1 Introduction

Gene regulatory networks consist of interactions among genes, transcription factors, proteins, and other cellular substances, and are generally thought to play an important role in many cellular functions. The ebdbNet package implements the Empirical Bayes Dynamic Bayesian Network (EBDBN) method [Rau et al., 2010], which estimates the posterior distributions of network parameters. The method also has the ability to infer gene regulatory networks from time-course data while simultaneously offering the flexibility to incorporate hidden states and/or driving inputs.

1.1 Model

Consider time-course gene expression data with \( P \) genes, \( K \) hidden states (optional), \( M \) known inputs (optional), \( T \) time points, and \( R \) biological replicates. Let \( \mathbf{y}_{tr}, \mathbf{x}_{tr}, \) and \( \mathbf{u}_{tr} \) represent the expression of the sets of genes, hidden states, and known inputs, respectively, in replicate \( r \) at time \( t \). The EBDBN method is based on the following state space model:

\[
\mathbf{x}_{tr} = A \mathbf{x}_{t-1,r} + B \mathbf{u}_{t,r} + \mathbf{w}_{tr}
\]

\[
\mathbf{y}_{tr} = C \mathbf{x}_{t,r} + D \mathbf{u}_{t,r} + \mathbf{z}_{tr}
\]

where \( \mathbf{w}_{tr} \sim N(0, I) \) and \( \mathbf{z}_{tr} \sim N(0, V^{-1}) \), with \( V = \{v_1, \ldots, v_P\} \) being a \( P \)-dimensional vector of gene precisions.

To investigate the role of feedback loops in a gene regulatory network, the expression level of genes from a previous time point (i.e., feedback) can be incorporated as inputs by setting \( \mathbf{u}_{1,r} = 0 \) and \( \mathbf{u}_{t,r} = \mathbf{y}_{t-1,r} \) for \( t = 2, \ldots, T \). In the context of gene regulatory networks, the primary parameter set of interest is typically contained in the matrix \( D \), which encodes the direct gene-to-gene interactions from one time to the next (in the case of feedback networks) or the input-to-gene interactions at a given time point (in the case of input networks).

1.2 Installation

The current version is ebdbNet version 1.1. It may be downloaded from CRAN at http://cran.r-project.org/web/packages/ebdbNet/index.html or by typ-
ebdbNet requires a recent version of R (at least version 2.8.1). The current version 1.1 of ebdbNet suggests prior installation of the packages GeneNet [Schäfer and Strimmer, 2006] and Rgraphviz [Carey et al., 2005], although this is not required for ebdbNet to work. GeneNet is suggested as it contains several example datasets as well as methods which may be easily compared to the EBDBN method in ebdbNet. Rgraphviz allows the user to visualize a network in R after implementing the EBDBN method.

2 Examples

The EBDBN algorithm is made up of three principal parts: 1) choice of hidden state dimension (the hankel function), 2) estimation of hidden states and calculation of posterior distributions (the ebdbn function), and 3) determination of edge significance and graph visualization (the zcutoff and visualize functions).

To illustrate the functionality of the ebdbNet package, we will use the T-cell activation data found in the GeneNet package [Schäfer and Strimmer, 2006]. These data consist of 44 replicates of the expression of 58 genes over 10 time points. We load these data, and format them for use in the EBDBN algorithm using the dataFormat function.

```r
> library(ebdbNet)
> library(GeneNet)
> tmp <- runif(1)
> set.seed(4568818)
> data(tcell)
> tc44 <- combine.longitudinal(tcell.10, tcell.34)
> tcell.dat <- dataFormat(tc44)
```

To allow the examples to run faster, we use only the first 5 replicates of the data.

```r
> R <- 5
> tcell.dat <- tcell.dat[1:R]
```

To illustrate the full functionality of the package, we will split these data so that the first 10 genes will act as the “known transcription factors”, and the remaining 48 genes will act as observed genes.

```r
> M <- 10
> tcell.TF.dat <- lapply(tcell.dat, function(x) x[1:M, ])
> tcell.sub.dat <- lapply(tcell.dat, function(x) x[-(1:M), ])
```

Four different types of network may be inferred using ebdbNet: 1) Input networks with hidden states, 2) Feedback networks with hidden states, 3) Input networks without hidden states, and 4) Feedback networks without hidden states.
2.1 Input networks with hidden states

For input networks with hidden states as in Equation (1), the user must first apply the `hankel` function to determine the appropriate number of hidden states.

```
> K <- hankel(tcell.sub.dat)$dim
> K
[1] 1
```

The hidden state dimension is chosen to be \( K = 1 \). The default values of `hankel` specify a biological time lag of 1 and a cutoff of 90% for dimension selection. These default values may be changed as desired by the user.

The network is then inferred at a 99.9% confidence level for the z-scores of the posterior distribution:

```
> net.1 <- ebdbn(input = tcell.TF.dat, tcell.sub.dat, K)

Running EBDBN algorithm ...

Iterations:
Max difference = 0.995960, ** Iteration 1 complete! **
Max difference = 7.534270, ** Iteration 2 complete! **
Max difference = 0.999527, ** Iteration 3 complete! **
Max difference = 0.722486, ** Iteration 4 complete! **
Max difference = 0.646409, ** Iteration 5 complete! **
Max difference = 0.642751, ** Iteration 6 complete! **
Max difference = 0.582332, ** Iteration 7 complete! **
Max difference = 0.492438, ** Iteration 8 complete! **
Max difference = 0.400488, ** Iteration 9 complete! **
Max difference = 0.333247, ** Iteration 10 complete! **
Max difference = 0.407237, ** Iteration 11 complete! **
Max difference = 0.271211, ** Iteration 12 complete! **
Max difference = 0.236107, ** Iteration 13 complete! **
Max difference = 0.165694, ** Iteration 14 complete! **
Max difference = 0.188530, ** Iteration 15 complete! **
Max difference = 0.166777, ** Iteration 16 complete! **
Max difference = 0.142897, ** Iteration 17 complete! **
Max difference = 0.117227, ** Iteration 18 complete! **
Max difference = 0.090793, ** Iteration 19 complete! **
Max difference = 0.065024, ** Iteration 20 complete! **
Max difference = 0.041329, ** Iteration 21 complete! **
Max difference = 0.020793, ** Iteration 22 complete! **
Max difference = 0.007931, ** Iteration 23 complete! **
EBDBN Algorithm complete!

> z.1 <- zCutoff(net.1$DPost, net.1$DvarPost)
> sum(z.1$z99.9)
```
In this case, after 23 iterations the algorithm converges and a total of 50 TF-to-gene edges are deemed to be significant.

2.2 Feedback networks with hidden states

For feedback networks with hidden states, Equation (1) reduces to

\[\begin{align*}
    x_{tr} &= Ax_{t-1,r} + By_{t-1,r} + w_{tr} \\
    y_{tr} &= Cx_{t,r} + Dy_{t-1,r} + z_{tr}
\end{align*}\] (2)

As in the example in Section 2.1, then \texttt{hankel} function is used to select the number of hidden states.

\[ K \leftarrow \texttt{hankel(tcell.dat)}\texttt{$dim}
\]

Then the network is inferred at a 99.9\% confidence level for the z-scores of the posterior distribution as follows:

\[ net.\texttt{2} \leftarrow \texttt{ebdbn(input = "feedback", tcell.sub.dat, K, conv.3 = 0.1)} \]

Running EBDBN algorithm ...

Iterations:
Max difference = 1.927085, ** Iteration 1 complete! **
Max difference = 0.298886, ** Iteration 2 complete! **
Max difference = 0.381050, ** Iteration 3 complete! **
Max difference = 0.163693, ** Iteration 4 complete! **
Max difference = 0.164684, ** Iteration 5 complete! **
Max difference = 0.083653, ** Iteration 6 complete! **
EBDBN Algorithm complete!

\[ z.\texttt{2} \leftarrow \texttt{zCutoff(net.\texttt{2}$\texttt{DPost}, net.\texttt{2}$\texttt{DvarPost})} \]

\[ \texttt{sum(z.\texttt{2}$\texttt{z99.9})} \]

In this case, we have adjusted the convergence criterion \texttt{conv.3} (\(\Delta_3\) in the algorithm), and after 6 iterations no gene-to-gene edges are selected.

2.3 Input networks without hidden states

For input networks without hidden states, Equation (1) reduces to

\[\begin{align*}
    y_{1,r} &= z_{1,r} \\
    y_{tr} &= Du_{tr} + z_{tr}, \ t = 2, \ldots, T
\end{align*}\] (3)

To infer this type of network in \texttt{ebdbNet} using a 99.9\% confidence level for the z-scores of the posterior distribution, the following commands are used:
> net.3 <- ebdbn(input = tcell.TF.dat, tcell.sub.dat, K = 0)

Running EBDBN algorithm ...

EBDBN Algorithm complete!

> z.3 <- zCutoff(net.3$DPost, net.3$DvarPost)
> sum(z.3$z99.9)

[1] 24

In this case, a total of 24 TF-to-gene edges are identified at the chosen level of significance.

### 2.4 Feedback networks without hidden states

For feedback networks without hidden states, Equation (1) reduces to

\[
Y_{1,r} = Z_{1,r} \\
Y_{tr} = D Y_{t-1,r} + Z_{tr,t}, t = 2, \ldots, T
\]

To infer this type of network in ebdbNet using a 95% confidence level for the z-scores of the posterior distribution, the following commands are used:

> net.4 <- ebdbn(input = "feedback", tcell.sub.dat, K = 0)

Running EBDBN algorithm ...

EBDBN Algorithm complete!

> z.4 <- zCutoff(net.4$DPost, net.4$DvarPost)
> sum(z.4$z99.9)

[1] 0

In this case, as in the feedback network with 1 hidden state inferred in Section 2.2, no gene-to-gene edges are identified.

### 3 Visualizing Networks

To visualize networks inferred by ebdbNet, we suggest one of two options:

- Cytoscape, an open source bioinformatics software platform
- Rgraphviz, a Bioconductor package

The function visualize in ebdbNet puts results in the appropriate format for either visualization method. For Rgraphviz, the user must install the graph visualization software graphviz in addition to Rgraphviz. See the README file in the source distribution of Rgraphviz for more details (particularly for Windows users, who may experience some difficulties during installation).
3.1 Cytoscape

As an example, we format the results from Section 2.3 (input network with no hidden states) for visualization in Cytoscape.

```r
> cytoscape.results <- visualize(z.3$z, z.3$z99.9, format = "Cytoscape", + type = "input")
> head(cytoscape.results)
```

```
[,1] [,2] [,3]
[1,]  "TF1"  "G9" "1"
[2,]  "TF1"  "G11" "1"
[3,]  "TF2"  "G3" "1"
[4,]  "TF2"  "G6" "1"
[5,]  "TF2"  "G12" "1"
[6,]  "TF2"  "G14" "1"
```

The first two columns of this table provide node names for parent and child nodes, respectively, and the third node provides the interaction type (1 for activations, -1 for inhibitions), with one row per edge. That is, the first edge of the network is a directed edge from TF1 to G9 representing an activation, and so on.

3.2 Rgraphviz

We may also visualize the results from Section 2.3 in R using the package Rgraphviz. We begin by loading Rgraphviz:

```r
> library(Rgraphviz)
```

Next we may use the visualize function to graph the network within R:

```r
> visualize(z.3$z, z.3$z99.9, format = "Rgraphviz", type = "input")
```
References


