LogTransform Documentation

Description: Apply a log transformation on a GCT file.

Author: Dr Mark Cowley (Garvan Institute of Medical Research, Sydney Australia), m.cowley@garvan.org.au

Kevin Ying (Garvan Institute of Medical Research, Sydney Australia), k.ying@garvan.org.au

Date: September, 2011

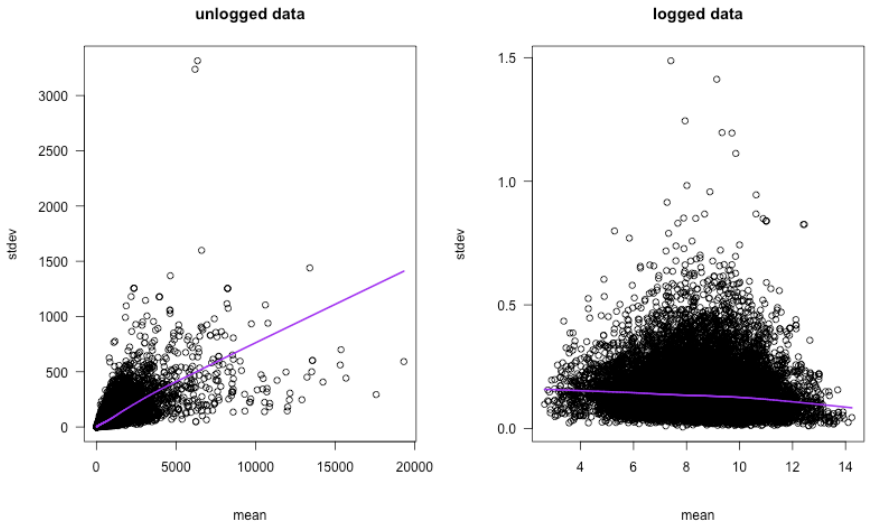
Release: 3.3

Summary

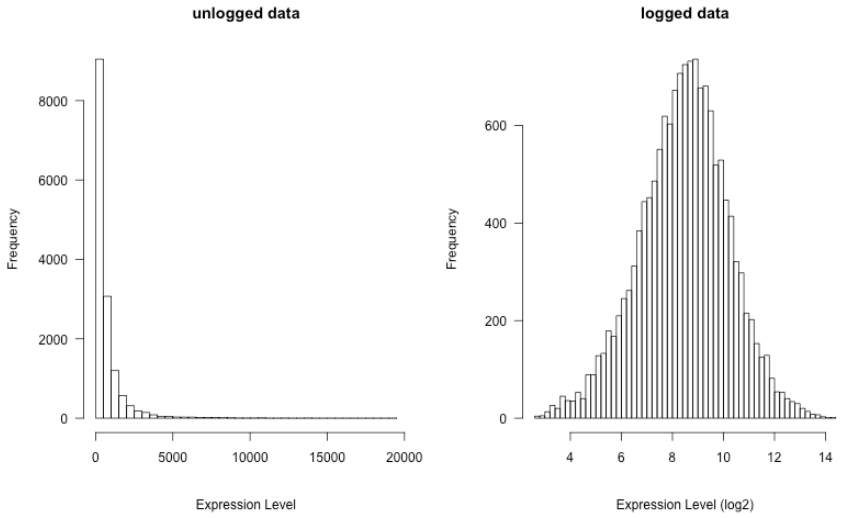
Apply a log base 2 transformation to data from a GCT file.

Why? Typically microarray data are log transformed. This is done for a number of reasons, including: it stabilizes the variance; it compresses the range of data; and it makes the data more normally distributed, which allows statistics to be applied to the data.

- ***Stabilizes the variance***: for data on the linear scale, as the expression level/signal gets higher, the variance also increases. Log transforming reduces this dependence, so that the variance is more consistent throughout the range of expression
- ***Compresses the range of data***: In a typical microarray experiment, most genes have levels <100, with very few having large levels >10,000. When this data is log transformed, the range shrinks from usually 1-65,000 to 0-16.



- ***Normally distributed data***: For performing t-statistics and ANOVA's one of the assumptions is that the errors* are normally distributed. This is absolutely not the case for unlogged data & this violation of the statistical assumption makes the statistical findings on unlogged data invalid.
    - picture a bell curve, with a mean, and symmetrically distributed 'errors', vs an "|_____" shaped curve with a mean way off to the left and errors that are not symmetric



If all data are >= 1.0, then this is simple – the data will just be log transformed.

However, you cannot log-transform values <= 0, so if your data has some negative values, then there are 3 simple strategies:

1. Truncate all values that are <= 1.0 to 1.0, and then log transform.
2. Add a positive value to all data points, so that the minimum value becomes 1.0, then log transform
3. Add a small positive value (eg 8) to all data points, which makes most data >= 1.0, and then truncate all remaining values <= 1.0 to 1.0

1) Is the simplest strategy. If the data is <= 1, then it's probably too low to be interesting anyway, so just truncate the data to 1.0

2) This method ensures all data is positive. The downside to this is just one very negative outlier, eg at -1200 causes a value of 1201 to be added to all data points which will have a big impact upon the lowly-moderately expressed genes

3) This is a good compromise between 1 & 2. I've found with Agilent microRNA data, adding +8 to all data pushes 85% of the values <=1 to >1, and the remaining 15% of data will become truncated.

HINT: Since log2(1.0) = 0.0, you may want to filter out those probes whose log-transformed values are all = 0. You can use the *PreprocessDataset* module (*number of columns above threshold*=1 and *column threshold*=0.1), or the *GeneFilter* module (coming soon).

Why do some microarray data have negative values anyway?

It's usually because of background subtraction. If a gene is expression close to, or just below the 'background' level on the array, then these data may become negative. This is certainly true is you use the *IlluminaExpressionFileCreator*, with the background subtraction mode = true, and is true for Agilent microRNA data.

Usage

To apply the strategies described above:

1. Truncating all values <= 1.0 to1.0
   - NB: This step will be automatically applied by the module before log transformation.
2. Add a positive value to all data points so that the minimum value will be at least 1.0.
   - Type "auto" in place of the "offset.value"
3. Add a small positive value to all data points and truncate remaining points.
   - By specifying a value yourself (eg. 8) as the "offset.value"

To help you decide how to handle values <= 1.0, LogTransform will print out a summary of the data before and after applying log transformation. Run the module once with your GCT file and open "stdout.txt". The summary should look something like:

```
negvals.unlogged.gct contains:
24 samples and 47323 genes
352589 (31.000 %) out of 1135752 are < 1
329203 (29.000 %) out of 1135752 are < 0
Min. : -47.846500
Max. : 38885.120000
```

References & Links

Parameters (* = required)

Name	Description
input file*	The GCT file to be log transformed
output file*	The name of the log transformed GCT file to write out. Default: <input.file_basename>_log2.gct

offset value	The numeric amount to add to the data before log transformation. "auto" – auto-offset: shift all data so that values will be ≥ 1 Default: 0

Input Files

1. **input file**
   An unlogged GCT file.
   All values <= 1.0 will be set as 1.0 before applying the log

Output Files

output file

The log transformed GCT file

stdout.txt

A file containing details of the run, including a summary of the data before and after log transformation (see above: Usage)

Warning/Error Messages:

"WARNING: data may already be log transformed"

The module checks if all values in the original GCT is below 60, if so, it is likely that the GCT file has already been log transformed.

Example Data

ftp://ftp.broadinstitute.org/pub/genepattern/datasets/all_aml/all_aml_test.gct

Citing this module

Cowley, M.J., Ying, K., *LogTransform* – a GenePattern module for applying a log transformation on GCT files (not published).

Platform Dependencies

Module type:	Preprocess & Utilities
CPU type:	any
OS:	any Tested on Ubuntu 10.10 Not tested on Windows
software	nil
Language:	R >= 2.5