inspect the fitting of the beta-mixture model:

```
> view(bm1, "fitBM")
```

We also see a very good fitting of three mixture components (blue, green and red dashed curve corresponds to the '-', 'x' and '+' distribution, respectively) to the association densities of real phenotyping screens (Fig. **??**).

PANR provides an S4 method *summarize* to print a summary of an object of class *BetaMixture* including input data and parameters, NULL fitting and beta-mixture model fitting results:

```
> summarize(bm1, what="ALL")
```

```
-input
phenotype partition     model    metric
  273 x 7      None    global    cosine

-NULL fitting
           No. of perm       summarize method     permutation method
                    10                  median                     all
shape1 (start->fitted) shape2 (start->fitted)
              4->2.990               4->2.938

-Beta-Mixture model fitting
--parameters:
                 z (init)            shapes(- init)
                     None        shape1=2, shape2=4
           shapes(x init)            shapes(+ init)
shape1=2.99, shape2=2.990        shape1=4, shape2=2
                    gamma
                     None
--control arguments:
     fitNULL    tolerance maxIteration
       FALSE          0.1          Inf
--results:
           shapes(- fitted)            shapes(x fitted)
shape1=3.031, shape2=11.404  shape1=2.990, shape2=2.938
           shapes(+ fitted)                 pi (fitted)
 shape1=8.429, shape2=1.650          0.260,0.491,0.249
                        NLL
                  16749.304
```

5.2 Inferring a posterior association network

Having finished the beta-mixture modelling of association densities of rich phenotyping screens, we obtain for each pair of genes a posterior probability belonging to the mixture components representing positive, negative association or lack of association. These posterior probabilities enable us to weigh edges for a *PANR* using:

- simply posterior probabilities for associations belonging to the positive

component;

- posterior odds in favor of positive association to lack of association or
- signal-to-noise ratios defined in Eq. (**??**);

Here we use the last method to weigh edges of a *PANR* and make a cutoff at 5 (meaning a 'substantial' strength of evidence according to the interpretation principles of Bayesian decision making [**?**]) to remove non-significant edges:

```
> pan<-new("PAN", bm1=bm1)
> pan<-infer(pan, para=list(type="SNR", log=TRUE, sign=TRUE,
+ cutoff=log(5)), filter=FALSE, verbose=TRUE)
```

The S4 method *infer* has an argument `filter` to filter out genes that have not any significant association with all the others.

PANR provides a method *buildPAN* to build an `igraph` object with a flexible framework for setting graph attributes. There are two graphics engines provided by *PANR* : *igraph* and *RedeR*. The latter software is more powerful for viewing a complex network or multiple modules at the same time.

```
> data(Bakal2007Cluster)
> pan<-buildPAN(pan, engine="RedeR",
+ para=list(nodeColor=nodeColor, hideNeg=TRUE))
```

To view the built graph or modules, we can use the method *viewPAN*:

```
> library(RedeR)
> viewPAN(pan, what="graph")
```

In Figure **??**, we highlighted genes (TCs) according to the clustering results in [**?**] (Table S8). We observe a high consistency between our functional network and their clustering results that TCs in the same cluster tend to enrich close to each other in the network.

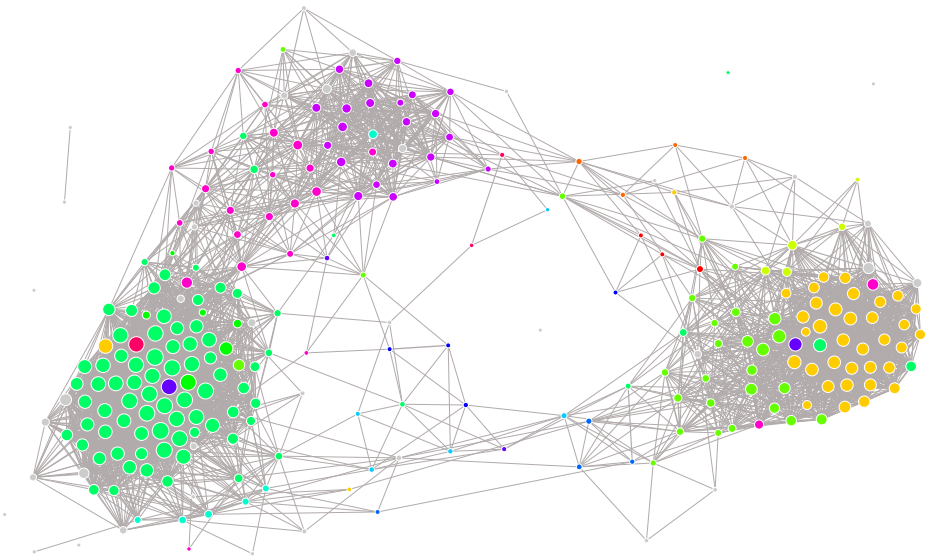

Figure 4: Inferred full posterior association network.

5.3 Searching for enriched modules

PANR performs unsupervised hierarchical clustering to search for modules. To quantify statistical significances of modules, *PANR* employs R package *pvclust* to do multiscale bootstrap resampling [**?**]. Please note that in this case study, to compare with the clustering results with [**?**] we still use first-order cosine similarities to do the clustering.

Parallel computing is supported by *PANR* powered by R package *snow*:

```
> library(pvclust)
> options(cluster=makeCluster(4, "SOCK"))
> pan<-pvclustModule(pan, nboot=1000, metric="cosine", hclustMethod=
+ "average", filter=TRUE, verbose=TRUE, r=c(5:12/7))
> if(is(getOption("cluster"), "cluster")) {
+         stopCluster(getOption("cluster"))
+         options(cluster=NULL)
+ }
```

In the above R codes, the argument `r` is appended in addition, as it is important for *pvclust* to specify the relative sample sizes of bootstrap replications (more details in *pvclust*). Similarly, the user can also append other arguments except `nboot`, `metric` and `hclustMethod` for *pvclust* to achieve the best performance. The argument `filter` is an option to exclude genes without any significant associations with all the other genes during the module searching.

A method is provided by *PANR* to retrieve ids for those significant gene modules, given a p-value cutoff, the minimal and maximal size cutoffs. The user can choose to sort modules by their sizes or p-values in an increasing or decreasing order. Using the same method *viewPAN* with the argument `what` set to 'pvclustModule', the user can view the modules by inputting their ids (Figure **??**).

```
> inds<-sigModules(pan,pValCutoff=0.01, minSize=3,
+ maxSize=100, sortby="pval", decreasing=FALSE)

> viewPAN(pan,what="pvclustModule", moduleID=inds)
```

The significant modules identified in this step can also be integrated with the inferred PAN. Again, we use *RedeR* to visualize gene modules in a hierarchical layout inside the PAN.

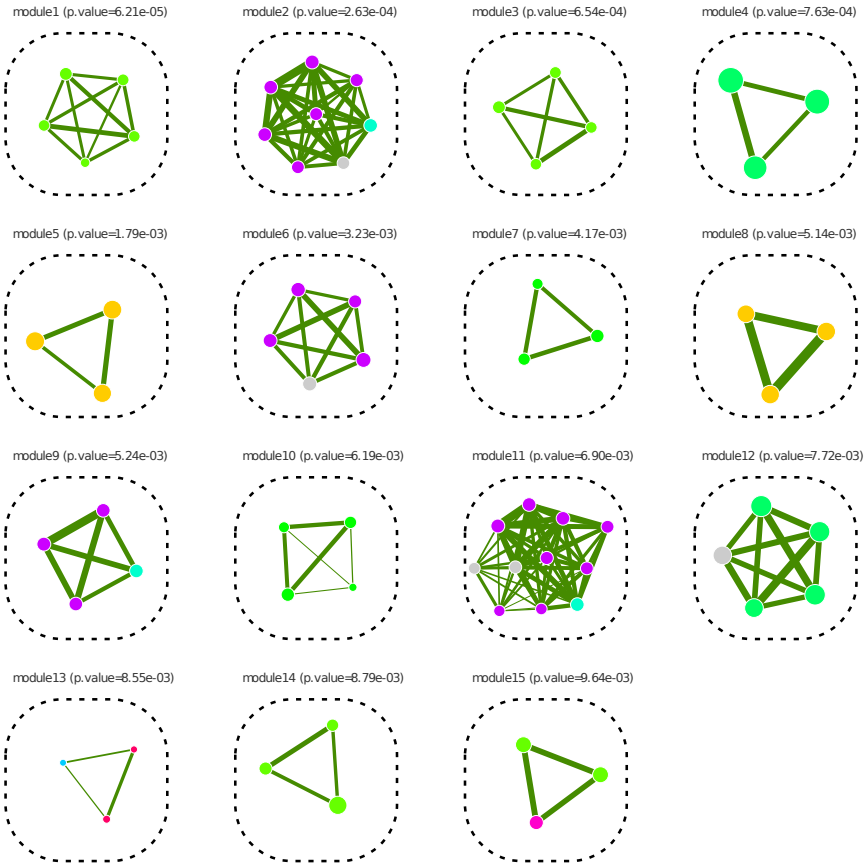

Figure 5: Top significant gene modules.

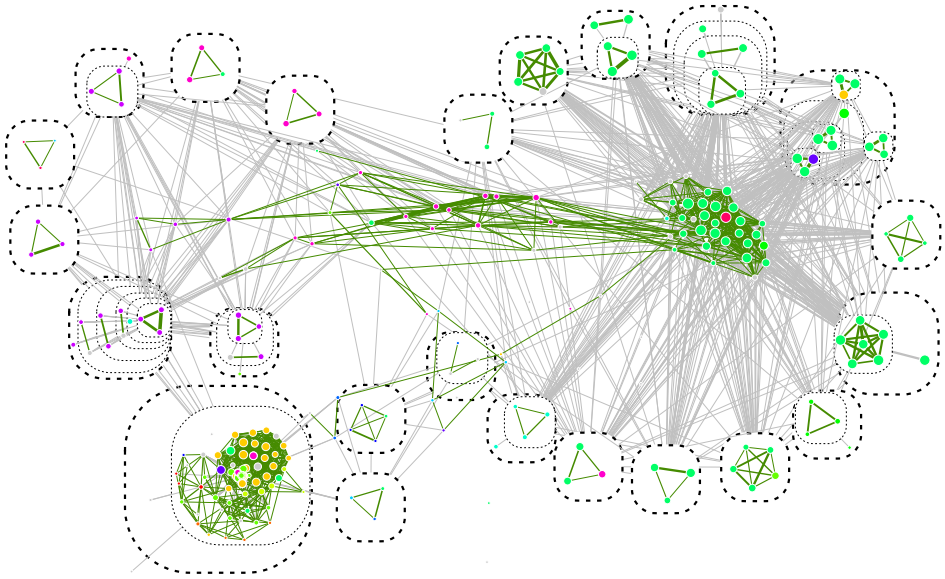

Figure 6: Significant gene modules found by *pvclust* (p-value<0.05).

```
> viewNestedModules(pan,pValCutoff=0.05,minSize=3,maxSize=100)
```

Figure **??** still presents the inferred *PANR* , but with all significant gene modules nested in containers (dashed rounded rectangles). Positive gene associations inside modules are colored in red, while inter-module associations are summed up and colored in gray. Comparing the gene modules found by *PANR* with clusters identified in [**?**], we also see a very high consistency. For example, the three big clusters responsible for cell adhesion complex formation (colored in yellow), adhesion disassembly (colored in green) and tail retraction (colored in purple and blue) are found to be highly significant modules or tightly connected submodules in our results.

For an object of class *PAN*, an S4 method is also provided to print a summary:

```
> summarize(pan, what="graph")
```

```
-graph
  edgeWeightType              log           signed edgeWeightCutoff
```

```
       SNR             TRUE             TRUE  1.6094379124341
     nodes            edges
       273             4363
```

6 Session info

This document was produced using:

```
> toLatex(sessionInfo())
```

- R version 3.3.0 RC (2016-04-26 r70550), x86_64-apple-darwin13.4.0
- Locale: C/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
- Base packages: base, datasets, grDevices, graphics, methods, stats, utils
- Other packages: PANR 1.18.0, igraph 1.0.1
- Loaded via a namespace (and not attached): MASS 7.3-45, RCurl 1.95-4.8, RedeR 1.20.0, XML 3.98-1.4, bitops 1.0-6, magrittr 1.5, pvclust 2.0-0, tools 3.3.0

7 References

[1] Bakal, C. and Aach, J. and Church, G. and Perrimon, N. (2007). Quantitative morphological signatures define local signaling networks regulating cell morphology. *Science*, **316**(5832), 1753.

[2] Ji, Y. and Wu, C. and Liu, P. and Wang, J. and Coombes, K.R. (2005). Applications of beta-mixture models in bioinformatics. *Bioinformatics*, **21**(9), 2118.

[3] Fuchs, F. and Pau, G. and Kranz, D. and Sklyar, O. and Budjan, C. and Steinbrink, S. and Horn, T. and Pedal, A. and Huber, W. and Boutros, M. (2010). Clustering phenotype populations by genome-wide RNAi and multiparametric imaging. *Molecular systems biology*, **6**(1).

[4] Suzuki, R. and Shimodaira, H. (2006). Pvclust: an R package for assessing the uncertainty in hierarchical clustering. *Bioinformatics*, **22**(12), 1540.

[5] McLachlan, G.J. and Peel, D. (2000). Finite mixture models. *Wiley-Interscience*, **299**.

[6] Dempster, A.P. and Laird, N.M. and Rubin, D.B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, **39**(1), 1–38.

[7] Lee, I. and Date, S.V. and Adai, A.T. and Marcotte, E.M. (2004). A probabilistic functional network of yeast genes. *Science*, **306**(5701), 1555.

[8] R Development Core Team (2008). R: A Language and Environment for Statistical Computing. *http://www.R-project.org*, ISBN 3-900051-07-0.

[9] Pan, W. (2006). Incorporating gene functions as priors in model-based clustering of microarray gene expression data. *Bioinformatics*, **22**(7), 795.

[10] Jeffreys, H. (1998). Theory of probability. *Oxford University Press, USA*, 432.

[11] Eisen, M.B. and Spellman, P.T. and Brown, P.O. and Botstein, D. (1998). Cluster analysis and display of genome-wide expression patterns. *Proceedings of the National Academy of Sciences*, **95**(25), 14863.

[12] Dadgostar, H. and Zarnegar, B. and Hoffmann, A. and Qin, X.F. and Truong, U. and Rao, G. and Baltimore, D. and Cheng, G. (2002). Cooperation of multiple signaling pathways in CD40-regulated gene expression in B lymphocytes. *Proceedings of the National Academy of Sciences*, **99**(3), 1497.

[13] Klaas W. Mulder, Xin Wang, Carles Escriu, Yoko Ito, Roland F. Schwarz, Jesse Gillis, Gabor Sirokmany, Giacomo Donati, Santiago Uribe-Lewis, Paul Pavlidis, Adele Murrell, Florian Markowetz and Fiona M.

Watt (2011). Diverse epigenetic strategies interact to control epidermal differentiation. *submitted*.

[14] de Hoon, M. and Imoto, S. and Miyano, S. (2002). A comparison of clustering techniques for gene expression data. in *Proc. of the 10th Intl Conf. on Intelligent Systems for Molecular Biology*.

[15] Ivanova, N. and Dobrin, R. and Lu, R. and Kotenko, I. and Levorse, J. and DeCoste, C. and Schafer, X. and Lun, Y. and Lemischka, I.R. (2006). Dissecting self-renewal in stem cells with RNA interference. *Nature*, **442**(7102), 533–538.

[16] Green, R.A. and Kao, H.L. and Audhya, A. and Arur, S. and Mayers, J.R. and Fridolfsson, H.N. and Schulman, M. and Schloissnig, S. and Niessen, S. and Laband, K. and others (2011). A High-Resolution C. elegans Essential Gene Network Based on Phenotypic Profiling of a Complex Tissue. *Cell/*, **145**(3), 470–482.

[17] Fuchs, F. and Pau, G. and Kranz, D. and Sklyar, O. and Budjan, C. and Steinbrink, S. and Horn, T. and Pedal, A. and Huber, W. and Boutros, M. (2010). Clustering phenotype populations by genome-wide RNAi and multiparametric imaging. *Molecular systems biology*, **6**(1).