BMC Bioinformatics

Methodology article

Empirical array quality weights in the analysis of microarray data

Matthew E Ritchie[1], Dileepa Diyagama[2], Jody Neilson[3], Ryan van Laar[2], Alexander Dobrovic[3], Andrew Holloway[2] and Gordon K Smyth*[1]

Address: [1]Bioinformatics Division, The Walter and Eliza Hall Institute of Medical Research, 1G Royal Parade, Parkville, Victoria 3050, Australia, [2]Ian Potter Foundation Centre for Cancer Genomics and Predictive Medicine, The Peter MacCallum Cancer Centre, St Andrews Place, East Melbourne, Victoria 3002, Australia and [3]Molecular Pathology Research, Department of Pathology, The Peter MacCallum Cancer Centre, St Andrews Place, East Melbourne, Victoria 3002, Australia

Email: Matthew E Ritchie - mritchie@wehi.edu.au; Dileepa Diyagama - dileepa.diyagama@petermac.org; Jody Neilson - alexander.dobrovic@petermac.org; Ryan van Laar - ryan.vanlaar@petermac.org; Alexander Dobrovic - alexander.dobrovic@petermac.org; Andrew Holloway - andrew.holloway@petermac.org; Gordon K Smyth* - smyth@wehi.edu.au

* Corresponding author

Published: 19 May 2006

BMC Bioinformatics 2006, **7**:261 doi:10.1186/1471-2105-7-261

Received: 28 September 2005
Accepted: 19 May 2006

This article is available from: http://www.biomedcentral.com/1471-2105/7/261

Abstract

Background: Assessment of array quality is an essential step in the analysis of data from microarray experiments. Once detected, less reliable arrays are typically excluded or "filtered" from further analysis to avoid misleading results.

Results: In this article, a graduated approach to array quality is considered based on empirical reproducibility of the gene expression measures from replicate arrays. Weights are assigned to each microarray by fitting a heteroscedastic linear model with shared array variance terms. A novel gene-by-gene update algorithm is used to efficiently estimate the array variances. The inverse variances are used as weights in the linear model analysis to identify differentially expressed genes. The method successfully assigns lower weights to less reproducible arrays from different experiments. Down-weighting the observations from suspect arrays increases the power to detect differential expression. In smaller experiments, this approach outperforms the usual method of filtering the data. The method is available in the limma software package which is implemented in the R software environment.

Conclusion: This method complements existing normalisation and spot quality procedures, and allows poorer quality arrays, which would otherwise be discarded, to be included in an analysis. It is applicable to microarray data from experiments with some level of replication.

Background

Assessment of data quality is an important component of the analysis pipeline for gene expression microarray experiments [1,2]. Although careful pre-processing and normalisation can ameliorate some problems with microarray data, including background fluorescence, dye effects or spatial artifacts [3], many sources of variation can affect the experimental procedure [4-7] and it is inevitable that variations in data quality will remain. In this article we demonstrate an approach in which variations in data quality are detected and adjusted for as part of the differential expression analysis. The method is widely applicable, easy to use and can have a high payoff.

Quality assessment procedures can be applied at the probe level or at the array level. Probe quality is influenced by local factors on the array such as printing irregularities or spatial artifacts. For spotted microarrays, spot-specific morphology and signal measurements obtained from image analysis software can be used to assign a quality score to each probe on the array [8-11]. Spots with low quality scores are commonly removed from further analysis. An alternative approach is to measure agreement between gene expression values from repeat probes directly and eliminate those spots with inconsistent replicate values [12,13]. For high-density oligonucleotide microarrays with multiple probes per gene, quality measures can be obtained from probe level models (PLMs). Image plots of robust weights or residuals obtained from robust PLMs can highlight artifacts on the array surface [2].

Probe quality assessment is not sufficient because some artifacts only become evident at the array level. Indeed the detection of problems is even more critical at the array level than at the probe level because a single bad array may constitute a sizeable proportion of the data from a microarray experiment. The quality of data from an entire array can be influenced by factors such as sample preparation and day-to-day variability [14]. Sub-standard arrays are typically identified using diagnostic plots of the array data [1,15-17]. The correlation between expression values of repeatedly spotted clones on an array is also used as an array quality measure [18]. Where large data sets are available, a statistical process control approach can identify outlier arrays [19]. In Affymetrix GeneChip experiments, array quality can be assessed using PLM standard errors or from RNA degradation plots [2].

Almost all the methods cited above classify the data as either "good" or "bad", and exclude "bad" probes or arrays from further analysis. In our experience however the "bad" arrays are usually not entirely bad. Very often the lesser quality arrays do contain good information about gene expression but which is embedded in a greater degree of noise than for "good" arrays. In this article, a graduated, quantitative approach is taken to quality at the array level in which poorer quality arrays are included in the analysis but down-weighted.

Quality assessment methods can be divided into those which are "predictive" and those which are "empirical". The operational meaning of quality is that high quality features produce highly reproducible expression values, while low quality features produce values which are more variable and hence less reproducible. Predictive quality assessment methods attempt to predict variability by comparing features such as spot morphology to normative measures. On the other hand, methods which compare duplicate spots within arrays are empirical in that they observe variability.

In this article we extend the empirical approach to multi-array experiments for which we measure the discrepancies between replicate arrays. In order to be as general as possible, we do not limit ourselves to simple replicate experiments, but work with a linear model formulation which allows us to handle experiments of arbitrary complexity including those with factorial or loop designs. The degree of replication in such experiments is reflected in the residual degrees of freedom for computing the residual standard errors. Our method is implemented by way of a heteroscedastic variance model. It is common for statistical models of microarray data to allow each probe to have its own individual variance. Our heteroscedastic model allows the variance to depend on the array as well as on the probe. The array variance factors then enter into the subsequent analysis as inverse array quality weights. Importantly, our method not only detects variations in data quality but adjusts for this as part of the analysis.

Our approach can be combined with predictive quality assessment methods and is an effective complement to them. Predictive methods can be used to filter spots or to provide quantitative prior spot weights which are incorporated into the linear model analysis. However the causes of poor quality data cannot always be clearly identified. The empirical array weight method described here estimates and accommodates any variation in quality which remains after the spot quality weights have been taken into account, i.e., after prediction has achieved as much as it can. Our approach is particularly effective when arrays vary in quality but the problems cannot be isolated to particular regions or particular probes on the offending arrays.

The presence of array-level parameters in our heteroscedastic model means that the statistical analysis can no longer be undertaken in a purely gene-wise manner. A naive approach to fitting the model would be computationally expensive. We propose two computationally efficient algorithms for estimating the model by the well-recognised statistical criterion of residual maximum likelihood (REML). These algorithms view the microarray data as many small data sets, one for each probe, with a small number of shared parameters corresponding to the array variance factors. An innovative gene-by-gene update procedure is proposed for particularly fast approximate REML estimation.

The array weight method developed here can be applied to any microarray experiment with array-level replication, including experiments using high-density oligonucleotide arrays, but our experience is mainly with experiments

using spotted microarrays. High density arrays allow the additional possibility of measuring reproducibility for multiple probes for each gene rather than relying on gene or probe-set summaries [2]. A full treatment of empirical array quality for these platforms is therefore likely to involve an analysis of reproducibility at both the probe level and probe-set level, a further development which is not investigated in this article.

In this paper, the linear model approach to microarray data analysis is reviewed and the heteroscedastic model which includes array weights is introduced. Next, the experimental and simulated data sets used in this study are explained and results for these data are presented. The computational algorithms for fitting the heteroscedastic model are then described, followed by discussion and conclusions. Supplementary materials including data, R scripts and additional plots are available [20].

Linear models for microarray data

Linear models provide a convenient means to measure and test for differential expression in microarray experiments involving many different RNA sources [21,22]. The linear model approach allows a unified treatment of a wide variety of microarray experiments, including dye-swaps, common reference experiments, factorial experiments and loop or saturated designs, with little more complication than simple replicated experiments. Although the statement of the linear model, given below, requires some mathematical notation, the application of the methods we describe is in practice very simple using available software. Consider a microarray experiment with expression values y_{gj} for genes $g = 1, ..., G$ and arrays $j = 1, ..., J$. The expression values could be log-ratios from two-colour microarrays or summarised log-intensity values from a single-channel technology such as Affymetrix GeneChips. We assume that the expression values have been appropriately pre-processed, background corrected and normalised. The term *gene* is used here in a general way to include any ESTs or control probes that might be on the arrays. Assume that the systematic expression effects for each gene can be described by a linear model

$$E(\mathbf{y}_g) = X\beta_g \quad (1)$$

where $\mathbf{y}_g = (y_{g1}, ..., y_{gJ})^T$ is the vector of expression values for gene g, X is a known design matrix with full column rank K, and $\beta_g = (\beta_{g1}, ..., \beta_{gK})^T$ is a gene-specific vector of regression coefficients. The design matrix will depend upon the experimental design and choice of parameterisation and the regression coefficients represent log-fold changes between RNA sources in the experiment [22,23]. For example, consider a two-colour microarray experiment with three replicate arrays comparing RNA sources A and B. The individual log-ratios $y_{gj} = \log_2(R_{gj}/G_{gj})$, where R_{gj} and G_{gj} are the Cy5 and Cy3 intensities, measure differences in gene expression between the two samples. For a simple replicated experiment with sample B always labelled Cy5, the design matrix would be a column of ones, and the coefficient β_g would represent the log-fold-change for gene g in sample B over A. Replicated experiments with dye-swaps would be the same except that minus ones would indicate the dye-swap arrays. Consider another example where samples A and B are compared through a common reference sample. If there are two arrays for each sample and the common reference is always Cy3, then the design matrix would be

$$X = \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 1 \\ 1 & 1 \end{pmatrix}.$$

Here the first coefficient β_{g1} estimates the log-fold-change between A and the common reference while the second coefficient β_{g2} estimates the comparison of interest between B and A. The design matrix can be expanded indefinitely to represent experiments of arbitrary complexity.

The linear model also assumes

$$\text{var}(y_{gj}) = \sigma_g^2 / w_{gj} \quad (2)$$

where w_{gj} is a prior spot quality weight and σ_g^2 is the unknown gene-specific variance factor. The spot quality weights will usually have arisen from a predictive spot quality assessment step, with large weights representing good quality spots and low weights representing poor quality spots. To avoid unnecessary complications we will assume throughout that all the y_{gj} are observed and that all the spot weights are strictly positive, $w_{gj} > 0$. In practice, the methods developed in this article can be modified to accommodate missing y-values or zero weights, but this complicates the presentation somewhat and will be omitted.

For simplicity we will assume that the y_{gj} are normally distributed and that expression values from different arrays are independent. The weighted least squares estimator of β_g is

$$\hat{\boldsymbol{\beta}}_g = (X^T \Sigma_g^{-1} X)^{-1} X^T \Sigma_g^{-1} \mathbf{y}_g \quad (3)$$

where $\Sigma_g = \text{diag}(w_{g1}, ..., w_{gJ})$ is the diagonal matrix of prior weights. The *t*-statistic for testing any particular β_{gk} equal to zero is

$$t_{gk} = \frac{\hat{\beta}_{gk}}{s_g \sqrt{c_{gk}}}$$

where s_g^2 is the residual mean square from weighted regression and c_{gk} is the *k*th diagonal element of $(X^T \Sigma_g^{-1} X)^{-1}$.

It is important to appreciate that the spot weights w_{gj} act in a relative fashion for each gene. The *t*-statistic t_{gk} and its associated *p*-value would be unchanged if all the w_{gj} for a given *g* were scaled up or down by a constant factor. Hence it is only the relative sizes of the w_{gj} across arrays *j* for any given *g* which are important.

The *t*-statistic has *J* - *K* degrees of freedom. In microarray analyses with a small to moderate number of arrays, for which *J* - *K* is small, it is usually beneficial to replace s_g^2 with a variance estimator which is shrunk or moderated across genes to obtain moderated *t*-statistics [22]. Genes can then be selected for differential expression based on large moderated *t*-statistics or small *p*-values.

A heteroscedastic model for probes and arrays

In this article we allow the unknown variance factors to depend on the array as well as on the gene,

$$\text{var}(\gamma_{gj}) = \sigma_{gj}^2 / w_{gj}. \quad (4)$$

We need a model for the variance factors σ_{gj}^2 which reflects the fact that the genes differ in variability and also that the arrays in the experiment may differ in quality in a way which increases or decreases the variability of all or most of the probes on a particular array. The simplest model which does this is the additive log-linear model

$$\log \sigma_{gj}^2 = \delta_g + \gamma_j \quad (5)$$

[24,25]. We impose the constraint $\sum_{j=1}^{J} \gamma_j = 0$ so that the $\sigma_g^2 = \exp \delta_g$ represent the gene-wise variance factors while the γ_j represent the relative variability of each array. Array *j* will have $\gamma_j < 0$ or $\gamma_j > 0$ depending on whether it is relatively better or poorer quality than the average. For instance, an array with $\exp \gamma_j = 2$ is twice as variable as a typical array and will be given half weight in an analysis. Note that the variances are assumed to depend multiplicatively on array quality. This is more appropriate than, say, an additive model of gene and array variances because it preserves relativities between the gene-wise precisions as array quality varies. The log-linear variance model also has substantial numerical and inferential advantages over other variance models in that positivity for the variances is ensured for any values of the δ_g and γ_j parameters.

The fact that all the genes contribute to the estimation of the γ_j means that, once estimated, the array weights can be taken to be fixed quantities when analysing each individual gene. The array weights $v_j = 1/\exp \hat{\gamma}_j$ can be incorporated into a differential expression analysis simply by combining them with the prior weights into modified weights $w_{gj}^* = w_{gj} v_j$. The weighted least squares calculations described in the previous section (Equation 3) can then be conducted with w_{gj}^* replacing w_{gj} throughout. The use of appropriate array weights will produce more precise estimates of the gene expression coefficients and improve power to detect differentially expressed genes.

Note that, although the scaling of the array weights is in principle arbitrary, our convention that $\sum_{j=1}^{J} \gamma_j = 0$ means we always choose the array weights v_j to have geometric mean equal to one.

Data

The use of array quality weights will be demonstrated on both real and simulated data sets. The first data set was acquired as a quality control step in the array fabrication process at the Peter MacCallum Cancer Centre, Melbourne. This data set contains 100 microarrays representing 4–6 arrays taken from the beginning, middle and end of 22 different print batches. The arrays were printed with a human 10.5 K cDNA library and six copies of the Lucidea Microarray ScoreCard (LMS) set of control probes [26]. Each array was hybridised with Cy3 labelled mRNA from the MCF7 breast cancer cell line and Cy5 labelled mRNA from the Jurkat T-cell leukemia cell line. Test and reference LMS spike-in mixes were added to the mRNA samples prior to labelling to produce predictable fold changes for the control spots (Table 1). The ratio control

Table 1: Summary of QC LMS controls. Theoretical fold-changes for the spike-in control probes in the QC LMS data set are shown. *M* values have been rounded to 2 decimal places.

Control (abbreviation)	Ratio R:G	$M = \log_2(R/G)$
Up-regulated 3-fold (U03)	3:1	$\log_2 3 = 1.58$
Up-regulated 10-fold (U10)	10:1	$\log_2 10 = 3.32$
Down-regulated 3-fold (D03)	1:3	$-\log_2 3 = -1.58$
Down-regulated 10-fold (D10)	1:10	$-\log_2 10 = -3.32$
Dynamic Range (DR)	1:1	$\log_2 1 = 0.00$

spots should show three-fold or ten-fold changes while the dynamic range spots should not be differentially expressed. The array images were analysed using Spot 2.0 [27] and the intensities were background corrected by subtracting morphological (*morph*) background values. The *morph* background treatment ensures that all intensities remain positive after background correction, and damps down the variability of the log-ratios for low intensity spots [28]. This eliminates the need for intensity-based filtering of spots in the subsequent analysis. We use this data for two purposes. Firstly, log-ratios were print-tip loess normalised [29]. Standardised residuals, $(y_{gj} - \bar{y}_g)/s_g$, were computed where y_{gj} are the normalised log-ratios and $\bar{y}_g$ and s_g are the probe-wise means and standard deviations. Standardised residuals from the 75% most highly expressed probes in the 10.5 K cDNA library were used as a population of non-normal deviates for generating simulated data sets. The other analysis of this data uses only the 120 LMS control spots on each array. Log-ratios from these control spots were global loess normalised [29],

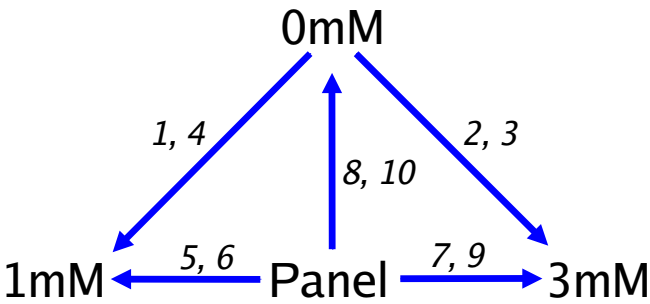

Figure 1
Design of the METH experiment. The METH experiment compared 3 mRNA sources of interest (0 mM, 1 mM and 3 mM) directly on the first 4 arrays and indirectly via a Panel reference on a further 6 arrays. The arrays are numbered from 1 to 10 in the order they were hybridised. Each arrow indicates a direct comparison made within an array, and points from the Cy3 labelled sample towards the Cy5 labelled sample.

using a relatively wide span of 0.7 because of the relatively small number of spots used. The resulting data will be referred to as the QC LMS data set in the remainder of the article.

The second data set arose from a study aimed at identifying novel methylated markers in myeloid malignancy using the leukemia cell line KG1a. Microarrays were printed with the same cDNA library and controls as the first data set. A known inhibitor of DNA methylation, 5-azacytidine, was added to KG1a cells in varying doses (0 mM, 1 mM and 3 mM). Both direct and indirect comparisons between the 1 mM and 3 mM treatments and the 0 mM treatment were made on a total of 10 arrays (Figure 1). The panel reference RNA consisted of a pool of RNA from 11 cancer cell lines. The arrays were scanned on a GenePix 4000B scanner and image analysed using GenePix Pro 4.0. The intensities were background corrected using the model-based 'normexp' method with an offset of 50 [30]. Again, this background correction method avoids negative intensities and the need for intensity-based filtering. Log-ratios were print-tip loess normalised [29]. This data set will be referred to as the METH experiment.

Simulations

For the simulation studies, normal and non-normal expression values (y_{gj}) from replicate arrays were generated with $G = 10000$ genes and $J = 3$ and 5 arrays in six different scenarios. For each simulation, different array variances ($\exp \gamma_j$) were assumed, and the gene-specific variances ($\exp \delta_g$) were sampled from the estimates (s_g^2) obtained from the QC data set. Non-normal deviates were sampled from the standardised residuals of the QC data set. These deviates are considerably more heavy-tailed than normal. In each data set, 5% (500) of genes were simulated to be differentially expressed at either 2-fold (250) or 3-fold (250), while the remaining 95% were simulated to have mean zero.

For the simulations with 3 arrays, the expression values for the third array were generated to be twice as variable as those from the first two arrays in simulation 1 (i.e., $v_1 = v_2 = 2v_3$), ten times as variable as the first two arrays in simulation 2 (i.e., $v_1 = v_2 = 10v_3$) or five times more variable on the second array and ten times more variable on the third array relative to the first in simulation 3 (i.e., $v_1 = 5v_2 = 10v_3$).

Simulations with 5 arrays were generated to have at least two more variable arrays. In simulation 4, expression values on the fourth and fifth arrays were simulated to be two

Table 2: Estimates of array weights obtained from 1000 simulated microarray data sets. Means and standard deviations of the array weights estimated from 1000 simulated data sets assuming six different array variance scenarios are shown for normal and non-normal data using the full REML algorithm and the gene-by-gene update algorithm. Accurate estimates with small standard deviations are obtained using the full algorithm. The gene-by-gene update algorithm recovers weights which are generally only slightly flattened towards equal weights.

Sim.	True weight	Mean (Standard deviation)			
		Normal		Non-Normal	
		Full	Gene-by-gene	Full	Gene-by-gene
1	1.26	1.26 (0.04)	1.23 (0.07)	1.25 (0.04)	1.22 (0.07)
	1.26	1.26 (0.04)	1.23 (0.07)	1.25 (0.04)	1.22 (0.07)
	0.63	0.63 (0.01)	0.66 (0.03)	0.64 (0.01)	0.67 (0.03)
2	2.15	2.16 (0.15)	2.07 (0.14)	2.13 (0.13)	2.04 (0.14)
	2.15	2.16 (0.14)	2.07 (0.13)	2.14 (0.14)	2.04 (0.14)
	0.22	0.22 (0.00)	0.24 (0.01)	0.22 (0.00)	0.24 (0.01)
3	3.68	3.72 (0.33)	2.07 (0.14)	3.54 (0.29)	2.05 (0.15)
	0.74	0.74 (0.04)	1.03 (0.06)	0.75 (0.04)	1.03 (0.06)
	0.37	0.37 (0.02)	0.47 (0.02)	0.38 (0.01)	0.48 (0.02)
4	1.52	1.52 (0.03)	1.50 (0.05)	1.51 (0.03)	1.50 (0.05)
	1.52	1.52 (0.03)	1.50 (0.04)	1.51 (0.03)	1.50 (0.05)
	1.52	1.52 (0.03)	1.50 (0.05)	1.51 (0.03)	1.50 (0.05)
	0.76	0.76 (0.01)	0.77 (0.02)	0.76 (0.01)	0.77 (0.02)
	0.38	0.38 (0.01)	0.38 (0.01)	0.38 (0.01)	0.39 (0.01)
5	2.19	2.19 (0.05)	2.16 (0.07)	2.17 (0.05)	2.14 (0.07)
	2.19	2.19 (0.05)	2.16 (0.07)	2.17 (0.05)	2.14 (0.07)
	2.19	2.19 (0.05)	2.15 (0.07)	2.17 (0.05)	2.14 (0.07)
	0.44	0.44 (0.01)	0.45 (0.01)	0.44 (0.01)	0.45 (0.01)
	0.22	0.22 (0.00)	0.22 (0.00)	0.22 (0.00)	0.23 (0.00)
6	3.44	3.44 (0.12)	3.00 (0.13)	3.37 (0.12)	2.95 (0.13)
	1.72	1.72 (0.04)	1.78 (0.06)	1.72 (0.04)	1.77 (0.06)
	0.86	0.86 (0.02)	0.89 (0.02)	0.86 (0.02)	0.89 (0.02)
	0.57	0.57 (0.01)	0.59 (0.01)	0.58 (0.01)	0.60 (0.01)
	0.34	0.34 (0.01)	0.36 (0.01)	0.35 (0.01)	0.36 (0.01)

times and four times more variable than those on the first three arrays (i.e., $v_1 = v_2 = v_3 = 2v_4 = 4v_5$). In simulation 5, expression values from the fourth and fifth arrays were five and ten times more variable than those on the first three arrays (i.e., $v_1 = v_2 = v_3 = 5v_4 = 10v_5$). For simulation 6, the expression values were two times, four times, six times and ten times more variable on arrays two to five respectively relative to the first array (i.e., $v_1 = 2v_2 = 4v_3 = 6v_4 = 10v_5$).

The six different scenarios and the true array weights in each case are listed in the first two columns of Table 2. Recall that only the relative sizes of the array weights are relevant so, by the convention described earlier ($\sum_{j=1}^{J} \gamma_j = 0$), we always scale the array weights so that they have geometric mean equal to one.

Results

Simulations

First we demonstrate the ability of the algorithms to return the correct array weights for simulated data sets where the true array variances are known. For each of the six simulation scenarios described in the previous section, 1000 independent data sets were generated and the variance model (Equation 5) was fitted to each. This was carried out for both normal and non-normal data. For each data set, estimates were obtained using the full REML algorithm and the approximate gene-by-gene update algorithm (see Methods section). Table 2 shows the means and standard deviations of the estimated array weights v_j. The full algorithm is shown to assign weights almost exactly consistent with the predicted values. The gene-by-gene update method returns array weights which are slightly less extreme, i.e., slightly flattened towards equal weights, although still broadly accurate. The gene-by-gene estimates are also somewhat more variable than those for full REML, a consequence of the fact that the REML estimators are theoretically optimal. All the standard deviations are small enough however that the variability is negligible, even for the approximate algorithm. The results are virtually unchanged whether the data is normal or non-normal. Although the accuracy of the full REML algorithm is impressive here, it is important to appreciate that very precise estimates of the array variances are not required for a weighted analysis to be effective, so that the gene-by-gene algorithm may be adequate in practice.

Note also that the REML algorithms are invariant with respect to the gene-wise means or standard deviations, so the results given in Table 2 remain the same regardless of how the gene specific means or standard deviations are generated.

Next we turn to the detection of differential expression and false discovery rates. For each simulated data set, differentially expressed genes were selected using ordinary *t*-statistics and using the empirical Bayes moderated *t*-statistics implemented in the limma software package [30]. These differential expression measures were used to compare three different array weighting schemes. We considered that an experimenter might choose (i) to use all the arrays equally in the analysis (equal weights), (ii) to use the array weights estimated by the REML algorithm, or (iii) to remove the worst one or two arrays from the analysis entirely (filtering). False discovery rates were calculated to compare the three weighting schemes. Figure 2 shows the average number of false discoveries plotted against the number of genes selected using ordinary *t*-statistics (solid lines) or moderated *t*-statistics (dashed lines) for the 3 array simulations listed in Table 2. Each line represents the average of 1000 simulations. Panels (a), (c) and (e) show the normal results for simulation 1, 2 and 3 respectively, while panels (b), (d) and (f) give the corresponding results for non-normal data. The same layout is used in Figure 3 for the 5 array simulations.

The black lines show the results obtained after removing the most variable array from simulations 1, 2 and 3 (Figure 2), or after removing the two most variable arrays in simulations 4, 5 and 6 (Figure 3). The light gray lines show the number of false positives obtained using equal weights and the dark gray lines indicate the false discovery rates when array weights from the full REML algorithm are used.

The first striking feature of Figures 2 and 3 is that the moderated *t*-statistics easily outperform the ordinary *t*-statistics regardless of the simulation assumptions, consistent with findings in other studies [22,31]. The second feature is that the use of array weights always gives the lowest false discovery rate of the three weighting schemes, regardless of which *t*-statistic is used. Array weighting outperforms both equal weighting and array filtering in all cases, although in simulation 1 equal weighting is nearly as good (the dark gray and light gray lines overlap in Figure 2, panels a and b). It is interesting that the strategy most commonly proposed in the literature, that of array-filtering, is generally the worst performer across the scenarios, except in simulation 5 with moderated *t*-statistics, when equal weighting is worst. The use of array-filtering with ordinary *t*-statistics is very poor indeed. This is despite the fact that the simulation results make array-filtering appear somewhat better than it could be in practice. This is because we always removed the one or two arrays which were known to be the most variable, whereas in real data situations the true status of each array is uncertain and must be inferred using diagnostic plots or other methods. The results in Figures 2 and 3 are for the full REML algorithm, however the results are virtually identical when the approximate gene-by-gene update algorithm is used instead [20]. This shows that the differences in estimated weights between the full and approximate REML algorithms observed in Table 2 are relatively unimportant from the point of view of evaluating differential expression.

QC LMS Data

The QC LMS data set provides an example of real data where we know the differential expression status of each spot. This example has the structure of a simple replicated experiment. The very large number of replicates allows us to assess accurately the effect of array weights. The variance model (Equation 5) was fitted to the log-ratios for the 120 LMS control spots across the 100 arrays. The array weights, $v_j = 1/\exp\ \hat{\gamma}_j$, are shown in Figure 4(a). The weights vary from a minimum of 0.11 for array 19 to a maximum of 3.68 for array 91. The least squares estimate of the log-fold change between the two RNA sources for each gene is the weighted mean of the individual array log-ratios with these weights. Inspection of MA-plots shows that arrays with lower estimated weights do indeed appear to return the theoretical fold changes more poorly than arrays with higher weights (Figure 5).

The differential expression status of the LMS control spots are known, so we can use them to assess our ability to distinguish probes which are differentially expressed from those which are not. Figure 6 plots *t*-statistics for testing differential expression for the 120 LMS controls. Ordinary *t*-statistics were calculated using either equal weights or using the array weights shown in Figure 4(a). The *t*-statistics for all classes of ratio controls (D03, D10, U03 and U10) move further from zero when array weights are used while the distribution of *t*-statistics for the dynamic range controls does not noticeably change. This demonstrates that the array quality weights increase statistical power to detect true differential expression without increasing the false discovery rate.

METH Data

In order to demonstrate our method on a smaller and more complex experiment, we now turn to the METH data. For this experiment, replication takes the form not only of duplicate arrays but also of redundancy between the direct and indirect comparisons available for each pair

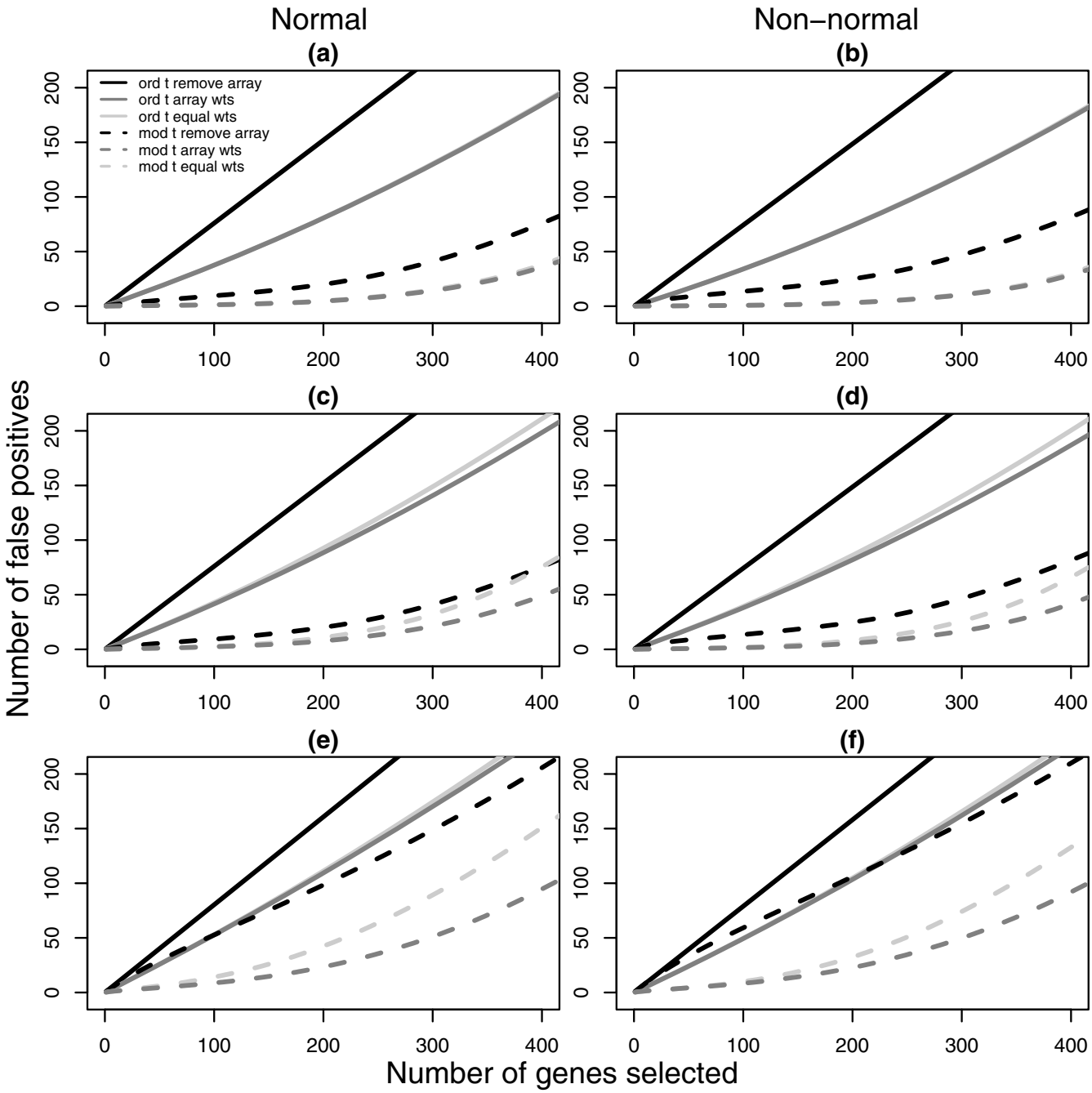

Figure 2

Number of false discoveries from simulated data sets with 3 arrays. For each 3 array simulation in Table 2 (1, 2 and 3), the average false discovery rates calculated using ordinary *t*-statistics (solid lines) or moderated *t*-statistics (dashed lines) are given. Panels (a), (c) and (e) show the results for normal data under simulations 1, 2 and 3 respectively while panels (b), (d) and (f) display the corresponding results for non-normal data. Black lines plot the false discovery rates when the most unreliable array is removed from the analysis. Light gray lines are the results obtained using equal weights and dark gray lines show the false discovery rates recovered with array weights. Each line is the average of 1000 simulated data sets. In nearly all cases, the use of array weights in the analysis gives fewer false positives than the other methods. Simulation 1 is the exception, with equal weighting and array weighting producing similar false discovery rates (overlapping curves) for both normal (a) and non-normal (b) data.

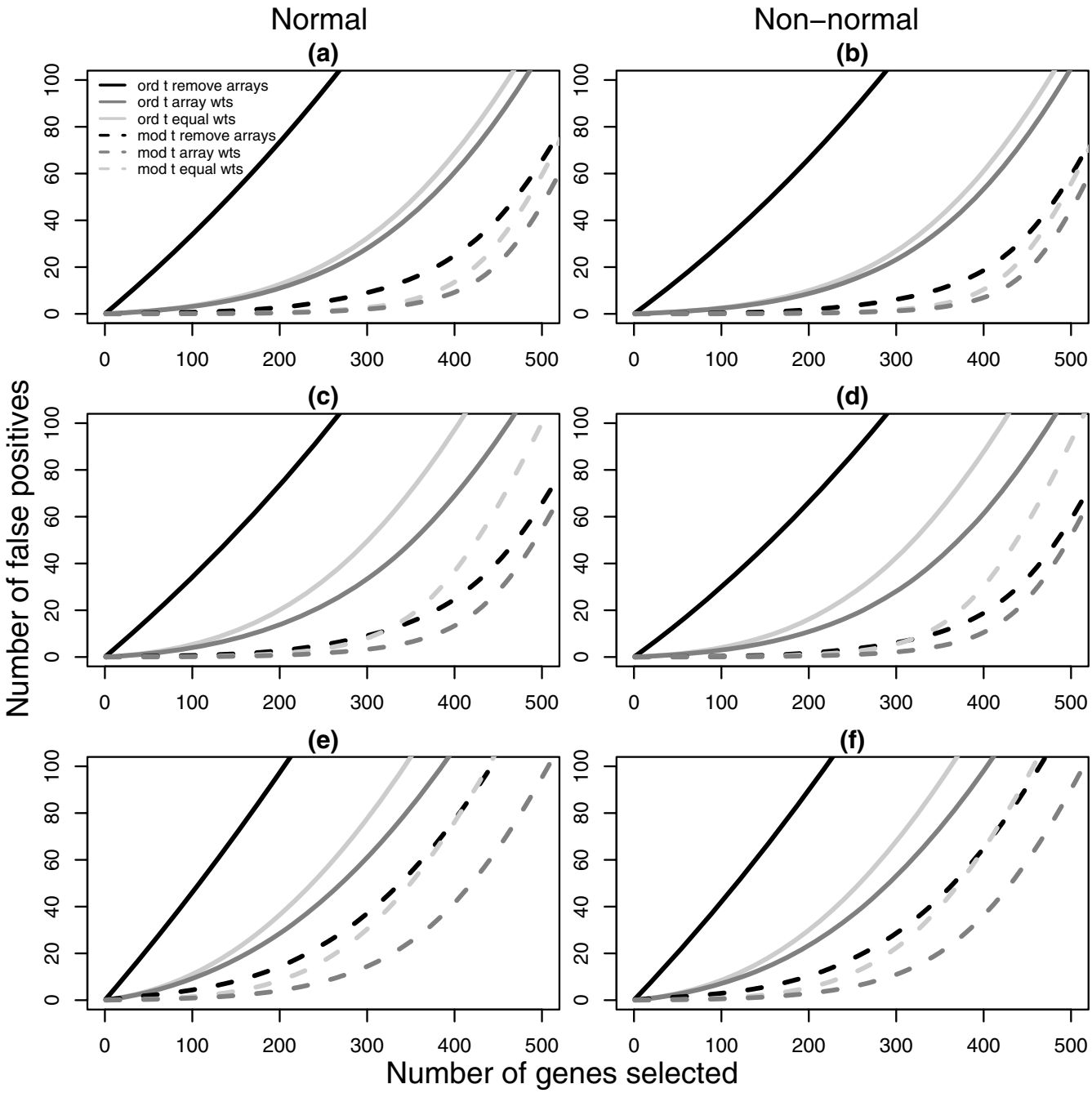

Figure 3

Number of false discoveries from simulated data sets with 5 arrays. For each 5 array simulation in Table 2 (4, 5 and 6), the average false discovery rates calculated using ordinary *t*-statistics (solid lines) or moderated *t*-statistics (dashed lines) are given. Panels (a), (c) and (e) show the results for normal data under simulations 4, 5 and 6 respectively while panels (b), (d) and (f) display the corresponding results for non-normal data. Black lines plot the false discovery rates when the two most unreliable arrays are removed from the analysis. Light gray lines are the results obtained using equal weights and dark gray lines show the false discovery rates recovered with array weights. Each line is the average of 1000 simulated data sets. In all cases, the use of array weights in the analysis gives fewer false positives than the other methods.

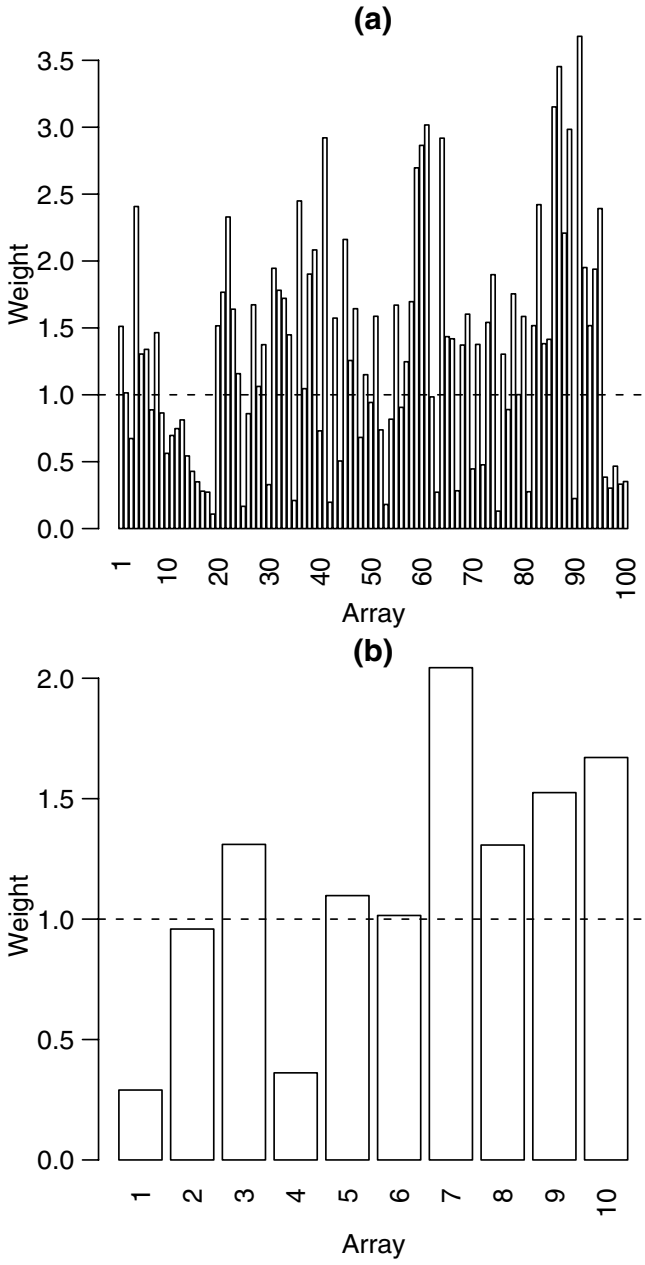

Figure 4

Array weights for the QC LMS and METH data sets. Array weights (v_j) calculated for the QC LMS data set (a) and the METH experiment (b). The arrays are ordered by time of hybridisation, and the dashed lines show the unit weight. In each experiment, there are arrays which receive higher and lower relative weights, corresponding to arrays which are more or less reproducible.

of treatments. The linear model requires three coefficients to represent differences between the three RNA treatments and the common reference leaving seven residual degrees of freedom. Of primary interest are the coefficients $\beta_{1\text{-}0}$ and $\beta_{3\text{-}0}$ which measure the gene expression differences 1mM-0mM and 3mM-0mM respectively. The design matrix was generated automatically using the limma software package. The linear model was fitted to all genes in the 10.5 K library and control probes were excluded.

Table 3: Number of differentially expressed genes for the METH experiment. Counts of candidate differentially expressed genes for the METH experiment obtained after (1) removing arrays with high background levels from the analysis or (2) keeping these arrays in the analysis, but down-weighting their expression values using the array quality weights from Figure 4 (b) are given for each comparison of interest. The false discovery rate was controlled to be less than 0.05 in each case. More candidate genes are discovered using array weights.

Analysis method	Number of genes	
	1mM-0mM	3mM-0mM
1. Remove arrays	0	1263
2. Array weights	654	1790

The experimenters who conducted the METH experiment were suspicious of the reliability of the first 4 arrays hybridised, which they believed were not giving consistent results with the last 6 arrays.

Figure 4(b) shows the array weights estimated from this data. Arrays 1 and 4 were assigned the lowest weights of 0.29 and 0.36 respectively. Diagnostic plots of the data [20] reveal that arrays 1 and 4 have high levels of background fluorescence in both channels, which does indeed indicate that these arrays are of poorer quality. The diagnostics do not identify a particular subset of problem spots which could be filtered out, so spot quality methods do not offer a solution. The usual method of dealing with this problem would involve removing these two suspect arrays from further analysis. We now consider the alternative of retaining these microarrays but down-weighting their expression values using empirical array weights. Differential expression was assessed for both methods using moderated *t*-statistics [22] adjusted for multiple testing using the false discovery rate method [33]. Table 3 shows the number of genes for the 1mM-0mM and 3mM-0mM treatment comparisons with adjusted *p*-values (*q*-values) less than 0.05. For the 1mM-0mM comparison, which has two poor quality arrays directly comparing these RNA sources, removal of the worst arrays throws away most of the information on this comparison and results in no differentially expressed genes. Using array weights gives 654 candidate differentially expressed genes for this comparison. Of these genes, 413 are also differentially expressed in the 3mM-0mM comparison and 237 show a monotonic response to dose with the 3mM-0mM fold-change being larger and in the same direction as the 1mM-0mM change. This suggests that many of these genes are worthy candidates for further validation.

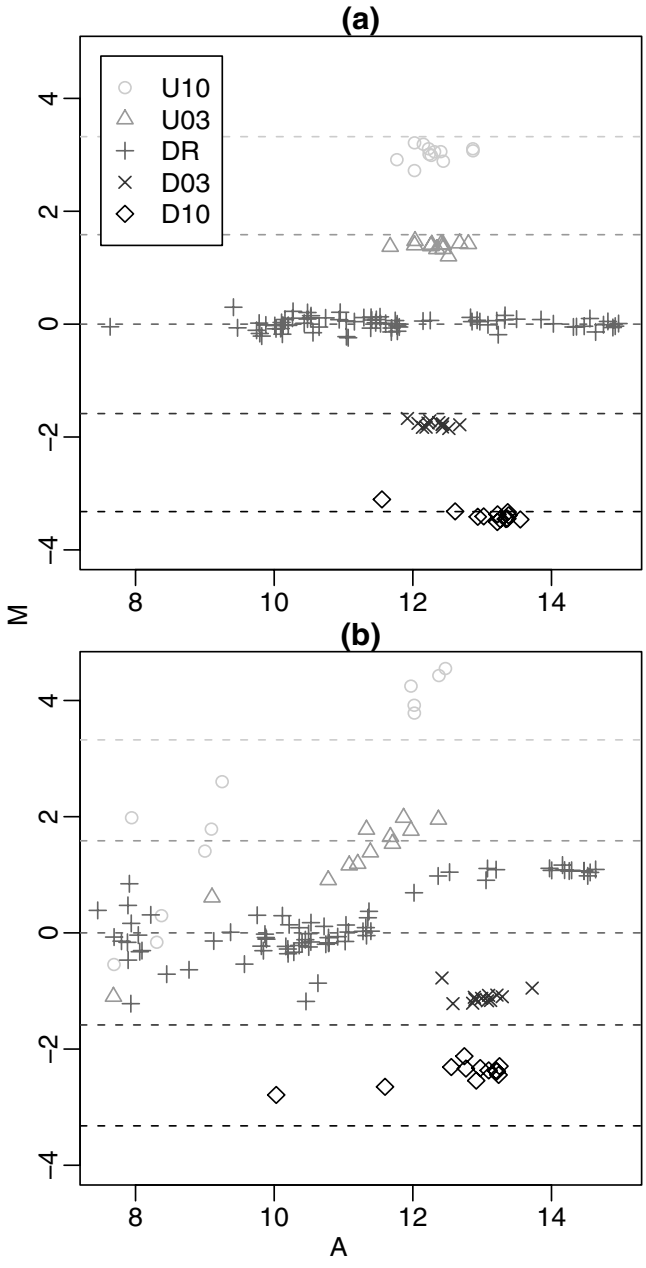

Figure 5
MA-plots of QC LMS controls for two arrays. MA-plots for arrays 91 (a) and 19 (b) which were assigned the highest and lowest quality weights respectively. Here $M = \log_2(R/G)$ is the spot log-ratio and $A = (\log_2 G + \log_2 R)/2$ is the spot log-intensity. Dashed lines show the theoretical *M* values from Table 1. The controls from array 91 consistently recover the true spike-in log-ratios and are assigned high weight (v_{91} = 3.68), whereas the log-ratios from array 19 are considerably more variable, resulting in a very low weight (v_{91} = 0.11).

Methods

Need for new algorithms

We now turn to the problem of computing REML estimates for the array variance parameters in the probe-array variance model (Equation 5). Algorithms for fitting heteroscedastic linear models are already available [34], however the high dimensionality of microarray data limits the usability of conventional algorithms. There are $G + J - 1$ parameters in the variance model and a further GK parameters in the linear model itself. The fact that the array parameters in the variance model are shared by all the genes means that the usual strategy of fitting models separately for each gene is not available. Even computers with many gigabytes of memory will run into memory limits using conventional algorithms with G much larger than around 50. Using a conventional algorithm for a typical microarray experiment with tens of thousands of genes is out of the question.

The basic difficulty from an algorithmic point of view is not the large number of expression values but rather the large number of parameters to be estimated. In the next section we develop a strategy for eliminating the gene-wise parameters β_g and δ_g from the estimation problem.

Nested iterations

Conditional on the array variance factors γ_j, the gene-wise coefficients β_g and variances δ_g can be computed in closed form using weighted least squares as described in the linear models section (Equation 3). The method of *nested iterations* is a strategy to reduce the dimension of an estimation problem by eliminating conditionally estimable parameters [35]. The idea is applied here to eliminate the gene-specific parameters from the REML likelihood function. This reduces the estimation problem to one involving just the $J - 1$ array weights.

Explicit expressions for the REML log-likelihood for heteroscedastic models such as ours can be found in [24] and [34]. Write $f(\mathbf{y}_g; \delta_g, \gamma)$ for the contribution to the REML log-likelihood for gene g with $\gamma = (\gamma_1, ..., \gamma_{J-1})^T$. The REML likelihood already has the property that the linear model parameters β_g are eliminated. The REML log-likelihood to be maximised is

$$\ell(\mathbf{y}_1, ..., \mathbf{y}_G; \delta_1, ..., \delta_G, \boldsymbol{\gamma}) = \sum_{g=1}^{G} f(\mathbf{y}_g; \delta_g, \boldsymbol{\gamma}) \qquad (6)$$

Rather than deal with this large dimensional problem, we eliminate the δ_g by considering the profile REML likelihood for γ. Write $\hat{\delta}_{g|\gamma}$ for the value of δ_g which maximises

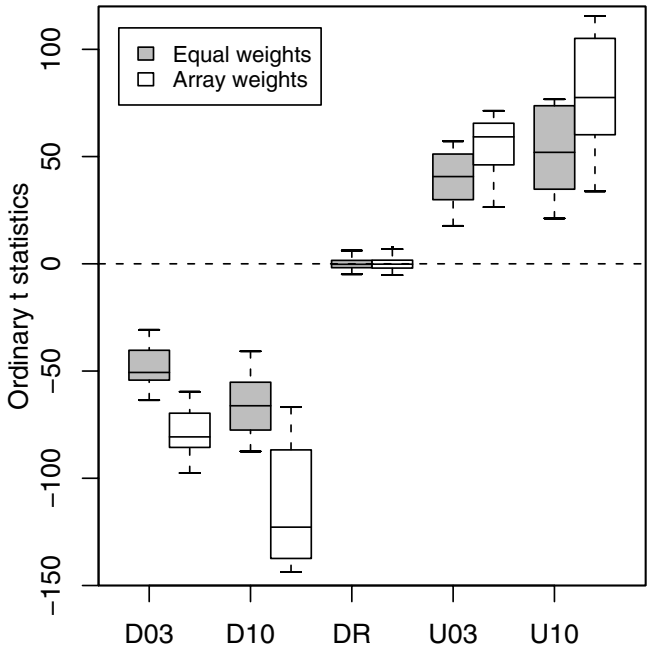

Figure 6
Ordinary *t*-statistics for the QC LMS controls. The *t*-statistics were calculated using either equal weights or the array weights (v_j) from Figure 4(a). Using array weights in the analysis results in more extreme *t*-statistics for the known differentially expressed controls (D03, D10, U03, U10) which represents a gain in power.

$f(\mathbf{y}_g; \delta_g, \gamma)$ for given γ. The profile REML log-likelihood for γ is

$$\ell_p(\mathbf{y}_1, \ldots, \mathbf{y}_G; \gamma) = \sum_{g=1}^{G} f(\mathbf{y}_g; \delta_{g|\gamma}, \boldsymbol{\gamma}) \quad (7)$$

We consider now the nested iteration for maximising the profile likelihood. Write

$$U_{g,\gamma} = \frac{\partial f(\mathbf{y}_g; \delta_g, \boldsymbol{\gamma})}{\partial \boldsymbol{\gamma}}. \quad (8)$$

Also let

$$A_g = \begin{pmatrix} A_{g,\delta\delta} & A_{g,\delta\gamma} \\ A_{g,\gamma\delta} & A_{g,\gamma\gamma} \end{pmatrix} \quad (9)$$

be the REML information matrix for gene *g*. The derivative of $f(\mathbf{y}_g; \hat{\delta}_{g|\gamma}, \gamma)$ with respect to γ is simply $U_{g,\gamma}$ evaluated at $\delta_g = \hat{\delta}_{g|\gamma}$. The information matrix for **γ** from gene *g*, conditional on $\delta_g = \hat{\delta}_{g|\gamma}$ is

$$A_{g,\gamma\cdot\delta} = A_{g,\gamma\gamma} - A_{g,\gamma\delta} A_{g,\delta\delta}^{-1} A_{g,\delta\gamma} \quad (10)$$

evaluated at $\delta_g = \hat{\delta}_{g|\gamma}$ [35].

The derivative of the profile REML log-likelihood ℓ_p therefore is

$$U_\gamma = \sum_{g=1}^{G} U_{g,\gamma} \quad (11)$$

and the information matrix associated with ℓ_p is

$$A_{\gamma\cdot\delta} = \sum_{g=1}^{G} A_{g,\gamma\cdot\delta} \quad (12)$$

evaluated at $\delta_g = \hat{\delta}_{g|\gamma}$. The REML estimate of γ can be evaluated by the nested scoring iteration

$$\gamma^{(i+1)} = \gamma^{(i)} + A_{\gamma\cdot\delta}^{-1} U_\gamma \quad (13)$$

where $\gamma^{(i)}$ is the *i*th iterated value and $A_{\gamma\cdot\delta}$ and U_γ are to be evaluated at $\gamma = \gamma^{(i)}$. The iteration will begin from a suitable starting value $\gamma^{(0)}$.

Full scoring iterations

In this section, convenient expressions will be derived for the quantities $A_{\gamma\cdot\delta}$ and U_γ. For any value of γ, the least squares estimator for β_g can be computed using weighted least squares computations (Equation 3) with working weights w_{gj}^* replacing the prior weights w_{gj}. The standardised residuals from this regression are

$$e_{gj} = w_{gj}^{*1/2}(y_{gj} - \mathbf{x}_j^T \boldsymbol{\beta}_g), \quad (14)$$

where $\mathbf{x}_j$ is the *j*th row of *X*.

Let

$$H_g = \Sigma_g^{-1/2} X (X^T \Sigma_g^{-1} X)^{-1} X^T \Sigma_g^{-1/2} = (h_{g,jk}) \quad (15)$$

be the projection matrix from the regression and write $h_{gj} = h_{g,jj}$ for the diagonal elements or leverages of H_g. Finally, let *Z* be the $J \times J$ design matrix

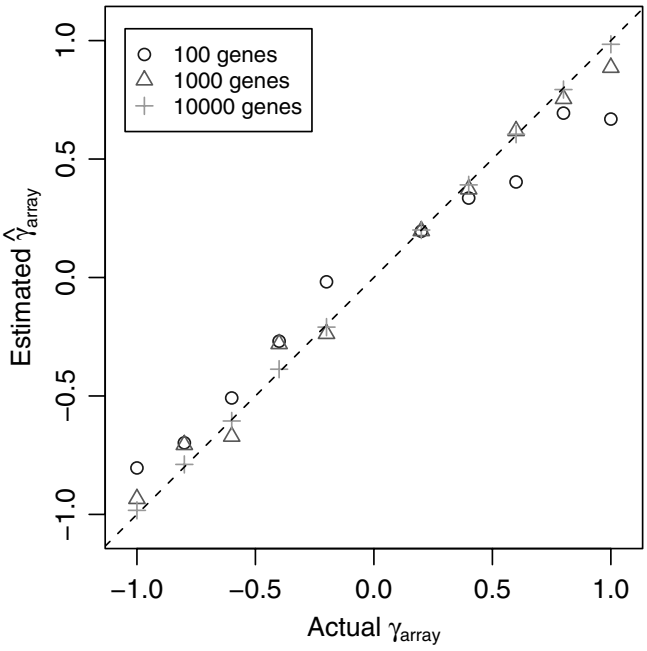

Figure 7

Estimated versus actual array variance parameters from simulated data. The gene-by-gene update algorithm was used to estimate the array variance parameters using 100, 1000 and 10000 genes from a simulated data set with 10 arrays. The estimates ($\hat{\gamma}_j$) are plotted against the true values (γ_j). As more genes are included in the iterations the accuracy of the estimates obtained from the gene-by-gene update algorithm improve, although with as few as 100 genes, the values recovered are broadly correct.

$$Z = \begin{pmatrix} 1 & 1 & 0 & \cdots & 0 \\ 1 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & 0 & \cdots & 1 \\ 1 & -1 & -1 & \cdots & -1 \end{pmatrix} \qquad (16)$$

Using these expressions we can write down computable expressions for quantities from the previous section. The conditional REML estimator of δ_g is $\hat{\delta}_{g|\gamma} = \log s^2_{g|\gamma}$ with

$$s^2_{g|\gamma} = \frac{1}{J-K}\sum_{j=1}^{J} e^2_{gj}. \qquad (17)$$

The score vector for γ is

$$U_{g,\gamma} = \frac{1}{2} Z_2^T \mathbf{z}_g \qquad (18)$$

where Z_2 is the last J - 1 columns of Z and $\mathbf{z}_g$ is the vector with components

$$z_{gj} = e^2_{gj} / s^2_{g|\gamma} - (1 - h_{gj}) \qquad (19)$$

for j = 1, ..., J. The information matrix is

$$A_g = \frac{1}{2} Z^T V_g Z \qquad (20)$$

where V_g is the $J \times J$ matrix with diagonal elements (1 - $h^2_{g,jj}$) and off-diagonal elements $h^2_{g,jk}$. Efficient algorithms exist to compute A_g [34]. Alternatively, it is often satisfactory to approximate the dense matrix V_g with the diagonal approximation V_{g1} = diag(1 - h_{g1}, ..., 1 - h_{gJ}) [25]. With this approximation, a straightforward calculation gives

$$2A_{g,\gamma\cdot\delta} = \text{diag}(1 - \mathbf{h}_{g(J)}) + (1 - h_{gJ})L - \frac{1}{J-K}(h_{gJ} - \mathbf{h}_{g(J)})(h_{gJ} - \mathbf{h}_{g(J)})^T \qquad (21)$$

where $\mathbf{h}_{g(J)} = (h_{g1}, ..., h_{gJ-1})^T$ and L is the J - 1 × J - 1 matrix of 1's. The nested information matrix $A_{\gamma\cdot\delta}$ therefore has diagonal elements given by

$$2A_{\gamma\cdot\delta,ll} = \sum_{g=1}^{G} \{1 - h_{gl} + 1 - h_{gJ} - (h_{gJ} - h_{gl})^2 / (J-K)\} \qquad (22)$$

and off-diagonal elements given by

$$2A_{\gamma\cdot\delta,lm} = \sum_{g=1}^{G} \{(1 - h_{gJ}) - \frac{1}{J-K}(h_{gJ} - h_{gl})(h_{gJ} - h_{gm})\} \qquad (23)$$

In matrix terms we can write

$$2A_{\gamma\cdot\delta} = \text{diag}(u_1, ..., u_{J-1}) + u_J L - N^T N/(J-K) \qquad (24)$$

where N is the matrix with ith row h_{gJ} - $\mathbf{h}_{g(J)}$ and u_j = $\sum_{g=1}^{G}(1 - h_{gj})(1 - h_{gj})$. With these quantities, the nested scoring iteration (Equation 13) is very memory efficient and can be carried out easily on a standard personal computer.

Gene-by-gene scoring iterations

Although memory efficient, the nested scoring iteration may still require a lot of computation for large G since G gene-wise regressions must be evaluated for every iteration. If the prior spot weights are equal, w_{gj} = 1, the gene-wise regressions can be computed very quickly but, if not,

a full set of least squares computations must be repeated for each gene and each iteration. In this section we explore a much lighter computation scheme in which only one pass is done through the genes and the array variance parameters are updated for each gene. This results in a very efficient gene-by-gene update algorithm which produces approximate REML estimators for the array weights.

The gene-by-gene update algorithm is given by

$$\gamma^{(g+1)} = \gamma^{(g)} + (A^*_{g,\gamma\cdot\delta})^{-1} U_{g,\gamma} \quad (25)$$

where $U_{g,\gamma}$ is as above (Equation 18) while A^* is an accumulating information matrix defined by

$$A^*_{g,\gamma\cdot\delta} = A^*_{g-1,\gamma\cdot\delta} + A_{g,\gamma\cdot\delta} \quad (26)$$

where $A_{g,\gamma\cdot\delta}$ is evaluated at $\gamma^{(g)}$ and $\hat{\delta}_{g|\gamma}$. The iteration is started from $\gamma^0 = \mathbf{0}$ and

$$A^*_{0,\gamma\cdot\delta} = \frac{10(J-K)}{J} Z_2^T Z_2. \quad (27)$$

These starting values begin the iteration from equal array variances with the information weight of ten genes. The effect of accumulating the information matrix in this way is to gradually decrease the step size of the iteration as the iteration passes through all the genes, resulting in a convergent iteration. The final value $\gamma^{(G)}$ is taken as the estimate of γ and is used to assign array weights. In our implementation in R [36], this algorithm calculates the array variance parameters in less than a second for the QC LMS data and in around 12 seconds for the METH data set on a 2.0 GHz Pentium M computer. The gene-by-gene nature of the algorithm means that minimal RAM is required for these computations.

While the gene-by-gene update algorithm is fast, it provides only an approximation to the REML estimators γ, and we need to check the accuracy of this approximation. To do this, expression values (y_{gj}) were simulated from normal distributions for J = 10 arrays and G = 10000 genes. The array variance parameters (γ_j) were equally spaced over the interval [-1, 1]. As already noted, the REML algorithm is invariant with respect to the gene-wise means and variances, so the gene-specific mean and variance parameters were set to zero in our simulations.

Figure 7 shows the estimated versus actual array variance parameters obtained from the gene-by-gene update algorithm after the first 100 genes, first 1000 genes and all 10000 genes respectively. The array variances are broadly correct even after 100 genes and after 1000 genes the accuracy is good. The root mean square errors between the update algorithm estimates and the true values averaged over the ten variance parameters were 0.17, 0.08 and 0.01 after 100, 1000 and 10000 genes respectively. These results indicate that the algorithmic short-cut taken by the gene-by-gene update algorithm does not seriously compromise the accuracy of the array variance estimates.

Discussion and Conclusion

This article has presented an empirical method for estimating quantitative array quality weights which is integrated into the linear model analysis of microarray data. Computationally efficient algorithms are developed to compute the array quality weights using the well-recognized REML criterion. As well as full REML estimation, a fast gene-by-gene update method which requires only one pass through the genes is described.

Examples of array quality weights which give less influence to the gene expression measurements from unreliable microarrays and relatively more influence to the measurements from reproducible arrays have been presented. In both simulated and real data examples, it has been demonstrated that array weights improve our ability to detect differential expression using standard statistical methods. The graduated approach to array quality has also been shown to be superior to filtering poor quality arrays both in simulations and for an experimental data set. In the simulations, filtering is shown to perform quite poorly, especially in combination with ordinary *t*-statistics. In the data example, filtering resulted in no significant genes to follow up, whereas the weighted analysis provided a few hundred sensible candidates.

The method is restricted for use on data from experiments which include replication with at least two residual degrees of freedom. For simple replicated experiments, a minimum of three arrays are needed and results from simulation studies show that this method is reliable in these situations, even in the presence of non-normally distributed data. Simulations were also used to show that array variance parameters are estimated with greater accuracy when more genes are available for the gene-by-gene update algorithm, and that these computational savings do not seriously compromise the accuracy of the final estimates. As a rule of thumb, we recommend that the full REML array weights be used when there are fewer than 1000 probes and that the gene-by-gene update method be used otherwise. The analysis of the control probes from the QC LMS data set showed that useful array weights can be obtained from the gene-by-gene algorithm with as few as 120 genes. The situation is different when there are no spot weights or missing expression values in the data. In this case the full REML algorithm can be implemented

very efficiently and so is recommended for any number of probes.

The empirical array weights form part of the quality and analysis pipeline and are not intended to replace the usual background correction, normalisation and quality assessment steps. In particular, array weights are not designed to account for spot-specific problems. The array weights method is instead designed to incorporate spot quality weights which might arise from gene filtering or from a predictive quality assessment step. The use of zero weights as prior weights ($w_{gj} = 0$) presents no problems for the method, although some special numerical treatment not discussed here is needed to ensure the sum to zero constraints are satisfied.

The array weight approach is also not intended to replace diagnostic array quality plots such as MA-plots, and arrays which are catastrophically poor quality should still be discarded. Taking a graduated approach to array quality, does however allow arrays of less than ideal quality, which would otherwise have to be discarded, to be kept in the analysis, but down-weighted.

The authors have applied the array weight method to very high quality data sets which featured arrays with low background, well-behaved controls and a good dynamic range of spot intensities. For such data sets, the method assigns approximately equal array weights to each array (data not shown). This indicates that the method does no harm when it is not required.

One further topic that deserves some attention is the use of robust linear models to estimate the gene expression coefficients. The array weights method has the same motivation as robust regression methods, but accumulates information on variability across genes on each array, which gene-wise robust regression methods are unable to do. Another consideration is sample size. While robust methods perform well on large sample problems, many microarray data sets such as the METH experiment consist of a small number of arrays and, in these situations, robust methods may not be suitable.

Authors' contributions

MER performed the data analyses, coded the algorithms and drafted the manuscript. GKS suggested the model and algorithms, helped with the coding and analyses and finalised the manuscript. The arrays from the QC data set were manufactured and hybridised by DD, RvL and AH. The METH experiment was planned and conducted by JN and AD.

Acknowledgements

Thanks to Terry Speed and Ken Simpson for their advice and for reading drafts of this manuscript. The anonymous reviewers are also thanked for their constructive comments on an earlier version of this manuscript.

References

1. Smyth GK, Yang YH, Speed TP: **Statistical issues in cDNA microarray data analysis.** *Methods Mol Biol* 2003, **224:**111-136.
2. Bolstad BM, Collin F, Brettschneider J, Simpson K, Cope L, Irizarry R, Speed TP: **Quality Control of Affymetrix GeneChip data.** In *Bioinformatics and Computational Biology Solutions Using R and Bioconductor* Edited by: Gentleman R, Carey V, Huber W, Irizarry R, Dudoit S. Springer; 2005:33-47.
3. Smyth GK, Speed TP: **Normalization of cDNA microarray data.** *Methods* 2003, **31:**265-273.
4. Schuchhardt J, Beule A, Malik E, Wolski H, Eickhoff H, Lehrach HH: **Normalization strategies for cDNA microarrays.** *Nucleic Acids Res* 2000, **28:**e47.
5. Wildsmith SE, Archer GE, Winkley AJ, Lane PW, Bugelski PJ: **Maximization of signal derived from cDNA microarrays.** *Biotechniques* 2001, **30:**202-208.
6. Spruill SE, Lu J, Hardy S, Weir B: **Assessing sources of variability in microarray gene expression data.** *Biotechniques* 2002, **33:**916-923.
7. Novak JP, Sladek R, Hudson TJ: **Characterization of variability in large-scale gene expression data: implications for study design.** *Genomics* 2002, **79:**104-113.
8. Wang X, Ghosh S, Guo SW: **Quantitative quality control in microarray image processing and data acquisition.** *Nucleic Acids Res* 2001, **29:**e75.
9. Tran PH, Peiffer DA, Shin Y, Meek LM, Brody JP, Cho KW: **Microarray optimizations: increasing spot accuracy and automated identification of true microarray signals.** *Nucleic Acids Res* 2002, **30:**e54.
10. Fan J, Tam P, Woude GV, Ren Y: **Normalization and analysis of cDNA microarrays using within-array replications applied to neuroblastoma cell response to a cytokine.** *Proc Natl Acad Sci USA* 2004, **101:**1135-1140.
11. Raffelsberger W, Dembele D, Neubauer MG, Gottardis MM, Gronemeyer H: **Quality indicators increase the reliability of microarray data.** *Genomics* 2002, **80:**385-394.
12. Jenssen TK, Langaas M, Kuo WP, Smith-Sørensen B, Myklebost O, Hovig E: **Analysis of repeatability in spotted cDNA microarrays.** *Nucleic Acids Res* 2002, **30:**3235-3244.
13. Yang IV, Chen E, Hasseman JP, Liang W, Frank BC, Wang S, Sharov V, Saeed AI, White J, Li J, Lee NH, Yeatman TJ, Quackenbush J: **Within the fold: assessing differential expression measures and reproducibility in microarray assays.** *Genome Biol* 2002, **3(11):**research0062.
14. Dumur CI, Nasim S, Best AM, Archer KJ, Ladd AC, Mas VR, Wilkinson DS, Garrett CT, Ferreira-Gonzalez A: **Evaluation of quality-control criteria for microarray gene expression analysis.** *Clin Chem* 2004, **50:**1994-2002.
15. Gollub J, Ball CA, Binkley G, Demeter J, Finkelstein DB, Hebert JM, Hernandez-Boussard T, Jin H, Kaloper M, Matese JC, Schroeder M, Brown PO, Botstein D, Sherlock G: **The Stanford Microarray Database: data access and quality assessment tools.** *Nucleic Acids Res* 2003, **31:**94-96.
16. Petri A, Fleckner J, Matthiessen MW: **Array-A-Lizer: A serial DNA microarray quality analyzer.** *BMC Bioinformatics* 2004, **5:**12.
17. Chen DT: **A graphical approach for quality control of oligonucleotide array data.** *J Biopharm Stat* 2004, **14:**591-606.
18. Steinfath M, Wruck W, Seidel H, Lehrach H, Radelof U, O'Brien J: **Automated image analysis for array hybridisation experiments.** *Bioinformatics* 2001, **17:**634-641.
19. Model F, Konig T, Piepenbrock C, Adorjan P: **Statistical process control for large scale microarray experiments.** *Bioinformatics* 2002, **18(Suppl 1):**S155-163.
20. **Supplementary materials** [http://bioinf.wehi.edu.au/resources/webReferences.html]
21. Kerr MK: **Linear Models for Microarray Data Analysis: Hidden Similarities and Differences.** *J Comput Biol* 2003, **10:**891-901.

22. Smyth GK: **Linear models and empirical Bayes methods for assessing differential expression in microarray experiments.** *Stat Appl Genet Mol Biol* 2004, **3(1):**Article 3.
23. Yang YH, Speed TP: **Design and Analysis of Comparative Microarray Experiments.** In *Statistical Analysis of Gene Expression Microarray Data* Edited by: Speed TP. CRG Press; 2003.
24. Verbyla A: **Modelling Variance Heterogeneity: Residual Maximum Likelihood and Diagnostics.** *J R Stat Soc [Ser B]* 1993, **55:**493-508.
25. Smyth GK, Huele AF, Verbyla A: **Exact and approximate REML for heteroscedastic regression.** *Statist Modelling* 2001, **1:**161-175.
26. Samartzidou H, Turner L, Houts T, Frorne M, Worley J, Albertsen H: **Lucidea Microarray ScoreCard: An integrated analysis tool for microarray experiments.** *Life Science News* 2001 [http://www4.amershambiosciences.com/].
27. Buckley MJ: **The Spot user's guide.** *CSIRO Mathematical and Information Sciences* 2000 [http://www.cmis.csiro.au/IAP/Spot/spotmanual.htm].
28. Yang YH, Buckley MJ, Dudoit S, Speed TP: **Comparison of methods for image analysis on cDNA microarray data.** *J Comput Graph Statist* 2002, **11:**108-136.
29. Yang YH, Dudoit S, Luu P, Speed TP: **Normalization for cDNA microarray data.** *Microarrays: Optical Technologies and Informatics, Proceedings of SPIE* 2001, **4266:**.
30. Smyth GK: **Limma: linear models for microarray data.** In *Bioinformatics and Computational Biology Solutions Using R and Bioconductor* Edited by: Gentleman R, Carey V, Huber W, Irizarry R, Dudoit S. Springer, New York; 2005:397-420.
31. Kooperberg C, Aragaki A, Strand AD, Olson JM: **Significance Testing for Small Sample Microarray Experiments.** *Stat Med* 2005, **24:**2281-2298.
32. **Limma** [http://bioinf.wehi.edu.au/limma]
33. Benjamini Y, Hochberg Y: **Controlling the false discovery rate: a practical and powerful approach to multiple testing.** *J R Stat Soc [Ser B]* 1995, **57:**289-300.
34. Smyth GK: **An Efficient Algorithm for REML in Heteroscedastic Regression.** *J Comput Graph Statist* 2002, **11:**836-847.
35. Smyth GK: **Partitioned algorithms for maximum likelihood and other non-linear estimation.** *Stat Comput* 1996, **6:**201-216.
36. R Development Core Team: **R: A Language and Environment for Statistical Computing.** 2006 [http://www.R-project.org]. R Foundation for Statistical Computing, Vienna, Austria