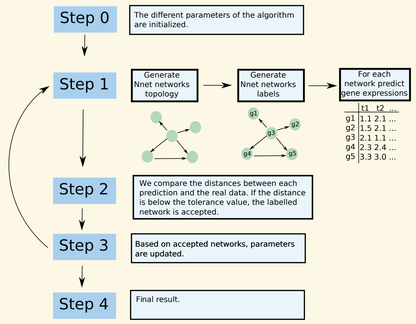
New insights in Approximate Bayesian Computation algorithms for network reverse-engineering

Nicolas Jung[a], Frédéric Bertrand[a], Myriam Maumy-Bertrand[a] and Khadija Musayeva[a]

[a]IRMA – Labex IRMIA – Université de Strasbourg & CNRS – France

Conclusion



Reverse-engineering consists in using **gene expression over time** or **over different experimental conditions** to discover the structure of the gene network in a targeted cellular process (Vallat et *al.*).

The fact that, for instance, gene expression data are usually noisy, highly correlated, and have high dimensionality explains the need for specic statistical methods to reverse engineer the underlying network.

Among known methods,
Approximate Bayesian Computation (ABC) algorithms have not been very well studied. Due to the **computational overhead** their application is also **limited to a small number of genes**.

In this work we have developed a **new multi-level ABC approach** that has **less computational cost**.

- At the **first level**, the method captures the **global properties** of the network, such as **scale-freeness** and **clustering coefficients**.
- Whereas the **second level** is targeted to capture **local properties**, including the **probability** of **each couple of genes being linked**.

Generation of a network topology

To generate a network, the **number of nodes** and the targeted **clustering coefficient** should be specified.

This algorithm is partially based on the algorithm by Di Camillo *et al*.

Let us call V the set of nodes to be connected in the graph G at the current iteration t and H the set of nodes to be connected at iteration $t + 1$. V is initialized as $V = \{1, ...,N\}$, that is, with all the N nodes in G, whereas H is initialized as the empty set H.

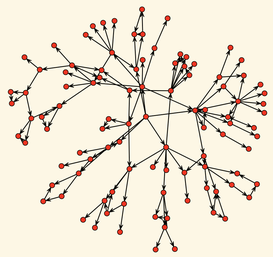
Nodes are then **linked to each other** through an **iterative procedure**, which consists of **three main steps**, explained in detail below.

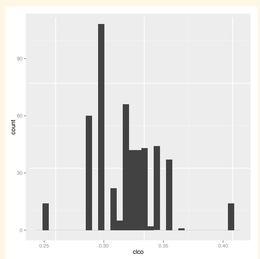
1. We generate **three candidate modules**. The structure is sampled from a **pool of motifs**, with possibility of **random changes**. The number of node of the module is set at random. In this algorithm we have : **feedback motif**, **feedforward motifs** and **loops**.
2. A score is assigned to each module, and one of the three modules is **sampled with probability proportional to this score**; let us denote the sampled module by M and the number of its nodes by m.
3. m **nodes are sampled** from V **and linked to each other** in the graph G, according to the **selected module structure** M.

At the end of this process, V is empty whereas H is composed of a lot of motifs.

To **link the motifs** together, we have to choose one node in each motif that is the first position. This set of nodes is then considered as set V .

V is updated by **deleting** the m **sampled nodes**; H is updated by **adding** the **nodes**.

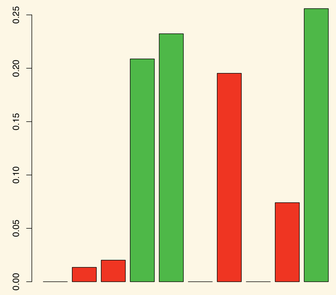




Running the ABC algorithm

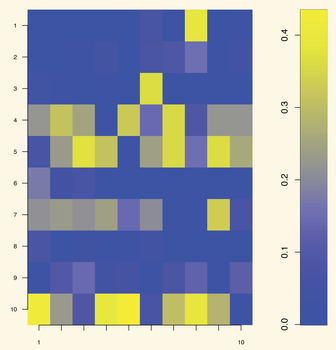
Hubs in the network?

This plot show the **probabilities** for each gene of being a **hub**.



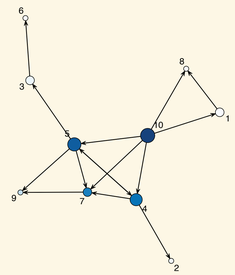
Probability of links?

The following shows the **probability** for **each couple** of **genes** of being **linked** :



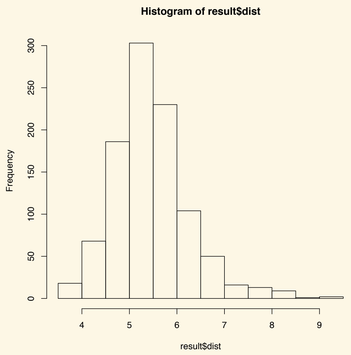
Plotting the network

The **diameter** of a node is linked to its **number of children** whereas the **color** is linked to the **probability** for each gene of **being a hub**.



Plotting the error

You can also have a **look** on the **error**:



Customizing the ABC algorithm

```
abc(data=M,
    clust_coeffs=0.33, #you can specify more
than one clustering coefficient
    tolerance=3.5, #maximal distance
between simulated and real data to accept
the network
    number_hubs=3, #the number of hubs
    iterations=10, #number of iterations
    number_networks=1000000, #number of
network simulated at each iteration
    hub_probs=NA, #specify the a priori
probabilty for each gene to be a hub
    neighbour_probs=NA, #specify the a pri-
ori probability for each couple of gene to
be linked
    is_probs=1)
```

References

Vallat, L. *et al.* (2013). Reverse-engineering the genetic circuitry of a cancer cell with predicted intervention in chronic lymphocytic leukemia. *PNAS*, **110**(2), 459–64.

Di Camillo, B., Toffolo, G. and Cobelli, C. (2009). A gene network simulator to assess reverse engineering algorithms. *Annals of the New York Academy of Sciences*. 1158(1) 125–142.

Rau, A. and Jaffrézic, F. and Foulley, J.-L. and Doerge, R.W. (2010). An empirical Bayesian method for estimating biological networks from temporal microarray data. *Statistical Applications in Genetics and Molecular Biology*. **9**(1).

Barabási, A.-L. (2002) Emergence of scaling in complex networks. *Handbook of graphs and networks: from the genome to the internet*. 69–84. Wiley Online Library.

Rau, A., Jaffrézic, F., Foulley, J.-L., and Doerge, R. W. (2012). Reverse Engineering Gene Regulatory Networks Using Approximate Bayesian Computation. *Statistics and Computing*. **22**(6) 1257–1271.