netresponse

Leo Lahti, Olli-Pekka Huovilainen and António Gusmão

March 18, 2011

1 Introduction

Condition-specific network activation is characteristic for cellular systems and other real-world interaction networks. If measurements of network states are available across a versatile set of conditions or time points, it becomes possible to construct a global view of network activation patterns. Different parts of the network respond to different conditions, and in different ways. Systematic, data-driven identification of these responses will help to obtain a holistic view of network activity [2, 1]. The NetResponse algorithm, implemented in this R package, provides efficient probabilistic tools for such analysis [1].

In summary, the algorithm detects local subnetworks that show particular activation patterns in a subset of conditions. The algorithm characterizes these responses, and identifies their activating conditions. The algorithm can be viewed as a type of subspace clustering where the feature space is constrained by the network. The method is based on nonparametric probabilistic modeling and variational learning, and it provides general exploratory tools for functional network analysis.

The current version provides the algorithmic implementation. The implementations are partially based on the agglomerative independent variable group analysis [3] and variational Dirichlet process mixture model algorithms [4]. Further tools for visualization and analysis will be provided in the later versions.

2 Loading the package and example data

Load the package and toy data set. The *toydata* object contains the variables D (gene expression matrix) and *netw* (network matrix). The data matrix D describes measurements of the network activation over multiple conditions. This simple toy data will be analyzed in the subsequent examples. Note that the method is potentially applicable to networks with thousands of nodes and conditions; the scalability depends on network connectivity.

```
> require(netresponse)
> data(toydata)
> D <- toydata$emat
> netw <- toydata$netw
```

3 Detecting network responses

Detect network responses across the conditions:

```
> model <- detect.responses(D, netw, verbose = TRUE)
```

```
Compute cost for each variable
Computing model for variable 1 / 10
Computing model for variable 2 / 10
Computing model for variable 3 / 10
Computing model for variable 4 / 10
Computing model for variable 5 / 10
Computing model for variable 6 / 10
Computing model for variable 7 / 10
Computing model for variable 8 / 10
Computing model for variable 9 / 10
Computing model for variable 10 / 10
done
Computing delta values for variable  1 / 10
Computing delta values for variable  2 / 10
Computing delta values for variable  3 / 10
Computing delta values for variable  4 / 10
Computing delta values for variable  5 / 10
Computing delta values for variable  6 / 10
Computing delta values for variable  7 / 10
Computing delta values for variable  8 / 10
Computing delta values for variable  9 / 10
Combining groups,  9  group(s) left...
Combining groups,  8  group(s) left...
Combining groups,  7  group(s) left...
Combining groups,  6  group(s) left...
Combining groups,  5  group(s) left...
Combining groups,  4  group(s) left...
Combining groups,  3  group(s) left...
Combining groups,  2  group(s) left...
Merging completed: no groups having links any more, or no improvement possible on level 10
```

4 Analyzing the results

Subnetwork statistics: size and number of distinct responses for each subnet

```
> stat <- model.stats(model)
> stat
```

```
         subnet.size subnet.responses
Subnet-1           7                1
Subnet-2           3                3
```

List the detected subnetworks (each is a list of nodes)

```
> get.subnets(model)
```

```
$`Subnet-2`
[1] "feat4" "feat5" "feat6"
```

Subnetworks can be filtered by size and number of responses. Subnetworks that have only one response are not informative of the differences between conditions, and typically ignored in subsequent analysis.

```
> get.subnets(model, min.size = 2, min.responses = 2)

$`Subnet-2`
[1] "feat4" "feat5" "feat6"
```

Each subnetwork response has a probabilistic association to each condition. Get the list of samples corresponding to each response (each sample is assigned to the response of the highest probability)

```
> subnet.id <- "Subnet-2"
> response2sample(model, subnet.id)
```

Retrieve model parameters of a given subnetwork (Gaussian mixture means, covariance diagonal, and component weights):

```
> pars <- get.model.parameters(model, subnet.id)
> pars

$mu
                 feat4      feat5       feat6
Response-1  0.08109344  2.9798441 -0.02807806
Response-2 -4.96945082  0.2076915  1.92109795
Response-3  4.78052024 -3.0683949 -3.17463435

$sd
               feat4     feat5     feat6
Response-1 1.0719085 1.0609530 1.0743968
Response-2 0.9217636 0.8725310 0.9572145
Response-3 1.0882094 0.9627731 0.8818674

$w
Response-1 Response-2 Response-3
 0.1950878  0.3655617  0.4393504

$nodes
[1] "feat4" "feat5" "feat6"
```

Probabilistic sample-response assignments for a given subnet is retrieved with:

```
> response.probabilities <- sample2response(model, subnet.id)
```

5 Extending the subnetworks

After identifying the locally connected subnetworks, it is possible to search for features (genes) that are similar to a given subnetwork but not directly interacting with it. To order the remaining features in the input data based on similarity with the subnetwork, type

```
> g <- find.similar.features(model, subnet.id = "Subnet-1")
> subset(g, delta < 0)
```

```
      feature.name      delta
feat5        feat5 -28.475118
feat6        feat6  -4.366608
```

This gives a data frame which indicates similarity level with the subnetwork for each feature. The smaller, the more similar. Negative values of delta indicate the presence of coordinated responses, positive values of delta indicate independent responses. The data frame is ordered such that the features are listed by decreasing similarity.

6 Variational Dirichlet process Gaussian mixture model

The package provides additional tools for nonparametric Gaussian mixture modeling based on previous work and implementations by [4, 3]. See the example in help(vdp.mixt).

7 Citing NetResponse

Please cite [1] when using the package.

8 Version information

This document was written using:

```
> sessionInfo()
```

```
R version 2.12.1 (2010-12-16)
Platform: x86_64-unknown-linux-gnu (64-bit)

locale:
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
 [3] LC_TIME=en_US.UTF-8        LC_COLLATE=C
 [5] LC_MONETARY=C              LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
 [9] LC_ADDRESS=C               LC_TELEPHONE=C
[11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C

attached base packages:
```

```
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] netresponse_1.1.23 Matrix_0.999375-46 lattice_0.19-13

loaded via a namespace (and not attached):
[1] grid_2.12.1  tools_2.12.1
```

References

[1] Leo Lahti *et~al.* (2010). Global modeling of transcriptional responses in interaction networks. *Bioinformatics.* Preprint: http://www.cis.hut.fi/lmlahti/publications/Lahti10bioinf-preprint.pdf

[2] Leo Lahti (2010). Probabilistic analysis of the human transcriptome with side information. PhD thesis. Aalto University School of Science and Technology, Department of information and Computer Science, Espoo, Finland, 2010. http://lib.tkk.fi/Diss/2010/isbn9789526033686/

[3] Antti Honkela *et~al.* (2008). Agglomerative independent variable group analysis. *Neurocomputing*, **71**, 1311–1320.

[4] Kenichi Kurihara *et~al.* (2007). Accelerated variational Dirichlet process mixtures. In B.~Schölkopf, J.~Platt, and T.~Hoffman, eds., *Advances in Neural Information Processing Systems 19*, 761–768. MIT Press, Cambridge, MA.