

10-810: Advanced Algorithms and Models for Computational Biology

Normalization

Gene Expression Analysis

Model



Pattern Recognition



Data Analysis



Experimental Design

Computational

Biological

information fusion

regulatory
networks

clustering,
classification

functional
assignment,
response
programs

normalization, miss.
value estimation

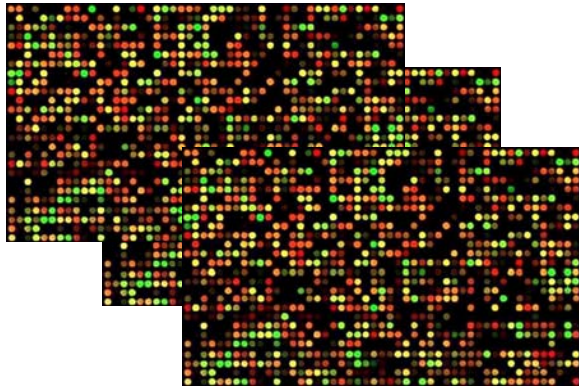
diff. expressed
genes

array design,
number of repeats

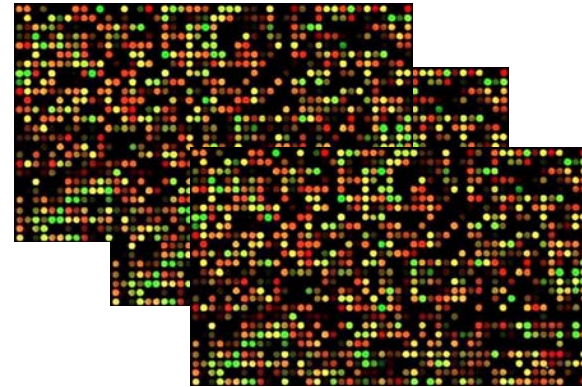
experiment
selection

Typical experiment: replicates

healthy



cancer



Technical replicates: same sample using multiple arrays

Dye swap: reverse the color code between arrays

Clinical replicates: samples from different individuals

Many experiments have all three kinds of replicates

Data analysis

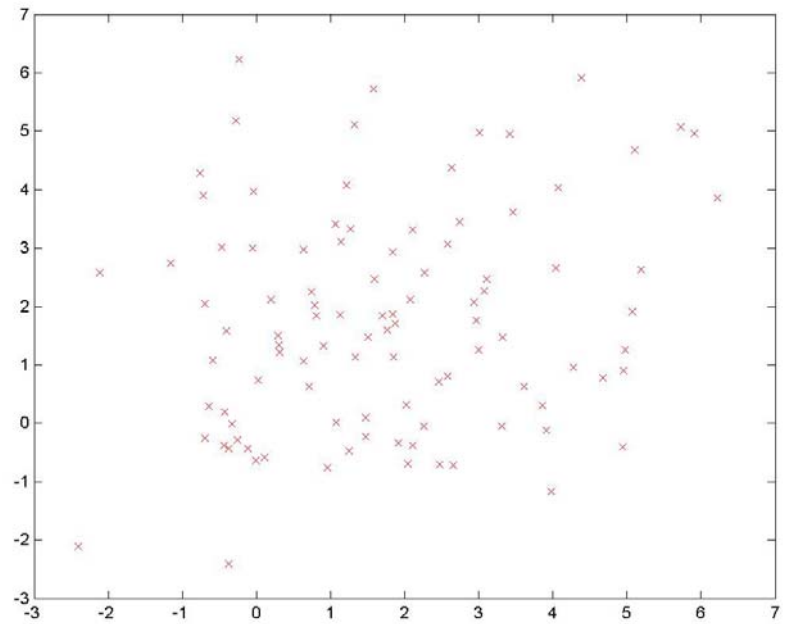
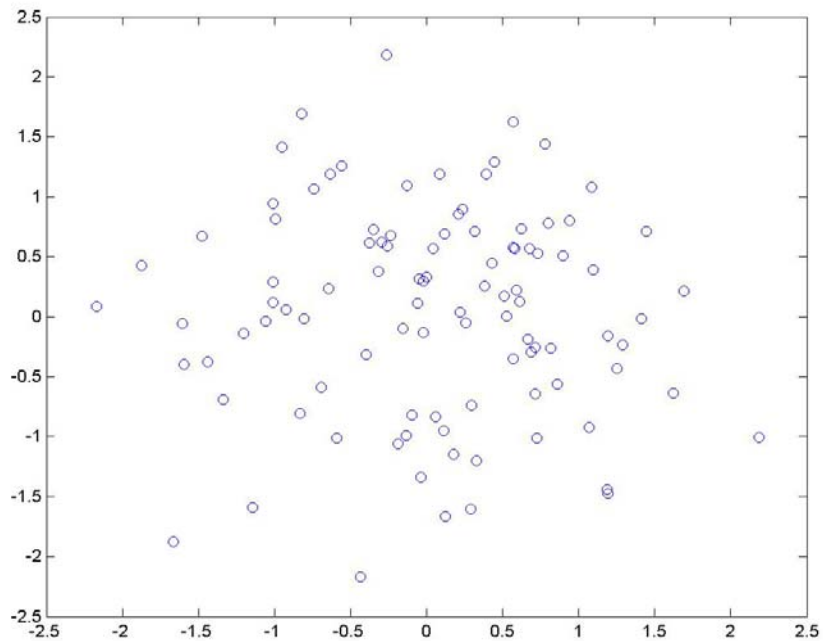
- Normalization
- Combining results from replicates
- Identifying differentially expressed genes
- Dealing with missing values
- Static vs. time series

Data analysis

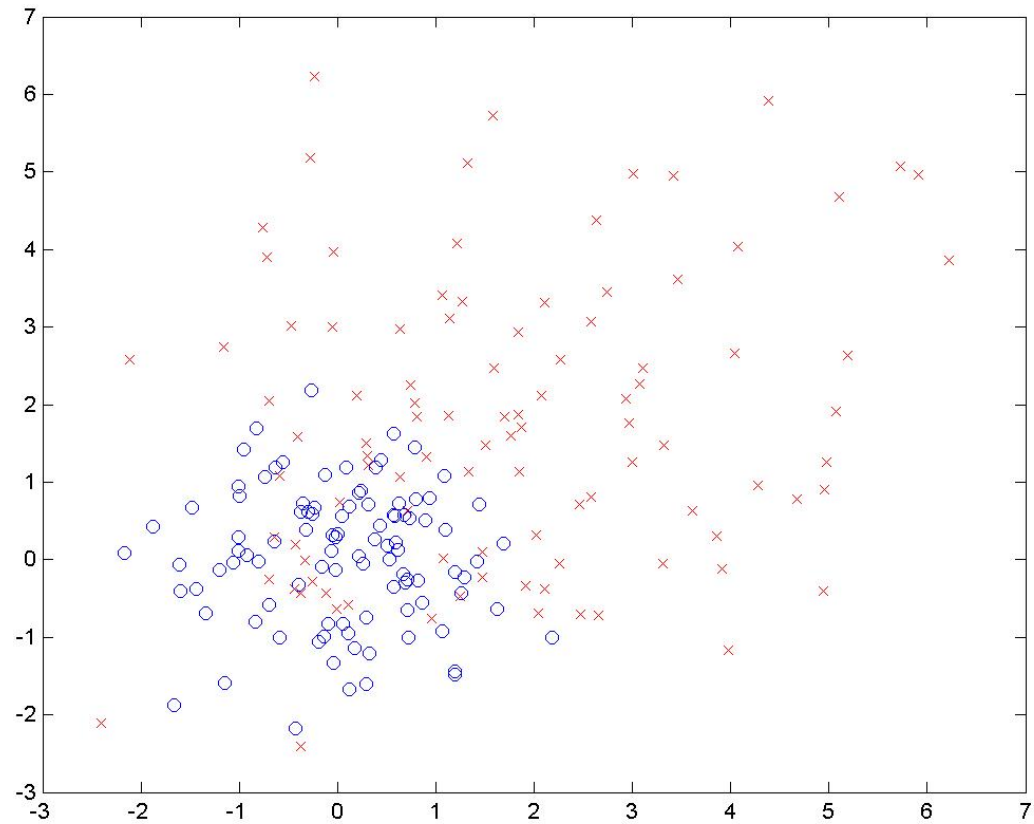
- Normalization
- Combining results from replicates
- Identifying differentially expressed genes
- Dealing with missing values
- Static vs. time series

Normalizing across arrays

- Consider the following two sets of values:



Lets put them together ...



Normalizing between arrays

The first step in the analysis of microarray data in a given experiment is to normalize *between* the different arrays.

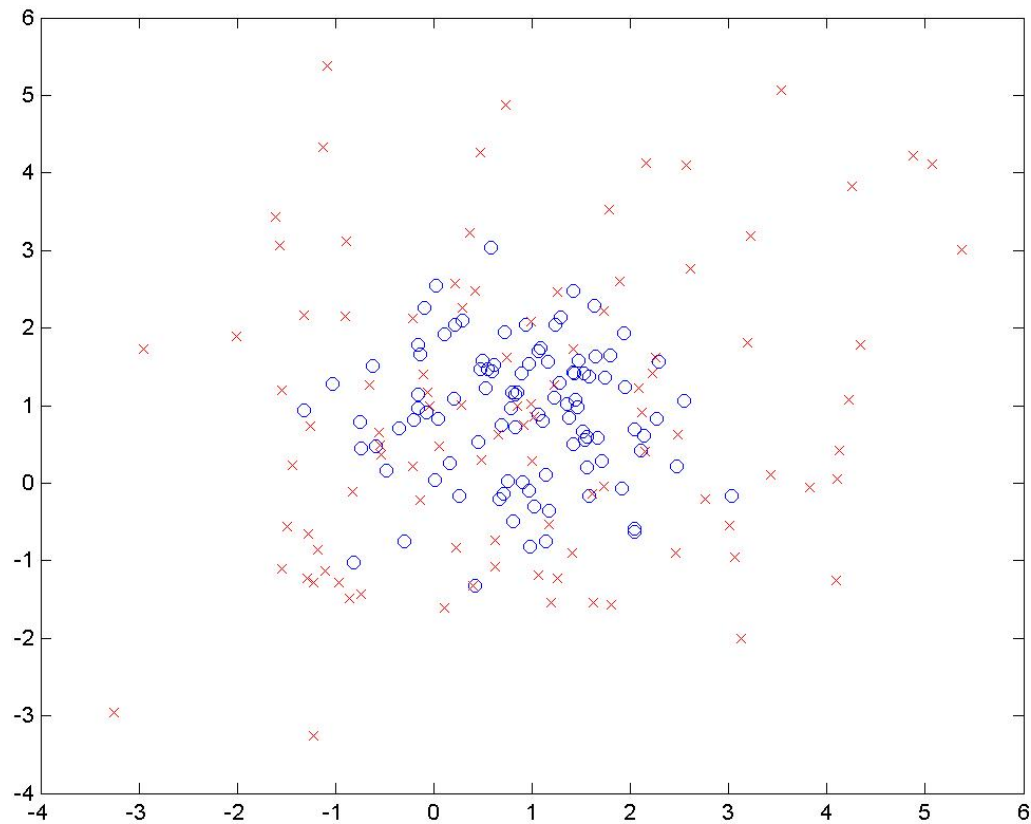
- **Simple assumption: mRNA quantity is the same for all arrays**

$$M^j = \frac{1}{n} \sum_{i=1}^n y_i^j \quad M = \frac{1}{T} \sum_{j=1}^T M^j$$

- Where n and T are the total number of genes and arrays, respectively. M^j is known as the **sample** mean
- Next we transform each value to make all arrays have the same mean:

$$\hat{y}_i^j = y_i^j - M^j + M$$

Normalizing the mean



Variance normalization

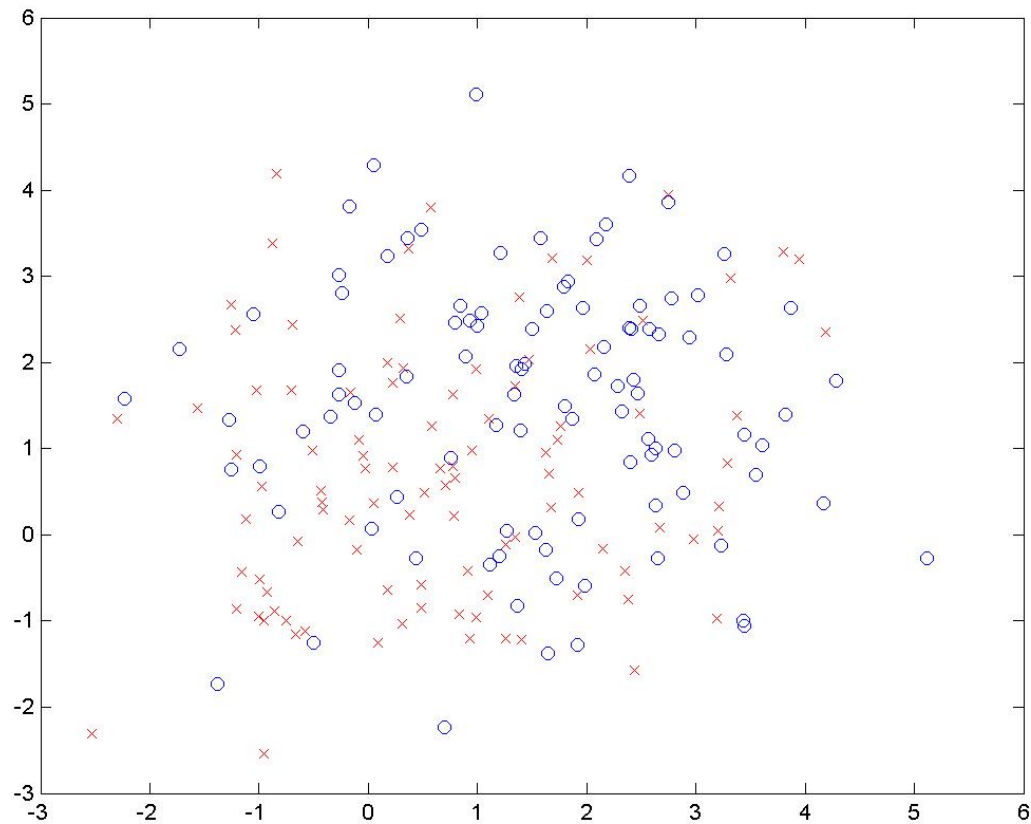
- In many cases normalizing the mean is not enough.
- We may further assume that the *variance* should be the same for each array
- **Implicitly we assume that the expression distribution is the same for all arrays** (though different genes may change in each of the arrays)

$$V^j = \frac{1}{n} \sum_{i=1}^n (y_i^j - M^j)^2 \quad V = \frac{1}{T} \sum_{j=1}^T V^j$$

- Here V^j is the sample variance.
- Next, we transform each value as follows:

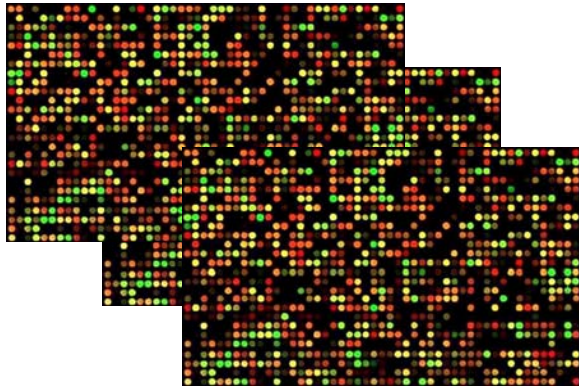
$$\hat{y}_i^j = \frac{(y_i^j - M^j + M)\sqrt{V}}{\sqrt{V^j}}$$

Normalizing mean and variance

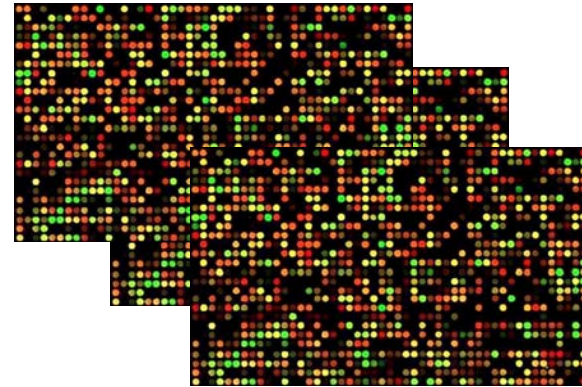


Typical experiment: ratios

healthy



cancer



- In many experiments we are interested in the *ratio* between two samples
- For example
 - Cancer vs. healthy
 - Progression of disease (ratio to time point 0)

Transformation

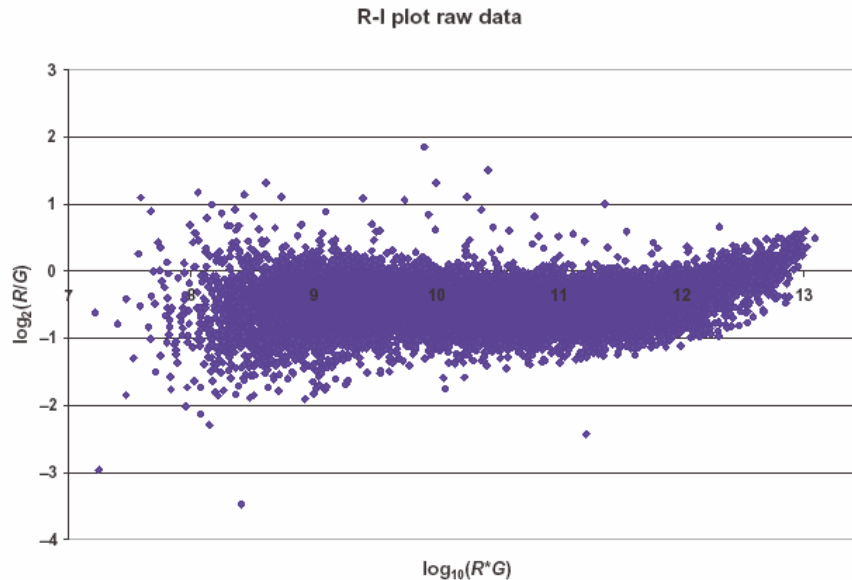
- While ratios are useful, they are not symmetric.
- If $R = 2 * G$ then $R/G = 2$ while $G/R = 1/2$
- This makes it hard to visualize the different changes
- Instead, we use a log transform, and focus on the *log ratio*:

$$y_i = \log \frac{R_i}{G_i} = \log R_i - \log G_i$$

- Empirical studies have also shown that in microarray experiments the log ratio of (most) genes tends to be normally distributed

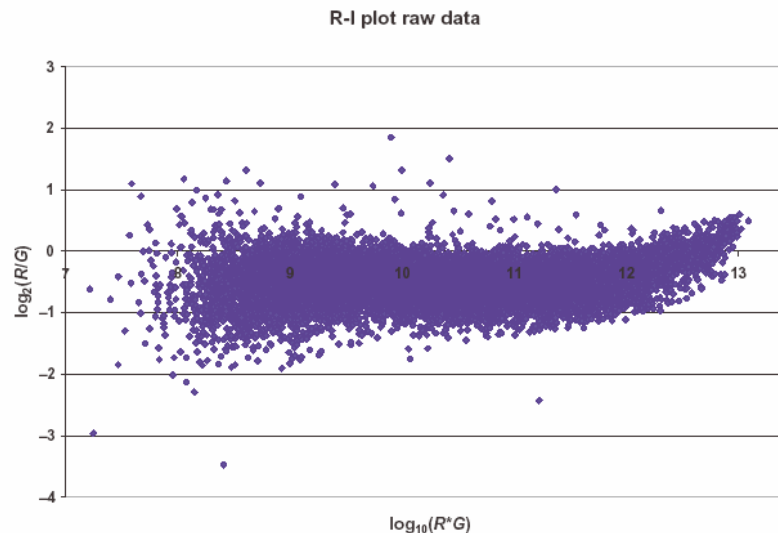
Normalizing between array: Locally weighted linear regression

- Normalizing the mean and the variance works well if the variance is independent of the measured value.
- However, this is not the case in gene expression.
- For microarrays it turns out that the variance is value dependent.



Locally weighted linear regression

- Instead of computing a single mean and variance for each array, we can compute different means and variances for different expression values.
- Given two arrays, R and G we plot on the x axis the (log) of their intensity and on the y axis their ratio
- We are interested in normalizing the average (log) expression ratio for the different intensity values



Computing local mean and variance

- Setting

$$m(x) = \frac{1}{k} \sum_{x_i=x} y_i \quad v(x) = \frac{1}{k} \sum_{x_i=x} (y_i - m(x))^2$$

may work, however, it requires that many genes have the same x value, which is usually not the case

- Instead, we can use a *weighted* sum where the weight is proportional to the distance of the point from x:

$$m(x) = \frac{1}{\sum_i w(x_i)} \sum_i w(x_i) y_i \quad v(x) = \frac{1}{\sum_i w(x_i)} \sum_i (w(x_i) y_i - m(x))^2$$

$$\hat{y}(x) = \frac{(y(x) - m(x) + M)\sqrt{V}}{\sqrt{v(x)}}$$

Determining the weights

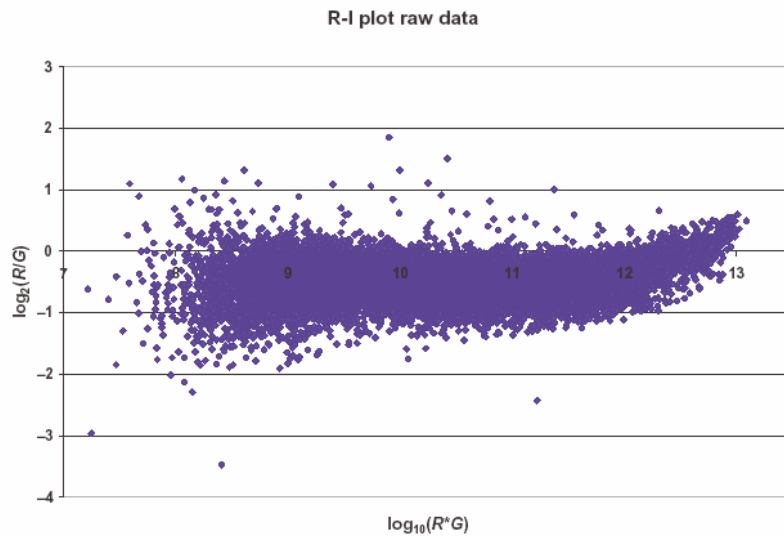
- There are a number of ways to determine the weights
- Here we will use a Gaussian centered at x , such that

$$w(x_i) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{(x-x_i)^2}{2\sigma^2}}$$

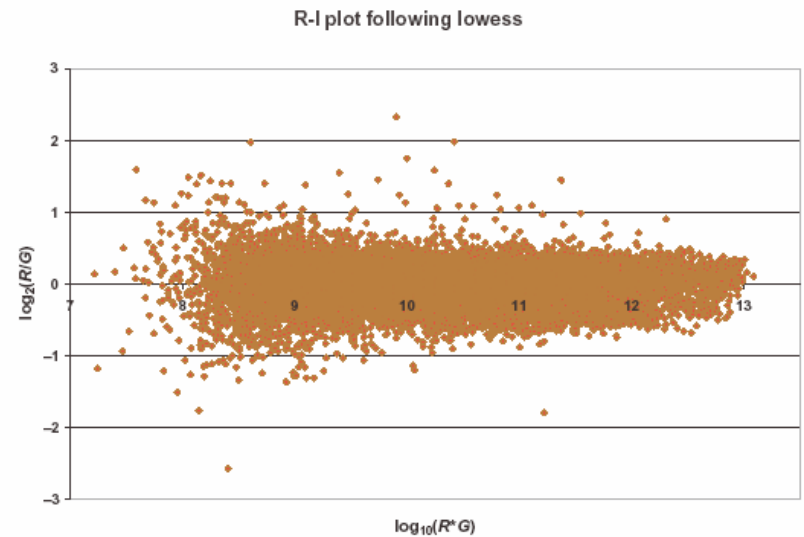
σ^2 is a parameter that should be selected by the user

Locally weighted regression: Results

Original values



normalized
values



Other normalization methods

- If you are not comfortable with the equal mRNA assumption, there are other possible normalization methods:
- We can use genes known as ‘house keeping genes’. **These genes are assumed to be expressed at similar levels regardless of the condition the cell is in.**
- Alternatively, we can use ‘controls’ . These are sequences that are manually inserted into the sample with *known* quantities (this is mainly useful for oligo arrays).

Using spike controls

- Suppose we have m raw measurements of spiked controls per chip and T chip experiments altogether
- We need to construct a model over these observations that disentangles the experiment dependent scaling and the underlying (supposedly fixed) control levels

x_1^1	x_1^T
.
.
x_m^1	x_m^T

Determining the underlying expression

We can try to learn the parameters of a model that attempts to disentangle the experiment dependent scaling and the underlying (fixed) control levels :

$$\begin{aligned}x_i^1 &= m_i r^1 e_i^1 \\ &\vdots \\ x_i^T &= m_i r^T e_i^T\end{aligned}$$

Here:

- x_i^j is the j 'th measurement for control i
- m_i is the fixed control amount
- r^j is the unknown experiment dependent scaling
- e_i^j is random multiplicative noise

Log transform

Log-transform all the variables

$$\log x_i^1 = \log m_i + \log r^1 + \log e_i^1$$

$$y_i^1 = \log x_i^1, \mu_i = \log m_i, \rho^1 = \log r^1, \varepsilon_i^1 = \log e_i^1$$

After the transformation we can express the model in the simple form
Observation = Model + noise

$$y_i^1 = \mu_i + \rho^1 + \varepsilon_i^1$$

Noise model

- We make some additional assumptions about the model

$$y_i^1 = \mu_i + \rho^1 + \varepsilon_i^1, \varepsilon_i \sim N(0, \sigma_i^2)$$

- Noise (ε) is independent across controls / experiments
- The noise is Gaussian (original multiplicative noise is log-normal)
- The noise variance does not depend on the experiment but may depend on the specific spiked control

Maximum likelihood estimate

- Maximum likelihood estimate (MLE) is a general and powerful techniques for fitting parameters of a probabilistic model.
- Given a parametric model (for example, Gaussian noise) and observed data, we look for the set of parameters (in our case, mean and variance) that *maximize the likelihood* of the model.
- If we observe data D , then we look for parameters that will maximize $p(D | M)$ where M is the model we assume

Maximum likelihood estimate: Example

- Assume a uniform distribution model $X \sim U(0, N)$.
- For such a model we have 1 parameter to fit (N)
- We now observe the values:

1.2, 0.5, 3.4, 2.4, 1.5, 0.8, 2.2, 3.2

what value should we use for N ?

- Recall that in a uniform model, $p(x) = 1/N$ for $0 < x < N$ and $p(x) = 0$ for $x > N$
- The likelihood of the data given N is thus:

$$\prod_{x < N} \frac{1}{N} \prod_{x > N} 0$$

Maximum likelihood estimate: Example

1.2, 0.5, 3.4, 2.4, 1.5, 0.8, 2.2, 3.2

- Recall that in a uniform model, $p(x) = 1/N$ for $0 < x < N$ and $p(x) = 0$ for $x > N$
- The likelihood of the data given N is thus:

$$\prod_{x < N} \frac{1}{N} \prod_{x > N} 0$$

- It is easy to see that to maximize this value we must pick an N that is at least as large as the maximum value we observed.
- On the other hand, the larger N the smaller $1/N$
- Thus, the value that maximizes the likelihood is $N = 3.4$, the largest value we observed.

Back to our model

- We want to fit our model to the (log transformed) raw data
- We first write log likelihood term for the observed expression values:

$$L(Y) = \sum_{i=1}^n \sum_j p(y_i^j | \mu_i, \rho^j, \sigma_i^2)$$

$$= \sum_i \sum_j -\frac{(y_i^j - \mu_i - \rho^j)^2}{2\sigma_i^2} - 0.5 \log(2\pi\sigma_i^2)$$

$$y_i^1 \sim N(\mu_i + \rho^1, \sigma^2)$$

y_1^1	y_1^T
.
.
y_m^1	y_m^T

Iterative solution

$$\mu_i = \frac{1}{n} \sum_{j=1}^n (y_i^j - \rho^j)$$

$$\sigma_i^2 = \frac{1}{n} \sum_{j=1}^n (y_i^j - \mu_i - \rho^j)^2$$

$$\rho^j = \frac{\sum_{i=1}^M (\sigma_i^2)^{-1} (y_i^j - \mu_i)}{\sum_{i=1}^m (\sigma_i^2)^{-1}}$$

We iterate until convergence

Normalizing using the estimated parameters

- Once we obtain the estimate for the scaling parameter ρ^j we re-scale each measured value as follows:

$$\hat{y}_i^j = y_i^j - \rho^j$$

so that all genes in all arrays will have a scaling factor of 1 (log scaling of 0)

Some additional notes

- The maximum likelihood estimates of the noise variances may become too small; would need MAP or Bayesian estimates for the variances in practice.
- The simple log-normal noise model may not be adequate

$$\sigma_i^2 = \frac{1}{n} \sum_{j=1}^n (y_i^j - \mu_i - \rho^j)^2$$

Oligo arrays: Negative values

- In many cases oligo array can return values that are less than 0 (Why?)
- There are a number of ways to handle these values
- The most common is to threshold at a certain positive value
- A more sophisticated way is to use the negative values to learn something about the variance of the specific gene

Data analysis

- Normalization
- Combining results from replicates
- Identifying differentially expressed genes
- Dealing with missing values
- Static vs. time series

Motivation

- In many cases, this is the goal of the experiment.
- Such genes can be key to understanding what goes wrong / or get fixed under certain condition (cancer, stress etc.).
- In other cases, these genes can be used as 'features' for a classifier.
- These genes can also serve as a starting point for a model for the system being studied (e.g. cell cycle, pheromone response etc.).

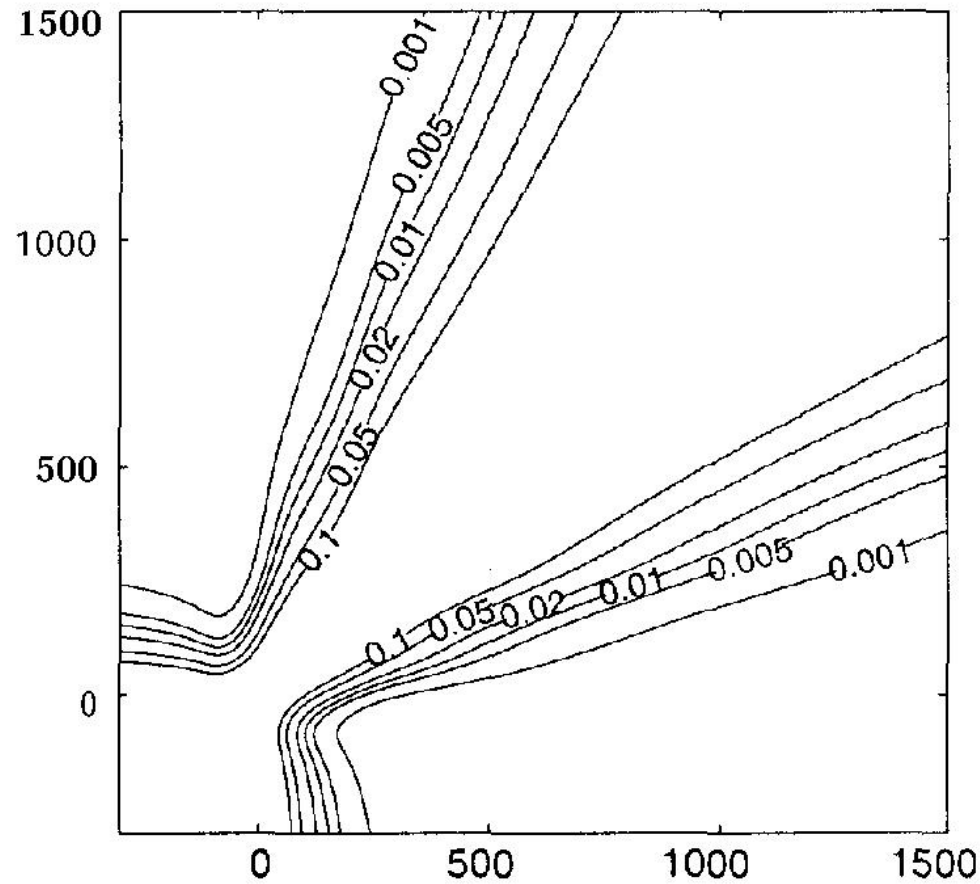
Problems

- As mentioned in the previous lecture, differences in expression values can result from many different noise sources.
- Our goal is to identify the 'real' differences, that is, differences that can be explained by the various errors introduced during the experimental phase.
- Need to understand both the experimental protocol and take into account the underlying biology / chemistry

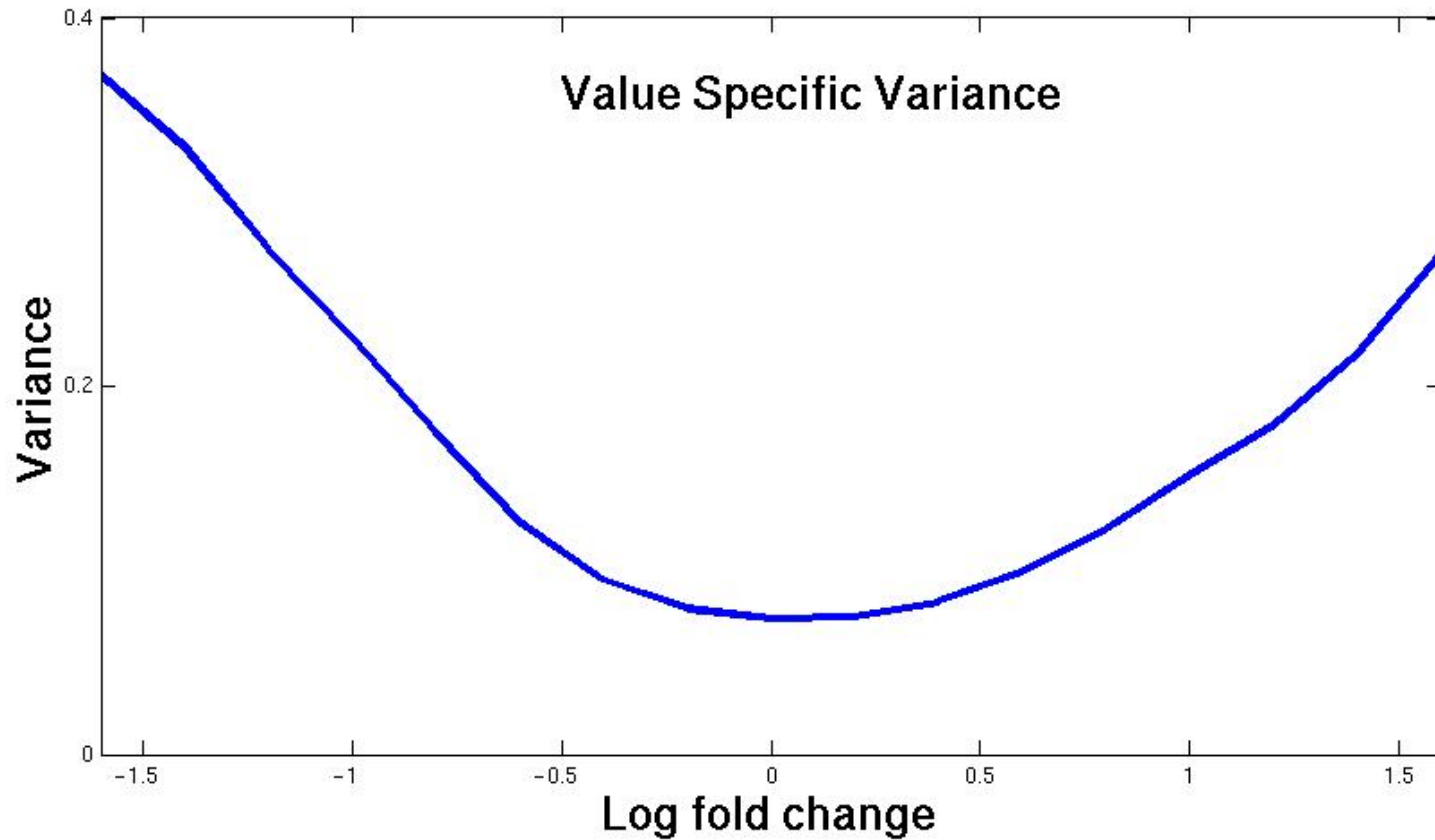
The 'wrong' way

- During the early days (though some continue to do this today) the common method was to select genes based on their fold change between experiments.
- The common value was 2 (or absolute log of 1).
- Obviously this method is not perfect ...

Significance bands for Affy arrays

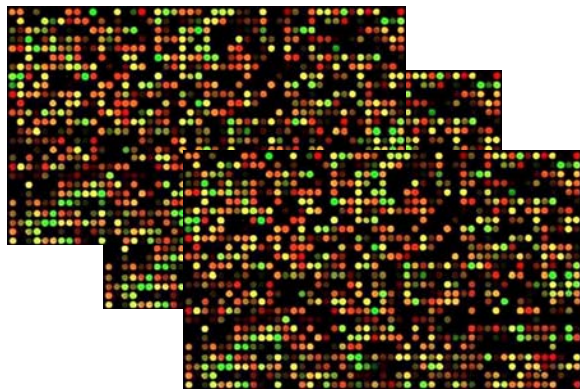


Value dependent variance

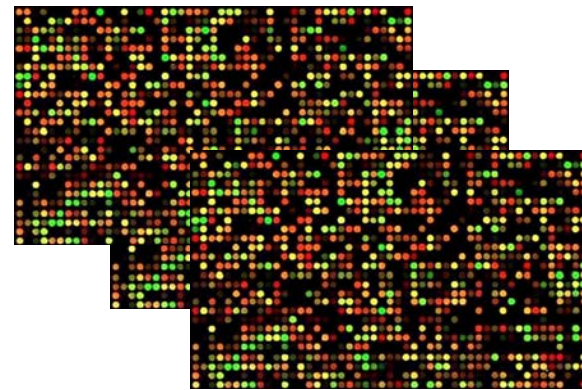


Typical experiment: replicates

healthy



cancer



Technical replicates: same sample using multiple arrays

Dye swap: reverse the color code between arrays

Clinical replicates: samples from different individuals

Many experiments have all three kinds of replicates

What you should know

- The different noise factors that influence microarray results
- The two major normalization methods:
 - Assuming the same mRNA quantity
 - Using spike controls or house keeping genes
- Maximum likelihood estimation (MLE) principal