— Nested Effects Models —
An example in *Drosophila* immune response

Florian Markowetz*

September 12, 2006

Abstract

Cellular signaling pathways, which are not modulated on a transcriptional level, cannot be directly deduced from expression profiling experiments. The situation changes, when external interventions like RNA interference or gene knock-outs come into play.

In Markowetz *et al.* (2005) and Markowetz (2006) we introduced an algorithm to infer non-transcriptional pathway features based on differential gene expression in silencing assays. The method is implemented in the Bioconductor package `nem`. Here we demonstrate its practical use in the context of an RNAi data set investigating the response to microbial challenge in Drosophila melanogaster.

We show in detail how the data is pre-processed and discretized, how the pathway can be reconstructed by different approaches, and how the final result can be post-processed to increase interpretability.

1 Drosophila RNAi data

We applied our method to data from a study on innate immune response in *Drosophila* (Boutros *et al.*, 2002). Selectively removing signaling components blocked induction of all, or only parts, of the transcriptional response to LPS.

Dataset summary The dataset consists of 16 Affymetrix-microarrays: 4 replicates of control experiments without LPS and without RNAi (negative controls), 4 replicates of expression profiling after stimulation with LPS but without RNAi (positive controls), and 2 replicates each of expression profiling after applying LPS and silencing one of the four candidate genes *tak*, *key*, *rel*, and *mkk4/hep*.

*Lewis-Sigler Institute for Integrative Genomics, Princeton, NJ 08544, USA. eMail: florian@genomics.princeton.edu; URL: http://genomics.princeton.edu/∼florian

Preprocessing and E-gene selection For preprocessing, we perform normalization on probe level using a variance stabilizing transformation (Huber *et al.*, 2002), and probe set summarization using a median polish fit of an additive model (Irizarry *et al.*, 2003). The result is included as a dataset in the package `nem`.

```
> library(nem)
> data("BoutrosRNAi2002")
```

The function `nem.discretize` implements the following two preprocessing steps: First, we select the genes as effect reporters (E-genes), which are more then two-fold upregulated by LPS treatment. Next, we transform the continuous expression data to binary values. We set an E-genes state in an RNAi experiment to `1` if its expression value is sufficiently far from the mean of the positive controls, *i.e.* if the intervention interrupted the information flow. If the E-genes expression is close to the mean of positive controls, we set its state to `0`.

Let C_{ik} be the continuous expression level of E_i in experiment k. Let μ_i^+ be the mean of positive controls for E_i, and μ_i^- the mean of negative controls. To derive binary data E_{ik}, we defined individual cutoffs for every gene E_i by:

$$E_{ik} = \begin{cases} 1 & \text{if } C_{ik} < \kappa \cdot \mu_i^+ + (1-\kappa) \cdot \mu_i^-, \\ 0 & \text{else.} \end{cases} \tag{1}$$

```
> res.disc <- nem.discretize(BoutrosRNAiExpression, neg.control = 1:4,
+     pos.control = 5:8, cutoff = 0.7)
```

```
discretizing with respect to POS and NEG controls
```

Estimating error probabilities From the positive and negative controls, we can estimate the error probabilities α and β. The type I error α is the number of positive controls discretized to state `1`, and the type II error β is the number of negative controls in state `0`. To guard against unrealistically low estimates we add pseudo counts. The error estimates are included into the discretization results:

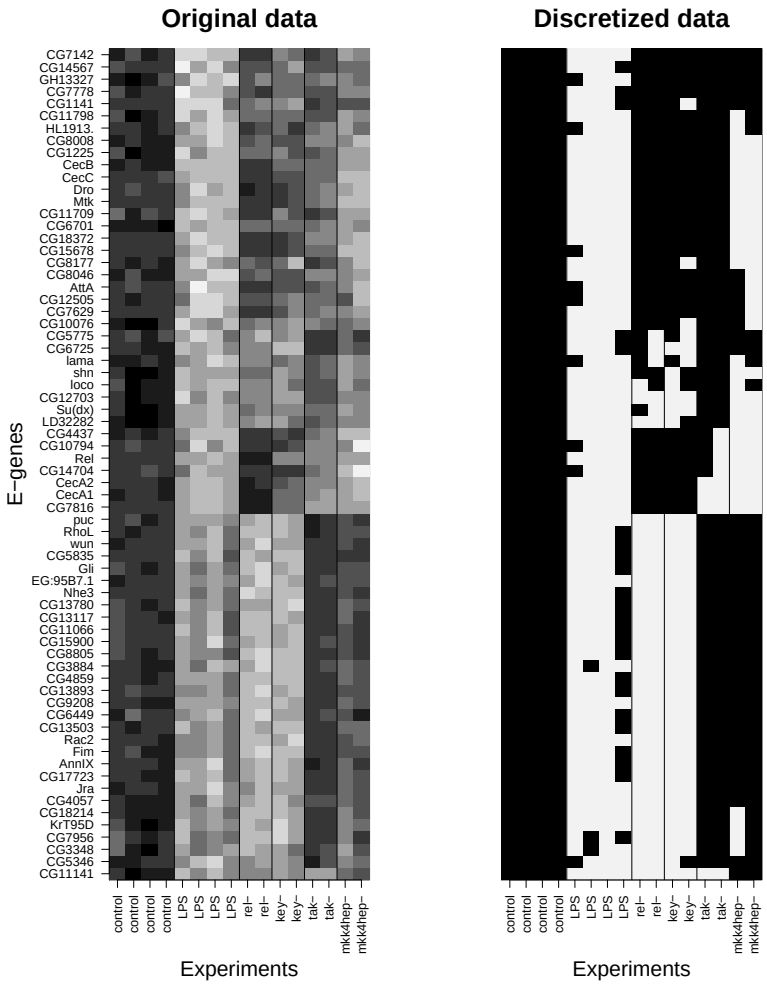

Figure 1: Continuous and discretized data

2 Applying Nested Effects Models

Which model explains the data best? With only four S-genes, we can exhaustively enumerate all pathway models and search the whole space for the best-fitting model. To score these models use either the marginal likelihood depending on α and β (details found in Markowetz et al (2005)) or the full marginal likelihood depending on hyperparameters (details in Markowetz, 2006). Additionally we show how to employ an edge-wise inference heuristic, which can also be applied to cases where exhaustive search over model space is infeasible (i.e. when we have more than 4 or 5 perturbed genes). An interface to all inference techniques is provided by the function `nem()`.

2.1 Exhaustive search by marginal likelihood

Scoring models by marginal log-likelihood is implemented in function `score`. Input consists of models and data, the type of the score (`"mLL"` or `"FULLmLL"`), the corresponding paramters (`para`) or hyperparameters (`hyperpara`) and a prior for E-gene positions (`P`).

```
> result <- nem(res.disc$dat, type = "mLL", para = res.disc$para,
+     inference = "search")

Generated 355 unique models ( out of 4096 )
Computing marginal likelihood for 355 models

> result

Object of class 'score' generated by 'score()'

$graph:  phenotypic hierarchy on genes
$mLL:    a vector of length 355
$pos:    a list   of length 355
$mappos: a vector of length 355
```

The output is the highest scoring model (`result$graph`), a vector of scores (`result$mLL`) and a list of E-gene position posteriors (`result$pos`), and a MAP estimate of E-gene positions (`result$mappos`). We can plot the results using the commands:

```
> plot(result, what = "graph")
> plot(result, what = "mLL")
> plot(result, what = "pos")
```

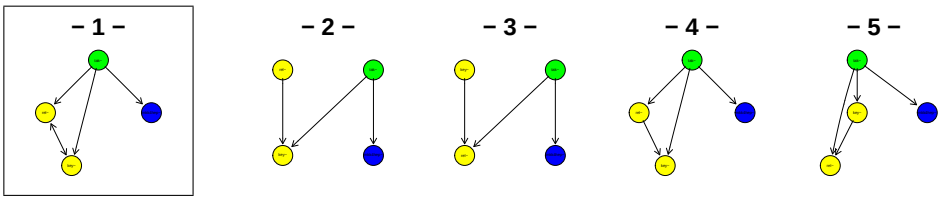

Figure 2: The five silencing schemes getting high scores in Fig. **??**. It takes a second to see it, but Nr.2 to 5 are not that different from Nr.1. The main feature, ie. the branching downstream of *tak* is conserved in all of them.

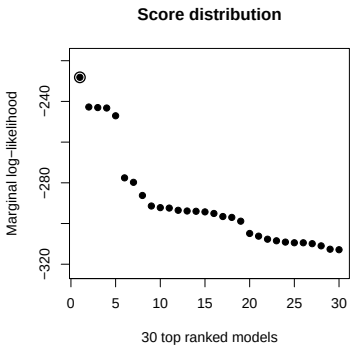

Figure 3: The best 30 scores

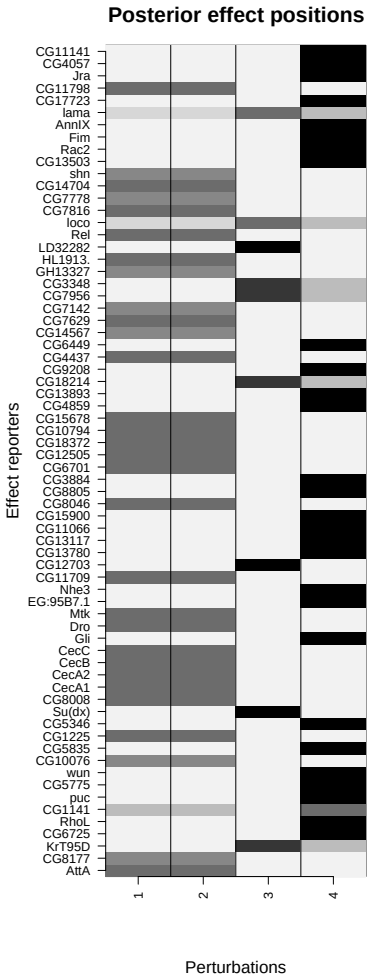

Figure 4: Posterior distributions of E-gene positions given the highest scoring silencing scheme (Nr. 1 in Fig. 2). The MAP estimate corresponds to the row-wise maximum.

2.2 Exhaustive search Full marginal likelihood

Additionally to what we did in the paper (Markowetz *et al.*, 2005) the PhD thesis (Markowetz, 2006) contains equations for a "full marginal likelihood" in which error probabilities α and β are integrated out. This section shows that using this score we learn the same pathways as before.

```
> result2 <- nem(res.disc$dat, type = "FULLmLL", hyperpara = c(1,
+     9, 9, 1), inference = "search")

Generated 355 unique models ( out of 4096 )
Computing FULL marginal likelihood for 355 models

> result2

Object of class 'score' generated by 'score()'

$graph:  phenotypic hierarchy on genes
$mLL:    a vector of length 355
$pos:    a list   of length 355
$mappos: a vector of length 355
```

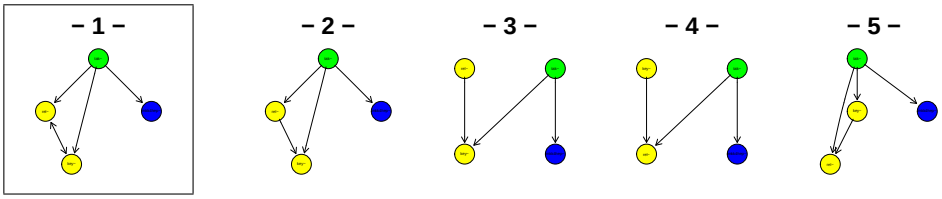

Figure 5: Same topologies as before.

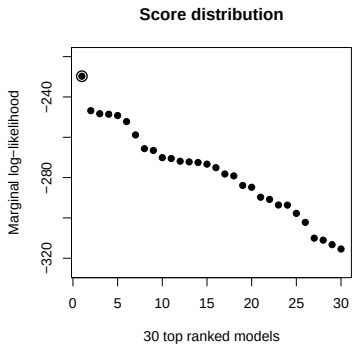

Figure 6: The best 30 scores by full marginal likelihood

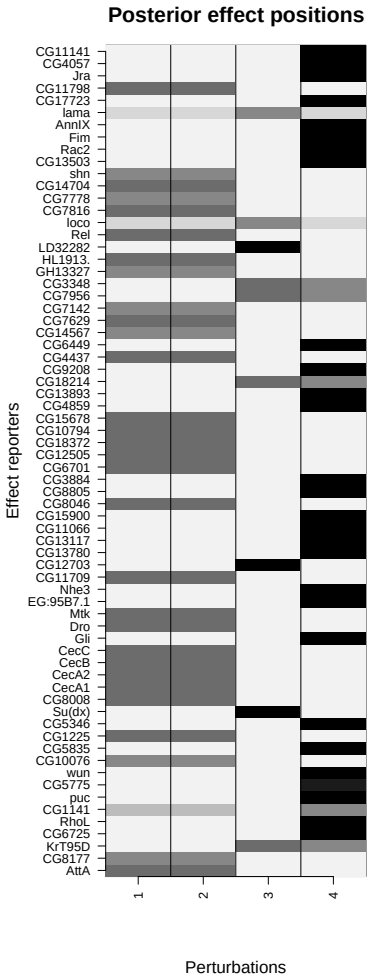

Figure 7: Posterior distributions of E-gene positions given the highest scoring silencing scheme (Nr. 1 in Fig. 5). The MAP estimate corresponds to the row-wise maximum.

2.3 Edge-wise learning

Instead of scoring whole pathways, we can learn the model edge by edge. For each pair of genes A and B we infer the best of four possible models: $A \cdot \cdot B$ (unconnected), $A \rightarrow B$ (effects of A are superset of effects of B), $A \leftarrow B$ (subset), and $A \leftrightarrow B$ (undistinguishable effects).

```
> result3 <- nem(res.disc$dat, para = res.disc$para, inference = "pairwise")

4 perturbed genes -> 6 pairwise tests
......
estimating effect positions

> result3

Object of class 'pairwise' generated by 'pairwise.posterior()'

$graph:  phenotypic hierarchy on genes)
$scores: posterior distributions of local models

Summary of MAP estimates:
all  ..  -> <->
  6   2   3   1
```

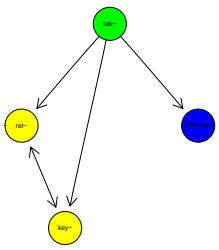

Figure 8: Result of edge-wise learning. Compare this to the result from global search. It looks exactely the same.

3 Visualization

```
> plot.effects(res.disc$dat, result)
```

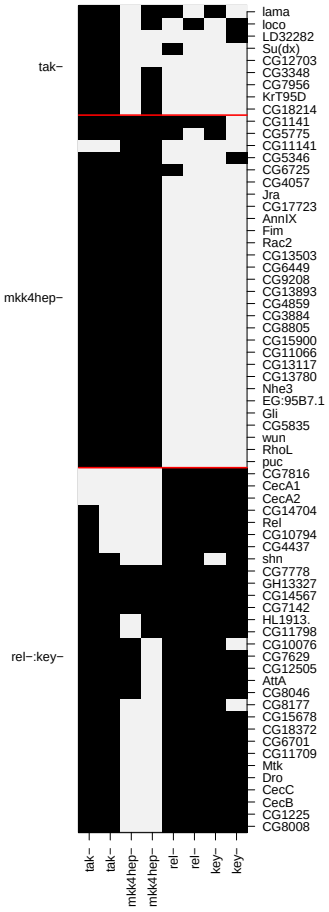

Figure 9: plotting data according to inferred hierarchy

4 Post-processing of results

Pruning spurious edges Function `prune.graph` ...

Combining strongly connected components First, we identify all nodes/genes which are not distinguishable given the data. This amounts to finding the strongly connected components in the graph. Relish and Key are now combined into one node.

```
> result3.scc <- SCCgraph(result3$graph, name = TRUE)
> plot(result3.scc$graph)
```

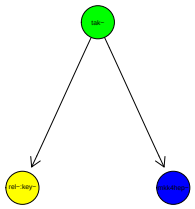

Figure 10: The undistinguishable profiles of *key* and *rel* are summarized into a single node.

Transitive reduction Additionally, in bigger graphs `transitive.reduction()` helps to see the structure of the network better. In this simple example there are no shortcuts to remove.

References

[Boutros *et al.* (2002)] Boutros M, Agaisse H, Perrimon N. Sequential activation of signaling pathways during innate immune responses in *Drosophila*. *Developmental Cell*, 3(5):711–722, 2002.

[Huber *et al.* (2002)] Huber W, Heydebreck A, Sültmann H, Poustka A, Vingron M. Variance Stabilization Applied to Microarray Data Calibration and to the Quantification of Differential Expression. *Bioinformatics*, 18:S96–S104, 2002.

[Irizarry *et al.* (2003)] Irizarry RA, Bolstad BM, Collin F, Cope LM, Hobbs B, Speed TP. Summaries of Affymetrix GeneChip probe level data. *Nucleic Acids Res.* 31(4):e15, 2003.

[Markowetz *et al.* (2005)] Markowetz F, Bloch J, Spang R. Non-transcriptional pathway features reconstructed from secondary effects of RNA interference *Bioinformatics*, 2005

[Markowetz (2006)] Markowetz F. Probabilistic Models for Gene Silencing Data. *PhD thesis*, Free University Berlin, 2005

Session Information

The version number of R and packages loaded for generating the vignette were:

- R version 2.4.0 alpha (2006-09-10 r39242), `x86_64-unknown-linux-gnu`
- Locale: `LC_CTYPE=en_US;LC_NUMERIC=C;LC_TIME=en_US;LC_COLLATE=en_US;LC_MONETARY=e`
- Base packages: base, datasets, graphics, grDevices, methods, stats, tools, utils
- Other packages: annotate 1.11.5, Biobase 1.11.35, class 7.2-29, e1071 1.5-13, geneplotter 1.11.9, graph 1.11.14, nem 1.0.1, RBGL 1.9.9, Rgraphviz 1.11.10, Ruuid 1.11.2