SpiecEasi - R package tutorial

Zachary Kurtz

April 28, 2015

Sparse InversE Covariance estimation for Ecological Association and Statistical Inference

This package will be useful to anybody who wants to infer graphical models for all sorts of compositional data, though primarily intended for microbiome relative abundance data (generated from 16S amplicon sequence data). It also includes a generator for [overdispersed, zero inflated] multivariate, correlated count data. Please see the paper preprint on arXiv (PLoS Comp Bio paper is in press).

One small point on notation: we refer to the method as "SPIEC-EASI" and reserve "SpiecEasi" for this package.

Installation

I assume that all auxillary packages are already installed - for example huge, MASS, etc. If you get an unexpected error, you may need to download and install a missing dependency.

From an interactive R session:

```
library(devtools)
install_github("zdk123/SpiecEasi")

## Downloading github repo zdk123/SpiecEasi@master
## Installing SpiecEasi
## '/usr/lib/R/bin/R' -vanilla CMD INSTALL  \
##  '/tmp/RtmpVEgSMx/devtools3d7974467a40/zdk123-SpiecEasi-cbb0170'  \
##  -library='/usr/local/lib/R/site-library' -install-tests
##
## Reloading installed SpiecEasi
##
## Attaching package:  'SpiecEasi'
##
## The following object is masked from 'package:MASS':
##
##    fitdistr

library(SpiecEasi)
```

Basic Usage

Lets simulate some multivariate data under zero-inflated negative binomial model, based on (high depth/-count) round 1 of the American gut project, with a sparse network. The basic steps are

1. load the data and normalize counts to to common scale (min depth)
2. fit count margins to the model

3. generate a synthetic network
4. generate some synthetic data
5. clr transformation
6. inverse covariance estimation
7. stability selection
8. evaluate performance

Obviously, for real data, skip steps 1-4.

```
data(amgut1.filt)
depths <- rowSums(amgut1.filt)
amgut1.filt.n <- t(apply(amgut1.filt, 1, norm_to_total))
amgut1.filt.cs <- round(amgut1.filt.n * min(depths))

d <- ncol(amgut1.filt.cs)
n <- nrow(amgut1.filt.cs)
e <- d
```

Synthesize the data

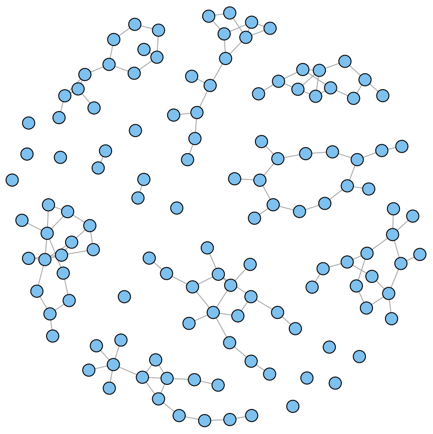
```
set.seed(10010)
graph <- make_graph('cluster', d, e)
Prec  <- graph2prec(graph)
Cor   <- cov2cor(prec2cov(Prec))

X <- synth_comm_from_counts(amgut1.filt.cs, mar=2, distr='zinegbin', Sigma=Cor, n=n)
```

the main SPIEC-EASI pipeline: Data transformation, sparse invserse covariance estimation and model selection

```
se.est <- spiec.easi(X, method='mb', lambda.min.ratio=1e-2, nlambda=15)

## Normalizing/clr transformation of data with pseudocount
## Inverse Covariance Estimation with mb ...
## Model selection with stars ...
## Done!
```

Visualize the network estimated and inferred network:

```
ig <- graph.adjacency(graph, mode='undirected')
# save layout for side-by-side plotting
g.coord <- layout.fruchterman.reingold(ig)
plot(ig, layout=g.coord, vertex.size=6, vertex.label=NA)
plot(graph.adjacency(se.est$refit, mode='undirected'),
      layout=g.coord, vertex.size=6, vertex.label=NA)
```


(a) True Graph

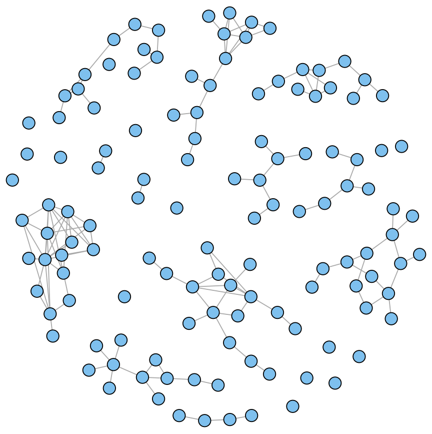
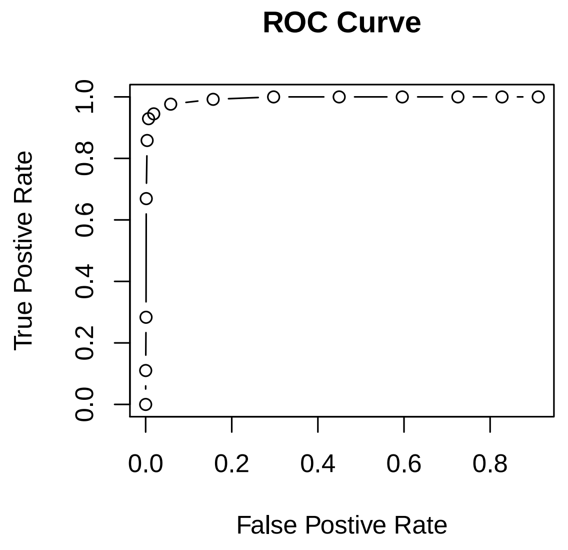

(b) S.E-MB graph

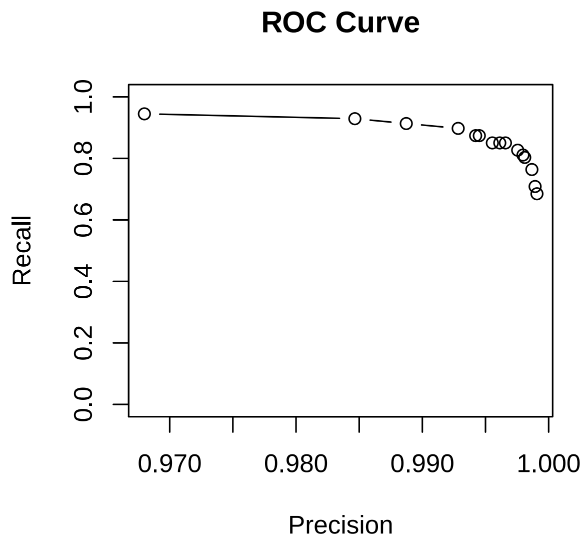
Figure 1: Network Visualization

examine ROC over lambda path and PR over the stars index for the selected graph.

```
huge::huge.roc(se.est$path, graph)

## Computing F1 scores, false positive rates and true positive rates....done.
## True Postive Rate: from 0 to 1
## False Positive Rate: from 0 to 0.9117348
## Area under Curve: 0.9054908
## Maximum F1 Score: 0.9623593

SpiecEasi:::stars.pr(se.est$merge[[se.est$opt.index]], graph, ll=15)

## Computing F1 scores, false positive rates and true positive rates....done.
## True Postive Rate: from 0 to 0.9448819
## False Positive Rate: from 0 to 0.03124206
## Area under Curve: 0.02833631
## Maximum F1 Score: 0.9562982
```



(a) ROC



(b) PR

Figure 2: Network Recovery

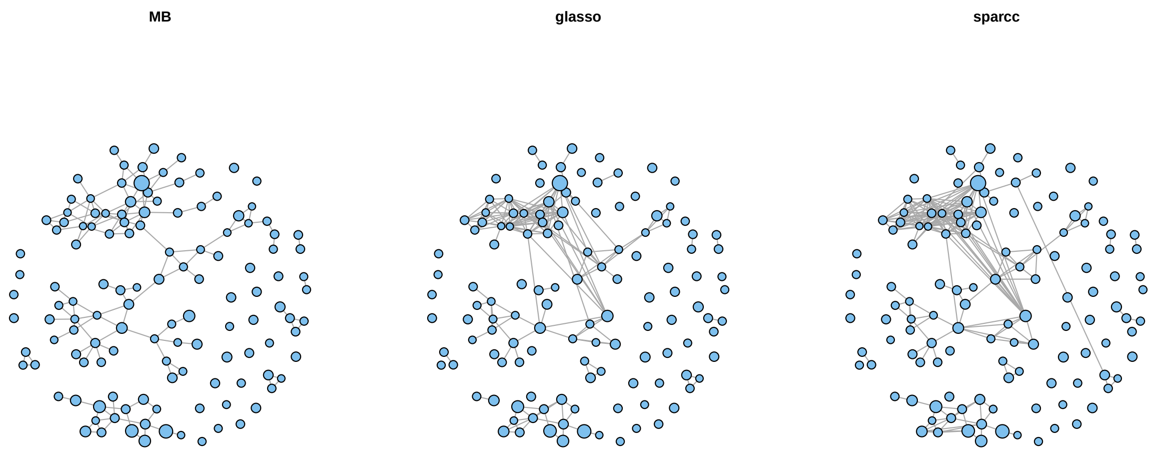
The above example does not cover all possible options and parameters. For example, other generative network models are available, the `lambda.min.ratio` (the scaling factor that determines the minimum sparsity/lambda parameter) shown here might not be right for your dataset, and its possible that you'll want more repetitions for stars selection.

Analysis of American Gut data

Now let's apply SpiecEasi directly to the American Gut data. Don't forget that the normalization is performed internally in the `spiec.easi` function. Also, we should use a larger number of stars repetitions for real data. We can pass in arguments to the inner stars selection function as a list via the parameter `icov.select.params`. If you have more than one processor available, you can also supply a number to `ncores`. Also, let's compare results from the MB and glasso methods as well as SparCC (correlation).

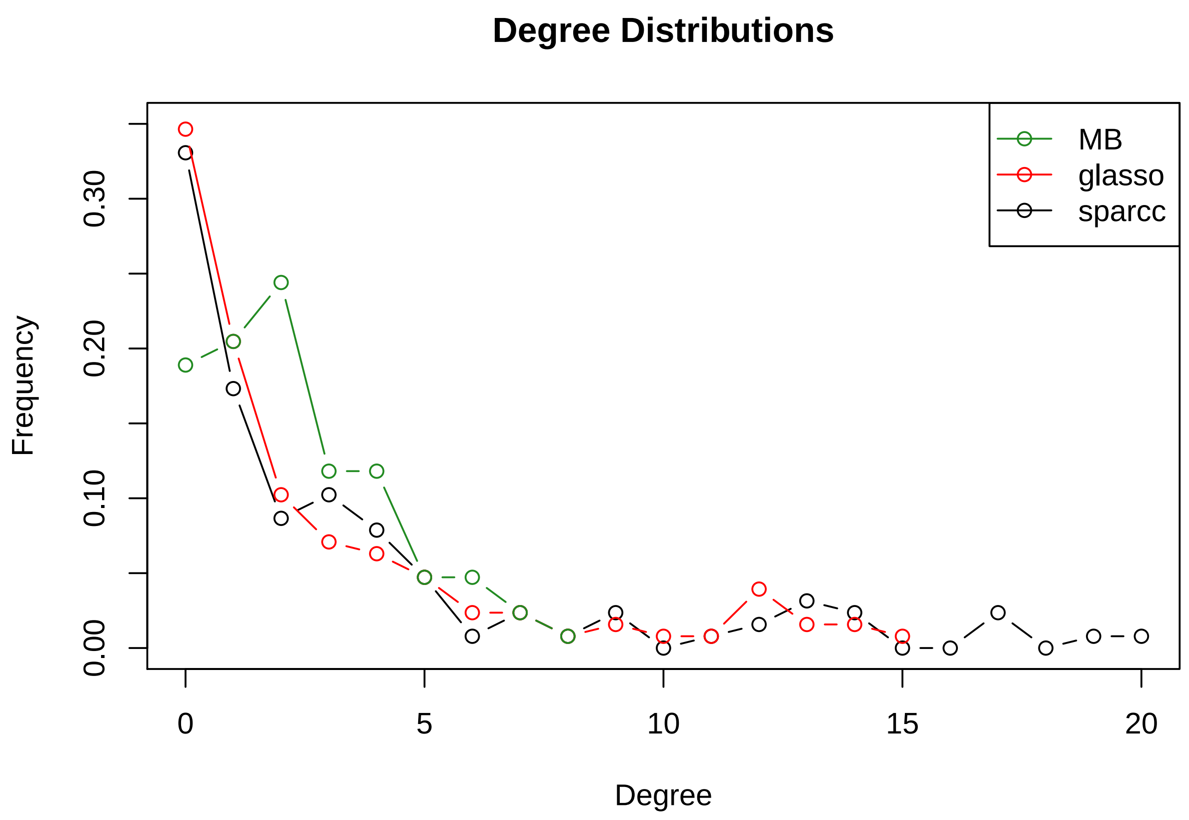
```r
se.mb.amgut <- spiec.easi(amgut1.filt, method='mb', lambda.min.ratio=1e-2,
                          nlambda=20, icov.select.params=list(rep.num=50))
se.gl.amgut <- spiec.easi(amgut1.filt, method='glasso', lambda.min.ratio=1e-2,
                          nlambda=20, icov.select.params=list(rep.num=50))
sparcc.amgut <- sparcc(amgut1.filt)
## Define arbitrary threshold for SparCC correlation matrix for the graph
sparcc.graph <- abs(sparcc.amgut$Cor) >= 0.3
## Create igraph objects
ig.mb <- graph.adjacency(se.mb.amgut$refit, mode='undirected')
ig.gl <- graph.adjacency(se.gl.amgut$refit, mode='undirected')
ig.sparcc <- graph.adjacency(sparcc.graph, mode='undirected', diag=FALSE)

## set size of vertex proportional to clr-mean
vsize <- rowMeans(clr(amgut1.filt, 1))+6
am.coord <- layout.fruchterman.reingold(ig.mb)

par(mfrow=c(1,3))
```

```
plot(ig.mb, layout=am.coord, vertex.size=vsize, vertex.label=NA, main="MB")
plot(ig.gl, layout=am.coord, vertex.size=vsize, vertex.label=NA, main="glasso")
plot(ig.sparcc, layout=am.coord, vertex.size=vsize, vertex.label=NA, main="sparcc")
```



Lets look at the degree statistics from the networks inferred by each method.

```
dd.gl <- degree.distribution(ig.gl)
dd.mb <- degree.distribution(ig.mb)
dd.sparcc <- degree.distribution(ig.sparcc)

plot(0:(length(dd.sparcc)-1), dd.sparcc, ylim=c(0,.35), type='b',
      ylab="Frequency", xlab="Degree", main="Degree Distributions")
points(0:(length(dd.gl)-1), dd.gl, col="red" , type='b')
points(0:(length(dd.mb)-1), dd.mb, col="forestgreen", type='b')
legend("topright", c("MB", "glasso", "sparcc"),
        col=c("forestgreen", "red", "black"), pch=1, lty=1)
```


Degree Distributions
Frequency
0.00
0.10
0.20
0.30
0
5
10
15
20
Degree
MB
glasso
sparcc