REVIEW

Fun with Microarrays Part II: Data Analysis

Paul C. Boutros

Division of Cancer Genomics and Proteomics; Ontario Cancer Institute; Toronto, Canada; M5G 2M9
Division of Signaling Biology; Ontario Cancer Institute; Toronto, Canada; M5G 2M9
Department of Medical Biophysics; University of Toronto; Toronto, Canada; M5G 2M9
Correspondence: paul.boutros@utoronto.ca

Microarray experiments are an important method for the high-throughput investigation of biological phenomenon. A wide variety of techniques and algorithms exist for analyzing and extracting information from microarrays. While the mathematical details of each algorithm are beyond the ken of most biologists, the basic steps and rationales underlying microarray data analysis need not be. This review focuses on nucleotide-based microarrays, by far the most popular type, and provides a simple overview of the steps involved in analyzing these experiments, the important algorithms used today, and the areas of active research.

Introduction

Over the past decade there have been two conceptual shifts in the way biology is studied. First, studies of single genes are being replaced by studies that probe many genes simultaneously. Second, different types of biological information, such as genomic alterations, mRNA levels, and protein levels, are being combined together in an attempt to give a comprehensive view of biological processes. Together, these two trends give what is usually called "Systems Biology" (*1-3*).

The emergence of systems biology has been driven by dramatic improvement in the equipment used to measure and study biological systems. For example, the advent of rapid DNA sequencers has led to the identification of thousands of new genes (*4*), while the emergence of affordable high-performance computers has revolutionized molecular modeling. Microarray-based technologies are another of these technological advances. Arguably, microarrays have had more impact than either DNA sequencing or increased computational capabilities.

This review is the second in a series of three articles describing microarray technologies. The first article discussed the underlying technology – the different types of microarray platforms, probes, and applications (*5*). This second part focuses on the analysis of microarray data, and the third will describe some of the most recent applications of this technology, with particular focus on systems biology.

If a researcher decides to embark on a microarray-based experiment, they may wonder why it is necessary to think deeply about the analysis of their microarray data. After all, "why should I care, won't the statistician or bioinformaticist take care of these details?" And indeed, if one is fortunate enough to have a collaborator to handle these details then both the sophistication and speed of the analysis can be greatly improved. Nevertheless, simple experiments are often analyzed by non-specialists, especially when specialists are not available for collaboration. And even if a dataset is analyzed by experts, an understanding of the steps they have taken will help in understanding the strengths and limitations of the conclusions reached. Furthermore, a greater understanding of the statistical analysis is critical for better understanding and communicating results, as when answering questions in local or international seminars.

Microarray analysis is a pipeline of sequential procedures, each motivated by a specific, physical aspect of a microarray experiment (**Figure 1**). Rather than focusing on the mathematical details of different algorithms, which have been reviewed by others (*6-9*), this review focuses on the underlying concepts. What steps need to be performed? Why? What options are available at each step? What are the major unresolved questions?

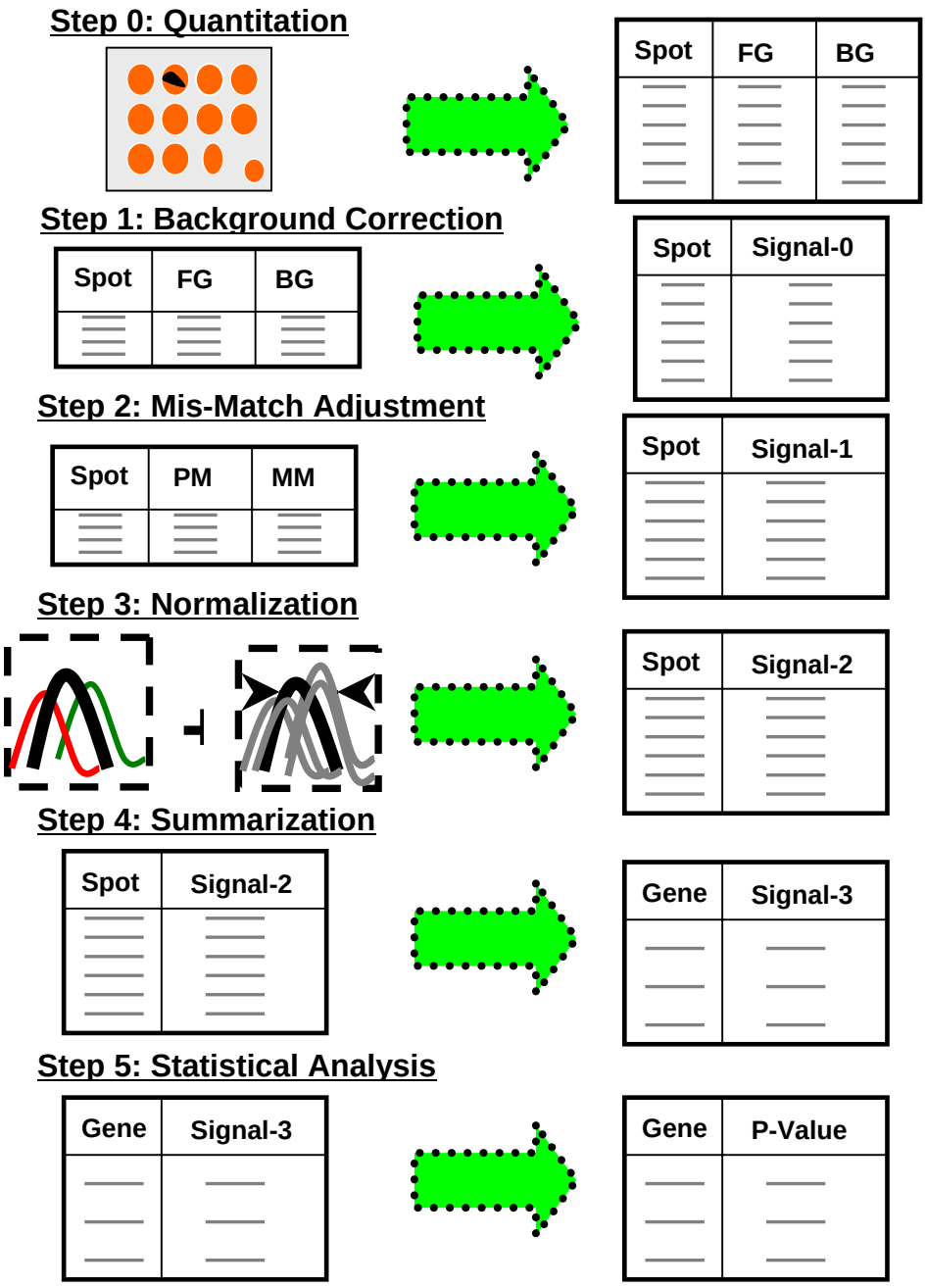

Figure 1. An overview of the microarray data analysis pipeline

Microarray analysis can be seen as a series of sequential steps. The output of one step feeds into the next step. All steps involve algorithms, implemented in various software packages (represented by the large arrows) that convert one type of data into another. As data moves along the pipeline, it becomes increasingly refined and representative of the underlying biology. The initial step of quantitation takes the raw image as an input, and converts it to estimates of signal in a spot (foreground signal, FG) and of non-specific signal in the surrounding regions to a spot (background signal, BG). In Step 1, these estimates are then combined into a single value of background-corrected signal (termed here Signal-0) by a background-correction algorithm. These adjusted signals are corrected for mismatches (if present on this array platform) in Step 2 to generate a new set of values, termed Signal-1. The data is then normalized twice (first within an array, then between arrays) to generate a set of normalized values (Signal-2). Replicate spots for a gene are then summarized (collapsed) into a single value, which is believed to be directly proportional to nucleotide values (Signal-3). Finally, statistical testing is done to assign a p-value to each gene, creating a dataset that can either be followed up with either wet-lab analyses or with integrative computational approaches (to be discussed in Part III of this series of reviews).

packages, but rather is a review of specific approaches and algorithms. If the underlying algorithm used in an analysis is inferior, then the results will be inaccurate regardless of how user-friendly the software interface is. In general, the highest quality and accuracy algorithms for microarray analysis are implemented in the open-source software packages. Commercial packages generally have a poor reputation for algorithmic quality, efficiency, and correctness. Most active research in microarray analysis uses the R statistical environment, generally as part of the BioConductor open-source project (*10*). Nevertheless, some robust commercial packages do exist.

Step 0: Quantitation
What is it?
The last experimental procedure in a microarray experiment is to hybridize labeled DNA onto the chip, then to wash away excess or weakly bound DNA. The resulting microarray is then excited with a laser and scanned. In essence, a "picture" of the array is taken, and areas with high binding will be visible as white spots on a black background. The first, mandatory step in analysis is to convert that image into a series of numbers. This process is called quantitation, and is absolutely essential for all microarray analyses. Indeed, people often fail to mention this step in their descriptions of methods because it is so essential to the analysis (hence the description of this step as "step zero").

Why do it?
All modern statistical methods involve the manipulation of numbers rather than images; that is, they require digital rather than analog information.

How is it done?
There are several different classes of algorithms for identifying spots and their edges. The most common types of algorithms integrate the signal along both the x- and y-axes, and use the peaks of this integration to make an initial estimate of the spot-location (**Figure 2**). This estimate is then refined either through iterative searching or "growing" around the initial point estimate (*11*).

What are the unresolved issues?
Strangely, little research is done into quantitation methods. In part this is because some studies have found that while existing algorithms vary dramatically in their underlying concepts, the final results change only slightly (*12*). Most research today does not go into improving estimates of spot-shape, but in automated and accelerating the process so that both arrays with more features, and experiments with more arrays may be processed more rapidly (*11*).

Recently, one group employed a continuous-wavelength scanner (termed a "hyperspectral scanner") to identify spurious fluorescence within experiments (*13*). Removing this source of noise greatly improved the fidelity of their studies. Should hyperspectral scanners become popular quantitation algorithms may have to be significantly changed to account for the extra dimension of wave-length dependence.

Step 1: Background Correction
What is it?
Each spot in a microarray has both specifically bound DNA and non-specifically bound DNA. The latter is thought to generate a "background" signal that must be subtracted or otherwise removed from the primary signal.

Why do it?
Background corrections are used to remove non-specific signal that arises from non-specific hybridization, the slide itself, or coatings or other materials on the slide. Many studies have demonstrated that careful removal of this signal can significantly increase the signal-to-noise ratio of a microarray experiment (*14-16*).

How is it done?
A large range of algorithms have been developed to remove this non-specific signal. The naïve approach is to simply subtract background signal from the foreground signal. Unfortunately this approach often leads to negative intensities. In fact, it has been shown that unbound spots can be less intense than the surrounding background (*17*).

For cDNA or long oligonucleotide arrays the two most successful methods both involve comprehensive models of the spot itself, either with Bayesian statistics (*18*) or by iterative fitting of a non-linear model (*19*).

For Affymetrix arrays, a seminal paper by Irizarry and coworkers evaluated several major background correction procedures and finds that the best method involves robustified fitting of a linear model (*20*). This method is part of the "RMA" procedure for processing Affymetrix

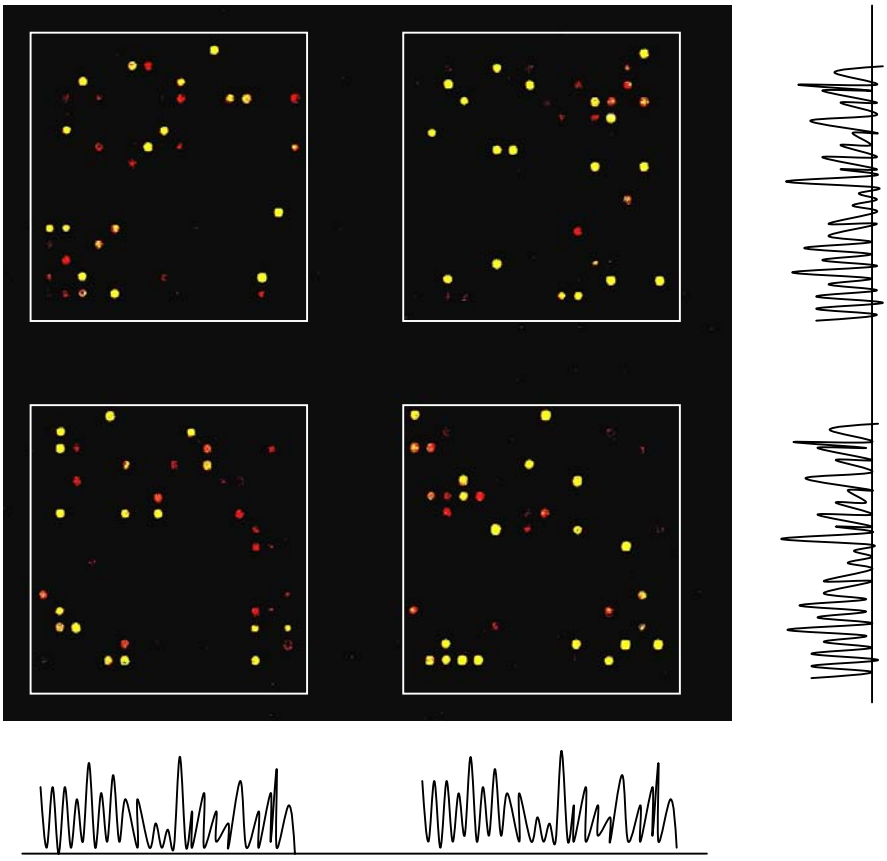

Figure 2. Spot-Finding Algorithms
In general, spot-finding algorithms work by integrating (adding) signal across the columns and rows of the array. In the figure this is shown for four grids of an array. Each grid is outlined in white; it should be emphasized that these lines are artificial; the actual array lacks this type of guidepost! Spots are thought to be located where column- and row-integrals simultaneously peak. A specialized spot-finding algorithm is used at the x-y locations of these peaks to determine the geometry and extent of each spot.

arrays, and is generally accepted as the optimal method for handling data from this platform (*21*).

What are the unresolved issues?
In general background corrections are well-characterized and easily handled by multiple robust algorithms. In particular, algorithms with very different mathematical backgrounds yield similar conclusions, suggesting that existing solutions are robust. Little active research appears to be underway in this area.

Step 2: Mis-Match Adjustment
What is it?
Some short oligonucleotide arrays – primarily Affymetrix GeneChip expression arrays – contain two types of sequences. They contain sequences designed to exactly match an mRNA transcript, called "perfect match" sequences. For each perfect match sequence, they also contain a "mismatch" sequence, which is an exact copy of the perfect match, but with a 1 bp mutation in the centre of the sequence. The idea is that this mutation will prevent binding of the actual transcript, but will allow any "non-specific hybridization" still to occur. As a result, the mismatch probes can be used to remove non-specific hybridization in a kind of secondary normalization process.

Why do it?
The concept is elegant: if the mismatch sequence performs as expected, this process can dramatically reduce the noise in an array experiment.

How is it done?
The caveat with this procedure is that the mismatch sequence rarely performs as expected. Early versions of the proprietary

Affymetrix analysis software (MAS4) would simply calculate: signal = perfect-match – mismatch. Unfortunately this often led to negative signals. Later versions of that software attempted to adjust for this problem by using non-parametric rank-statistics (*22*). Affymetrix unfortunately chose to assign a p-value to this comparison, which has led to countless researchers making the error of using this p-value for selection of differentially expressed genes. If a researcher reports a p-value from an n=1 microarray experiment, a misinterpretation of the mismatch correction data is the likely culprit! These p-values should not be interpreted as estimates of differential expression, but rather of ProbeSet quality. Because they are provided by slick commercial software packages, they are very frequently abused by non-expert analyzers of microarray data.

Work by Irizarry and coworkers – again, implemented into the RMA algorithm – showed that the mismatch data only added noise, and contributed very little signal (*21*). As a result, the RMA algorithm actually ignores all mismatch probes and only utilizes the perfect-match probes.

What are the unresolved issues?
While this approach is theoretically elegant, it has not performed well in practice. It is likely that future microarrays will no longer carry mismatch controls for non-specific hybridization.

Step 3: Normalization
What is it?
Normalization is the process of removing "non-biological variability" from a dataset. Common causes of this variability are simple fluctuations in experimental procedure: variable loading of DNA onto arrays, incomplete mixing of DNA across the array, or variability in the efficacy of the labeling reactions. Additionally, arrays are known to be more accurate at measuring higher-intensity transcripts leading to an "intensity bias".

Why do it?
Simply put: normalization is the single most important step in the analysis of a microarray experiment (*23, 24*). The selection of normalization method can triple the sensitivity and selectivity of an experiment.

How is it done?
There are literally hundreds of algorithms for the normalization of microarray data. Broadly, they can be divided into two classes. Within-array normalization algorithms seek to remove variability within a single array. Between-array normalization algorithms seek to remove variability across a set of arrays (*9*).

For Affymetrix data extensive work has shown that a between-array normalization algorithm called quantile smoothing is most efficient (*25, 26*). This algorithm merges information from all the arrays in an experiment to generate one averaged array (termed the reference distribution). Each individual array is then transformed so that it shares the same distribution as the reference distribution. This technique is embodied in the RMA pre-processing algorithm, and is very widely used today. Earlier techniques for Affymetrix datasets are less accurate and have generally been replaced by RMA.

There is less agreement about which normalization methods should be applied to cDNA microarray data. Many groups apply semi-parametric normalization methods (*27*) in the belief that they will introduce less bias than highly parameterized methods. One algorithm that appears to perform reasonably well for virtually all datasets is variance-stabilizing normalization (*19*). This approach attempts to compensate for the signal-bias, while simultaneously adjusting other sources of bias by iterative fitting of a non-linear model. While certainly not optimal for all datasets, variance-stabilizing normalization is perhaps the safest choice of algorithm for the analysis of cDNA microarray data.

What are the unresolved issues?
As microarray experiments expand beyond mRNA to measurements of genomic DNA many groups are trying to determine if the methods used for mRNA arrays are equally applicable to genomic arrays. At least in some cases, they appear not to be (*28*).

With the advent of large tiling arrays, much work has gone into improving the computational performance of these algorithms. Performing a variance-stabilizing normalization or a quantile-smoothing to a series of very large arrays is often not feasible on modern desktop

computers. In fact, most desktop computers are unable to normalize a moderated sized study of 100 human tumour samples. The limiting factor in many such analyses is not the speed of the processor, but its ability to load the entire dataset into memory for processing (i.e. the quantity of RAM). While 64-bit computers can alleviate these concerns, they are not yet as common and cheap as standard desktop machines. As a result, improving the efficiency of the underlying algorithms is an ongoing and important challenge in the field.

Step 4: Summarization
What is it?
A single gene might be represented by many sequences on an array. Different sequences might represent different parts of the gene, or they may be replicates. Summarization combines the signal from all sequences into a single number.

Why do it?
It is more convenient to work with a single number per gene, and this represents the underlying biology – one would generally expect different regions of the gene to be co-regulated. Splicing of course is an exception to this, and specialized arrays and algorithms are used to analyze splice-variants (*29, 30*).

How is it done?
For Affymetrix arrays, summarization is usually done, surprisingly, as part of the RMA pre-processing algorithm. A statistical procedure known as a median-polish is used to combine the signals from multiple Probes together. This procedure is called "robust" because it is relatively insensitive to outliers. That is, one "noisy" or "bad" sequence will contaminate the entire gene.

Interestingly, I am unaware of any published methods for summarization of cDNA arrays. In general duplicate spots are treated as duplicates and allowed to occur multiple times in the downstream analysis.

What are the unresolved issues?
The problem with summarization algorithms is their handling of splice-variants. Sometimes parts of the same gene behave differently because of biologically regulated and important splicing, and blindly running a summarization algorithm will obscure this data. Many groups are working on techniques for combining multiple probes together into a single model of a gene. The most advanced techniques in this area employ factor graphs and were developed in Toronto to study exon-specific signals with the aim of determining how many genes really exist in a mammalian genome (*30*).

Step 5: Statistical Analysis
What is it?
Statistical analysis is used to identify genes that change between experimental conditions.

Why do it?
The ultimate goal of nearly all microarray experiments is to identify a set of genes that are related to some biological phenomenon. For example, one might study genes that change expression in response to a drug, or genes whose expression is correlated with patient survival, or genes that are associated with cell-cycle progression. In each of these examples, a statistical analysis needs to be performed to select the relevant genes out of the bulk of the microarray data.

How is it done?
Standard statistical techniques are generally used, but employed sequentially on each gene in the microarray. To continue the examples above, to identify genes that change in response to a drug, one would employ a t-test to look for differential expression between drug-treated and vehicle-treated samples. This t-test would be repeated separately for every gene on the array. To identify genes that are associated with patient survival one would fit a Cox proportional hazards model relating patient survival to gene-expression and clinical factors. This Cox model would be fitted separately for each gene on the array. Similarly, to identify genes associated with cell-cycle progression a multivariate ANOVA model might be fit separately to each gene on the array.

While these are standard statistical procedures, their repeated use on thousands or tens of thousands of genes can lead to a major problem. Imagine that we are considering an experiment on a new drug treatment using an array with 10,000 genes. We have analyzed 20 drug-treated and 20 placebo-treated samples, so we have 10,000 sets of 40 numbers to consider. Unfortunately a competitor gains access to our data and replaces the data for all 10,000 genes with random numbers. Unaware of this mischief, we proceed with the analysis

and perform t-tests on these 10,000 sets of random numbers. If we then decide that any gene that shows $p < 0.05$ by a t-test is significant, we will find, by random chance alone 10,000 x 0.05 = 500 hits. The results of our competitor's malevolency and our use of naïve statistical analysis are 500 false-positive hits from our microarray study. Each of these 500 false positives will need to be tracked down, wasting much time and resources.

This problem – where repeating a test many times will generate large numbers of hits by random chance alone – is called the multiple-testing problem. A variety of "adjustments" have been proposed by statisticians to remove this danger. The most common adjustment is known as the "false discovery rate" (FDR) adjustment for multiple-testing (*31, 32*). In general, all statistical analyses of microarray data need to be followed by some sort of a multiple-testing adjustment.

What are the unresolved issues?

The statistical methods outlined above all involve taking a standard test and applying separately to each individual gene. This approach ignores the fact that many genes are correlated. That is, when one is high so are the others, or vice versa. These genes have natural relationships, and treating them as independent variables obscures this data. For example, proteins in large complexes such as the ribosome or proteasome are well-known to be co-expressed (*33*). The development of statistical methods that can take advantage of these correlations is an on-going challenge (*34*).

Step 6: Clustering & Integration Analysis

Clustering is the process of finding those patterns inherent to a dataset – the "natural trend" of the data. It does not exploit knowledge about what types of samples are studied, or what their clinical or biological annotation might be. Instead, it focuses in an unbiased way on the raw data itself. When properly used, clustering can be a very powerful tool, but unfortunately it is frequently misused in the analysis of microarray data. A recent review in this journal outlines the uses and misuses of clustering algorithms for microarray data analysis in great detail (*35*).

Integration analysis refers to the combination of microarray data with other data-types, such as mass-spectroscopic evaluations of protein levels, experimentally-derived protein-protein interaction networks, or analyses of upstream regulatory elements in the genome. A number of major integration approaches will be described in the third part of this series.

Conclusions

The analysis of microarray data is often treated and described like it is a "mysterious art" whose depths are only comprehensible to a select few experts. Paradoxically, this attitude is taken by experts and novices alike! Experts often wish to avoid complex and time-consuming explanations of the approaches available and their rationale for selecting particular methodologies. Indeed, often algorithms are selected because they are well accepted in the field, rather than for theoretical or empirical advantages. Further, an aura of "mystery" around the field generates prestige and dependency. On the other hand, novices may be unfamiliar with the broad range of algorithms available, the often contradictory studies of algorithm-efficacy, and of course the underlying mathematical basis. Because the mathematics of most technical papers in the field is inaccessible, it is natural to think that the basic concepts underlying those papers would also be inaccessible.

Fortunately, this is not the case. The basic concepts of microarray analysis are not inherently complex (**Figure 1**). An image file is first transformed into a series of numbers based on the intensity of pixels in the image. These numbers are then transformed by a series of algorithms to remove non-biological sources of variability, such as non-specific background signal. Following these "pre-processing" steps, statistical techniques are used to identify genes of interest. The statistics themselves are often very familiar – t-tests or ANOVA models – but are applied repeatedly to thousands of genes. Following a statistical adjustment for this "repeated testing", a gene-list is generated. Finally, data can be clustered (*35*) or integrated with other information, as will be discussed in part three of this review.

Acknowledgments

The author thanks Dr. Allan Okey and Ms. Ivy Moffat for helpful suggestions on earlier versions of this manuscript and Dr. Lyle Burgoon for his insightful conversations on microarray data analysis in particular, and bioinformatics in general.

References

1. E. C. Butcher, E. L. Berg, E. J. Kunkel, *Nat. Biotechnol.* **22**, 1253 (Oct, 2004).
2. D. W. Selinger, M. A. Wright, G. M. Church, *Trends Biotechnol.* **21**, 251 (Jun, 2003).
3. D. Noble, *Nat. Rev. Mol. Cell Biol.* **3**, 459 (Jun, 2002).
4. P. C. Boutros, *Hypothesis* **3**, 26 (2005).
5. P. C. Boutros, *Hypothesis* **4**, 15 (2006).
6. J. Quackenbush, *Nat. Genet.* **32 Suppl**, 496 (Dec, 2002).
7. I. V. Yang *et al.*, *Genome Biol.* **3**, research0062 (Oct 24, 2002).
8. M. Bilban, L. K. Buehler, S. Head, G. Desoye, V. Quaranta, *Curr. Issues Mol. Biol.* **4**, 57 (Apr, 2002).
9. G. K. Smyth, T. Speed, *Methods* **31**, 265 (Dec, 2003).
10. R. C. Gentleman *et al.*, *Genome Biol.* **5**, R80 (2004).
11. A. N. Jain *et al.*, *Genome Res.* **12**, 325 (Feb, 2002).
12. A. Lehmussola, P. Ruusuvuori, O. Yli-Harja, *Bioinformatics (Oxford, England)* **22**, 2910 (Dec 1, 2006).
13. M. J. Martinez *et al.*, *Nucleic Acids Res.* **31**, e18 (Feb 15, 2003).
14. C. S. Brown, P. C. Goodwin, P. K. Sorger, *Proc. Natl. Acad. Sci. U.S.A.* **98**, 8944 (Jul 31, 2001).
15. X. Wang, S. Ghosh, S. W. Guo, *Nucleic Acids Res.* **29**, E75 (Aug 1, 2001).
16. D. M. Rocke, B. Durbin, *J. Comput. Biol.* **8**, 557 (2001).
17. P. H. Tran *et al.*, *Nucleic Acids Res.* **30**, e54 (Jun 15, 2002).
18. C. Kooperberg, T. G. Fazzio, J. J. Delrow, T. Tsukiyama, *J. Comput. Biol.* **9**, 55 (2002).
19. B. P. Durbin, J. S. Hardin, D. M. Hawkins, D. M. Rocke, *Bioinformatics* **18 Suppl 1**, S105 (Jul, 2002).
20. R. A. Irizarry *et al.*, *Biostatistics* **4**, 249 (2003).
21. R. A. Irizarry *et al.*, *Nucleic Acids Res.* **31**, e15 (Feb 15, 2003).
22. W. M. Liu *et al.*, *Bioinformatics* **18**, 1593 (Dec, 2002).
23. R. Hoffmann, T. Seidl, M. Dugas, *Genome Biol.* **3**, RESEARCH0033 (Jun 14, 2002).
24. S. E. Choe, M. Boutros, A. M. Michelson, G. M. Church, M. S. Halfon, *Genome Biol.* **6**, R16 (2005).
25. B. M. Bolstad, R. A. Irizarry, M. Astrand, T. P. Speed, *Bioinformatics* **19**, 185 (Jan 22, 2003).
26. C. Workman *et al.*, *Genome Biol.* **3**, research0048 (Aug 30, 2002).
27. J. E. Eckel *et al.*, *Bioinformatics* **21**, 1078 (Apr 1, 2005).
28. F. D. Gibbons, M. Proft, K. Struhl, F. P. Roth, *Genome Biol.* **6**, R96 (2005).
29. J. M. Johnson *et al.*, *Science* **302**, 2141 (Dec 19, 2003).
30. B. J. Frey *et al.*, *Nat. Genet.* **37**, 991 (Sep, 2005).
31. B. Efron, R. Tibshirani, *Genet. Epidemiol.* **23**, 70 (Jun, 2002).
32. J. D. Storey, R. Tibshirani, *Proc. Natl. Acad. Sci. U.S.A.* **100**, 9440 (Aug 5, 2003).
33. K. R. Brown, I. Jurisica, *Bioinformatics* **21**, 2076 (May 1, 2005).
34. G. K. Smyth, *Statistical Applications in Genetics and Molecular Biology* **3**, 1 (December 2003, 2003).
35. P. C. Boutros, *Hypothesis* **4**, 28 (2006).