bbcflib's RNA-seq module: methods

Julien Delafontaine

October 24, 2011

1 Introduction

The RNA-seq module takes as input BAM files resulting of the alignment of reads on the *exonome*. It counts the number of times each read mapped to each exon, and calculates the RPKM (Reads per Kilobase per Million) ratios defined as follows:

$$RPKM = 10^3 \times 10^6 \times \frac{\text{total number of reads on the exon}}{\text{exon length} \times \text{number of reads in the experiment}}$$

Counts and RPKM values will be globally called "scores" in the following. However, their analysis is different. In particular, the hypothesis we make later that counts follow approximately a negative binomial distribution cannot be applied to RPKM data.

The score of a gene is obtained by summing the respective scores of the exons it contains. The score of transcripts is harder to calculate. Since an exon can be used to build several different transcripts, it is not clear to which transcript one should attribute a read mapping on that exon. One can map directly on the transcriptome; in this case, if a read maps to k different transcripts, usually a count of 1/k will be attributed to every of them. This way one can also reconstruct the scores of a gene by summing the scores of all its transcripts, but the scores of the transcripts themselves are obviously wrong. We instead use a mathematical approach of inferring transcripts expression from exons scores, in order to optimize the distribution of the reads among transcripts.

Note that junction reads are "lost" during the mapping on exons. For the moment they are not taken into account, although there may be a non-negligible quantity of them. Also they could help the redistribution of reads telling how many times at least a transcript was produced. Bowtie can save them into a separate BAM file that one could remap on the transcriptome to improve the results. Some biologists take it as a necessary condition to trust the program. Just to give an idea, from a typical experiment [KAP1 with MEF cells, 62M mm9 reads], we got 26% of unaligned reads on the exonome, 45% of which mapped to the transcriptome, meaning that there are about 12% of reads from splice junctions. Moreover, about half of the remaining (6% of total reads) mapped somewhere else in the genome.

At the end of the pipeline, scores are normalized [DESeq]. Then we look for differential expression of genome regions across samples using a negative binomial generalized linear model (GLM) [R's `glm.nb`] with the count data.

2 Transcripts expression

2.1 Problem formulation

Suppose a gene G possesses n exons E_1,...,E_n. There can be at most $m = \sum_{k=1}^{n} \binom{n}{k} = 2^n - 1$ alternative transcripts, say T_1,...,T_m. Call e_1,...,e_n the (known) scores of exons, and t_1,...,t_m the (unknown) scores of transcripts. For each transcript T_i, define t_{ij} as the score of the part of the transcript corresponding to exon E_j, so that $t_i = t_{i1} + \ldots + t_{1n} = \sum_{k=0}^{n} I_{ik} \cdot t_{ik}$, where $I_{ik} = 1$ if transcript T_i contains exon E_k, 0 otherwise.

Conversely, we have $e_j = \sum_{k=1}^{m} I_{kj} \cdot t_{kj}$, $\forall j$. In matrix form,

$$\begin{pmatrix} e_1 \\ \vdots \\ e_n \end{pmatrix} = \begin{pmatrix} I_{11} & \cdots & I_{m1} & 0 & \cdots & & \cdots & 0 \\ \vdots & & & & & & & \vdots \\ 0 & \cdots & & \cdots & 0 & I_{1n} & \cdots & I_{mn} \end{pmatrix} \begin{pmatrix} t_{11} \\ t_{21} \\ \vdots \\ t_{m,n-1} \\ t_{mn} \end{pmatrix}$$

This is equivalent to just equally distribute the score of each read over the overlapping transcripts (what some usual alignment programs do), which is wrong because all parts of a same transcript must be equally transcribed.

One can solve

$$\begin{pmatrix} e_1 \\ \vdots \\ e_n \end{pmatrix} = \begin{pmatrix} I_{11} & \cdots & I_{m1} \\ \vdots & & \vdots \\ I_{1n} & \cdots & I_{mn} \end{pmatrix} \begin{pmatrix} \bar{t_1} \\ \vdots \\ \bar{t_m} \end{pmatrix},$$

and then multiply $\bar{t_1}, \ldots, \bar{t_m}$ by the number of exons each transcript contains, to obtain the RPKM values for $t_1, \ldots, t_m$. This method would only hold for count data if all exons had the same length, but typically it is very variable - RPKM values by definition are normalized with respect to the length of the exons.

To get counts as well, we consider the relative sizes of the different parts of a transcript and include this information in the matrix instead of binary values. For a given transcript T_i made from exons $e_1, \ldots, e_r$ (change indices if necessary), define $l_1, \ldots, l_r$ the lengths of the respective exons, and $L = l_1 + \ldots + l_r$ the total length of the transcript. We have $t_i = t_{i1} + \ldots + t_{1n} = \alpha_{i1}t_{i1} + \alpha_{i2}t_{i1} + \ldots + \alpha_{in}t_{i1}$, where $\alpha_{ij} = \frac{l_j}{L}$ is the relative length of the j-th exon with respect to the whole transcript. Now defining $\bar{t_i} := t_{i,min\{j|E_j \in T_i\}}$, the linear system then becomes

$$\begin{pmatrix} e_1 \\ \vdots \\ e_n \end{pmatrix} = \begin{pmatrix} \alpha_{11}I_{11} & \alpha_{21}I_{21} & \cdots & \alpha_{m1}I_{m1} \\ \vdots & & & \vdots \\ \alpha_{1n}I_{1n} & \alpha_{2n}I_{2n} & \cdots & \alpha_{m-1,n}I_{mn} \end{pmatrix} \begin{pmatrix} \bar{t_1} \\ \vdots \\ \bar{t_m} \end{pmatrix},$$

Calling $E = (e_1, \ldots, e_n)^T$, $T = (\bar{t_1}, \ldots, \bar{t_m})^T$, and M the matrix in the middle, one has to solve for T the system of linear equations $E = MT$. Usually M will not be invertible, and m is greater than n (overdetermined system), so the solution can only be an approximation. The problem becomes to solve

$$\min_T ||E - MT||_2^2$$

One can show that the optimal solution is given by the Moore-Penrose *pseudo-inverse* of M, as follows.

2.2 Pseudo-inverse

Let $M = UDV^T$ be the singular value decomposition of the $n \times m$ matrix M. U is $n \times n$ unitary, D is $n \times m$ diagonal, and V is $m \times m$ unitary. D contains the singular values of M on its diagonal left block, and zeros on the right - if $m > n$ - or at the bottom - if $n > m$. Then the matrix $M^+ = V(D^{-1})^T U^T$ is called the pseudo-inverse of M, and $\tilde{T} = M^+E$ is the optimal solution of our problem, i.e. $\forall x,\ ||E - Mx||_2 \geq ||E - M\tilde{T}||_2$.

This method is guaranteed by some theorem to converge to the global minimum. However, it arises a very annoying issue: $\tilde{T}$ can have - often has - negative components, which are incompatible with the notion of counts or RPKM.

2.3 Non-negative least-squares

Adding a constraint of positiveness on the scores, the problem becomes

$$\min_T ||E - MT||_2^2 \quad \text{, with constraint} \quad T \geq 0$$

This can be solved by Alternative Nonnegative Least-Squares (ANLS). This kind of problem occurred rather recently in applications, and faster algorithms are still under development. Most of them depend a lot on initialization and insure to converge to a *local* minimum only. Note that a local minimum can be sufficient for most applications (?).

For our particular case, we used a Python equivalent of Matlab's `lsqnonneg` function [`http://diffusion-mri.googlecode.com/svn/trunk/Python/lsqnonneg.py`], itself inspired by an algorithm of Lawson and Hanson [C.L. Lawson and R.J. Hanson, Solving Least-Squares Problems, Prentice-Hall, 1974, ch.23, p.161].

2.4 Analysis of the error

Since we are trying distribute the exon counts among transcripts, the sum of scores over all exons in a gene should be approximately the sum over all its transcripts.

The relevance of the least-squares approach is given by the differences $||E - MT||_2$.

The rank of M is bounded by $\min(n, m)$. Usually, there are less alternative transcripts than exons ($m < n$). This is equivalent to project - when estimating M^+E - n-dimensional data onto an m-dimensional subspace. Therefore, scores on transcripts can be only inferior to exon scores, and there is a - minimized - loss of information. But for usual analyses that consider ratios between different conditions, we can hope that the list of differentially expressed transcripts remains the same.

A typical experiment [still KAP1 with MEF cells] lead to the following results.

The sum of negative components generated by the pseudo-inverse solution accounts for 16% of the total counts (6% of RPK) on transcripts.

The ratio between the total exon RPK and transcript RPK with NNLS is about 93% - so 7% of the initial counts are lost.

Surprisingly, the size of least-squares errors are not correlated with the ratio n/m.

The unconstrained least-squares problem is the maximum likelihood estimator for linear regression of *normally distributed* data. Count data must use different models. Weighted least-squares applied to the exponential of our data (?) correspond to the generalized linear model we want to apply for differential expression analysis. However we believe that the NNLS solution is equivalent.

2.5 Normalization

3 GLM analysis module

Encouraged by an article showing the comparison between different kinds of analysis modules [ref], we chose to replace the `R` package `DESeq` by a Generalized Linear Model (GLM), i.e. a non-linear regression model. The idea is to take into account the effect of the different parameters that define each of the different conditions, instead of only computing the ratios between scores in one condition and another. Since our observations (counts) seem to have a negative binomial distribution [ref], a standard (normal) regression is not appropriate.

3.1 Remainder about GLMs

A regression model tries to explain a response variable Y by covariates $X = (X_1, X_2, \ldots)$. For instance, the normal linear model is defined as $Y|X \sim N(X\beta, \sigma^2)$, β being a vector of parameters to estimate.

A GLM is determinate by the distribution of $Y|X$, and a *link function* g that connects the mean of Y, μ, and the linear term:

$$\mu|X = g(X\beta)$$

Once chosen the distribution of Y, one must choose a link function that maps the domain of Y to that of X. In particular there is a unique *canonical* link function that guarantees the existence of a minimal sufficient statistic for β, and thus makes more sense with distributions of the exponential family (such as negative binomial *only if the dispersion parameter θ in known and constant*). In the negative binomial case, it is $\log(\frac{\mu}{\mu+\theta})$, but is rarely used. One usually prefers the logarithm. Square root or even identity link functions are also supported in `R`. We are using the logarithm for our application.

3.2 Our data

One usually wants to compare samples from different experimental conditions. Each condition is called a "group", called $g_1, \ldots, g_l$. Samples from a same group ("runs") are biological or technical replicates. A typical output of the RNA-seq module is a tab-delimited file of the form

	g1.1	**g1.2**	**g1.3**	**g2.1**	**g2.2**	**g2.3**	**g3.1**	**g3.2**	**g3.3**
feat1	178	138	83	1935	53	120	263	928	492
feat2	159	142	112	504	442	528	577	156	1244
feat3	59	19	166	58	1130	826	103	463	852
...									

Each numeric column is a different sample, labeled `group_name.run_index`. Each line represents a different feature (gene, exon, or transcript). We build a different GLM for each feature.

3.3 Design

The design matrix indicates which combination of parameters defines which condition. Our covariates are the parameters that determine our conditions, e.g. temperature, treatment, peak in ChIP-seq data, a.s.o. An example of design matrix could be

	g1	**g2**	**g3**
Temperature (T)	30	40	60
Treatment (tr)	1	1	1
ChIP_peak (Ch)	0	0	1

3.4 Formulation of the GLM

Let us take one of our genomic features, say f. Our response variable Y is the score vector of f in d different samples:

$$Y = (f_1, \ldots, f_d)^T$$

We can write the model as

$$\begin{pmatrix} f_{g1.1} \\ f_{g1.2} \\ f_{g1.3} \\ f_{g2.1} \\ f_{g2.2} \\ f_{g2.3} \\ f_{g3.1} \\ f_{g3.2} \\ f_{g3.3} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} \beta_{T30} \\ \beta_{T40} \\ \beta_{T60} \\ \beta_{tr0} \\ \beta_{tr1} \\ \beta_{Ch0} \\ \beta_{Ch1} \end{pmatrix}$$

For each feature, we get an estimate $\beta = (\beta_{T40}, \ldots, \beta_{Ch1})$ that gives the contribution of each covariate to the total variance of Y.

3.5 Contrasts

The contrasts matrix indicates which tests we want to perform. An example contrasts matrix looks like

3.6 Interpretation of the results

The p-value associated to each feature in each group is the result of a test of the null hypothesis $\beta_C = 0$ for the corresponding covariate C. A small p-value means that the coefficient accounts for a non-negligible part of the total variance.