`mash` No Baseline

Sarah Urbut

September 20, 2016

Contents

1 Purpose

The purpose of this document is to propose a method for extending `mash` to estimate 'true' effects across conditions in a setting in which no obvious baseline exists. We assume that we observe noisy, uncentered averages $\hat{\boldsymbol{C}}_{jr}$ in each of R conditions, and seek to estimate the underlying true 'deviations' from average measurement across conditions and can be seen as the effects in `mash`.

Here, the use of bold-face notation indicates a vector, while matrix quantities are typeset in capital but unboldface letters.

2 Defining the Model

Now, we observe for each gene j a vector of uncentered noisy average feature expression $\hat{\boldsymbol{C}}$ across R conditions:

$$\hat{\boldsymbol{C}}|\boldsymbol{C} \sim N_R(\boldsymbol{C}, \hat{V}) \tag{1}$$

where the 'true' uncentered averages $\boldsymbol{C}$ can be written as follows:

$$\boldsymbol{C}|\mu, \boldsymbol{v} = \mu\mathbf{1} + \boldsymbol{v} \tag{2}$$

Where μ is a scalar that is the mean of the 'true' uncentered averages $\boldsymbol{C}$.

$\boldsymbol{v}$ is a zero-centered mixture of multivariate normals:

$$\boldsymbol{v}|\boldsymbol{\pi}, \mathbf{U} \sim \sum_{\mathbf{k,l}} \pi_{\mathbf{k,l}}\, N_{\mathbf{R}}(\mathbf{0}, \omega_{\mathbf{l}}\mathbf{U_k}) \tag{3}$$

Critically, our quantity of interest now, $\boldsymbol{v}$ represents the true 'deviations' from average gene expression across each condition and can be seen as the effects in `mash`.

3 Applications

We will again apply a two-step process to our selection of covariance matrices, where we select a set of denoised 'pattern' matrices U_k by using the EM algorithm on the max effects across conditions, and then expanding this list by a fixed grid of scalar weights ω_l such that we conclude with a list of $P = KxL$ covariance matrices Σ. We can then:

- estimate the P prior weights $\boldsymbol{\pi}$ on this fixed P-list of covariance matrices from a training matrix of randomly selected feature expression measurements across conditions
- compute the posterior distribution $\boldsymbol{v}|L\hat{\boldsymbol{C}}, \boldsymbol{s_j}$

  Let

$$L\boldsymbol{C} = L\mu\mathbf{1} + L\boldsymbol{v} \tag{4}$$

  L is the RxR centering matrix $L_r = I_r - \frac{1}{r}\mathbf{1}\mathbf{1}^\top$ which removes the mean of each R column vector.

  Then :

$$\begin{aligned} L\boldsymbol{C} &= L\mu\mathbf{1} + L\boldsymbol{v} \\ L\boldsymbol{C} &= 0 + L\boldsymbol{v} \\ L\hat{\boldsymbol{C}} &= L\boldsymbol{v} + E \end{aligned} \tag{5}$$

  Where $E \sim \mathcal{N}(0, L\hat{V}L')$

4 Previous Model

If we were to apply the old `mash` to data simulated as in (5) in which we assumed the mean was known, we would write

$$\begin{aligned} \boldsymbol{C}|\mu, \boldsymbol{v} &= \mu\mathbf{1} + \boldsymbol{v} \\ \hat{\boldsymbol{C}} &= \mu + \boldsymbol{v} + E \\ \hat{\boldsymbol{C}} - \mu &= \boldsymbol{v} + E \end{aligned} \tag{6}$$

where $E \sim \mathcal{N}(0, V)$

5 Comparing Models

In the new model, we incorporate the centering matrix L into our selection of covariance matrices by modeling $L\hat{\boldsymbol{C}} = L\boldsymbol{v} + E$ where $E \sim N(0, LVL')$.

Using old mash, we assume that the mean centered estimates directly approximate $\boldsymbol{v}$, that is $\hat{\boldsymbol{C}} - \mu = \boldsymbol{v} + E$ where $E \sim N(0, V)$.

We want to estimate effects with no baseline, such that null effects aren't forced to be negative (by a competitive default, given that non-null effects are seen as positive, or vice versa). We hope that this method will simply shrink null effects to zero and recognize the direction of the non-null effects, instead of making both null and non-null positive (negative) relative to some intermediate average.

5.1 Selecting The Covariance Matrices

We initiate our set of covariance matrices for the denoising step as before in `mash`, where now we compute the empirical covariance matrices and a variety of dimensional reductions on the feature-centered JxR *matrix* of maximum average, $L\hat{C}'$ instead of $\hat{C}$ alone. In practice, we actually use the matrix of maximum uncentered T statistics. Three critical things to note:

1. Here, L will be RxR because we need U_k to be RxR
2. When denoising with Bovy, our previous approach used the matrix of maximum T statistics T_{Mxr} to both initialize and train the BovyEM. Now, we will initalize with the MXR matrix of t($L_{R,R}T'$) (or alternatively, TL) and train the EM on the MxR-1 matrix of maximum $t(L_{R-1,R}T')$
3. When choosing ω, we will use the diagonal of LVL', where V is $D(s.j^2)$, and select from the MxR matrix of centered T statistics and their centered standard errors.
4. We will choose the maxes as those that have a maximum centered t statistic of at least some threshold in at least one (or averaged across tissues) rather than those that satisfy an ash criteria in at least one tissue because we know that choosing uncorrelated LVL' as the standard errors with which to input to ash is incorrect.

6 Likelihood

Now we will replace the RxR matrix L with the R-1xR matrix $L*$, effectively removing a data point from the observed uncentered statistics, such that the rank of the marginal variance of w is guaranteed to be equal to the dimension of w.

Now for each gene J at each component k, integrating over $\boldsymbol{v}$,

$$
\begin{aligned}
L_{R-1,R}\boldsymbol{C} &\sim \mathcal{N}(0, L_{R-1,R}U_k L'_{R-1,R}) \\
L_{R-1,R}\hat{\boldsymbol{C}} &\sim \mathcal{N}(0, L_{R-1,R}U_k L'_{R-1,R} + L_{R-1,R}\hat{V}L'_{R-1,R})
\end{aligned} \tag{7}
$$

And thus we can use the Bovy et al algorithm invoked in both the Extreme Deconvolution package and in 'Sarah's MixEm' where according to Bovy's language.

For each gene, and $w_j = L_{R-1,R}\hat{\boldsymbol{C}}_j$.

$$
\begin{aligned}
\mathbf{w} &= L_{R-1,R}\hat{\boldsymbol{C}} \\
\mathbf{w} &\sim \mathcal{N}(0, L_{R-1,R}\boldsymbol{v}) \\
\boldsymbol{v}|\boldsymbol{\pi}, \mathbf{U} &\sim \sum_{\mathbf{k,l}} \pi_{\mathbf{k,l}}\, N_{\mathbf{R}}(\mathbf{0}, \omega_{\mathbf{l}}\mathbf{U_k})
\end{aligned} \tag{8}
$$

Recall that our previous approach was simplified by the fact that $\boldsymbol{w}_j$ was simply $\hat{\boldsymbol{b}_j}$ and the projection matrix was simply the I_r identity matrix. Our inference on $\boldsymbol{b}$ was analogous to their inference on $\boldsymbol{v}_j$.

As before, we are interested in returning the prior covariance U_k matrices of the 'true' deviations $\boldsymbol{v}$, which we will then rescale by choosing a set of ω that are appropriate to $L\hat{\boldsymbol{C}}$ to comprise a set of $P = KxL$ prior covariance matrices Σ.

and choose the set of π that maximizes compute the following likelihood at each of the P components:

$$
\begin{aligned}
L_{R-1,R}\hat{\boldsymbol{C}}_j &\sim \mathcal{N}(0, L_{R-1,R}\Sigma_p L'_{R-1,R} + L_{R-1,R}\hat{V}_j L'_{R-1,R}) \\
T_{jp} &= L_{R-1,R}U_k L'_{R-1,R} + L_{R-1,R}\hat{V}_j L'_{R-1,R} \\
L_{R-1,R}\hat{\boldsymbol{C}}_j &\sim \mathcal{N}(0, T_{jp})
\end{aligned} \tag{9}
$$

7 Posteriors

Now, as before we can compute a posterior distribution such that:

$$
\boldsymbol{v}|L_{R-1,R}\hat{\boldsymbol{C}}, \pi, \Sigma, \boldsymbol{s} \sim N(\mu^1, U^1) \tag{10}
$$

Where at each of the P components for each gene J

$$\begin{aligned} \mu_{jp}^1 &= \Sigma_p L'_{R-1,R} T_{jp}^{-1} L_{R-1,R} \hat{\boldsymbol{C}}_j \\ U_{jp}^1 &= \Sigma_p - \Sigma_p L'_{R-1,R} T_{jp}^{-1} L_{R-1,R} \Sigma_p \end{aligned} \tag{11}$$

8 Differences required over `mash` implementation

- We will now work with a matrix of observed column-centered gene averages, $L\hat{C}'$ in order to:
  1. initialize our choice of U_k;
  2. choose the maxes by which to denoise,
  3. choose our set of scales, ω_l
  4. compute our hierarchical weights, $\boldsymbol{\pi}_p$ as well as our posteriors.
- It is critical to note that here **L will need to be RxR because U_k must be RxR**
- The new distribution we seek to estimate for each j is then $v|L_{R-1,R}\hat{\boldsymbol{C}}, \boldsymbol{s}_j$
- To choose the maxes, I think we ought to use a w_j cutoff since computing the univariate lfsr on w_j and the diagonal of LVL' assumes that LVL' is diagonal when we know it cannot be.

9 Simulation

In this simulation framework, there are 1000 real associations in 10000 null across 44 tissues.

Each 'real association' is simulated in the following manner:

```
{function(n=1000,d=44,betasd=1,esd=0.1,K=10){
  library("MASS")
  library("mvtnorm")
  J=0.10*n

  configs = matrix((rnorm(d*K)),byrow=T,ncol=d) # A matrix of K classes (patterns) acr
  F=as.matrix(configs);
  covmat=lapply(seq(1:K),function(k){
```

```
  A=F[k,]%*%$t(F[k,]);
  A/max(diag(A))})
## each entry of F is the the factor of decomposition of covariance of effect sizes
z = sample(K,J,replace=TRUE) # randomly sample factor to be loaded on for each real


mus=rnorm(n)  ###generate a list of n mus
mumat=matrix(rep(mus,d),ncol=d)##generate a matrix of mus for each gene
omega=abs(rnorm(J,mean=0,sd=betasd))##effect size variance can be big or small
beta=t(sapply(seq(1:J),function(j){
  k=z[j]
  mvrnorm(1,mu=rep(0,d),Sigma=omega[j]*covmat[[k]])
  #rmvnorm(1,mean = rep(0,d),sigma=omega*covmat[[k]])
}))

beta=rbind(beta,matrix(rep(0,(n-J)*d),ncol=d))
c=beta+mumat
sj=abs(matrix(rnorm(n*d,esd,0.001),ncol=d))##use uniform to simulate 'shrunken'
e=t(apply(sj,1,function(x){rmvnorm(1,mean=rep(0,d),sigma=diag(x)^2)}))
chat=c+e
t=chat/sj
return(list(beta=beta,chat=chat,covmat=covmat,components=z,t=t,mumat=mumat,
shat=sj,error=e,ceff=c,F=F,omega=omega))}
```

Such that for every true associations a factor is chosen and 'standardized' such that the maximum value across the diagonal is one. The true effects are then simulated according to the assigned component, scaled by some factor ω, and then and this scaling is added to a chosen mean for the gene, centered at o with σ^2 of 1.

The true $ceff$ is then computed as

$$ceff = \mu + \beta$$

and

$$chat = ceff + E$$

where $E \sim N(0, V)$ and V is diagonal.

This function reports the true μ, the true β for the 1000 real genes and their associated componenent, as well as the standard error.