Signalling Entropy: an introductory tutorial

Andrew Teschendorff

January 3, 2014

Introduction

The aim of this vignette is to illustrate the use of a number of R-scripts for computing signalling entropy measures. The rationale and motivation for signalling entropy is explained in our previous publications [1–3]. Briefly, signalling entropy refers to a measure of the uncertainty in signalling patterns within a cellular signal transduction network. Its computation requires two inputs: (1) a PPI network, and (2) a gene expression data matrix.
There are different ways in which signalling entropy can be computed [1–3]. Here, we will describe one approach which does it at the level of individual samples, in which case the gene expression data must be positively valued (e.g. the expression data must derive from Affy, Illumina or RNA-Seq count data) and the mass action principle is used to construct the stochastic matrix over the network.
In the first step we integrate our PPI network with the gene expression data to derive a maximally connected subnetwork.

The Input Data

Let us first load in the PPI data:

```
> load("hprdAsigH-13Jun12.Rd");
```

To check what R objects have been uploaded, do:

```
> ls()
```

```
[1] "hprdAsigH.m"   "sigHclassA2.v" "sigHclassA.v"
```

which should list all the objects in the current session. The adjacency matrix specifying the PPI network is *hprdAsigH.m*. Note that the rownames of this matrix represent Entrez gene IDs. Now, let us load in the gene expression data matrix:

```
> load("dataSCM2.Rd");
```

This is Illumina gene expression data for 191 samples, with the rows of *avdataSCM2.m* annotated to Entrez gene IDs. The samples are human embryonic stem cell (hESCs) samples, induced pluripotent stem cells (iPSCs) and somatic differentiated tissue. Which of the three is a given sample is given by the *phenoSCM2.lv* list object. Specifically, the first entry of this list object specifies this: 1=hESC,2=iPSC,3=differentiated.
The first step is to integrate the gene expression data matrix with the PPI network. For this we can use the *DoIntegPIN* function:

```
> source("DoIntegPIN.R");
> pin.o <- DoIntegPIN(hprdAsigH.m,avdataSCM2.m);
```

We can check the dimensions of the resulting maximally connected subnetwork:

```
> print(dim(pin.o$a));
```

```
[1] 7757 7757
```

Similarly,

```
> print(dim(pin.o$e))
```

```
[1] 7757  191
```

should yield a matrix with the same number of rows but 191 samples, since it is the expression data matrix for the genes represented in the maximally connected subnetwork.

Estimating the Entropy Rate

Here we shall now estimate the Entropy Rate (SR) in a sample specific fashion. Before doing so, it is convenient to compute the maximum possible entropy rate for the given integrated network. To do this, we can use the function *CompMaxSR*. Note that this maximum SR only depends on the adjacency matrix, i.e. it is independent of gene expression and which sample we consider [3].

```
> source("CompSR.R");
> maxSR <- CompMaxSR(pin.o$a);
> print(maxSR);
```

```
[1] 6.279403
```

Now, let us compute the SR for each sample. Computing the SR is computationally expensive, hence, we shall only compute it for 6 samples (3 hESCs and 3 differentiated). First select the samples and extract the gene expression submatrix:

```
> hesc.idx <- which(phenoSCM2.lv[[1]]==1)[1:3];
> diff.idx <- which(phenoSCM2.lv[[1]]==3)[1:3];
> selS.idx <- c(hesc.idx,diff.idx);
> tmpE.m <- pin.o$e[,selS.idx];
```

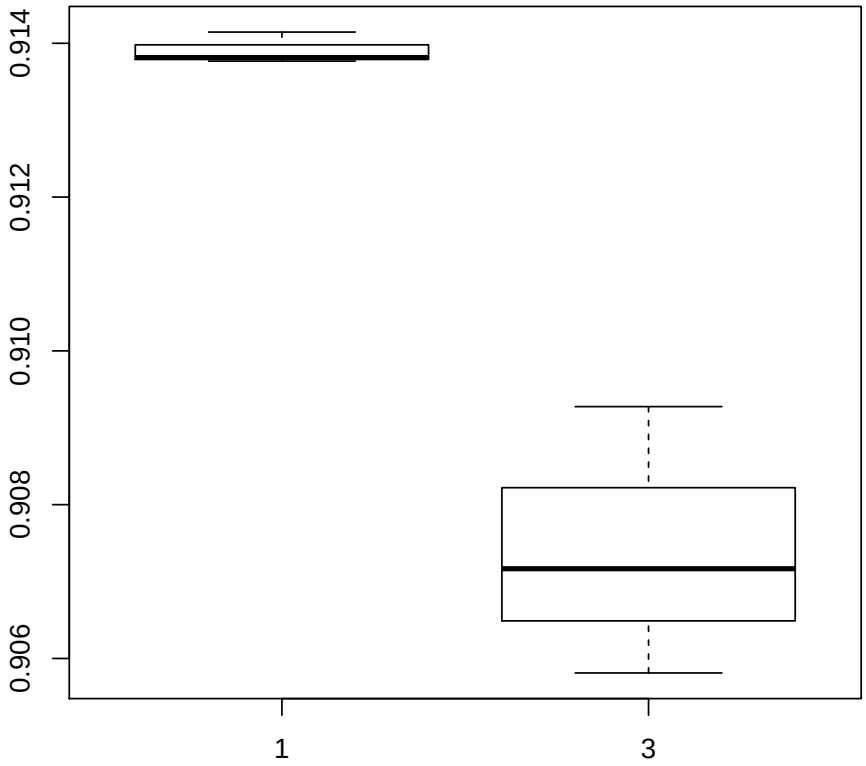
Now, we can run the *CompSR* function. Since this function acts on a gene expression profile of one sample, we can loop through (alternatively you may speed up things significantly using the *parallel* and *multicore* R-packages). Alternatively, we just load in the results:

```
> sr.lo <- list();
> k.v <- rowSums(pin.o$a);
> for(s in 1:ncol(tmpE.m)){
+ #   sr.lo[[s]] <- CompSR(tmpE.m[,s],pin.o$a,k.v,local=TRUE,method=1,maxSR=maxSR);
+ #   print(s);
+ }

> load("sr.Rd");
```

Note that we computed the degree vector *k.v* since this needs to be passed on to the function. We also specified *local=TRUE*, which means that the function will also return the local entropies for each gene in the network, not just the global entropy rate. We also specified the option *method=1*, which refers to how the stochastic matrix is estimated [3]. Specifying the option=1 means that the local stochastic vector around a gene is independent of the gene's expression value. This could be changed using the method=2 option, in

which case the other argument (*quants.v*) becomes effective. This vector specifies the quantiles for diving the gene expression values into 3 states of no or low expression, intermediate expression and high expression. Finally, we pass on the maximum SR computed earlier so that the returned global entropy rate is normalised relative to this max SR and so lies between 0 and 1.
In order to check the calculation, we extract the global entropy rates out and generate a boxplot:

```
> sr.v <- vector();
> for(s in 1:length(sr.lo)){
+   sr.v[s] <- sr.lo[[s]]$sr;
+ }

> boxplot(sr.v ~ phenoSCM2.lv[[1]][selS.idx]);
```



As expected, the entropy rates are all less than 1, and they are higher for the hESCs compared to the differentiated (somatic tissue) samples [3]. Estimating the SRs for more samples would then allow P-values to be computed, say using a Wilcoxon rank sum test, telling us if the difference in SR between the two phenotype is statistically significant or not. More generally, computing the SR of a sample can tell us about the overall level of intracellular signalling promiscuity in the sample, or, since the samples typically represent cell populations, it tells us about the overall level of gene expression heterogeneity in the sample [3].

We have seen that the global entropy rate is different between the hESCs and the somatic differentiated tissue samples. Recalling that the global entropy rate, SR, is a weighted average of the unnormalised local entropies, i.e.

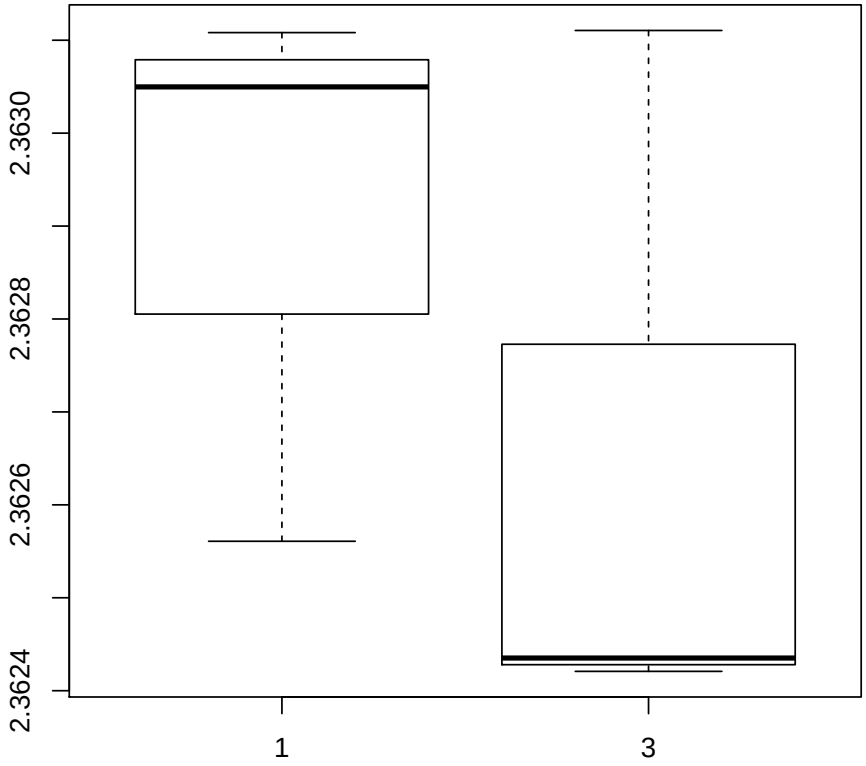
$$SR = \sum_i \pi_i S_i = -\sum_{ij} \pi_i p_{ij} \log p_{ij} \tag{1}$$

where π denotes the stationary distribution of the stochastic matrix P (P has elements p_{ij}), it should be clear that a comparison of SR between two phenotypes corresponds to a comparison of the equilibrium entropy rates, as these involve the stationary (steady-state) distribution π. This stationary distribution is largely dependent on the global topology of the network. However, in principle, one could also consider the unweighted average of the local unnormalised entropies, which would correspond to a "non-equilibrium entropy rate", i.e.

$$SR_{neq} = \sum_i S_i/N = -\frac{1}{N}\sum_{ij} p_{ij} \log p_{ij} \tag{2}$$

Is this also different between hESCs and somatic tissue? The function CompSR also returned the local entropies so this is easy to assess:

```
> srneq.v <- vector();
> for(s in 1:length(sr.lo)){
+   srneq.v[s] <- mean(sr.lo[[s]]$s);
+ }

> boxplot(srneq.v ~ phenoSCM2.lv[[1]][selS.idx]);
```

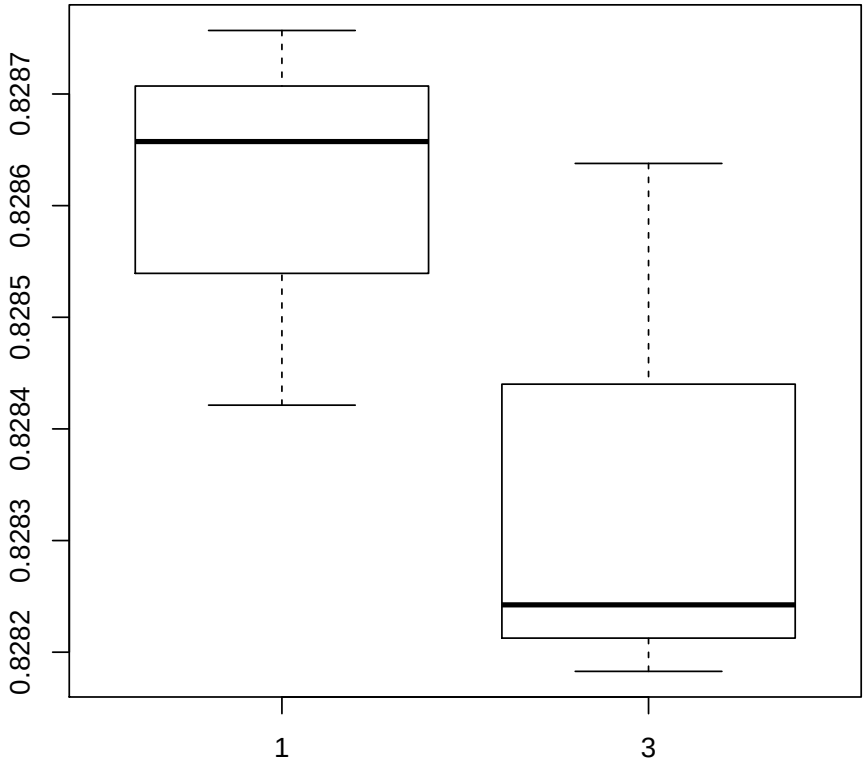


So, based on these 6 samples it would appear that the non-equilibrium entropy rate, i.e the mean of the local entropies is still different between the two phenotypes, but the discriminatory ability is reduced. The problem of using a mean of local entropies as defined above is that the dynamic range of the local entropies could vary significantly depending on the degree of the node [1]. Thus, taking the mean over nodes which could give rise to widely different entropy values is not sensible. Instread, one could consider another type of non-equilibrium entropy, defined as the average of the normalised local entropies:

$$\tilde{SR}_{neq} = \sum_i \tilde{S}_i/N = -\frac{1}{N}\sum_{ij}\frac{1}{\log k_i}p_{ij}\log p_{ij} \tag{3}$$

where each S_i has been normalised to lie between 0 and 1 (which is achieved by diving each S_i by $\log k_i$ with k_i the degree of node i).

```
> srneq2.v <- vector();
> for(s in 1:length(sr.lo)){
+   srneq2.v[s] <- mean(sr.lo[[s]]$ns);
+ }

> boxplot(srneq2.v ~ phenoSCM2.lv[[1]][selS.idx]);
```



Thus, we can see that the mean of these local normalised entropies improves the discriminatory power.
In principle, genes (nodes) can now be ranked according to differential entropy, for instance, using Wilcoxon rank sum tests to derive for each gene a P-value, which reflects the statistical significance of the difference in the local normalised entropy of the given gene between the two phenotypes.

References

[1] Teschendorff AE, Severini S (2010) Increased entropy of signal transduction in the cancer metastasis phenotype. BMC Syst Biol 4:104.

[2] West J, Bianconi G, Severini S, Teschendorff AE (2012) Differential network entropy reveals cancer system hallmarks. Sci Rep 2:802.

[3] Banerji CR, Miranda-Saavedra D, Severini S, Widschwendter M, Enver T, et al. (2013) Cellular network entropy as the energy potential in waddington's differentiation landscape. Sci Rep 3:3039.