Joint Bayesian Inference of miRNA and Transcription Factor Activities

Benedikt Zacher, Khalid Abnaof, Stephan Gade, Erfan Younesi,
Achim Tresch, Holger Fröhlich

April 24, 2017

1 Introduction

Expression levels of mRNA molecules are regulated by different processes, comprising inhibition or activation by transcription factors (TF) and post-transcriptional degradation by microRNAs (miRNA). *birta* (Bayesian Inference of Regulation of Transcriptional Activity) uses the regulatory networks of TFs and miRNAs together with mRNA and miRNA expression data to infer switches of regulatory activity between two conditions. A Bayesian network is used to model the regulatory structure. In the model, mRNA expression levels depend on the activity states of its regulating miRNAs and TFs and the miRNA expression is dependent on the associated miRNA activity. *birta* uses Markov-Chain-Monte-Carlo (MCMC) sampling to infer these activity states, using one of the conditions as a reference. During MCMC, switch moves - toggling the state of a regulator between active and inactive - and swap moves - exchanging the activitiy states of either two miRNAs or two TFs - are used to sample from the posterior distribution. [10]
This vignette presents the application of the *birta* package in different scenarios including a simulated and a real data set. The package can be loaded by typing:

```
> library(birta)
```

2 Joint inference of transcription factor and miRNA activities

The main function of the package, `birta`, provides a flexible and easy-to-use interface to the method. Before the function is applied to an artifical and a real data set, the most important options are explained in the following. For a thorough description of all options, see the `birta` help page.

- **model**. There are two different models available to infer activity states. "all-plug-in" models mRNA and miRNA expression as gaussian distributions with (limma) estimates for its parameters from the data, whereas the "no-plug-in" model implements a fully bayesian approach, which uses gamma distributions as priors for the unknown parameters. For a detailed description and comparison of these models, see [10].
- **limmamRNA** and **limmamiRNA** are the output of the function `limmaAnalysis` for mRNA and miRNA expression data. *limma* is used to calculate differentially expressed mRNAs and miRNAs [7]. Its output is used for the initialization of the plug-in parameters of the model. The initialization of ω, describing the effect of an active regulator on mRNA expression, as well as the parameterization of the probability distributions is estimated based

on the limma output. `limmaAnalysis` is intended to give an easy-to-use interface to differential expression analysis using *limma*. However, a customized analysis can be passed to `birta` as well.

- **sample.weights**. In both models, the initial ω vector of the regulator-target graph may be sampled together with the activity states. This is realised by setting a prior probability for ω and slightly altering ω with samples from a gaussian distribution (parameters `weightSampleMean` and `weightSampleVariance`) in each iteration.

- **potential_swaps** is the output of `get_potential_swaps`. In a swap move, two TFs or two miRNAs, having different activity states, exchange these. This is especially useful for highly overlapping regulator-target graphs. If not specified, `birta` automatically calls `get_potential_swaps` to calculate all potential swaps with the default threshold of a minimal overlap of targets between regulators. However, if it needs to be pre-computed differently, it can be directly passed to `birta`.

- If **run.pretest** is *true*, miRNA and TF states are initialized with the result of a hypergeometric test in order to improve convergence. Each target gene set is tested for overrepresentation of differentially expressed genes. The corresponding regulator is set active, if the gene set shows an enrichment with a p-value < 0.05 (default). This option should only be used in case of observed convergence problems. Otherwise the inference starts with all activity states set to zero.

- **nrep** is an integer vector of length four, which specifies the number of replicates for miRNA and mRNA expression experiments: c(#miRNA-reps-condition1, #miRNA-reps-condition2, #mRNA-reps-condition1, #mRNA-reps-condition2).

2.1 Application to a simulated data set

A simulated expression data set of 1000 genes is used together with a human TF- and miRNA-target graph. The TF-target gene network was compiled by computing TF binding affinities to promoter sequences of all human genes according to the TRAP model [6] using TRANSFAC matrices. The miRNA-target graph includes miRNA-target interactions, which are either experimentally confirmed (Tarbase) [5] or predicted at least by two of the following methdos: miRanda [1], miRBase [4] and miRDB [8]. For details on the simulation and construction of the regulatory networks, see [10].
`data(humanSim)` loads the objects `genesets`, which holds the regulator-target graphs, as well as the simulated expression data. The two target networks are named lists associating each TF, resp. miRNA with its target gene sets. The expression data is stored in a matrix. In this example, there are five replicates for the "treated" case and three for the "control" case for mRNA and miRNA expression measurements.

```
> data(humanSim)
> str(head(genesets$TF))

List of 6
 $ V$AIRE_01   : chr [1:821] "10368" "51087" "10272" "22846" ...
 $ V$AP3_Q6    : chr [1:871] "10368" "51087" "10272" "22846" ...
 $ V$CEBPA_01  : chr [1:852] "10368" "51087" "10272" "93408" ...
 $ V$EN1_01    : chr [1:841] "10368" "51087" "10272" "22846" ...
 $ V$FOXJ2_02  : chr [1:858] "10368" "51087" "10272" "268" ...
 $ V$HELIOSA_01: chr [1:790] "10368" "51087" "10272" "22846" ...

> str(head(genesets$miRNA))
```

```
List of 6
 $ hsa-miR-548a-3p: chr [1:42] "114818" "1306" "145773" "169200" ...
 $ hsa-miR-766    : chr [1:15] "10071" "1087" "11007" "158747" ...
 $ hsa-miR-15a    : chr [1:26] "10611" "1264" "1399" "145773" ...
 $ hsa-miR-15b    : chr [1:30] "10611" "114757" "1264" "145773" ...
 $ hsa-miR-16     : chr [1:30] "1399" "145773" "169026" "192670" ...
 $ hsa-miR-195    : chr [1:26] "10492" "1399" "145773" "169026" ...
```

```
> head(sim$dat.mRNA)
```

```
          control.1    control.2  control.3   control.4   control.5  treated.1
114818 -0.131363521 -0.106038466 -0.3083316 -0.15966962 -0.15487331 -0.4292146
1306   -0.184995472 -0.514438355 -0.4312150 -0.34521239  0.06595186 -0.8549238
145773 -0.584408159 -0.416174064 -0.4950412 -0.08120659 -0.13051279 -2.3737003
169200 -0.012647299 -0.013087558 -0.8544657 -0.19007296 -0.15861887 -0.1464592
1823    0.009880467 -0.003646771 -0.6783295 -0.58769338 -0.02450959 -0.1580861
1982   -0.021820271  0.151964598 -0.8124917 -0.51106594 -0.15358637  0.3049217
         treated.2  treated.3     treated.4  treated.5
114818  0.17680336 -0.4920621  0.3057739177 -0.2496781
1306   -0.09336761 -0.4404861 -0.0004008015 -0.1225919
145773 -2.31265679 -2.2920097 -2.2225012098 -2.3626999
169200  0.12669181 -0.1547725 -0.1841533338 -0.4108952
1823   -0.15493929 -0.4758073 -0.2910310564  0.1876621
1982   -0.04991545 -0.3105898 -0.0831039217  0.2711129
```

`limmaAnalysis` fits a linear model to all mRNA and miRNA expression values and computes log fold changes and p-values for differential expression for comparisons of two groups. A design matrix must be generated and passed to `limmaAnalysis` together with contrasts (see the *limma* vignette for details). The output contains estimates of the variance and fold changes, which are used to parameterize and initialize the model.

```
> design = model.matrix(~0+factor(c(rep("control", 5), rep("treated", 5))))
> colnames(design) = c("control", "treated")
> contrasts = "treated - control"
> limmamRNA = limmaAnalysis(sim$dat.mRNA, design, contrasts)
> limmamiRNA = limmaAnalysis(sim$dat.miRNA, design, contrasts)
```

Since miRNA expressions are available, miRNAs are assumed to be active under the condition, where it is upregulated and its targets are downregulated. In general, transcription factor expression is not informative for its activity, thus a switch in regulatory acitvity is predicted. However, a condition specific model, considering expression values of differentially expressed TFs can be applied with *birta* and is discussed in section 2.2.
Now, the data is passed to `birta`. To keep the computations low in this example, the ω vector is not sampled (`sample.weights=FALSE`), potential swaps were pre-computed and the number of iterations is very low. In a real application, the number of iterations should be much higher to make sure, that the Markov-Cahin has converged.

```
> sim_result = birta(sim$dat.mRNA, sim$dat.miRNA, limmamRNA=limmamRNA,
+ limmamiRNA=limmamiRNA, nrep=c(5,5,5,5), genesets=genesets,
+ model="all-plug-in", niter=50000, nburnin=10000,
+ sample.weights=FALSE, potential_swaps=potential_swaps)
```

Figure 1 shows the log-likelihood during the sampling to check the convergence.

```
> plotConvergence(sim_result, nburnin=10000, title="simulation")
```

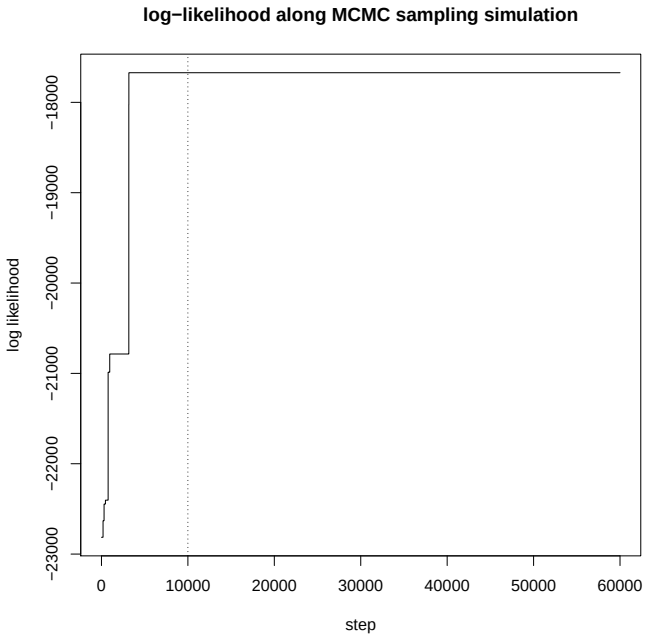

Figure 1: Log-likelihood during MCMC sampling for the simulated data set.

The `sim_result` object is a *list* and the sampled activity states can be accessed via `TFActivitySwitch`, `miRNAstates1` and `miRNAstates2` respectively. Each vector contains the frequency, with which a specific regulator was sampled from the posterior distribution. A value of 0 means, that the regulator was never sampled as active, meaning switching it to active fits the model very badly. A value of 1 means, that the regulator of interest is most certainly active, since switching its state to active substantially increases the likelihood of the model.

```
> sim$TFstates[sim$TFstates == 1]

V$GATA3_03
         1

> sim_result$TFActivitySwitch[sim_result$TFActivitySwitch > 0]

V$GATA3_03
         1

> sim$miRNAstates[sim$miRNAstates == 1]

hsa-miR-155  hsa-miR-96
          1           1

> sim_result$miRNAstates1[sim_result$miRNAstates1 > 0]

hsa-miR-155
          1

> sim_result$miRNAstates2[sim_result$miRNAstates2 > 0]

hsa-miR-96
         1
```

***birta* with a miRNA-target graph only**

It is possible to apply *birta* only to either miRNA-target or TF-target graph. To do this, the miRNA-target graph from the above example is simply extracted and passed to `birta` with the expression data. An example application to a TF-target graph without miRNAs is shown in the next section.

```
> genesetsmiRNA = genesets["miRNA"]
> result_miRonly = birta(sim$dat.mRNA, sim$dat.miRNA, limmamRNA=limmamRNA,
+ limmamiRNA=limmamiRNA, nrep=c(5,5,5,5), genesets=genesetsmiRNA,
+ model="all-plug-in", niter=50000, nburnin=10000,
+ sample.weights=FALSE, potential_swaps=potential_swaps)

> result_miRonly$miRNAstates1[result_miRonly$miRNAstates1 > 0]

hsa-miR-155
          1

> result_miRonly$miRNAstates2[result_miRonly$miRNAstates2 > 0]

hsa-miR-96
         1
```

2.2 Application to an E. Coli data set

Preprocessed microarray data [3], as well as a filtered TF-target graph [2] is used to demonstrate the utility of *birta* on a real data set to infer TF activity states. The expression experiment consists of three replicates from E. Coli during aerobic growth and four replicates during anaerobic growth. The TF-target graph contains annotations for 160 transcription factors. Expression values are stored in an *ExpressionSet*.

```
> data(EColiOxygen)
> EColiOxygen

ExpressionSet (storageMode: lockedEnvironment)
assayData: 4205 features, 7 samples
  element names: exprs
protocolData: none
phenoData
  rowNames: GSM18261 GSM18262 ... GSM18289 (7 total)
  varLabels: Strain GrowthProtocol GenotypeVariation Description
  varMetadata: labelDescription
featureData: none
experimentData: use 'experimentData(object)'
  pubMedIds: 15129285
Annotation: org.EcK12.eg.db

> head(exprs(EColiOxygen))

        GSM18261 GSM18262  GSM18263  GSM18286  GSM18287  GSM18288  GSM18289
947315 10.277125 10.22119 10.410919 10.208393 10.179176 10.186009 10.009045
945490 10.138638 10.17328 10.215396 10.170649  9.993040 10.277822  9.968522
944896 11.016805 11.28574 11.308092 11.287854 11.582083 11.632015 11.463312
945321  8.726455  9.00633  8.973156  9.149897  9.245039  9.298647  9.113609
944895 11.179725 11.09959 11.270414 10.792218 10.750200 11.289802 10.960788
947758 12.399980 12.50940 12.043803 12.460848 12.531210 12.440010 12.510939
```

Differentially expressed genes are calculated using `limmaAnalysis`, which is then passed to `birta`, together with the TF-target graph `EColiNetwork`.

```
> design = model.matrix(~0+factor(pData(EColiOxygen)$GrowthProtocol))
> colnames(design) = c("aerobic.growth", "anaerobic.growth")
> contrasts = "anaerobic.growth - aerobic.growth"
> limmamRNA = limmaAnalysis(EColiOxygen, design, contrasts)
> ecoli_result = birta(EColiOxygen, nrep=c(0, 0, 3, 4),
+ genesets=EColiNetwork, limmamRNA=limmamRNA,
+ model="all-plug-in", niter=50000, nburnin=10000,
+ sample.weights=FALSE)

> plotConvergence(ecoli_result, nburnin=10000, title="E. Coli")
```

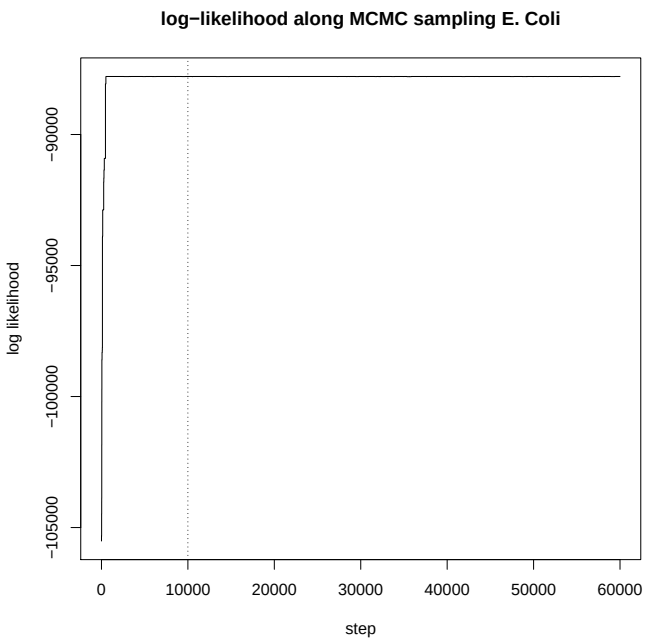

Figure 2: Log-likelihood during MCMC sampling for the E. Coli data set.

The log-likelihood is shown in Figure 2. For active TFs, a cutoff of 0.9 is chosen. The total number of target genes is shown together with the number of differentially expressed target genes for the predicted active TFs:

```
> activeTFs = ecoli_result$TFActivitySwitch[ecoli_result$TFActivitySwitch > 0.9]
> sort(activeTFs)

 adiY  galS  appY  argR  arsR  betI  birA  cusR  dcuR  fruR  gcvA  glcC  lrhA
0.989 0.997 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
 mhpR  mntR  nrdR  pdhR  phoB  slyA  soxR  trpR  ydeO  zntR   zur
1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000

> DEgenes = limmamRNA$pvalue.tab$ID[limmamRNA$pvalue.tab$adj.P.Val < 0.05]
> DEgenesInTargets = sapply(ecoli_result$genesetsTF[names(activeTFs)],
+ function(x) c(length(which(x %in% DEgenes)), length(x)))
> rownames(DEgenesInTargets) = c("#DEgenes", "#targets")
> DEgenesInTargets[,order(DEgenesInTargets["#targets",], decreasing=T)]
```

```
         argR fruR phoB pdhR trpR appY galS nrdR dcuR mhpR birA cusR glcC betI
#DEgenes    0    0    0    0    0    0    0    0    0    0    0    0    0    0
#targets   36   36   36   21   11    9    8    8    7    6    5    5    4    3
         gcvA lrhA ydeO zur arsR soxR adiY mntR slyA zntR
#DEgenes    0    0    0   0    0    0    0    0    0    0
#targets    3    3    3   3    2    2    1    1    1    1
```

Using transcription factor expression

In accordance with recent findings [9], the default model of *birta* does not suppose that the mRNA expression levels of a TF and its (putative) target genes are generally correlated. However, assuming a correlation of TF expression and its targets might be correct in some cases. Thus, an extended model of *birta* allows to integrate TF expression of differentially expressed TFs into the model in a similar way as it models miRNA expression.
`TFexpr` contains an excerpt of `EColiOxygen`, containing the mRNA expression for all 160 TFs in `EColiNetwork`. The row names of the expression matrix were converted to the corresponding TF identifiers in `EColiNetwork`.

```
> head(exprs(TFexpr))
```

```
      GSM18261  GSM18262  GSM18263  GSM18286  GSM18287  GSM18288  GSM18289
acrR  8.277473  8.309069  8.504610  7.857166  7.686808  8.111077  7.915678
ada   9.277946  9.540328  9.186303  9.578132  9.646316  9.444881  9.384217
adiY  6.330554  6.555999  6.686157 10.801038 10.986309  8.788498  8.612713
agaR 10.854649 10.726303 10.782988 10.936007 11.041171 11.200971 11.130323
allR 11.324718 11.193124 11.389784 11.102606 11.274170 11.273160 10.936546
allS  8.520564  8.764251  8.693574  8.806117  8.806997  8.705715  8.272239
```

`limmaAnalysis` is used to assess differential expression of these TFs. Both objects are then passed to `birta`, which automatically extracts differentially expressed TFs from the matrix using `lfc.mRNA` and `fdr.mRNA` as log fold change and p-value cutoff respectively. The expression of these selected TFs is then used in the model. Activities of non-differentially expressed TFs are modeled with the default model.

```
> limmaTF = limmaAnalysis(TFexpr, design, contrasts)
> ecoli_TFexpr = birta(EColiOxygen, nrep=c(0, 0, 3, 4),
+ genesets=EColiNetwork, TFexpr=TFexpr, limmaTF=limmaTF,
+ limmamRNA=limmamRNA, model="all-plug-in", niter=50000,
+ nburnin=10000, sample.weights=FALSE)
```

If the TF expression is considered - like for miRNAs - the TF is assumed to be active under the condition, where it is higher expressed. Therefore, it is possible to make a condition specific prediction for the activity of these TFs. For TFs, which are not differentially expressed, the prediction refers again to a switch in activity between both conditions.

```
> sort(ecoli_TFexpr$TFstates1[ecoli_TFexpr$TFstates1 > 0.9])
```

```
glcC betI mhpR cusR mntR pdhR
   1    1    1    1    1    1
```

```
> sort(ecoli_TFexpr$TFstates2[ecoli_TFexpr$TFstates2 > 0.9])
```

```
 zntR  appY  fruR   zur  adiY  nikR  lrhA
0.985 1.000 1.000 1.000 1.000 1.000 1.000
```

```
> sort(ecoli_TFexpr$TFActivitySwitch[ecoli_TFexpr$TFActivitySwitch > 0.9])
```

```
 galS  argR  arsR  birA  dcuR  gcvA  nrdR  phoB  slyA  soxR  trpR  ydeO
0.989 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000 1.000
```

3 Conclusion

birta integrates miRNA and mRNA data in a statistical framework (namely a Bayesian Network) to make inference on TF and miRNA activities in a condition specific way. It is a step towards the important goal to unravel causal mechanisms of gene expression changes under specific experimental or natural conditions.
This vignette was generated using the following package versions:

- R version 3.4.0 (2017-04-21), `x86_64-pc-linux-gnu`
- Locale: `LC_CTYPE=en_US.UTF-8`, `LC_NUMERIC=C`, `LC_TIME=en_US.UTF-8`, `LC_COLLATE=C`, `LC_MONETARY=en_US.UTF-8`, `LC_MESSAGES=en_US.UTF-8`, `LC_PAPER=en_US.UTF-8`, `LC_NAME=C`, `LC_ADDRESS=C`, `LC_TELEPHONE=C`, `LC_MEASUREMENT=en_US.UTF-8`, `LC_IDENTIFICATION=C`
- Running under: `Ubuntu 16.04.2 LTS`
- Matrix products: default
- BLAS: `/home/biocbuild/bbs-3.5-bioc/R/lib/libRblas.so`
- LAPACK: `/home/biocbuild/bbs-3.5-bioc/R/lib/libRlapack.so`
- Base packages: base, datasets, grDevices, graphics, methods, parallel, stats, utils
- Other packages: Biobase 2.36.0, BiocGenerics 0.22.0, MASS 7.3-47, birta 1.20.0, limma 3.32.0
- Loaded via a namespace (and not attached): compiler 3.4.0, tools 3.4.0

References

[1] D. Betel, M. Wilson, A. Gabow, D. S. Marks, and C. Sander. The microrna.org resource: targets and expression. *Nucleic Acids Res*, 36(Database issue):D149–D153, Jan 2008.

[2] R. Castelo and A. Roverato. Reverse engineering molecular regulatory networks from microarray data with qp-graphs. *J Comput Biol*, 16(2):213–227, Feb 2009.

[3] M. W. Covert, E. M. Knight, J. L. Reed, M. J. Herrgard, and B. O. Palsson. Integrating high-throughput and computational data elucidates bacterial networks. *Nature*, 429(6987):92–96, May 2004.

[4] S. Griffiths-Jones, H. K. Saini, S. van Dongen, and A. J. Enright. miRBase: tools for microRNA genomics. *Nucleic Acids Res*, 36(Database issue):D154—-D158, Jan. 2008.

[5] G. L. Papadopoulos, M. Reczko, V. A. Simossis, P. Sethupathy, and A. G. Hatzigeorgiou. The database of experimentally supported targets: a functional update of tarbase. *Nucleic Acids Res*, 37(Database issue):D155–D158, Jan 2009.

[6] H. G. Roider, A. Kanhere, T. Manke, and M. Vingron. Predicting transcription factor affinities to dna from a biophysical model. *Bioinformatics*, 23(2):134–141, Jan 2007.

[7] G. K. Smyth. Limma : Linear Models for Microarray Data. *Bioinformatics*, (2005):397–420.

[8] X. Wang and I. M. E. Naqa. Prediction of both conserved and nonconserved microrna targets in animals. *Bioinformatics*, 24(3):325–332, Feb 2008.

[9] M. Wu and C. Chan. Learning transcriptional regulation on a genome scale: a theoretical analysis based on gene expression data. *Brief Bioinform*, May 2011.

[10] B. Zacher, K. Abnaof, S. Gade, E. Younesi, A. Tresch, and H. Frohlich. Joint Bayesian Inference of Condition Specific miRNA and Transcription Factor Activities from Combined Gene and microRNA Expression Data. *submitted*, 2012.