MIC653: Advanced Molecular Genetics Bioinformatics & Computational Genomics

Projects: Computation to Solve Problems

Analysis of gene expression data (uses CyanoBIKE)

About 20 years ago microarrays were introduced to measure simultaneously the expression of thousands of genes. With the drastic decrease in cost of sequencing, RNAseq has largely supplanted microarrays as a tool to measure gene expression. Despite the shift in methodology, certain problems remain, for example the issue reproducibility and the problem of how to compare different experiments performed under different conditions. There are many programs available to help researchers work through these problems and analyze gene expression data. However, what can you do if you want to analyze the data in a way that was not anticipated by those who wrote those programs. For that, you need to be able to program the computer yourself.

A. Introduction to computational analysis of microarray data in BioBIKE

A tour entitled Combining pathways with microarrays* takes you through a specific problem of gene expression analysis, using BioBIKE functions. Go through this tour, replicating each step within CyanoBIKE.

B. How to compare the results of different microarray experiments?

The main purpose of measuring gene expression is usually to determine a ratio of some sort, the change of expression at some time point or condition relative to another. It is also useful at times to know absolute levels of expression – perhaps expression went up by a factor of two, but did it rise from a high level to a higher level or a very low level to a still low level. Attempts to do this are bedeviled by the variability in expression measurements, which is only partially addressed by replicate measurements. While measurements of gene expression via RNAseq generally have less variability than measurements via microarrays, the issue still remains, and considering how it can be handled with microarrays may offer general insight into the problem.

1. Look at the raw expression data from the Ehira experiment, i.e. the fluorescence intensity measurements before statistical manipulation. To do this, bring down INSIDE-MICROARRAY, and specify ehira_2006 as the experiment, 2 as the condition, and CODING-GENES-OF A7120 for the gene (all the genes measured by the microarray). Choose the options RAW and +CONTROL-VALUES, getting the following:

   Each line gives you the six replicate measurements for one gene. Note the range of values from amongst replicates of one gene and amongst genes.

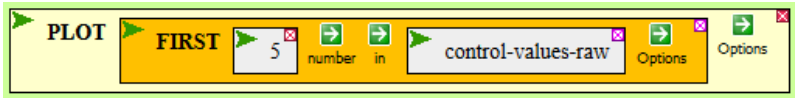
2. Of course there is variation from one replicate to the next, but Are the replicates systematically different from one another? To determine this DEFINE a variable (maybe call it *control-values-raw* and drag the above function into the *value* box. Before executing, choose the –LABELS option in addition to the others. This will suppress

* Available by clicking the link or by going to the BioBIKE portal (http://biobike.csbc.vcu.edu), clicking the link to the guided tours, and choosing the tour called *Combining Pathways with Microarray Data*.

listing of the names of the genes. You'll get just the numbers, which will simplify matters. Execute the function and verify that the result (in the Result Pane) has the same numbers as before.

3. Now to plot the distribution of values for each replicate. Right now, the numbers are grouped horizontally by gene. We want the same numbers grouped horizontally by replicate. You can swap vertical for horizontal by using the TRANSPOSE-LIST function. Bring that function down into the workspace and put *control-values-raw* into the *list* box. Execute the function, and examine the result to verify that the transposition has taken place.

4. The second step in plotting the distributions is to bring down the PLOT function into the workspace. Unfortunately, this function can display no more than five distributions at one time, so we'll use just the first five replicates. In the *list-or-table* box bring down the FIRST function, mouse over the *number* icon and click *number*, and enter 5 for *n*. Then drag into the *entity* box the TRANSPOSE-LIST function you made in Step 3. Finally execute the function, which should look something like what you see below:



5. This plot shows the raw expression value for each gene, from the first at the left to the 5336th at the right. That's not what we want. We want a distribution, where the X-axis is the expression value. To change the nature of the plot in this way, go back to the PLOT function and choose the BIN-INTERVAL. If you enter 1 as the BIN-INTERVAL *value*, then the function will count how many values lie between 0 and 1, 1 and 2, 2 and 3, etc. Execute the function…

6. Still not very helpful, because almost all of the values are scrunched near 1. To spread out the distributions, go back to the PLOT function and choose the options MAX, specifying 300 as the maximum value on the X axis. Now when you execute the function, you should see 5 distributions. Do you see them? Are they the same?

7. That was the raw values. Now repeat Steps 1 through 6, replacing RAW in Step 1 with NORMALIZED-BY-MEDIAN. What do you make of the plotted distributions?

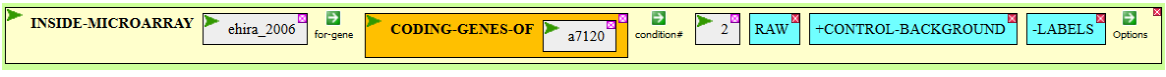
8. The replicates are now much more comparable to each other. How was this achieved? NORMALIZED-BY-MEDIAN achieves the compression in the following steps for each replicate:

    a. Subtract from each raw value the corresponding background value. Call this the *net expression*.

    b. Find the median value for the net expression

    c. Divide each net expression by the median value

We can do this ourselves. First DEFINE *control-background* as the raw background values, obtained by INSIDE-MICROARRAY:



Execute this function.

9. Now subtract the control background from the control values, using DIFFERENCE-OF, and call it *net-expression*:

Check the first result in the Result Pane – Are the numbers indeed what is expected from the subtraction?

10. Calculate the median net expression using the MEDIAN function. Bring it into the work space and… problem! MEDIAN calculates the median of a simple list of numbers, but *net-expression* contains a list of lists, 5336 lists, each containing the values of the 6 replicates for the expression of a single gene. We need 6 lists, each containing the 5336 values for a single replicate. This part of the problem was already solved – see Step 3, and execute the function with *net-expression* as the *list*.

11. The second part of the problem is feeding MEDIAN the six lists one at a time. Learning how to do this sort of thing – iteration – is perhaps the biggest single hurdle in going from a non-programmer to a programmer. To do the iteration, construct the following function:

APPLY-FUNCTION gives MEDIAN the simple list it requires (called *replicate-list*), taken one at a time from the transposed *net-expression.* Executing this function should give you 6 medians, one for each replicate. Does it? Do the numbers make sense, from your observations of the numbers in *net-expression*?

12. Almost there. Divide *net-expression* by medians, using QUOTIENT-OF. Execute the function and compare the first result with the first result of the normalized values (Step 7). How do they compare? Why?

If you've made it this far, then you have gone through the process of normalizing a microarray, a procedure required in gene expression analysis whether you use microarrays or RNAseq.