

# Bayesian Inference of Regulatory influence on Expression (biRte)

Holger Fröhlich

October 29, 2019

## 1 Introduction

Expression levels of mRNA is regulated by different processes, comprising inhibition or activation by transcription factors (TF) and post-transcriptional degradation by microRNAs (miRNA). *biRte* (Bayesian Inference of Regulatory influence on Expression (biRte)) uses the regulatory networks of TFs and miRNAs together with mRNA and miRNA expression data to infer the influence of regulators on mRNA expression. Furthermore, *biRte* allows to consider additional factors such as CNVs. *biRte* has the possibility to specify Bayesian priors for the activity of each individual regulatory factor. Moreover, interaction terms between regulators can be considered. *biRte* relies on a Bayesian network model to integrate data sources into a joint likelihood model. In the model mRNA expression levels depend on the activity states of its regulating factors via a sparse Bayesian linear regression using a spikes and slab prior [?]. Moreover, miRNA expression levels depend on miRNA activity states. *biRte* uses Markov-Chain-Monte-Carlo (MCMC) sampling to infer activity states of regulatory factors. During MCMC, switch moves - toggling the state of a regulator between active and inactive - and swap moves - exchanging the activity states of either two miRNAs or two TFs - are used [8].

*biRte* is meant as a replacement for the earlier package *birta*. *biRte* offers several advantages compared to *birta*.

- possibility to include additional regulatory factors and data apart from TFs and miRNAs
- possibility to include target specific regulation strength values
- possibility to define a prior probabilities for activity of each individual regulator and even regulator pairs.
- significantly faster inference (about 15 fold speed-up)
- significantly higher accuracy of inference due to improved likelihood calculation
- inference of regulatory networks as a follow-up step
- possibility to work with arbitrarily complex statistical designs, if log fold changes are used.

The package can be loaded by typing:

```
> rm(list=ls())  
> library(birte)
```

## 2 Usage of biRte

The two main functions of the package are `birteRun` and `birteLimma`. `birteLimma` is a convenience function, which passes the output of `limmaAnalysis` to `birteRun`. The most important input arguments to `birteRun` are

- **dat.mRNA**. Matrix of mRNA expression data with row names indicating genes.
- **affinities**. A weighted regulator-target graph. This is a list with at most three components (TF, miRNA, other). Each of these lists again contains a weighted adjacency list representation. See **affinities** for more information and `humanNetworkSimul` for an example. Per default weights are ignored in the inference process. IMPORTANT: gene names used in this network have to match with row names of `dat.mRNA`.
- **nrep.mRNA** is an integer vector, which specifies the number of replicates per condition for mRNA data

## 3 Applying biRte to RNAseq Data

*biRte* relies on the assumption that data are (multivariate) normally distributed. Application to RNAseq data is thus not immediately possible. Data should thus be transformed appropriately, e.g. via the voom + limma mechanism [5].

## 4 Example: Aerobic vs. anaerobic growth in E. Coli

To demonstrate the use of *biRte* we here show a most basic application to a microarray dataset by [2] together with a filtered TF-target graph [1]. The gene expression data comprises 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*.

```
> library(Biobase)
> 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
  featureNames: 1 2 ... 4205 (4205 total)
  fvarLabels: symbol Entrez
  fvarMetadata: labelDescription
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

Before starting our biRte analysis we try to simplify the TF-target by clustering regulators with highly overlapping target gene sets. Then we determine possible interactions between regulators by looking for regulators, which have an overlap that is large enough to be considered, but not as large that the effect is indistinguishable from main effects by individual regulators.

Afterwards, differentially expressed genes are calculated using `limmaAnalysis`. The result is then passed to `biRteLimma`, together with the TF-target graph `EColiNetwork`. As a final step we use biRte to look for regulator activities that can explain differential gene expression between anaerobic and aerobic growth. In a real application the number of MCMC iterations should be increased significantly:

```
> # prepare network
> affinities = list(TF=sapply(names(EColiNetwork$TF), function(tf){w = rep(1, length(EColiNetwork$TF))
> affinities = simplify(affinities)
> affinities$other = proposeInteractions(affinities)
> # prepare data
> mydat = exprs(EColiOxygen)
> colnames(mydat) = make.names(paste(pData(EColiOxygen)$GenotypeVariation, pData(EColiOxygen)$GrowthCondition))
> limmamRNA = limmaAnalysis(mydat, design=NULL, "wild.type.anaerobic - wild.type.aerobic")
> mydat = cbind(mydat[,colnames(mydat) == "wild.type.aerobic"], mydat[,colnames(mydat) == "wild.type.anaerobic"])
> ecoli_result = birteLimma(dat.mRNA=mydat, limmamRNA=limmamRNA, affinities=affinities, niter=5000)

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

The log-likelihood is shown in Figure 1. Below we show those TFs, who reveal a marginal activity probability of larger than a cutoff corresponding to an expected false positive rate of 0.001. We look at the total number of target genes together with the number of differentially expressed target genes for the predicted TFs:

```
> tau = suggestThreshold(ecoli_result$post[,1])

start values: (alpha, beta) = 1.5 5 lambda = 0.5 0.5
logLik = -1038.598 (alpha, beta) = 149.4132 149.4132 lambda = 0.1021743 0.8978257
logLik = -1038.61 (alpha, beta) = 149.4132 149.4132 lambda = 0.09917355 0.9008264
[1] "converged!"

> activeTFs = rownames(ecoli_result$post)[ecoli_result$post[,1] > tau]
> activeTFs

[1] "caiF"      "dcuR"      "fnr"       "fhfA"      "fruR"      "fur"
[7] "galR"      "gatR"      "gcvA"      "glpR"      "iclR"      "narL"
[13] "glnG"      "arcA_crp"  "arcA_fnr"  "crp_fnr"   "gadE_gadW" "fhfA_hyfR"
[19] "modE_narL" "fnr_narP"  "modE_narP" "argP_nrdR" "dnaA_nrdR" "gadE_pdhR"
[25] "gadE_torR"

> if(length(activeTFs) > 0){
+   DEgenes = rownames(limmamRNA$pvalue.tab)[limmamRNA$pvalue.tab$adj.P.Val < 0.05 & abs(limmamRNA$log2FC) > 1]
+   genesetsTF = c(sapply(affinities$TF, names), sapply(affinities$other, names))
+   DEgenesInTargets = sapply(genesetsTF[intersect(activeTFs, names(genesetsTF))],
```

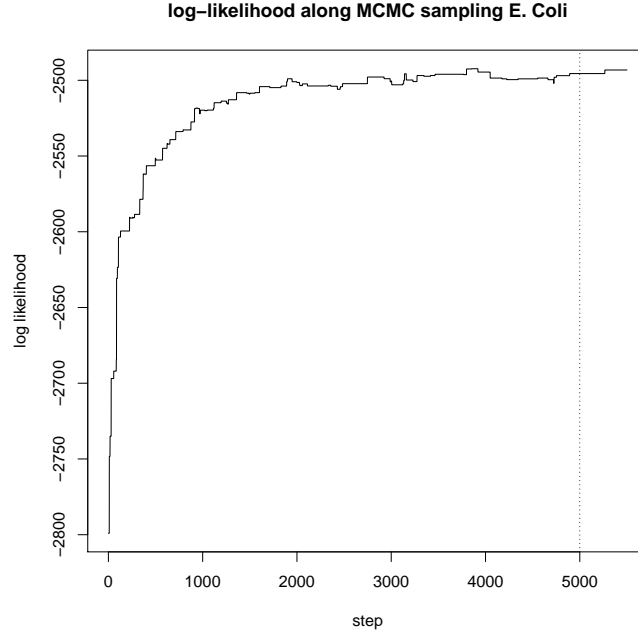


Figure 1: Log-likelihood during MCMC sampling for the E. Coli data set.

```
+ function(x) c(length(which(x %in% DEgenes)), length(x))
+ rownames(DEgenesInTargets) = c("#DEgenes", "#targets")
+ DEgenesInTargets[,order(DEgenesInTargets["#targets",], decreasing=TRUE)]
+ }
```

	fnr	narL	fur	crp_fnr	arcA_fnr	arcA_crp	glnG	fruR	fnr_narP	fhlA
#DEgenes	35	31	1	10	6	4	0	3	6	15
#targets	264	101	78	77	74	57	43	36	35	29

	modE_narL	modE_narP	fhlA_hyfR	caiF	galR	glpR	dcuR	gatR	gadE_gadW
#DEgenes	2	0	0	0	0	0	5	0	2
#targets	18	15	11	10	9	9	7	6	6

	gadE_pdhR	gadE_torR	gcvA	iclR	argP_nrdR	dnaA_nrdR
#DEgenes	0	1	0	0	0	0
#targets	5	5	3	3	2	2

We can ask, how well log fold changes predicted by our biRte model agree with observed log fold changes:

```
> pred = birtePredict(ecoli_result, rownames(mydat))
> cor(pred[[1]][[1]]$mean, limmamRNA$pvalue.tab[rownames(mydat), "logFC"])

[1] 0.4818549
```

Once again it should be noted that in a real application the MCMC sampler should run much longer and hence better results are expected.

## 5 Using Regulator Expression Data

One of the strength of *biRte* is that measurements of regulators can be integrated smoothly into the inference process.

In our example situation no miRNA expression data is available, but some transcription factors have been measured on the microarray. In accordance with published results [7], *biRte* does not suppose that the mRNA expression levels of a TF and its (putative) target genes are correlated. However, differential TF expression on mRNA level might still give a hint on activity differences on protein level. Thus, *biRte* allows to integrate expression data of differentially expressed TFs. In our case *TFexpr* contains an excerpt of *EColiOxygen*. It comprises 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

Differential expression of these TFs can be assessed by subsetting our previous *limmamRNA* object. We use the obtained results to define an informative prior for each regulator and regulator-regulator interaction, before running a *biRte* analysis, and to set up a reasonable initial state for the sampler:

```
> limmaTF = limmamRNA
> limmaTF$pvalue.tab = limmaTF$pvalue.tab[rownames(limmaTF$pvalue.tab) %in% fData(TFexpr)$Entrez,]
> names(limmaTF$lm.fit$sigma) = as.character(fData(EColiOxygen)$symbol[match(names(limmaTF$lm.fit$sigma),
> rownames(limmaTF$pvalue.tab) = as.character(fData(EColiOxygen)$symbol[match(rownames(limmaTF$pvalue.tab),
> diff.TF = rownames(limmaTF$pvalue.tab)[limmaTF$pvalue.tab$adj.P.Val < 0.05 & abs(limmaTF$pvalue.tab$diff) > 1])
> theta.TF = rep(1/length(affinities$TF), length(affinities$TF))
> names(theta.TF) = names(affinities$TF)
> theta.other = rep(1/length(affinities$other), length(affinities$other))
> names(theta.other) = names(affinities$other)
> theta.other[unique(unlist(sapply(diff.TF, function(tf) grep(tf, names(theta.other)))))] = 0.5 #
> init.TF = theta.TF
> init.TF = (init.TF >= 0.5)*1
> init.other = theta.other
> init.other = (init.other >= 0.5)*1
> # note that niter and nburnin are much too small in practice
> ecoli_TFexpr = birteLimma(dat.mRNA=mydat, data.regulators=list(TF=exprs(TFexpr)), limmamRNA=limmamRNA)
> tau = suggestThreshold(ecoli_TFexpr$post[,1])

start values: (alpha, beta) = 1.5 5 lambda = 0.5 0.5
logLik = -850.2386 (alpha, beta) = 4.214676 149.4132 lambda = 0.1980839 0.8019161
logLik = -850.5199 (alpha, beta) = 4.182509 149.4132 lambda = 0.217441 0.782559
logLik = -850.5199 (alpha, beta) = 4.182238 149.4132 lambda = 0.2176202 0.7823798
[1] "converged!"

> activeTFs = ecoli_TFexpr$post[ecoli_TFexpr$post[,1] > tau,1]
> activeTFs
```

adiY	appY	arcA	caiF
1.000	1.000	1.000	1.000
dicA	gadE	lrhA	narP

0.482	1.000	1.000	0.302
rstA	uhpA	acrR	betI
1.000	0.646	1.000	1.000
cusR	fnr	feaR	fur
1.000	1.000	1.000	1.000
glcC	hcaR	iscR	lldR
1.000	1.000	1.000	1.000
marA	mhpR	mntR	narL
1.000	1.000	1.000	1.000
pepA	soxS	dcuR	fhfA
1.000	1.000	1.000	1.000
fruR	galS	gatR	gcvA
0.482	1.000	1.000	1.000
arcA_crp	argP_dnaA	crp_fnr	gadE_gadW
1.000	1.000	0.990	0.428
gadE_gadX	fhfA_hyfR	arcA_ihfA_U_ihfB	appY_iscR
1.000	1.000	1.000	0.140
iscR_modE	fnr_narL	modE_narL	appY_narP
0.466	0.870	1.000	0.086
iscR_narP	modE_narP	argP_nrdR	csgD_ompR
0.398	0.986	0.666	0.198
iscR_oxyR	gadE_pdhR	ompR_rstA	rcaA_U_rcaB_rstA
1.000	1.000	0.702	0.438
hupA_U_hupB_tyrR			
1.000			

## 6 Network Inference

After having determined active regulators one may ask, in which way these regulators influence each other. Bayesian Networks are a principal possibility, but would usually require direct measurements of regulators, which is difficult to obtain for TFs. Moreover, the typically small sample size imposes a principal limitation. We thus restrict ourselves to subset relationships between differentially expressed target genes. These subset relationships can have two possible interpretations: One possibility is that regulator A acts upstream of regulator B, if differential targets of B are a subset of those of A. Another possibility is that A and B jointly co-regulate certain target genes. The idea of (noisy) subset relationships has striking similarities to Nested Effects Models (NEMs) [?, 4], which have been introduced for causal network inference from perturbation data. Although in our case we do not have targeted perturbations of individual regulators, probabilistic inference of subset relationships between differentially expressed targets of regulator pairs can be effectively solved via NEM inference. *biRte* uses the pair-wise inference algorithm discussed in [6] as default.

*biRte* offers a convenience function `estimateNetwork` for this purpose. The function decomposes clusters of active regulators into individual regulators and performs appropriate calls to functions from *nem* [3]. The output is a network indicating subset relationships between differential targets of active regulators. In our example this would be done as follows:

```
> DEgenes = rownames(limmamRNA$pvalue.tab)[limmamRNA$pvalue.tab$adj.P.Val < 0.05 & abs(limmamRNA$
> net = estimateNetwork(ecoli_TFexpr, thresh=tau, de.genes=DEgenes)
> library(nem)
> if(require(Rgraphviz)){
+   plot(net, transitiveReduction=TRUE)
+ }
```

This yields the network shown in Figure 2. In addition to the network structure we can investigate

the estimated dependencies regulator-gene dependencies in more depth. This may give additional insights whether a particular gene is a direct target of a particular transcription factor or not and hence allow for filtering out false positive target predictions:

```
> net$mappos
```

In our case there are several totally unspecific target genes (assigned to "null"), which may indicate false positive target gene predictions.

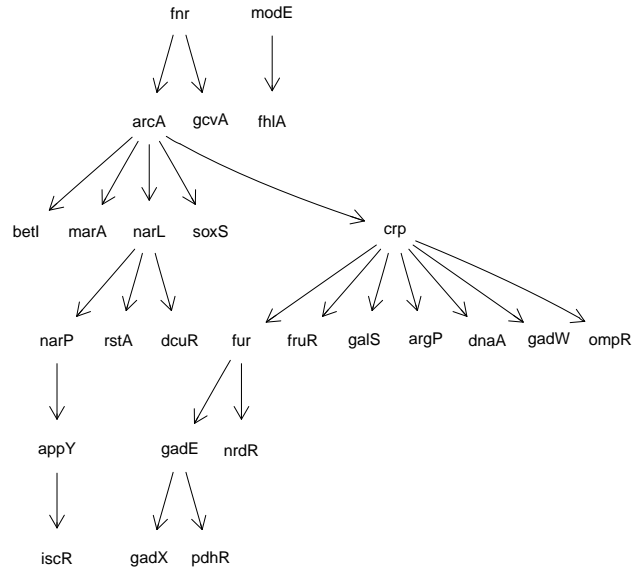


Figure 2: Inferred network between active TFs.

## 7 Regulator Activities in Single Samples

One may ask, in how far activity of regulators differs from one sample to another. *biRte* now allows for addressing such a question. Essentially, the idea is to run *biRte*'s Bayesian inference procedure independently for each single sample. That means for each individual sample we look for the regulator combination that could explain the observed expression level of genes. The output is a samples x regulators matrix containing either the marginal posterior probabilities for each regulator to be active in a certain sample, or the most likely configurations of active regulators in each sample. In our example we run single sample analysis as follows:

```
> ss_ecoli_TFexpr = birteLimma(dat.mRNA=mydat, data.regulators=list(TF=exprs(TFexpr)), limmamRNA=
> ss_ecoli_TFexpr[,colSums(ss_ecoli_TFexpr) > 0] # regulators that are active in at least one sam
```

## 8 Conclusion

*biRte* integrates regulator expression and mRNA data into a probabilistic framework to make inference on regulator activities. It is a step towards the important goal to unravel causal mechanisms of gene expression changes under specific experimental or natural conditions. A unique feature is the combination with network inference.

This vignette was generated using the following package versions:

- R version 3.6.1 (2019-07-05), x86\_64-w64-mingw32
- Locale: LC\_COLLATE=C, LC\_CTYPE=English\_United States.1252, LC\_MONETARY=English\_United States.1252, LC\_NUMERIC=C, LC\_TIME=English\_United States.1252
- Running under: Windows Server 2012 R2 x64 (build 9600)
- Matrix products: default
- Base packages: base, datasets, grDevices, graphics, grid, methods, parallel, stats, utils
- Other packages: Biobase 2.46.0, BiocGenerics 0.32.0, Rcpp 1.0.2, RcppArmadillo 0.9.800.1.0, Rgraphviz 2.30.0, birte 1.22.0, graph 1.64.0, nem 2.60.0
- Loaded via a namespace (and not attached): MASS 7.3-51.4, Matrix 1.2-17, RBGL 1.62.0, RColorBrewer 1.1-2, boot 1.3-23, class 7.3-15, codetools 0.2-16, compiler 3.6.1, e1071 1.7-2, foreach 1.4.7, glmnet 2.0-18, iterators 1.0.12, knitr 1.25, lattice 0.20-38, limma 3.42.0, plotrix 3.7-6, statmod 1.4.32, stats4 3.6.1, tools 3.6.1, xfun 0.10

## References

- [1] 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.
- [2] 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.
- [3] H. Fröhlich, T. Beißbarth, A. Tresch, D. Kostka, J. Jacob, R. Spang, and F. Markowetz. Analyzing gene perturbation screens with nested effects models in R and bioconductor. *Bioinformatics*, 24(21):2549–2550, Nov 2008.
- [4] H. Fröhlich, A. Tresch, and T. Beissbarth. Nested effects models for learning signaling networks from perturbation data. *Biom J*, 51(2):304–323, Apr 2009.
- [5] C. W. Law, Y. Chen, W. Shi, and G. K. Smyth. voom: Precision weights unlock linear model analysis tools for rna-seq read counts. *Genome Biol*, 15(2):R29, 2014.
- [6] F. Markowetz, D. Kostka, O. Troyanskaya, and R. Spang. Nested effects models for high-dimensional phenotyping screens. *Bioinformatics*, 23:i305 – i312, 2007.
- [7] M. Wu and C. Chan. Learning transcriptional regulation on a genome scale: a theoretical analysis based on gene expression data. *Brief Bioinform*, May 2011.
- [8] B. Zacher, K. Abnaof, S. Gade, E. Younesi, A. Tresch, and H. Fröhlich. Joint bayesian inference of condition-specific mirna and transcription factor activities from combined gene and microrna expression data. *Bioinformatics*, 28(13):1714–1720, Jul 2012.