The Department of Statistics
The University of Auckland
New Zealand

Bayes Analysis for Microarrays

Marcus Davy

March 2004

Supervisor: Dr Mik Black

A thesis submitted in partial fulfillment of the requirements of Master of Science

The University of Auckland

Thesis Consent Form

This thesis may be consulted for the purpose of research or private study provided that due acknowledgement is made where appropriate and that the author's permission is obtained before any material from the thesis is published.

I agree that the University of Auckland Library may make a copy of this thesis for supply to the collection of another prescribed library on request from that Library; and

1. I agree that this thesis may be photocopied for supply to any person in accordance with the provisions of Section 56 of the Copyright Act 1994.

   Or

2. This thesis may not be photocopied other than to supply a copy for the collection of another prescribed library.

   (*Strike out 1 or 2*)

Signed: ....

Date: ....

Created: 5 July 2001
Last updated: 9 August 2001

Abstract

Microarrays are a recently developed technology designed to make inferences about the expression levels of thousands of genes simulateously. Dual channel cDNA microarrays measure relative expression of unknown mRNA fragments (gene expression information) after competitive hybridization to known spots of immobilized cDNA. Their design complexity allows for many sources of variation to infiltrate experiments, both within, and between, array slides. The nature of the data, and the experimental complexity has created considerable interest in the statistical community. The number of genes placed on cDNA microarrays is increasing, while the amount of information collected per gene is low. Coupled with this problem, the proportion of genes likely to differentially express is a small fraction of the spots on an array. Many practitioners have adopted multiple comparison procedures in hypothesis testing to analyze cDNA microarrays, controlling the proportion of Type I errors detected in a list of differentially expressing genes.

In this work simulations of microarray datasets were used to investigate the performance of popular analysis methods over a range of experimental settings. Empirical Bayes analysis was compared to t-statistic (within gene) analysis using a comparable multiple comparison procedure to contrast the two approaches. Empirical Bayes was found to be an extremely powerful analysis method when considering the observed proportion of false discoveries in predicted gene lists. When spot replicate information was low, Empirical Bayes analysis significantly outperformed t-statistic within gene analysis. From this, it is concluded here that information sharing between genes in variance estimation is an important factor in low replicate microarrays.

Acknowledgements

The author would like to thank Gavin Ross and Ross Crowhurst, who were both instumental in creating the opportunity for this study to take place. Many thanks go to Mik Black for his supervision, direction and commitment in this piece of study. Acknowledgements go to Tamara Sirey for her work editing Chapters 1, and 2. Finally, I would like to thank my wife Suzanne for her support throughout the year.

Contents

List of Figures

List of Tables

It has not escaped our notice that the specific pairing we have postulated immediately suggests a possible copying mechanism for genetic material - J.D Watson & F.H.C. Crick

1
Introduction

Molecular Biology is fast becoming a multidisciplinary field with underpinnings involving Biology, Chemistry, Physics and Mathematics. It involves the study of molecular processes within cells, the smallest fundamental unit within organisms. There are two major classifications of cells; *prokaryotes* and *eukaryotes*. Prokaryotic organisms are unicellular and have relatively simple anatomies. They have a circular genome to carry genetic information inside the cell, and simple organelle structures. Eukaryotic organisms can be unicellular or multicellular, and generally possess radically more complex structures. They have a nuclear membrane, encapsulating a nucleus organelle containing genetic information inside the cell. Specialized structures present in eukaryotes include chloroplasts in plants for photosynthesis of light into energy, and mitochondria in animals for respiration transforming glucose molecules under aerobic conditions into energy and the bi-products carbon dioxide and water.

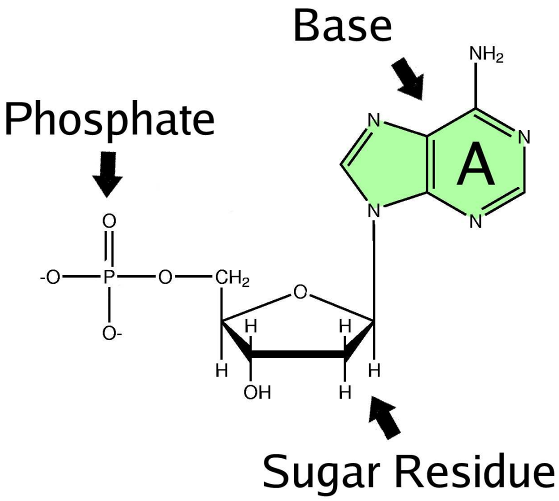
Cells contain a collection of structures in which controlled enzyme-catalysed chemical reactions maintain everything essential for organism survival. Proteins are the molecules which, based on their unique structure, catalyze specific chemical reactions in organisms. They are formed by the polymerization of repeating structural units called amino acids. Amino acids are joined head to tail by peptide bonds in a condensation chemical reaction to form polypeptide molecules. There are twenty different choices available for each amino acid residue allowing a massive number of various length polypeptide molecules to exist. Organisms synthesize thousands of different protein molecules whose vast range of physiochemical characteristics stem from the varied properties and combinations of the amino acid residues.

In 1944 Oswald Avery and his colleagues [1] demonstrated that deoxyribonucleic acid (DNA), is the molecule which carries genetic information inside the cell, and is the fundamental unit of inheritance within living organisms. In 1953 Watson and Crick [2] revealed the structure of DNA as a symmetrical double stranded helical duplex possessing complementary strands. Specific coding regions of DNA molecules are called *genes* which are the “genetic blueprint” encoding for proteins in the cell. *Chromosomes* are large DNA molecules which contain thousands of genes, packaged up with highly conserved histone proteins into a structured molecule. In 1958 Fancis Crick published his *central dogma* [3], defining perceived relationships of DNA molecules to protein molecules via intermediate ribonucleic acid (RNA) molecules, the paradigm of Molecular Biology.

1.1 Nucleic acids

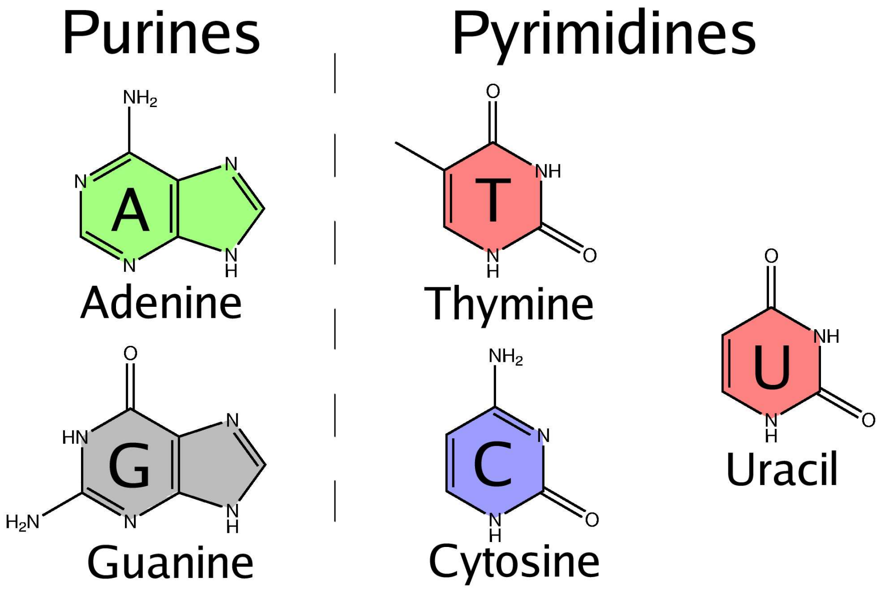
Nucleic acids are made up of the atomic elements Hydrogen (H), Carbon (C), Nitrogen (N), Oxygen (O), and Phosphorus (P). Each molecular strand is formed by the polymerization chemical reaction of repeating structural units called nucleotides of which there are three components, a phosphate group bonded to a sugar component which in turn is bonded to a nitrogenous base.

Figure 1.1: Components of DNA, the nucleotide base in the illustration is Adenine. Note that the deoxyribose sugar component is oriented almost perpendicular to the nucleotide base.



Nucleotides link to adjacent units to form long chains between the phosphate and sugar components held together by strong phosphodiester bonds. Phosphate groups bridge between the $3'$ and $5'$ carbon positions (labelled 1 to 5) of successive sugar residues, forming a strong backbone to the molecule. The sugar component structure differs slightly between DNA and RNA. In DNA the sugar component is deoxyribose, and in RNA the sugar component is ribose which is hydroxylated (containing an extra hydroxyl group). The ribose sugar in the backbone makes RNA strands more unstable than DNA strands, susceptible to base catalysed hydrolysis reactions cleaving apart the RNA backbone into smaller chain fragments. The nitrogenous base component can vary. There are five different base structures found in nucleic acids; Adenine (A), Guanine (G), Cytosine (C), Thymine (T), and Uracil (U). The five different nucleic acid bases are classified as either *purines* or *pyrimidines* depending on their structural similarity to their parent molecules, purine and pyrimidine. Purines have two carbon nitrogen rings in their structure, and pyrimidines have one carbon nitrogen ring. DNA chains are made up of 4 possible nitrogenous base components $\{AGCT\}$. In RNA Uracil replaces the structurally similar nitrogenous base Thymine, the 4 possible bases are $\{AGCU\}$.

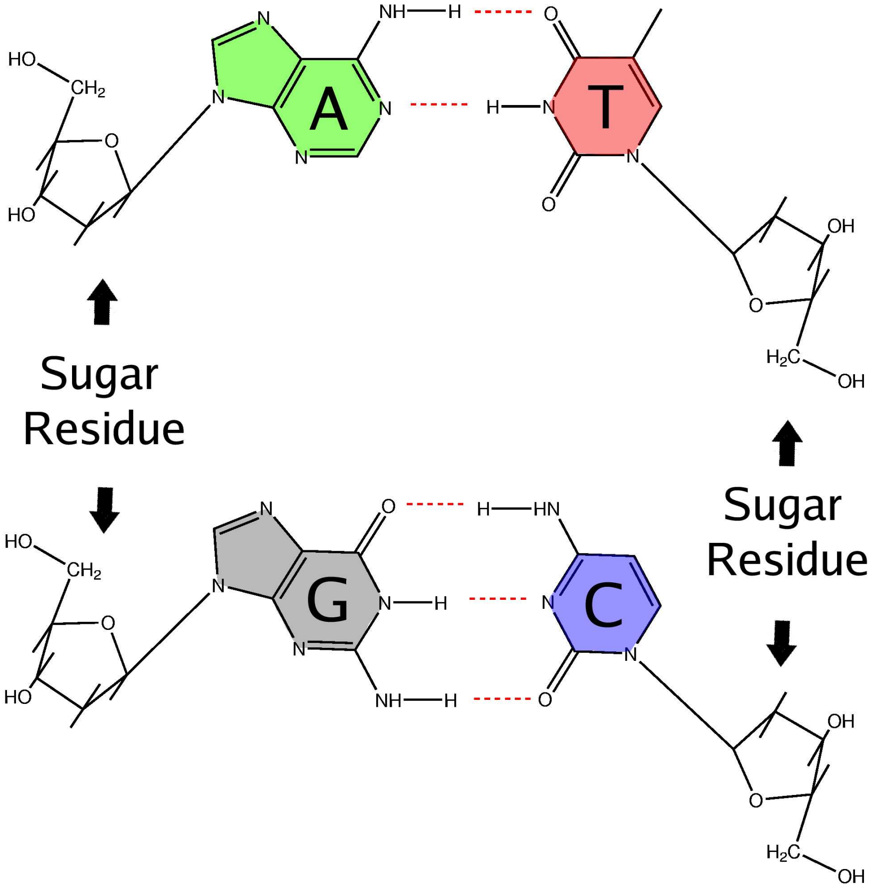
Figure 1.2: Purine and pyrimidine nucleotide bases in DNA, and RNA. Uracil replaces the base Thymine in RNA.



The allowable base components of nucleic acids can be polymerized in any order giving the molecules a high degree of uniqueness. The primary structure of nucleic acid molecules can be summarized by the sequence of nitrogenous bases starting with a phosphate group attached to the 5′ carbon atom of the first nucleotide, and ending with a hydroxyl group attached to the 3′ carbon atom. DNA in its native form is double stranded. The highly regular symmetrical corkscrew structure consists of two backbone chains on the outside of the helix, and nucleotide bases on the inside. Specific complementary base pairing between purine and pyrimidine nucleotides exists via hydrogen bonds: Adenine binds to

Thymine forming two hydrogen bonds, Guanine binds to Cytosine forming three hydrogen bonds.

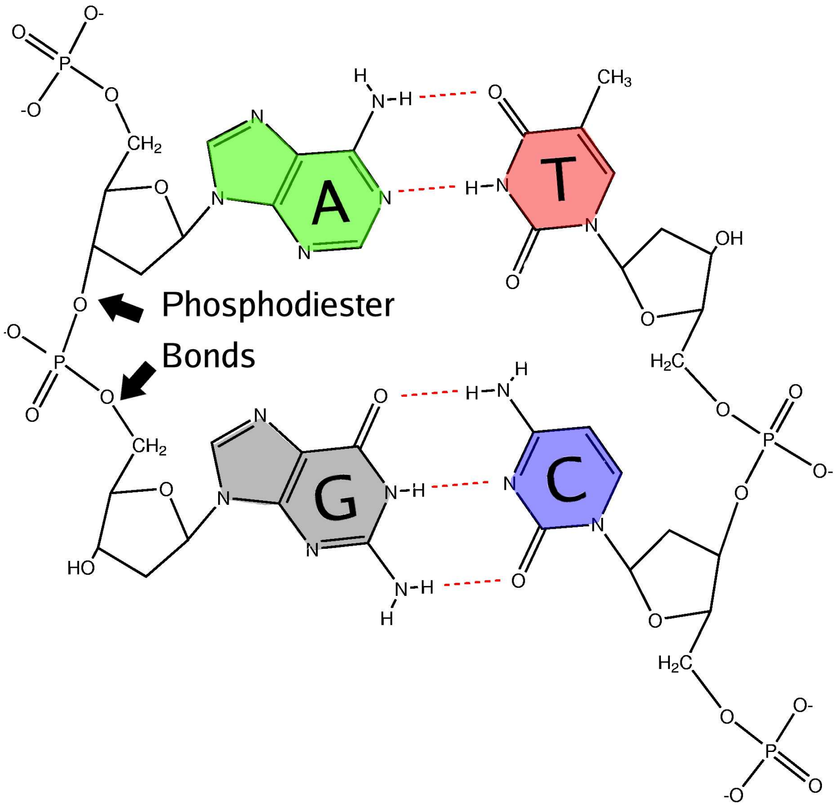
Figure 1.3: Schematic illustrating DNA purine-pyrimidine base pairing, hydrogen bonds are dashed lines.



The strand identified starting with a phosphate group attached to the 5′ carbon atom in the deoxyribose backbone is called the *sense* strand, the reverse complement strand starting with a phosphate group attached to the 3′ carbon atom in the deoxyribose backbone is called the *antisense* strand. The symmetrical structure of DNA strands allows for a mechanism of replication of the molecule.

The hydrogen bonds between paired nucleotides are weaker than the phosphodiester links in the backbone of the molecule, so that if DNA is heated above a characteristic melting temperature, T_m (typically around 72℃), thermal energy will break the hydrogen bonds between purine and pyrimidine nucleotides, and the helix will denature into separate single stranded DNA molecules in solution. The T_m value is defined to be the temperature at which 50% of identical denatured sense and antisense DNA molecule strands in equilibrium realign into double helix molecules. Sufficient thermal energy in the molecule breaks short runs of misaligned nucleotide base paired regions allowing global alignment of nucleotide base pairs to form under an S shaped probability melting curve, denaturation increases with temperature and available thermal energy. The T_m value is derived from the molecule length and the amount of Cytosine-Guanine content which forms stronger hydrogen bonds. Renaturation of the molecule can occur at temperatures 5° - 10℃ below the melting temperature. Under these annealing conditions denatured DNA almost completely renatures in a process called *hybridization*.

Figure 1.4: Schematic illustrating DNA Double Helix structure, adjacent purine-pyrimidine bases are actually planar to one another, almost perpendicular to the backbone.



The structure of RNA can be single or double stranded. Double stranded RNA does not form a helical structure due to sterical clashes in its hydroxyl group specifically in the second carbon position. Single stranded RNA can form secondary structures as intramolecular complementary purine and pyrimidine bases bond back on themselves within the chain to form loops. Double stranded RNA structures can form secondary structures with intermolecular complementary purine and pyrimidine bases bonding between the strands, and intramolecular bonding.

1.2 Protein synthesis

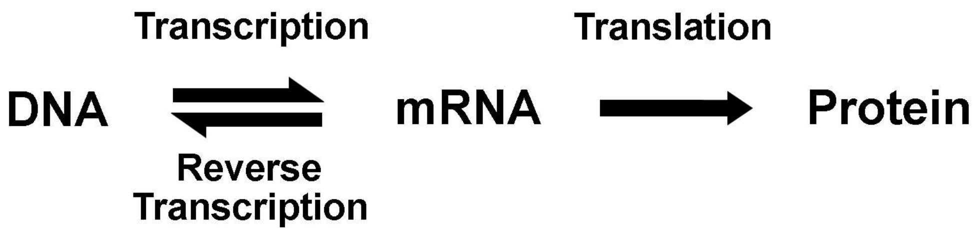
The central dogma for the functional roles of genes in DNA chromosomes occurs via a complex process. DNA molecules carry genetic information directing their own replication and transcription of genes. Double stranded DNA molecules self replicate during cellular division. Intermediate RNA molecules are templates for synthesis of polypeptide chains which undergo conformational tertiary structure changes to form proteins. Each molecule strand separates, so its complementary strand can be enzymatically synthesised in a semi-conservative manner to produce a complete copy of the molecule for each daughter cell.

DNA directs protein synthesis in a 2 stage process using complex molecular machinery to create polypeptide chains. In the first stage, single stranded messenger RNA (mRNA) is synthesised in a process called *transcription* from specific sections of the double stranded DNA templates by the enzyme RNA polymerase. An enzyme in *Escherichia Coli (E. Coli)* bacteria was discovered in 1970 by Temin & Baltimore [4, 5] making it possible to reverse this process in the laboratory. Under the process *reverse transcription*, mRNA can be synthesised back into single stranded complementary DNA (cDNA), an exact copy of the genetic template involved in protein synthesis. Messenger RNA contains the complement of the genetic template of a specific gene coding for a specific polymerized sequence of polypeptides for protein synthesis. After transcription the messenger RNA molecule is transported by the cell to the cytoplasm for protein synthesis. There are two structural classes of RNA involved in protein synthesis, transfer RNA, and ribosomal RNA. Transfer RNA and ribosomal RNA contain evolutionary conserved secondary structures which perform functional roles during protein synthesis.

In the second stage, mRNA encodes for amino acids in sequential order under an irreversible process called *translation*. Sequential triplets of nucleotides in mRNA called *codons* encode for specific amino acids. There are $4^3 = 64$ combinations of nucleotides encoding for twenty possible amino acids. Redundancy in the third position of triplet combinations allows more than one codon to encode for a particular amino acid. Transfer RNA recognizes codons and carries corresponding amino acids for sequential polymerization synthesis in conjunction with ribosomal RNA. The synthesised polypeptide transcript products undergo post-translational processing to form functional entities. This involves protein folding to create functional tertiary and quaternary structures (complexes of multiple protein structures) capable of carrying out different functions in the life cycle of an organism.

The conversion of encoded information from genes to mRNA, and then to the resulting protein is known as gene expression. The rate at which this occurs can vary between genes. Intermediate mRNA molecules degrade within minutes in the cytoplasm of cells, making abundance highly correlated with the rate that a cell synthesises proteins. If a gene is

Figure 1.5: Diagram of Francis Crick’s Central Dogma.



highly expressed, then its corresponding mRNA template will be highly abundant in the cytoplasm of the cell.

In the later part of the 20th century various experimental breakthroughs were discovered in the laboratory to aid in the scientific manipulation of nucleic acid molecules. Restriction Endonucleases were discovered in certain bacteria species by Hamilton Smith and Daniel Nathans in the late 1960’s [6]. These enzymes recognize specific nucleotide base patterns four to eight bases in length within double stranded DNA. They cleave the backbone of both strands of the duplex in specific positions creating two molecular fragments, with either blunt ends or sticky overhang ends depending on the Restriction Endonuclease used. Type II Restriction Endonucleases cut at points within the recognition site, enabling scientists to cut double stranded DNA molecules into many smaller fragments, all cut at the same recognition site.

In 1975 Ed Southern published a technique [7] for analyzing gene structure and measuring the quantity of specific DNA products of interest. This was accomplished by utilizing hybridization properties of DNA to detect specific sequences in complex populations of DNA fragments. The technique (dubbed Southern blotting) involves using electrophoresis to separate genomic DNA fragments in a gel by molecular size, blotting the fragments onto a nitrocellulose filter, then hybridizing a fluorescent or radioactively labelled complementary DNA probe of known composition to the gel in order to identify specific fragments of interest. Soon after in 1977, Alwine et al. [8] published a similar technique named Northern blotting to manipulate mRNA isolated from cells. In this technique electrophoresis separates the mRNA fragments based on their molecular size, the fragments are then hybridized with a fluorescent or radioactively labelled complementary mRNA probe of known composition to identify specific fragments of interest. These methods are reasonably sensitive and can detect small amounts of DNA or mRNA. The disadvantage is that they are labour intensive and can only investigate a small number of genes at a time.

At the beginning of the 21st century, molecular biology moved into the genomics era. Automated sequencing methods were developed to read the genetic code of genomic DNA and reverse transcribed cDNA, minimizing human labour and error in the laboratory. *Expressed sequence tag* (EST) sequencing involves the cellular extraction of specific expressed mRNA from cell samples in organisms. The cDNA versions of these transcripts are obtained by reverse transcription and sequenced to identify the underlying bases in the mRNA transcript. EST sequence base composition information is stored in databases such as the *Genbank*[1] flatfile database, in extensively annotated records. The volume of information in these databases from EST sequencing and genomic sequencing has increased in the last decade at an exponential rate. The underlying annotation information is extremely useful in the microarray experiments discussed in the next section.

1.3 Microarray technology

1.3.1 Overview

Macroarrays, microarrays and oligonucleotide array technologies were all developed within the last decade. These technologies are an extension of Northern blotting, and are designed to measure mRNA expression levels for thousands of genes simultaneously in a single experiment. This technique can be used to look at differential gene expression in specific tissues of an organism. An array based approach is one of the first screening steps in identifying candidates for further laboratory experimentation to make inferences about gene function. The general methodology involves taking advantage of the specific sense and antisense hybridization binding properties of nucleic acid molecules.

Consider a unique mRNA molecule of unknown abundance to be a *target* species of expressing RNA of interest. The antisense copy of the target mRNA is a cDNA *probe* which will bind to the target mRNA under controlled hybridization conditions. The idea is to separately immobilize thousands of antisense probes of known sequence identity, complementary to target mRNA transcripts of interest. Unique antisense probes are placed in an ordered array of spots bound to a glass slide, with each spot containing millions of identical copies of the unique antisense probe. Probes are selected in a carefully constructed design so that there will be high specificity to target mRNA molecules under hybridization. Each probe must be gene and organism specific across the microarray otherwise hybridization by sufficiently similar mRNA species will compromise the experimental results.

The cDNA constructs are created and amplified by either reverse transcription or oligonucleotide synthesis (synthetic constructs of short runs of nucleotide sequences). A tissue sample of expressing target mRNA transcripts is collected, within which the abun-

[1] National Center for Biotechnology Information, http://www.ncbi.nlm.nih.gov

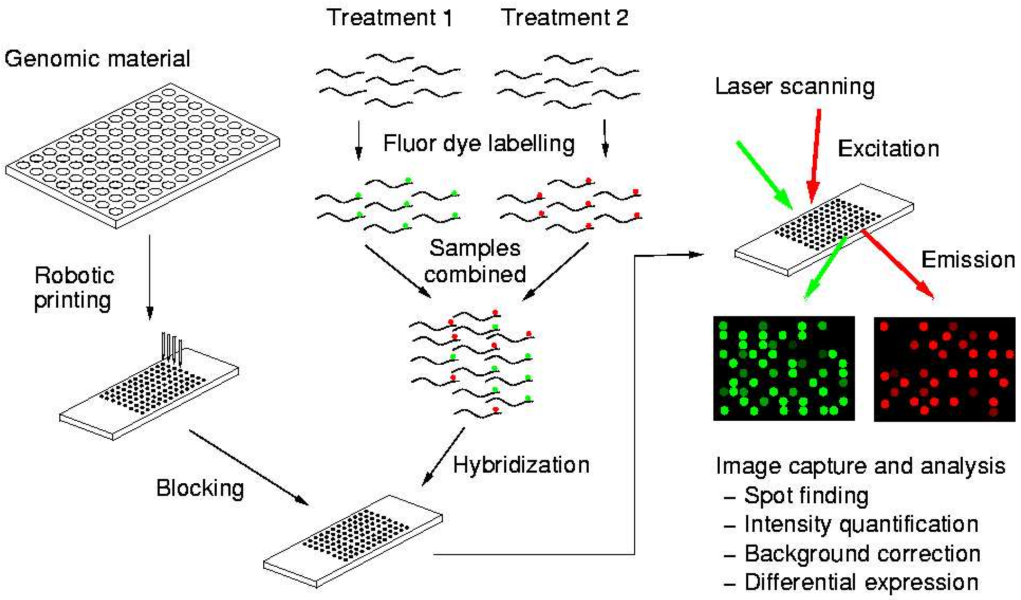
dance of each target species is unknown. All of the collected mRNA transcripts are subsequently labelled with a fluorescent dye molecule. The labelled tissue sample is washed over the glass slide under carefully controlled hybridization conditions allowing an equal probability of annealing by all the mRNA transcripts present. Target mRNA species will have specific affinity to hybridize to their complementary cDNA probes at appropriate annealing temperature conditions. After hybridization, the sample is carefully washed off the slide surface leaving only the hybridized complementary mRNA bound to the probe by hydrogen bonding.

Image analysis is used to quantify the amount of fluorescence emitted from each hybridized spot. The recorded intensity is proportional to the abundance of the transcript bound to the probe from the tissue sample. This approach can be extended to many thousands of probes immobilized in an ordered array of spots limited only by the space available on the glass slide.

1.3.2 Complementary DNA microarrays

Complementary DNA microarrays allow competitive hybridization to take place between multiple target samples identified with different fluorescent markers. Current two target channel technology uses the fluorescent dyes, cy5 (red) and cy3 (green), to identify mRNA transcripts from either target sample. These dye molecules emit fluorescence upon excitation with laser light at the wavelengths, 635 nm, and 532 nm respectively. Usually a control sample and a treatment sample are analyzed, with interest focussed on detecting differences in the abundance of the mRNA transcripts in each sample. Simultaneous competitive hybridization of the two target samples is allowed to take place. Messenger RNA species from either sample under optimal experimental conditions have an equal probability of hybridizing to spotted cDNA probes. The intensity of the emitted fluorescence by each hybridized spot is proportional to the relative abundance of hybridized mRNA transcripts derived from both target samples. This is a crucial property of cDNA microarrays since quantifying the absolute amounts of starting target mRNA in tissues samples is difficult. If both mRNA transcripts are of equal abundance for a particular gene, the spot(s) corresponding to that gene will emit equivalent fluorescence in both channels. The resulting combined colour after image analysis contains varying shades of yellow depending on transcript abundance. The equal abundance of mRNA transcripts between target samples is known as *equivalent expression.* If each mRNA transcript is present in significantly different amounts the resulting combined colour after image analysis will be varying shades of red or green respectively, depending on differential signal abundance. The differing abundance of mRNA transcripts between target samples is known as *differential expression.*

Figure 1.6: Overview of cDNA spotted array experimental process (Courtesy of Mik Black).



1.3.3 Array construction

Public and private sequencing projects contain vast amounts of useful EST sequence and annotation information. Researchers use these databases to construct cDNA libraries of expressed mRNA cells for a selected organism. Informatics algorithms are then used to construct a suitable set of cDNA probes of interest which are stored in multiple 384 well microtitre plates. Each 384 well microtitre plate is a 16×24 grid of wells exactly 4.5mm apart. The position of each probe in each well is tracked for spotting onto glass slides. Coatings are added to the slides to make the surface hydrophobic and positively charged, so cDNA probes will stick to the surface.

The total printable area of an experiment covers about 22.5×22.5 millimetres of a single glass microscope slide. Printing tips of an open capillary design similar in principle to a quill pen are used to deposit cDNA probes onto the slide surface. Current technology uses 16 print tips in a 4×4 grid, with each tip spaced 4.5 millimetres apart for dipping into the 384 well microtitre plates. Highly accurate robotics control the print tip movements, surface tension loads 16 probes into the print tips, and precise robotic control taps them onto the glass slides depositing less than $1 \mu l$ of the cDNA probe. The open capillary design allows rapid rinsing and drying of tips before spotting the next 16 samples onto the slide. Each spotted probe is approximately 130 μm in diameter and contains millions of copies of identical cDNA. Each print tip spots a grid matrix of probes within a 4.5^2 millimetre area limited by the distance to the next print tip grid. The complete process can take several hours to complete depending on the number of probes and arrays to be spotted. The slides are then “snap” cooled to ensure each cDNA probe is uniformly distributed throughout the spot, and denatured so each probe is able to be hybridized. Finally, blocking agents are applied to stop DNA hybridizing over the unspotted regions of the slide surface.

1.3.4 Hybridization

Each fluorescently labelled treatment sample is placed in the same hybridization mixing chamber along with the spotted microarray slide at suitable annealing temperatures. The hybridization solution must be well mixed to ensure mRNA targets from both samples are evenly distributed throughout the solution. Identical mRNA species from each treatment condition competitively hybridize to their complementary cDNA probes. Mixing ensures that each target mRNA species has an equal probability of binding to the complementary probes. This process takes several hours under carefully controlled conditions. After hybridization the excess liquid is washed off the microarray slide to remove unhybridized mRNA. Blocking agents ensure that unhybridized target mRNA does not bind to unspotted areas of the glass slide.

1.3.5 Fluorescence detection

Fluorescence is caused in certain molecules by the absorption of light at a certain wavelength which matches the energy required to excite electrons to a higher transitional state. Electrons within fluorescent molecules fall back to their ground state releasing lower energy fluorescent light at a longer wavelength lower in energy than the excitation wavelength in a phenomenon called *Stokes shift* [9]. The difference in excitation energy and emission energy is dissipated as heat within the molecule. In competitive hybridization experiments (e.g., cDNA microarrays) fluorescent dyes are chosen which have different emission wavelengths so their signals can be detected independently of one another within the same experiment. The emission energy is linearly proportional to the amount of fluorescently labelled cDNA hybridized to the probe.

The two fluorescent dyes, cy5, and cy3, display fluorescence upon light excitation at 635nm, and 532nm respectively. Lasers are used as the excitation light source for the wavelengths required, with the emitted fluorescent light detected using a photomultiplier tube which measures a voltage proportional to the amount of photons absorbed in the detector. The cy5 dye requires less excitation energy and is more susceptible to photodegredation. The newer generation of laser scanners are able to simultaneously scan and detect fluorescence at both wavelengths.

The microarray slides are processed using a 16 bit fluorescent scanner integrated with computer software, and image files are stored in the lossless TIFF format for each fluorescent channel. Image analysis software creates a raw intensity data file to quantify spot brightness, and quality, for each of the fluorescent dyes as representative summary statistics. Median summary statistics of intensity values are usually most representative of spot images, as they are more robust measures of the distributions of pixels than mean summaries. Median intensity values are proportional to median photon counts for each spot detected with a photo multiplier tube in the scanning process. The theoretical raw intensity values range from 1 to 65536, or 2^0 to 2^{16} on the binary scale.

1.3.6 Normalization

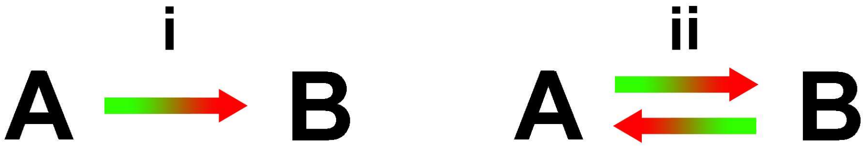
Throughout the process of conducting a cDNA microarray experiment there are several sources of variation introduced by the experimental process that need to be taken into account. It is likely that there will be unequal amounts of expressing mRNA transcripts between target samples, as it is difficult to accurately measure expressed mRNA during cellular extraction. The fluorescent labeling reactions of target mRNA and the hybridizations to cDNA probes may also have different efficiencies for each species of mRNA. Fluorescent dyes have been found to display intensity dependence [10], and printing tips may behave slightly differently due to imperfections as a result of age and wear. Any of these factors can bias the results. To account for this, numerical techniques are used to normalize the raw intensity data after image analysis. The total amount of expressing mRNA in target samples is assumed to be the same in cDNA microarrays using competitive hybridization so the ratio of total measured expression in each channel should be equal to 1. If abundance of labelled mRNA transcripts is significantly higher in either channel, naive total intensity normalization can be used, compensating up the total intensity in the other channel so the ratio of total measured expression in each channel equals 1. A subset of housekeeping genes essential for cell survival within both target mRNA samples that are known to always have high expression can be normalized to 1, especially useful in experiments where a treatment drastically changes the amount of gene expression relative to the other treatment. Print tip normalization techniques using *loess* [11] non-parametric smoothing have been used to account for variation between print tips. Further reading about different normalization techniques is available in [12].

2 Methods

2.1 Visualizing microarray designs

Graphical diagrams are a visually interpretable way of representing a set of related two colour microarray experiments. *Edge directed graph* illustrations simplify complex microarray experimental designs on the same spotted arrays. There are two parts to their structure, *nodes* and *edges*. Target mRNA samples are assigned as labelled nodes for a set of microarray experiments on the same spotted cDNA probes. Edges are lines connecting any two nodes corresponding to actual competitive hybridizations between two target mRNA samples on a microarray slide. For visual enhancement bi-coloured arrows depict edges incorporating dye labeling information to each target mRNA, one convention labels green at the base of the arrow representing the cy3 dye, and red at the arrowhead representing cy5 the dye (Figure 2.1i).

Figure 2.1: (i) Directed graph of a single microarray experiment.(ii) A dye reversal experiment. Target mRNA samples are labelled A and B, usually the control target mRNA is labelled using cy3 green dye.



Direct and indirect comparisons can be made between any two target mRNA samples via single and multiple nodes as long as a path can be traced via edges through nodes.Each arrow depicts an actual microarray hybridization experiment with its own sources of variation, so the shorter the path across multiple microarrays the higher the precision will be for statistical analysis [13]. Multiple competitive hybridizations using the same dye allocations between two target mRNA samples can be summarized by a single arrow labelled with an integer representing the number of repeated microarray hybridizations. A dye reversal experiment where fluorescent dyes are swapped between samples on two microarrays can be visually summarized with two arrows connecting target treatment nodes.

2.2 Measured intensities

Genepix scanner image analysis software such as the proprietary *Genepix Pro* software [14] bundled with the *Genepix 4000B* scanner creates a *genepix results format* (GPR) text file representative of the two 16 bit TIFF file images from the laser scanned slides. The first 31 lines of this file are diagnostic scanner settings. The GPR data following the diagnostic information consists of a matrix of multiple columns of microarray annotation, grid coordinates, measured intensities, and diagnostics for the two channels. The red and green foreground intensities, and the background red and green intensities are the responses of interest. Background correction by subtracting neighbouring background intensities from the adjacent foreground spot intensities is a standard technique to adjust for uneven background anomalies caused by factors such as uneven blocking, and slide surface non uniformity. The resulting raw intensities are denoted $\underset{\sim}{R}$ and $\underset{\sim}{G}$, representing the intensity channel derived from red and green fluorescent images.

2.3 Notation

2.3.1 Single microarray

In a microarray experiment involving a single array with cDNA probes spotted on the microarray complementary to m genes of interest (as in Figure 2.1), each treatment is assigned to one of two fluorescent dyes. This causes complete confounding between fluorescent dyes and the treatments themselves. Often channels are labelled by dye colour. This is not capable of being extended to experimental designs incorporating multiple microarrays where dye reversals take place in some of the target mRNA treatment hybridizations. For this reason the following notation, defining $\underset{\sim}{x} = \underset{\sim}{G}$, and $\underset{\sim}{y} = \underset{\sim}{R}$ will be used to represent the measured intensities in a single microarray experiment where $\underset{\sim}{x}$ and $\underset{\sim}{y}$ refers to the intensity vectors of actual target mRNA treatments,

$$\underset{\sim}{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_m \end{pmatrix} \quad \underset{\sim}{y} = \begin{pmatrix} y_1 \\ \vdots \\ y_m \end{pmatrix}. \tag{2.1}$$

After normalization to adjust the channel intensities of the microarray slide for effects due to the technological process rather than biological differences in target mRNA or printing artefacts, the labeling of treatments is arbitrary, as dye bias effects are assumed to have been removed. Designs that have no replication of probed mRNA spots eliminate the ability for assessing any within-gene intensity measurement error.

In competitive hybridization of single cDNA microarrays, the primary comparison of interest is the relative expression ratio T, for each of the m genes on the array. By convention the treatment labelled with the cy5 (red) dye is the numerator of the expression ratio,

$$T_i = \frac{R_i}{G_i} = \frac{y_i}{x_i}\ , i = 1, \ldots, m, \tag{2.2}$$

where m is the number of spots on the array. Theoretical scanned intensities in each channel range in magnitude between $2^0 = 1$ and $2^{16} = 65536$. The distribution of the relative expression ratio distribution, T, under equivalent expression should be centred around 1. Upward regulated genes in treatment y_i will have expression ratio greater than 1 covering a range between (1, 65536), downward regulated genes in treatment y_i will have an expression ratio less than 1 covering a range of (0, 1), the reciprocal ratio. The problem with examining measured intensity ratios on an untransformed scale is that the extremely right skewed distributions of each intensity channel cause the range of downward regulated genes to be much smaller than upward regulated genes. For example, genes which exhibit twice the expression in one of the channels, will either have a relative expression ratio of 0.5 , or 2. The log transformation is commonly used to down-weight intensities adjusting values relative to signal size. This adjusts the scale for comparison of logged reciprocals to one that is symmetric. The log_2 transformation has further interpretation significance as it relates to the base 2 of 16 bit scanner binary numerical storage in computers. The log ratio is given by,

$$\begin{aligned} log_2(T_i) &= log_2\left(\frac{y_i}{x_i}\right) \\ &= log_2(y_i) - \log_2(x_i), \text{ where } i = 1, \ldots, m \text{ genes.} \end{aligned}$$

Under equivalent expression, signal in both channels is the same, and thus, $log_2(T_i) = 0$. Upward and downward regulated ratios are differentiated by the sign of the log transformed intensities, as illustrated in Figure 2.2.

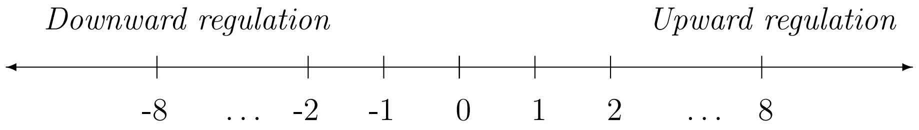


Figure 2.2: Theoretical number line of the log ratio, $log_2(T_i)$ of measured intensities.

2.3.2 Multiple microarrays

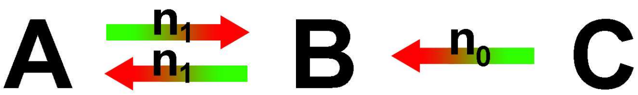
Assuming there is no replication of probed mRNA spots on any of the microarray slides in the series of experiments, suppose we are interested in comparisons between cDNA target samples A and B, as shown in Figure 2.3. Let n_1 and $n_2 = n_1 + n_0$ denote the number of replicate measurements in each condition, where n_0 provides auxiliary information between target samples B and C.

The notation in 2.1 can be extended to the matrices, X and Y, of intensities from a collection of related microarray experiments,

$$\mathbf{X} = \begin{pmatrix} x_{11} & \ldots & x_{1n_1} \\ \vdots & \ddots & \vdots \\ x_{m1} & \ldots & x_{mn_1} \end{pmatrix} \qquad \mathbf{Y} = \begin{pmatrix} y_{11} & \ldots & y_{1n_2} \\ \vdots & \ddots & \vdots \\ y_{m1} & \ldots & y_{mn_2} \end{pmatrix}.$$

In single, and multiple array experiments, after appropriate within-slide and between-

Figure 2.3: Directed graph of n_1 hybridizations between A and B, and n_0 hybridizations between B and C in a balanced dye swap microarray experiment.



slide normalization, intensities derived from either channel could have been arbitrarily labelled as the vector $\underset{\sim}{z}$, or the matrix **Z** respectively. This notation will be used for calculations involving the same computation in both channels where,

$$\underset{\sim}{z} = \begin{pmatrix} z_1 \\ \vdots \\ z_m \end{pmatrix}, \text{ or } Z = \begin{pmatrix} z_{11} & \dots & z_{1n_1} \\ \vdots & \ddots & \vdots \\ z_{m1} & \dots & z_{mn_1} \end{pmatrix}.$$

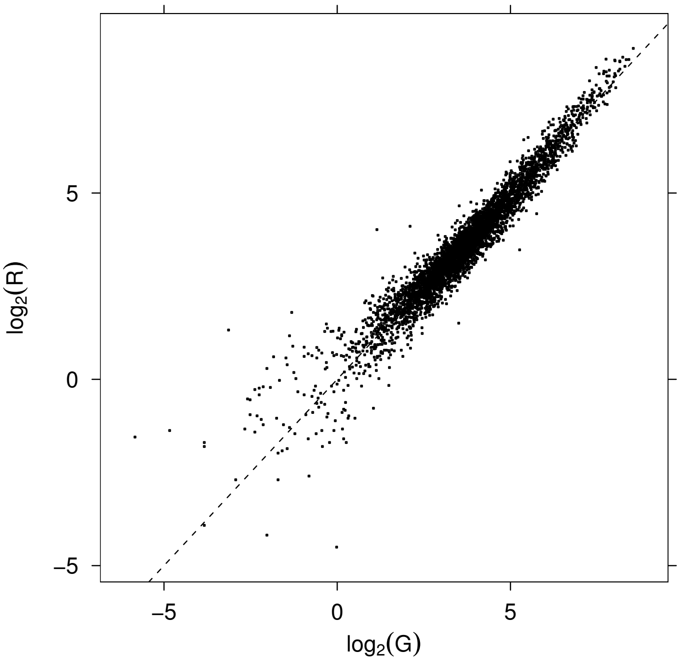
In experimental designs where there is replication of spotted mRNA on some of the microarrays, the amount of intensity information will differ depending on the gene of interest. Matrix notation is inappropriate under these designs as n_1 and n_2 are variables depending on the amount of spot replication. In these designs, the data for each gene i can be represented by $x_i = (x_{i1}, \dots, x_{in_1})$, and $y_i = (y_{i1}, \dots, y_{in_1})$.

2.4 Exploratory plots

Researchers are continually devising new types of diagnostic exploratory plots to graphically represent artefacts of interest within pre and post-normalized microarray experiments, such as scatter plots, boxplots, and image contour plots of single and multiple array slides [15]. Spot statistics of interest are generally derived from red and green foreground, and background log-intensities. Each type of plot has certain advantages in the patterns of the information that can visually illustrated. Examples of scatter plots in Figures 2.4, and 2.5, come from two control samples on an *E. coli* cDNA microarray, [16]. A scatter plot of R versus G on the log_2 transformed scale visually relates the relationship in the ratio T in equation 2.2 for each spot. Equivalently expressed genes are expected to lie on the 45° line after normalization of each slide.

Scatter plots commonly called “MA-plots” (as in Figure 2.5) are helpful in identifying spot artefacts and intensity dependent patterns in single microarray experiments [10]. These are plots on the log transformed scale where the intensities undergo a rotation

Figure 2.4: Scatterplot of $log_2(R)$ versus $log_2(G)$ intensities. The 45° dotted line depicts expected equivalent expression.



through the transformation equations,

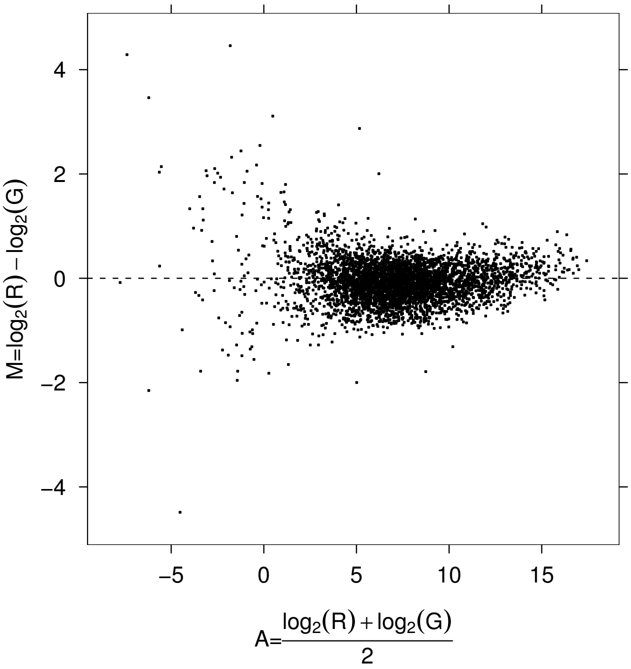
$$\begin{aligned} M &= log_2(R/G) \\ &= log_2(R) - log_2(G), \end{aligned} \tag{2.3}$$

and

$$\begin{aligned} A &= log_2(\sqrt{R * G}) \\ &= \frac{log_2(R) + log_2(G)}{2}. \end{aligned} \tag{2.4}$$

The notation M and A refers to the *minus*, and *add* operators involved. The quantity A acts as a measure of the average total abundance, while M acts as the difference in relative intensity, revealing intensity dependent information as the signal increases in both channels. Equivalently expressed genes are expected to be centered about 0 on the y axis. Points which lie above 0 on the y axis have higher expression in the red dye, points which lie below 0 on the y axis have higher expression in the green dye.

Figure 2.5: MA-plot of $M = log_2(R) - log_2(G)$ versus $A = \frac{1}{2}(log_2(R) + log_2(G))$ intensities. The dotted line at $M = 0$ depicts expected equivelent expression.



Nonlinear smoothers are used to fit a smoothed trend line to the data to identify trends across the whole microarray, or within print tip groups to identify individual variation in printing tips. The similar nonlinear smoothers *lowess* and *loess* are generally used. Due to its more simplistic approach, lowess is computationally faster.

2.5 Bioconductor

Bioconductor[1] [17] is an open source software development project which started in 2001, providing software tools for the analysis and comprehension of genomic data. The founding core development team is based in the Biostatistics Unit of the Dana Farber Cancer Institute in the Harvard Medical School/Harvard School of Public Health. The open source nature of the project under a General Public License as published by the Free Software Foundation, encourages developers anywhere in the world to contribute useful software tools to the evolving development for the benefit of world wide genomic research.

The underlying language of the bioconductor project is **R** [18], which uses packages of bundled software code to design and distribute tools for multiple operating systems within an environment allowing rapid development of extensible, scalable software. The

[1] http://www.bioconductor.org

bioconductor website is used to distribute software packages, documentation, and public scientific data. The distributions are available as two snapshots, the current stable release, and a developmental release incorporating the latest packages, with an associated risk of software instability. Static snapshots of scientific data are provided for analytical analysis since public data available over the internet changes quickly over time. For reproducibility of research results collaborators across the world need to be analyzing the same datasets independently to validate and compare analytical methods.

The documentation provided is in portable document format (PDF), and as pseudo-LaTeX source code text files under the Sweave[2] framework [19]. A graphical widget environment recognises code chunks within pseudo-LaTeX files allowing users to validate results in a computing environment while reading the pdf documentation. One of the goals of the bioconductor project is to encourage reproducible research through the Sweave [19] system, and to provide stable snapshots of example datasets for software validation. Bioconductor packages are written within an object oriented programming framework. An *object* structure is an access mechanism to locate and modify data, *class* structures define a blueprint of variables and methods accessible to an object, *methods* are programming routines which access, modify, and process content within the class structures.

There are four main parts involved in microarray dataset access and processing within bioconductor. The first part provides methods handling the loading of scanned intensity files from various microarray image analysis formats. These methods create an *expression set* object encapsulating a complete experiment, each containing the untransformed intensity responses and auxiliary experiment annotation information. Further information can be subsequently added to the expression set. The second and third parts provide normalization methods and exploratory plot methods to process pre and post-normalized expression sets. In the final part of analysis, bioconductor provides statistical tools to analyze expression sets filtering out genes of interest.

In cDNA microarray packages for bioconductor, either Spot [20] or GPR file formats derived from image scanners are loaded into the expression set which provides storage slots for relevant intensity and microarray annotation information. The red and green untransformed raw intensities are read in and stored into slots named *maRf*, and *maGf*. The data can be either in the form of vectors representing one microarray, or matrices representing multiple microarrays from the same experiment. Functions performing normalization methods create a similar object which is now a normalized class. The intensity information is stored in slots *maM* and *maA*, representing the transformations in equations 2.3 and 2.4. Multiple hypothesis comparison procedures require information stored in the normalized slots *maRf*, and *maGf* to be back transformed onto normalized R' and

[2] Sweave is a system to generate PDF documentation in a framework to allow the reader to recreate and modify the results reported within the document, http://www.ci.tuwien.ac.at/∼leisch/

G' log-intensity scales. Solving two simultaneous equations with two unknowns yields the inverse transformation,

$$\begin{aligned}
log_2(G') &= log_2(R') - M \qquad (A) \\
log_2(G') &= 2A - log_2(R') \qquad (B) \\
\text{Substituting (A) into (B)} & \\
log_2(R') - M &= 2A - log_2(R') \\
2log_2(R') &= 2A + M \\
log_2(R') &= A + \frac{M}{2} \qquad (2.5) \\
log_2(G') &= A + \frac{M}{2} - M \\
&= A - \frac{M}{2}. \qquad (2.6)
\end{aligned}$$

Using equations 2.5 and 2.6 it is straight forward to calculate the normalized values of R′ and G′ on the transformed scales. As of undertaking this work, functions did not exist in bioconductor to back transform *maM* and *maA* values into *maRf* and *maGf* intensities. Example functions within the bioconductor framework were written (Appendix A.1) to extract this information from the data structure objects depending on their class; marrayRaw, and marrayNorm.

2.5.1 Reading GenePix files

Bioconductor has a library called marrayInput [15] to load output files from GenePix software into R [18]. The GenePix array list format (GAL) is a standard spot description format identifying layout characteristics of oligo or cDNA spots in Blocks, Rows and Columns, alongside Names and Identifiers of printed substances. GAL files can be generated from third party sources for any microarray facility, and are used as input into the microarray robotics. If GAL files are derived from third party sources the nesting order of Blocks, Rows and Columns is not necessarily the same as the nesting order of the information from the GPR files from scanning. This can create a problem in bioconductor, as the code to upload the data merges description information (names and identifiers) contained within GAL files to intensity information contained within GPR files. In this scenario it is possible to scramble description information with respect to the intensity data. GPR files also contain Gene annotation information for Names and Identifiers of printed substances, but this is not read by default in bioconductor. Using this auxiliary information, modification was made to the provided **R** functions in the marrayInput library to upload the Gene annotation information directly from the Names and Identifiers columns within GPR files already synchronized to intensity information (Appendix

A.2). Since this work it was discovered that the bioconductor function *read.marrayRaw* in the marrayInput library can parse both GPR and GAL files. This allows synchronized description information to be added into the expression set via a two step process.

2.5.2 Speed issues

The amount of information stored within microarray experiments is continually increasing, that is, in the number of arrays within an experiment, and the number of spots on each array. This is challenging computer hardware required for data storage (disk space and memory) and analysis (processing speed). The sheer volume of information in raw intensity format files must be cut down to remove redundancy and promote scalability as datasets increase in size. In GPR files, there are many columns of auxiliary spot statistic information which are not necessary for downstream microarray analysis, and can be discarded. Only about 7 columns of data are required for downstream analysis, as opposed to the 48 columns[3] contained in the data field section of each GPR file.

Functions within the bioconductor package *marrayInput* were found to be quite inefficient in the speed of loading multiple GPR files into a single expression set. They use the *scan* function to read each row of intensity data from each array line by line. In the process, *scan* can be optimized by discarding columns which the expression set doesn't need to read into memory, significantly improving upload speed. The **R** code in the function read.marrayRaw was modified to improve uploading speed of Spot and GPR text files. The approach proposed utilizes the "what" argument within the scan function. Any field that is not required is given a list entry of "NULL", *scan* then skips the field until it finds a field it requires (See code Appendix A.3).

A test was conducted on the four *swirl*[4] data set Spot files provided in the *Vignette*[5] documentation example. On a 1.8Ghz machine running R version 1.8.1 on linux[6], the four files were uploaded 10 times comparing the original *read.marrayRaw* code to the modified code. The original code took $\sim$ 36 seconds to upload the spot files, the modified code took $\sim$ 12 seconds to upload. In this example a 300% speed increase was obtained, so the improvement will certainly be significant when loading hundreds of microarrays into a single expression set. The optimization methodology used was submitted to the package co-author and maintainer (Y. Yang) for incorporation into the package *marrayInput* in future releases of bioconductor.

[3] GPR Version 3.0, results acquired using Genepix Pro 4.1.1.4 software

[4] Dataset provided by Katrin Wuennenberg Stapleton from the Ngai Lab at UC Berkeley.

[5] Sweave [19] pseudo-LaTeX PDF documentation

[6] Redhat 8.0 operating system

3
False Discovery Rate Control

The genomics era of the last decade has produced an explosion of biological sequence information. Advances in microarray technologies are likely to see increasing numbers of transcripts for genes of interest (under hybridization) packed onto a single microarray slide in a dense ordered array of spots. Due to the nature of this data, an important question of interest is "which genes on a chip are undergoing differential expression between target mRNA samples?". This can be addressed as a problem in multiple hypothesis testing, a simultaneous test of the null hypothesis that there is no association between the expression levels within each gene and the target mRNA responses of interest. In biological systems it is likely that the number of genes which change will be small, likewise the proportion of genes whose expression levels are unaffected will be large. This problem has lead to the application of *false discovery rate*(FDR) controlling procedures [21] in microarray analysis [22] as a suitable method of controlling the amount of error when determining genes undergoing significant changes. This chapter examines the FDR controlling procedure of Benjamini and Hochberg, which controls the expected proportion of Type I errors in a list of rejected hypotheses. The level of control in this procedure is an expectation, the experimental variance is unknown for any realization of the procedure. To assess the variability of this procedure, the general operating characteristics of FDR control are examined.

3.1 Error rate control

Consider the problem of simultaneously testing m null hypotheses H_j, $j = 1, \ldots, m$. Let R be the number of rejected null hypotheses. The testing situation is summarized in Table 3.1, using the notation of Benjamini and Hochberg [21]. The m specific hypotheses of interest are assumed to be known in advance, but the numbers of true null hypotheses m_0, and alternative hypotheses, m_1, are unknown. R is an observable random variable, while S, T, U, and V are all unobservable random variables. V is the number of Type I errors, (hypotheses declared significant when they are actually from the null distribution), and T is the number of Type II errors, (hypotheses declared not significant when they are actually from the alternative distribution).

Table 3.1: Classification of m hypothesis tests (Benjamini and Hochberg [21]).

	# declared not significant	# declared significant	total
# true null hypotheses	U	V	m_o
# non-true null hypotheses	T	S	m_1
	$m - R$	R	m

Standard notation in statistical literature defines α as the probability of committing a Type I error, and β as the probability of committing a Type II error. The power in statistical hypothesis testing is defined as the probability, $1 - \beta$, of correctly identifying non true null hypotheses. As the number of simultaneous hypothesis tests increases, the α significance threshold must be modified to account for the increasing number of expected rejections due to chance, so as to maintain a specified level of error rate control. Multiple comparison procedures (MCPs) are used to determine α based on various criteria. The observed probabilities are conditional on which hypotheses are actually true. Control of the Type I error rate under *any* combination of true and false hypotheses is referred to as strong control. Weak control refers to control of the Type I error rate when *all* null hypotheses are true.

3.1.1 Family wise error rate

The family wise error rate (FWER) as described in [23] is defined as the probability of making at least one Type I error,

$$\begin{aligned} \text{FWER} &= P(V \geq 1) \\ &= 1 - P(V = 0). \end{aligned}$$

In multiple hypothesis testing under FWER control, as the number of hypothesis tests, m, increases, the p-value rejection threshold, α, decreases toward 0, thus providing a high level of certainty in the rejected null hypotheses at the expense of the rejection threshold being overly conservative. In the context of microarray experiments, FWER control against a single false positive is typically too strict, which leads to many missed detections [24]. Multiple comparison procedures which identify as many significant genes as possible while minimizing the proportion of false positives are likely to be far more powerful.

3.1.2 False discovery rate

The false discovery rate (FDR) proposed in [21] is the expected proportion of incorrectly rejected Type I errors in the list of rejected hypotheses. It is a less conservative multiple comparison procedure with greater power than FWER control, at a cost of increasing the likelihood of obtaining Type I errors. Using the notation of Benjamini and Hochberg [21],

$$\text{FDR} \;=\; E\left(\frac{V}{R} \mid R > 0\right) P(R > 0). \tag{3.1}$$

The procedure proposed by Benjamini and Hochberg [21] for controlling the FDR involves a stepwise adjustment of ordered p-value statistics obtained from the hypothesis tests. If $P_1, P_2, \ldots, P_m$ are the ordered p-values for the hypothesis tests $H_1, H_2, \ldots, H_m$, the stepwise procedure controlling the FDR below α^* is given by,

$$\begin{aligned}&\text{Reject all } H_i : i = 1, 2, \ldots, k,\\ &\text{where } k \text{ is the largest } i \text{ for which } P_i \leq \frac{i}{m}\alpha^*.\end{aligned} \tag{3.2}$$

This is a *step-up* procedure controlling the FDR on the sorted p-values from smallest to largest, a similar *step-down* procedure achieves almost identical results [25] by controlling the FDR on the sorted p-values from largest to smallest. In a comparison of Type I error rates, it can be shown that under for any combination of true null hypotheses, FDR $\geq$ FWER. The FDR controlling procedure also provides weak control of the FWER at level α^*.

Benjamini and Hochberg [21] showed that their stepwise adjustment procedure provided the following level of FDR control,

$$\text{FDR} \;\leq\; \frac{m_0}{m}\alpha^* \;\leq\; \pi_0\alpha^*, \tag{3.3}$$

where π_0 is the proportion of true null hypotheses in m hypotheses of interest. In situations when π_0 is small, the expected FDR will be well below α^*. Depending on the power of the hypothesis tests, as π_0 approaches 1 the number of rejected hypotheses, R, will decrease since m_0 is large, but the probability of Type I errors in the rejected hypotheses will increase. Analogous to the FDR, a measure of statistical power in multiple hypothesis testing can be defined as the expected proportion of correct discoveries, the true discovery rate (TDR) [26] out of R rejected hypotheses,

$$\begin{aligned}\text{TDR} &= E\left(\frac{S}{R} \mid R > 0\right) P(R > 0) \\ &= \left[1 - E\left(\frac{V}{R} \mid R > 0\right)\right] P(R > 0) \\ &= P(R > 0) - \text{FDR}. \end{aligned} \qquad (3.4)$$

In 2001 Finner and Roters [27] proved equality of equation 3.3,

$$\text{FDR} = \frac{m_0}{m}\alpha^* = \pi_0\alpha^*. \qquad (3.5)$$

Perfect knowledge of the proportion π_0 of true null hypotheses (i.e. how many m_0, out of m hypothesis tests were true), allows constant FDR control over a range of $\pi_0 \in (\alpha^*, 1)$ so that FDR $= \alpha^*/\pi_0$. This is known as *adaptive control* of the FDR [28, 29]. In practice, π_0 is generally unknown during hypothesis testing, therefore adaptive control relies on good estimation of π_0.

FDR control at the level α^* provides a list of rejected hypotheses where the expected proportion of false discoveries is controlled at $\pi_0\alpha^*$ under non-adaptive control, and at the constant threshold level α^* under adaptive control. The false non-discovery rate (FNDR) [30] investigates the expected proportion of false non-discoveries (Type II errors) obtained in the list of rejected hypotheses under the FDR procedure in 3.2. The FNDR equation is similar to the FDR equation in 3.1 where,

$$\text{FNDR} = E\left(\frac{T}{(m-R)} \mid (m-R) > 0\right) P((m-R) > 0). \qquad (3.6)$$

This is the expected proportion of missed findings when testing m null hypotheses.

3.1.3 Simulation sufficient statistics

Hypothesis testing simulation studies have been used to generate observations under multiple hypothesis testing conditions. FDR controlling multiple comparison procedures are then applied to the observations to characterise operating characteristics of FDR control, the observed proportion of false discoveries (OPFD). [31]. The approach uses parametric distributions to generate observations under the null and the alternative hypotheses, statistical methods are then applied to the generated observations. With knowledge about the underlying generating distributions, the unobservable random variables under hypothesis testing in Table 3.1 become observable random variables under simulation. The sufficient statistics required to generate all the information within Table 3.1 are the observed counts v and t for the number of Type I and Type II errors, along with m, the number of hypothesis tests, and either π_0 or m_0 where,

$$\begin{aligned} U &= m_0 - V \\ &= \pi_0 m - V, \\ S &= (m - m_0) - T \\ &= (1 - \pi_0)m - T. \end{aligned} \tag{3.7}$$

In multiple comparison procedures with acceptable power, usually $V < U$ and $T < S$, so it is most computationally efficient to store the observed counts v and t (**R** code in Appendix A.5).

3.1.4 Observed proportion of false discoveries

From equation 3.1, the OPFD is defined as,

$$\text{OPFD} = \begin{cases} \frac{V}{R} & R > 0 \\ 0 & R = 0. \end{cases} \tag{3.8}$$

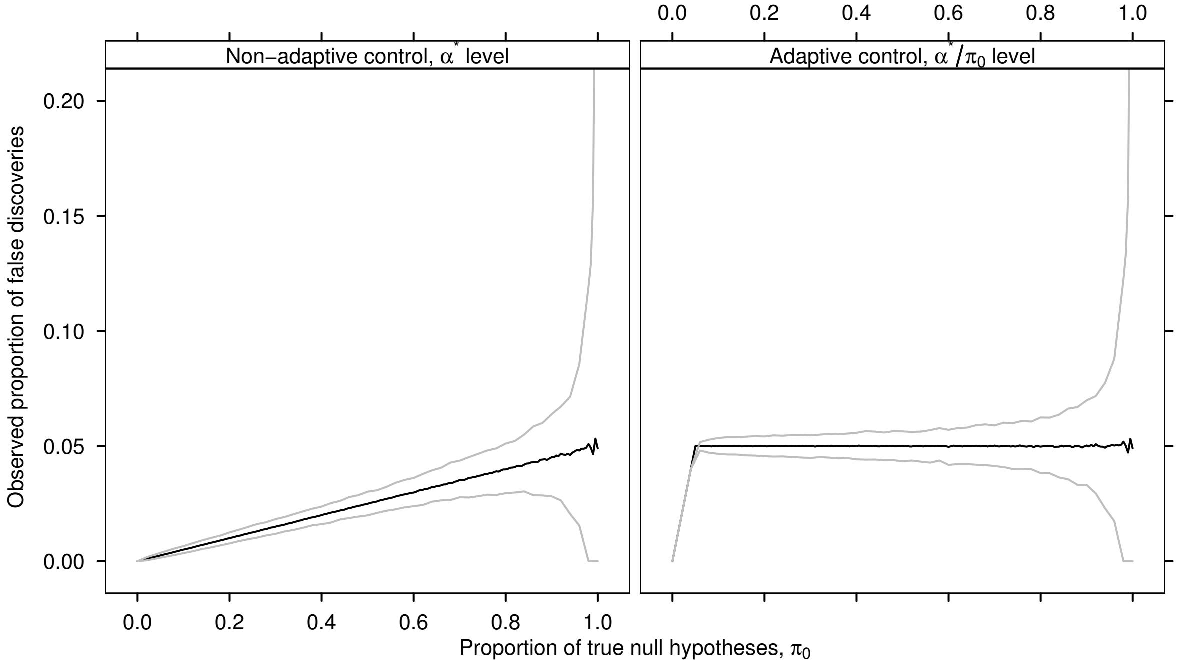
Figure 3.1 illustrates the general characteristics of the OPFD distribution under non-adaptive and adaptive control as a function of π_0 using the step-up procedure at the $\alpha^* = 0.05$ level. The distributions were estimated by 1000 simulations of $m = 10,000$ independent observations, at each value of $\pi_0 = (0, 0.005, \ldots, 1)$, generated from a mixture of two normal distributions as follows,

$$\begin{aligned} x_i &\sim N(\mu_i, 1), \quad i = 1, \ldots, m, \\ \mu_i &= \begin{cases} 0 & \text{if } x_i \text{ from } H_0 \\ \mu & \text{if } x_i \text{ from } H_A. \end{cases} \end{aligned} \tag{3.9}$$

The hypothesis H_0: $\mu = 0$ was tested against $\mu \neq 0$ for each of the m observations at the $\alpha^* = 0.05$ level.

Under non-adaptive control the E[OPFD] is centered about $\pi_0\alpha^*$, which agrees with the result of Finner and Roters [27], while under adaptive control (assuming perfect knowledge of π_0) the E[OPFD] is centered about α^* over the range of $\pi_0 \in (\alpha^*, 1)$. Under both methods of FDR control variability increases in the OPFD distribution as $\pi_0 \to 1$. An interesting artefact in the E[OPFD] under non-adaptive control is that $\pi_0 = \alpha^*$ when

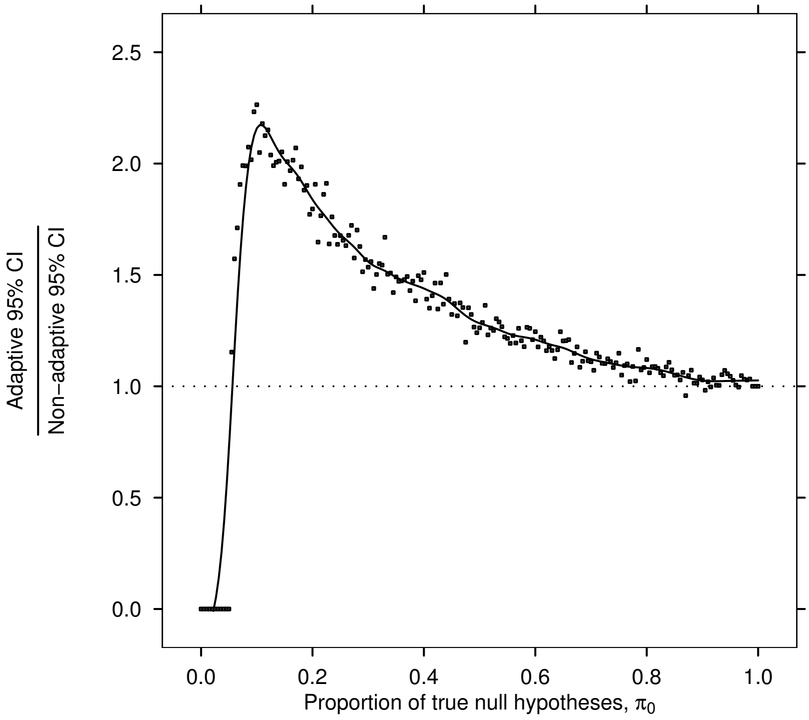
Figure 3.1: The OPFD distribution as a function of π_0 under non-adaptive and adaptive control at the $\alpha^* = 0.05$ level, simulated from a mixture of normal distributions over a range of π_0 where $\mu = 3$ under H_A in 3.9. Black lines are the E[OPFD], grey lines are 2.5 and 97.5 percentiles of the OPFD distribution.



$\pi_0 \leq \alpha^*$ with certainty over a line with gradient equal to 1. This is because the threshold α^* in the FDR step up procedure 3.2, rejects all null hypotheses when $\pi_0 \leq \alpha^*$.

The 2.5 and 97.5 percentiles of the OPFD distribution were calculated to illustrate observed confidence intervals at the 95% level, they are significantly larger under adaptive FDR control than under non-adaptive control. The E[OPFD] becomes less stable as $\pi_0 \to 1$. As the 95% confidence interval increases in size, the proportion of incorrectly rejected hypotheses either has a high probability of either being large, or is equal to 0 so as to maintain FDR control at α. Figure 3.2 describes the observed ratio of the adaptive 95% confidence interval divided by the non-adaptive 95% confidence interval. The smooth spline fit to the simulated ratios infers that under adaptive control, when $\pi_0 = 0.1$ the 95% confidence interval is approximately 2.25 times larger than under non-adaptive control. The observed ratio decreases continuously displaying smooth curvature, as π_0 increases. When $\pi_0 = 1$, the ratio equals 1 (since $\hat{\pi}_0 = 1$).

Figure 3.2: Ratio of the OPFD distribution adaptive 95% confidence interval width divided by the OPFD distribution non-adaptive 95% confidence interval width as a function of π_0. Observed confidence intervals are derived from Figure 3.1, simulated from a mixture of normal distributions over a range of π_0 where $\mu = 3$ under H_A in 3.9. The smooth curve is a natural spline fit.



3.1.5 Characteristics of non-adaptive control of the FDR

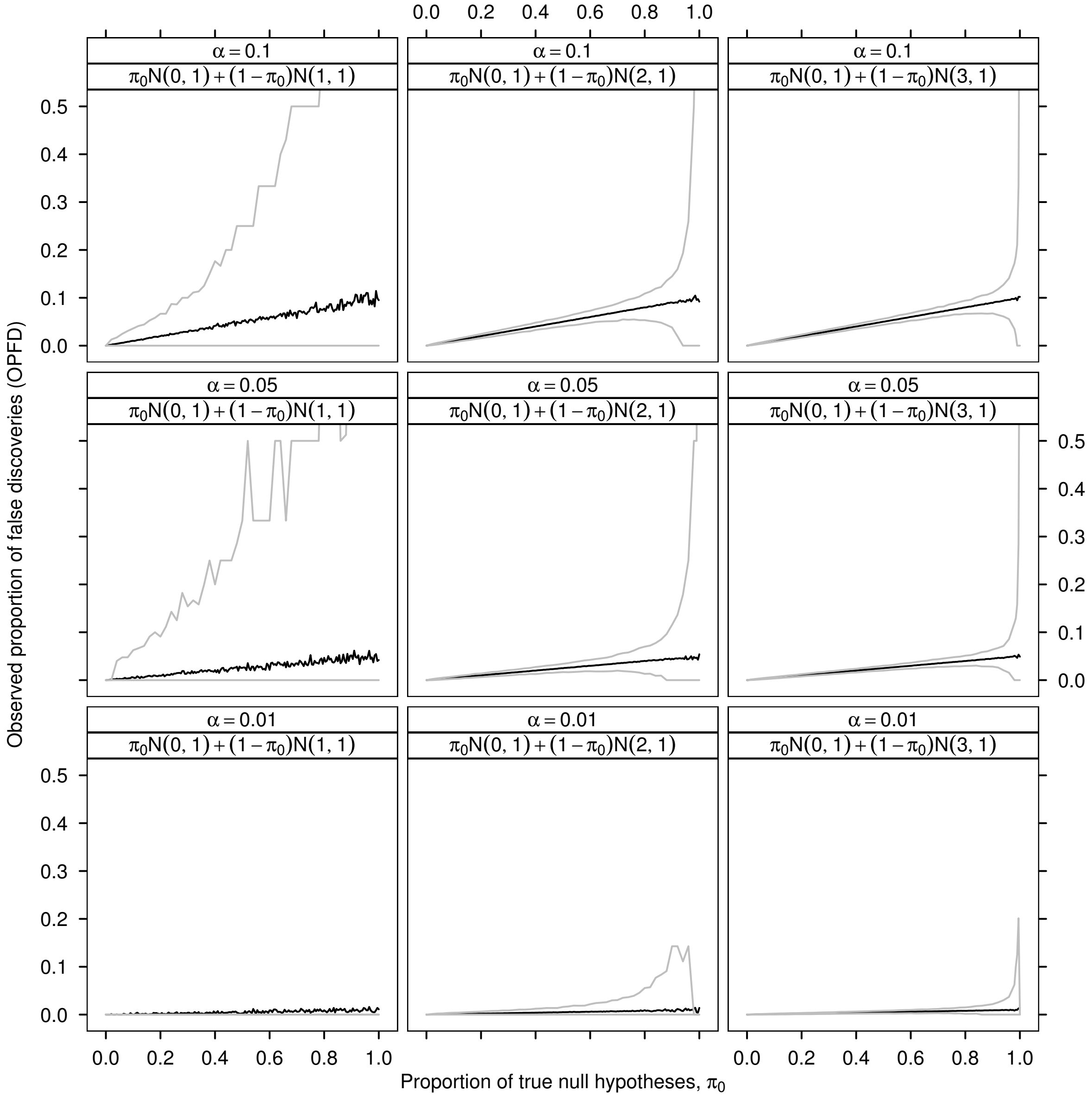
The simulation in subsection 3.1.4 was extended to investigate the general characteristics of the OPFD distribution for three different distances between the null and alternative distributions, $\mu \in (1, 2, 3)$, and different levels of FDR control, $\alpha^* \in (0.01, 0.05, 0.1)$. Results are presented in Figures 3.3 to 3.8, in a 3×3 matrix of plot panels of the OPFD distribution, and other useful measures such as the observed proportion of true discoveries (OPTD) distribution. From equation 3.4, the OPTD is defined as,

$$\text{OPTD} \;=\; \begin{cases} \frac{S}{R} & R > 0 \\ 0 & R = 0. \end{cases} \tag{3.10}$$

In each row of plots α^* remains constant, and in each column of plots the distance μ_i, between the null and alternative distributions remains constant.

Figure 3.3 illustrates how the OPFD changes as a function of π_0, conditioned over different levels of μ and α^*. As μ increases, there is a decrease in the variability of the OPFD due to the improvement in hypothesis testing sensitivity. As the α^* threshold level

Figure 3.3: OPFD distribution for the non-adaptive step-up FDR controlling procedure [21], at $\alpha^* \in (0.01, 0.05, 0.1)$ levels, simulated from a mixture of normal distributions over a range of π_0 where $\mu \in (1, 2, 3)$ under H_A in equation 3.9. Black lines are the E[OPFD], grey lines are 2.5 and 97.5 percentiles of the OPFD distribution.



increases, variability increases in the OPFD. In the first column of plots when $\mu = 1$, the FDR procedure has limited sensitivity. In plot panels where the $\alpha^* = 0.05$, and $\alpha^* = 0.1$, the 95% confidence intervals in the OPFD become highly variable as π_0 increases. The poor sensitivity causes the OPFD to become highly polarized, with the OPFD likely to be either 0, or large as $\pi_0 \to 1$. The observed distribution v, is also highly variable in this situation (data not shown). When $\alpha^* = 0.01$, even though the OPFD is as expected along the line $E[\text{OPFD}] = \text{FDR} = \alpha^*\pi_0$, greater than 95% of the OPFD observations in the confidence interval equal 0 across the whole range of π_0. This illustrates that there is a tradeoff between the choice of α, and the sensitivity in the hypothesis testing.

The OPTD is illustrated in Figure 3.4. In the first column of plots where $\mu = 1$, the OPTD is very poor, and gets progressively worse as α^* decreases. When $\mu = 2$, the sensitivity in hypothesis testing increases dramatically (the E[OPTD] is almost 1 over most of the range of π_0), but drops off quickly as π_0 approaches 1. The factor governing the OPTD in Figure 3.4 is the $P(R > 0)$ term of 3.4, which is much larger than the FDR term.

Figure 3.5 illustrates the situation when the $P(\text{OPFD} = 1)$. The OPFD can only equal 1 if the multiple comparison procedure makes at least one rejection ($R > 0$). Any rejections that are made are all Type I errors. The $P(\text{OPFD} = 1)$ increases as $\pi_0 \to 1$, and the number of rejected truly null hypotheses decreases.

In the first column of plots when $\mu = 1$, the $P(\text{OPFD} = 1)$ is significantly higher, and more variable for moderate to large values of π_0. This is due to lower sensitivity in the multiple comparison procedure. When $\mu = 2$ or greater, the $P(\text{OPFD} = 1) = 0$, over most of the range of π_0, increasing as $\pi_0 \to 1$. As α^* is increased, a higher proportion of Type I errors in the rejected hypotheses is observed, and the $P(\text{OPFD} = 1)$ increases.

3.1.6 Characteristics of adaptive control of the FDR

Figure 3.6 shows the effect of adaptive control of the OPFD, where π_0 is assumed to be known. Under adaptive control, the OPFD has similar features as under non-adaptive FDR control, except that FDR control is centered about α^* for $\pi_0 \in (\alpha^*, 1)$. In the bottom row of plots when $\alpha^* = 0.01$, the distribution of the upper 97.5 percentile of the OPFD distribution shifts to the right in the plot panels (increasing in π_0) as μ increases. The 95% confidence interval is unstable because the number of rejections, R, is moderate in size, with around 2.5% of simulations containing false discoveries for fixed π_0. As $\pi_0 \to 1$ there is a point at which $P(R = 0) \approx 1$, causing the upper 95% confidence interval to drop sharply. In Figure 3.7, as π_0 increases, the OPTD is initially controlled at $1 - \alpha^*$. In the first column of plots when $\mu = 1$, the E[OPTD] decreases rapidly as π_0 increases. There is a point along the range of π_0 where the constant E[OPTD] control drops below $1 - \alpha^*$. As the level of FDR control α^* is increased, the location of the threshold point on π_0 increases. The OPTD distribution is extremely variable for π_0 larger than this threshold.

The $P(\text{OPFD} = 1)$ under adaptive FDR control (Figure 3.8) displays almost identical pattern to Figure 3.5 using non-adaptive control. This implies that the average proportion of all incorrect hypotheses is similar using either form of FDR control.

Figure 3.4: OPTD distribution for the non-adaptive step-up FDR controlling procedure [21], at $\alpha^* \in (0.01, 0.05, 0.1)$ levels, simulated from a mixture of normal distributions over a range of π_0 where $\mu \in (1, 2, 3)$ under H_A in equation 3.9. Black lines are the $E[\text{OPTD}]$, grey lines are 2.5 and 97.5 percentiles of the OPTD distribution.

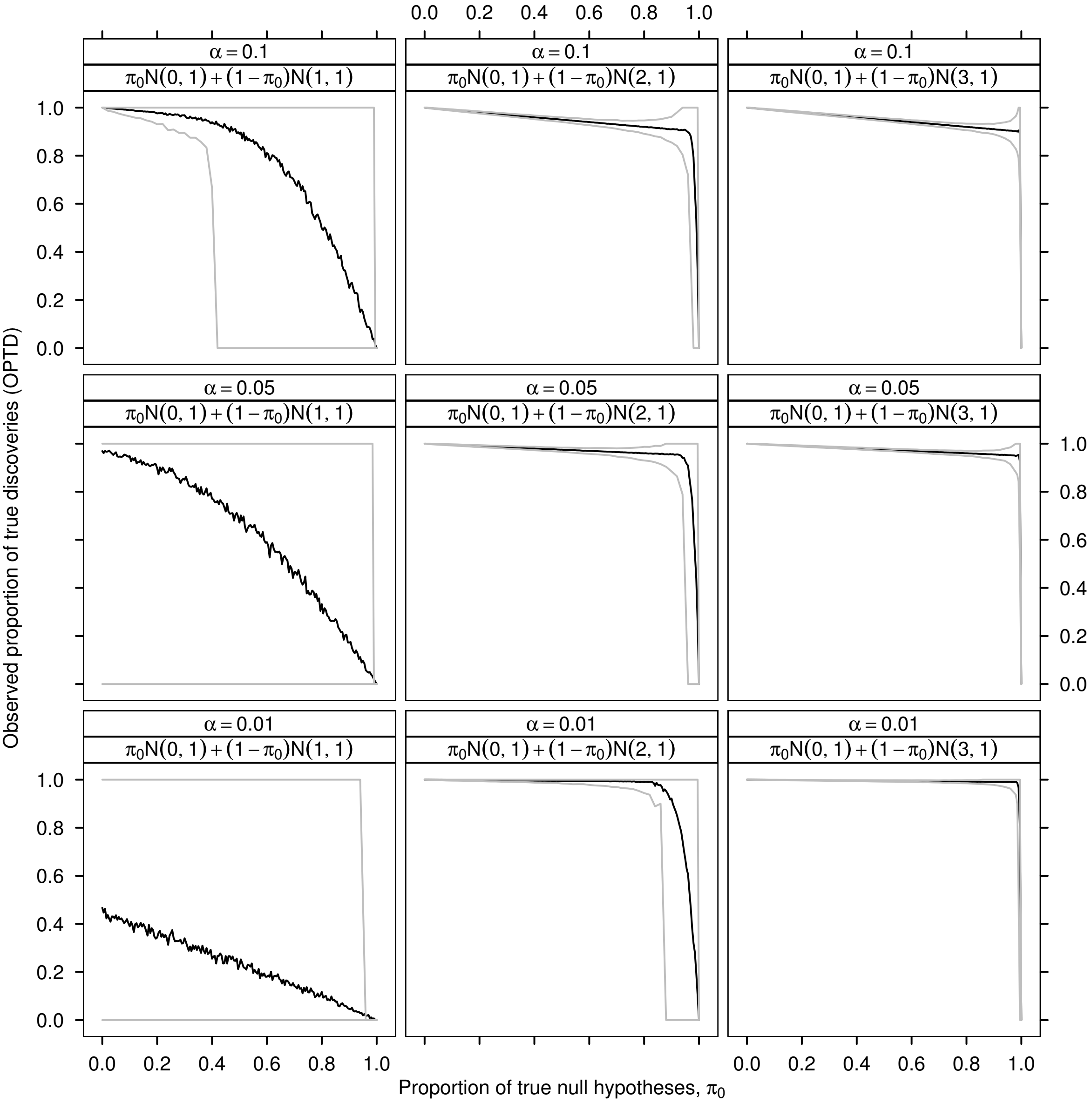


Figure 3.5: Probability that the OPFD equals 1 under the non-adaptive step-up FDR controlling procedure [21], at $\alpha^* \in (0.01, 0.05, 0.1)$ levels, simulated from a mixture of normal distributions over a range of π_0 where $\mu \in (1, 2, 3)$ under H_A in equation 3.9. Black lines are the $P(OPFD = 1)$.

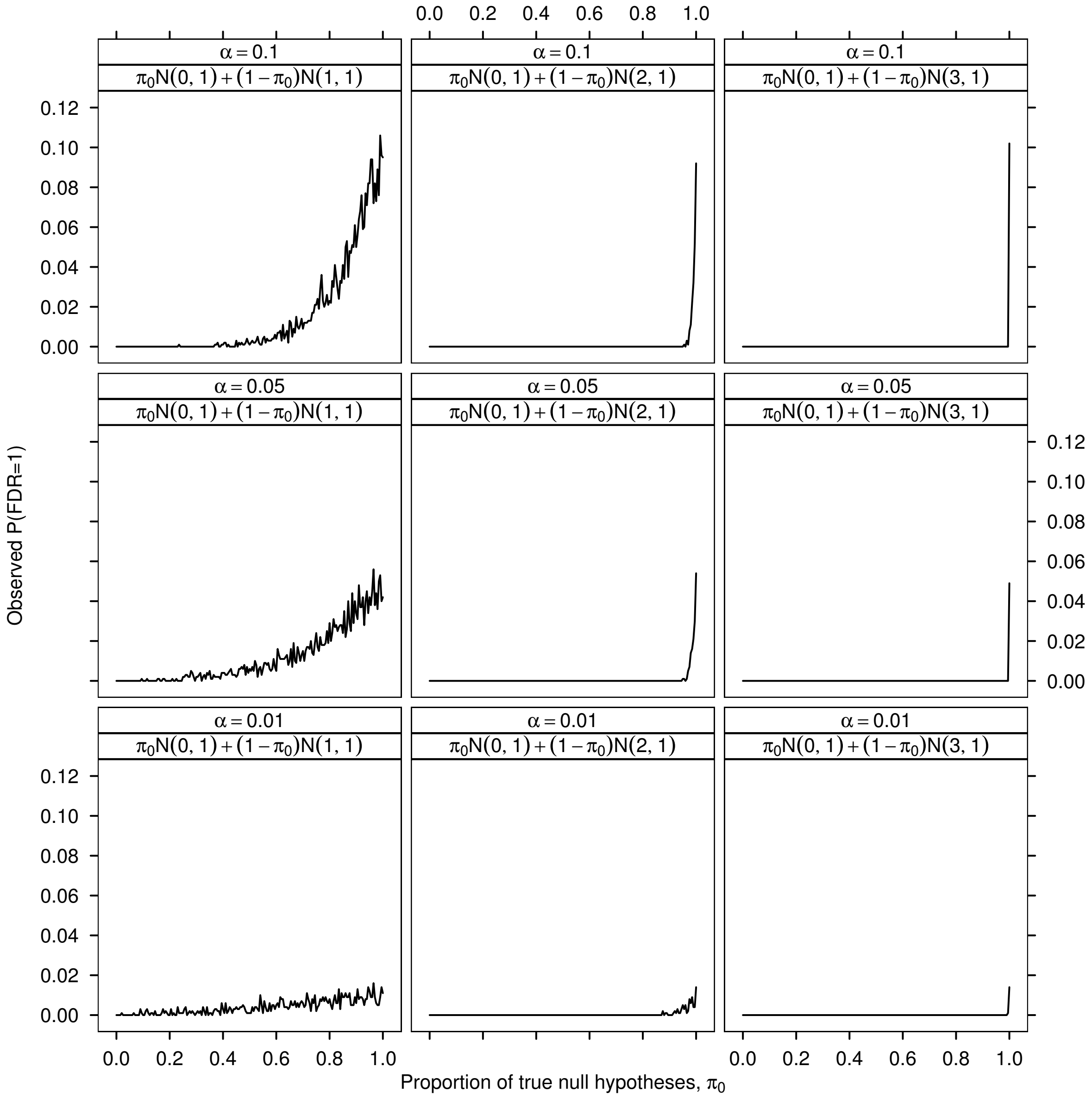


Figure 3.6: OPFD distribution for the adaptive step-up FDR controlling procedure [28], at $\alpha^* \in (0.01, 0.05, 0.1)$ levels, simulated from a mixture of normal distributions over a range of π_0 where $\mu \in (1, 2, 3)$ under H_A in equation 3.9. Black lines are the $E[\text{OPFD}]$, grey lines are 2.5 and 97.5 percentiles of the OPFD distribution.

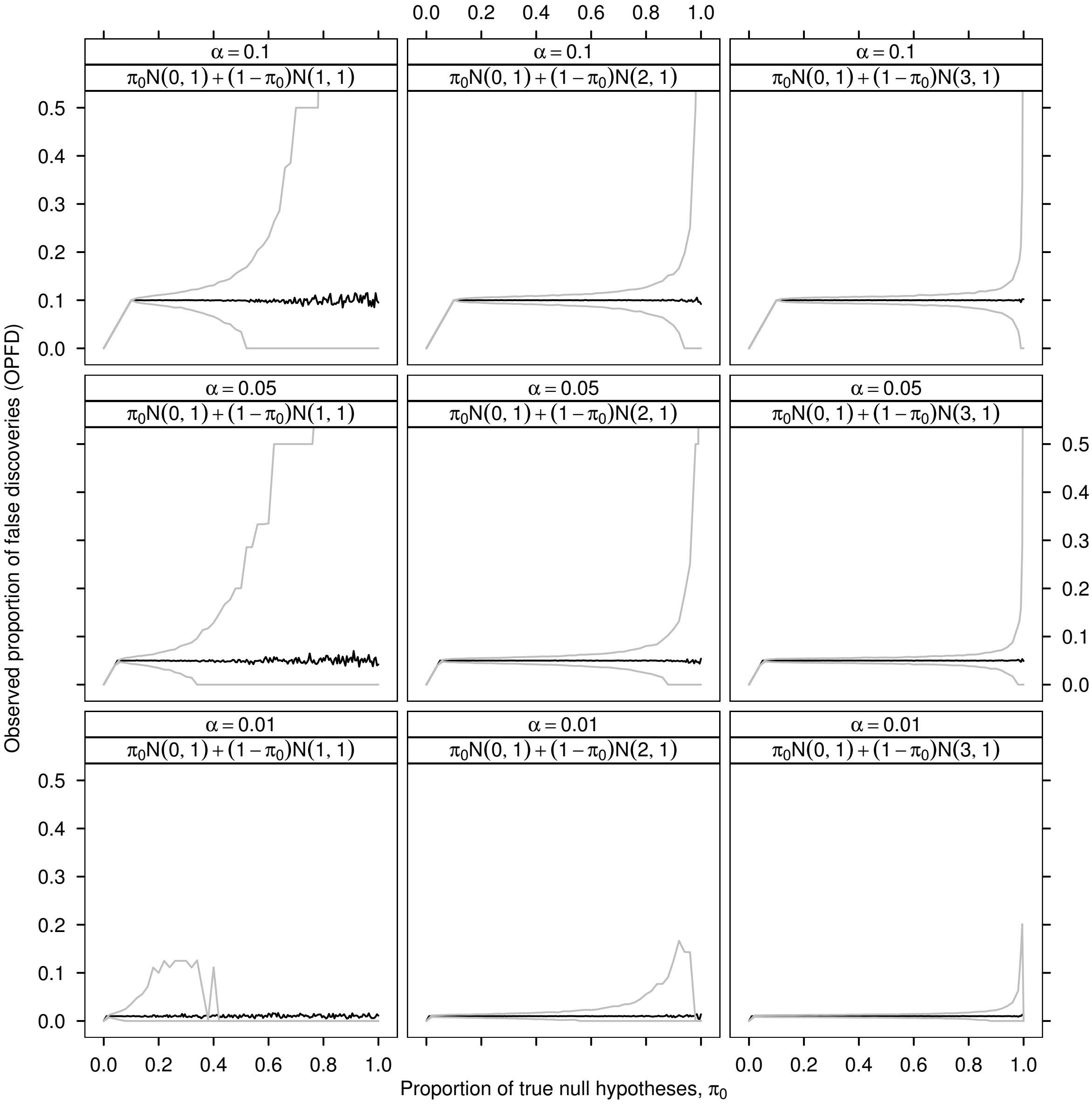


Figure 3.7: OPTD distribution for the adaptive step-up FDR controlling procedure [28], at $\alpha^* \in (0.01, 0.05, 0.1)$ levels, simulated from a mixture of normal distributions over a range of π_0 where $\mu \in (1, 2, 3)$ under H_A in equation 3.9. Black lines are the $E[\text{OPTD}]$, grey lines are 2.5 and 97.5 percentiles of the OPTD distribution.

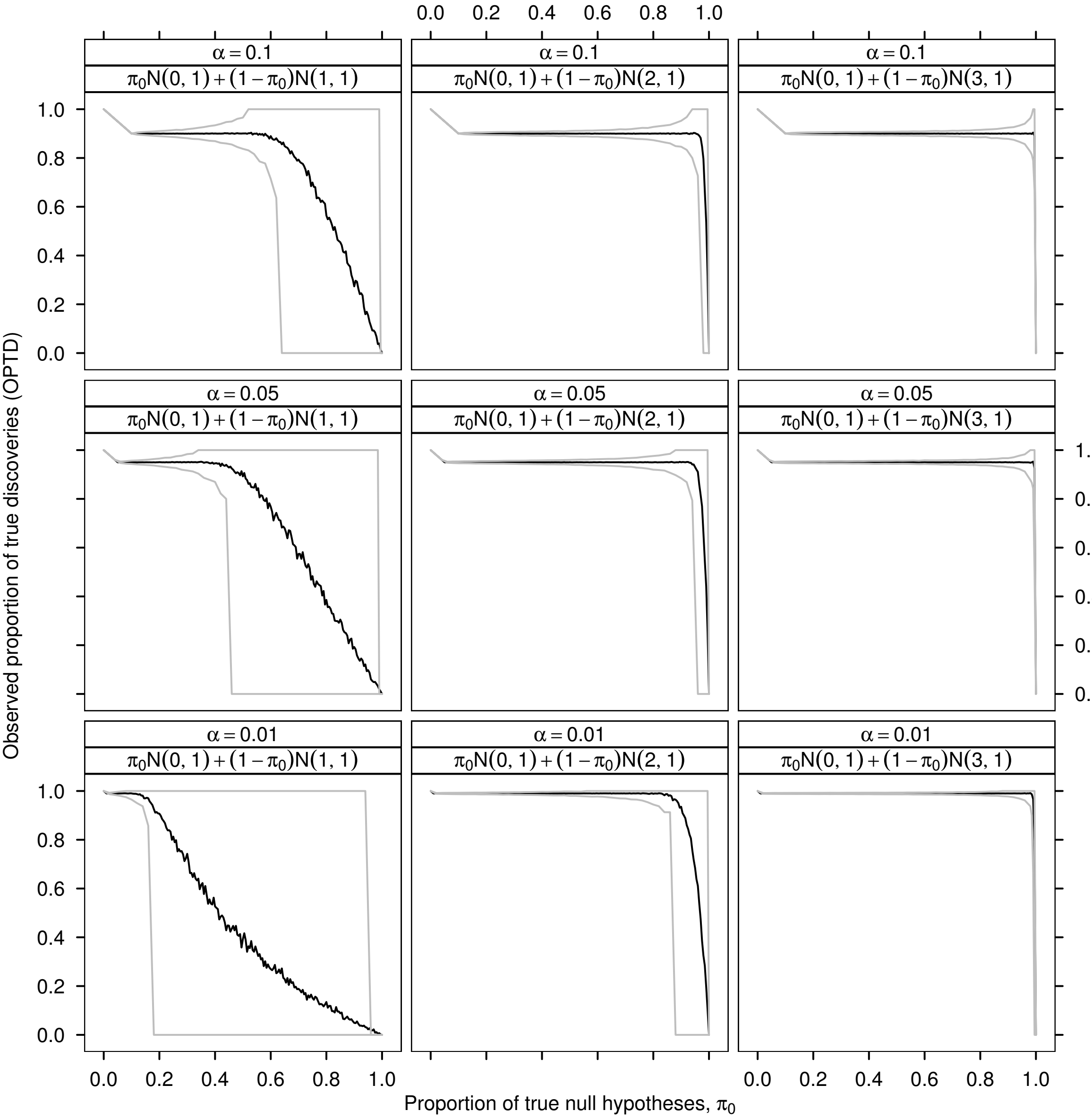
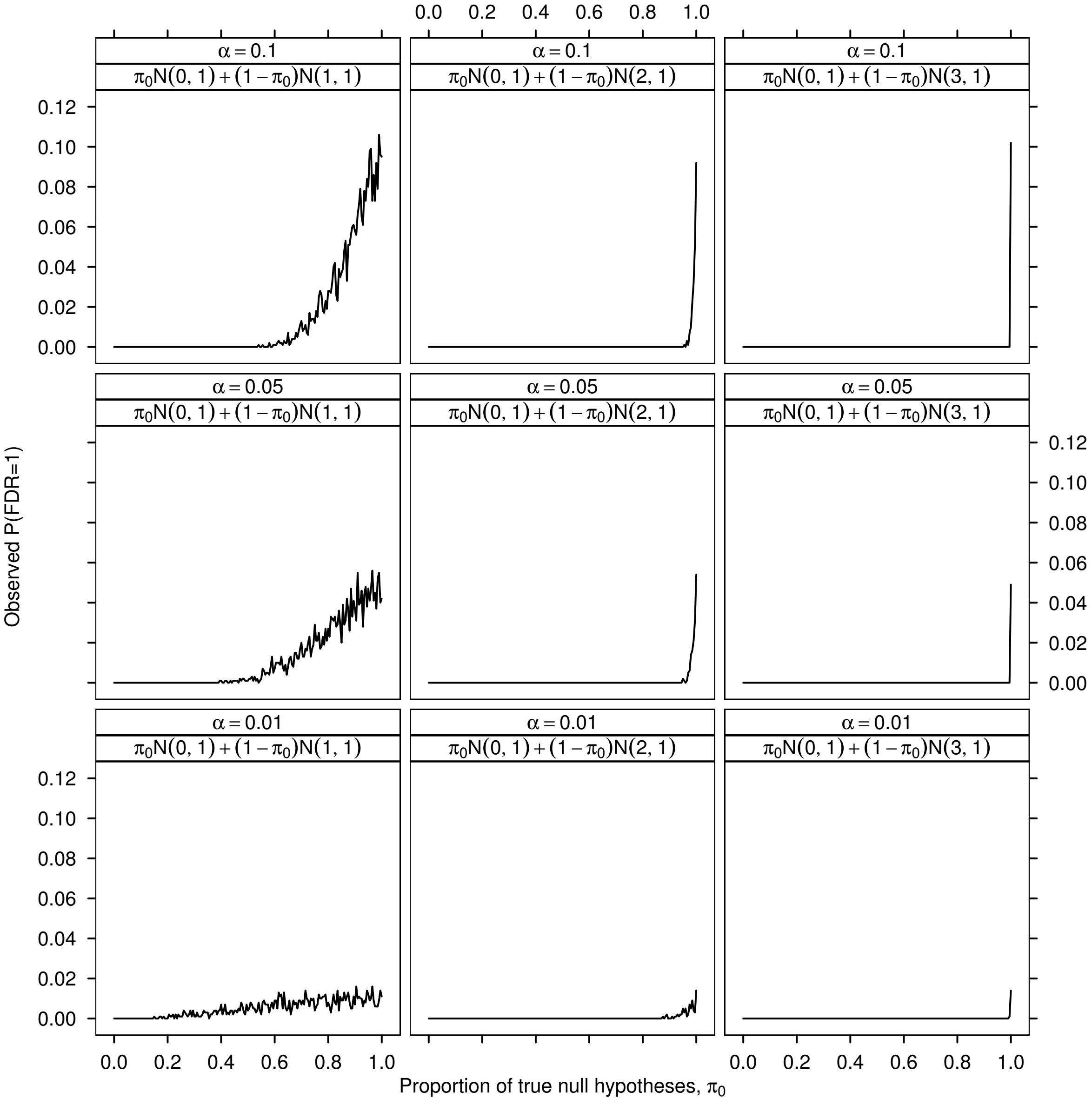


Figure 3.8: Probability the OPFD equals 1 under the adaptive step-up FDR controlling procedure [28], at $\alpha^* \in (0.01, 0.05, 0.1)$ levels, simulated from a mixture of normal distributions over a range of π_0 where $\mu \in (1, 2, 3)$ under H_A in equation 3.9. Black lines are the $P(OPFD = 1)$.



3.2 Estimation of π_0

In hypothesis testing, m_0, the number of true null hypotheses out of m hypotheses tested is unknown, therefore the proportion of true null hypotheses π_0 must be estimated.

3.2.1 Estimating π_0 using natural splines

The distribution of p-values obtained from m hypothesis tests is an unknown mixture of $m_0 = m\pi_0$ p-values from the null distribution, and $m_1 = m(1 - \pi_0)$ p-values from the alternative distribution. A method for estimating $\hat{\pi}_0$ directly from the mixture distribution of p-values was proposed by Storey [32]. It involves exploiting the fact that under the null distribution p-values are uniformly distributed, without having to quantify the distribution of alternatively distributed p-values.

In this method an estimate of $\hat{\pi}_0$ as a function of λ (a tuning parameter estimating a threshold of significance), is obtained from the distribution of observed p-values $p_1, \ldots, p_m$,

$$\hat{\pi}_0(\lambda) \quad = \quad \frac{\#\left\{P_i > \lambda\right\}}{m(1-\lambda)}. \tag{3.11}$$

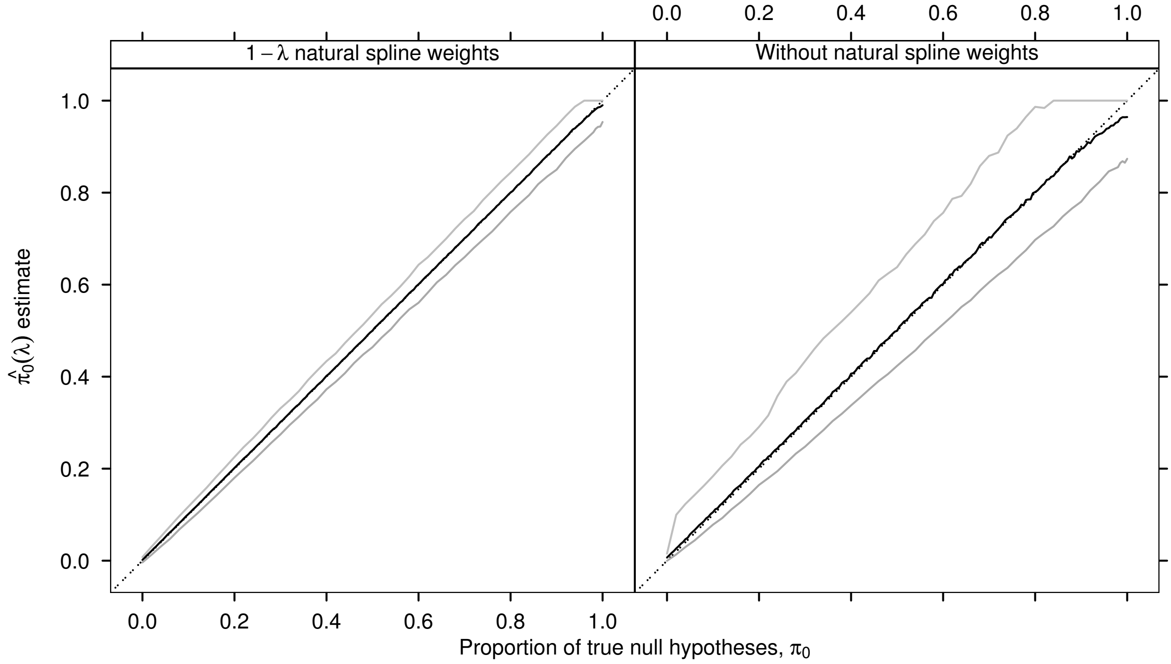
If all the p-values are from H_0, then $P_i \sim U(0,1)$, which implies $E[\#\left\{P_i > \lambda\right\}] = m(1-\lambda)$, and $E[\hat{\pi}_0(\lambda)] = 1$. Under any mixture of p-values, as $\lambda \to 1$ the estimate of $\hat{\pi}_0(\lambda)$ becomes unbiased, with a trade-off of increased variance. Storey [32] uses a bootstrap approach to select λ for the estimation of $\hat{\pi}_0(\lambda)$. This method, however has been shown to possess unfavourable estimation properties [33]. Estimation of π_0 in 3.11 can also be made subjectively by eye directly from a histogram of the mixture distribution of p-values, or in a fully automated way that utilizes support for the estimate at $\hat{\pi}_0(\lambda = 1)$ over a range of λ [34]. Consider a plot of $\hat{\pi}_0(\lambda)$ versus λ over a range of $\lambda = 1, 0.01, 0.02, \ldots, (m-1)/m$. The method of Storey & Tibshirani [34] estimates $\lim_{\lambda \to 1}\hat{\pi}_0(\lambda) = \hat{\pi}_0(1)$ by fitting a natural spline to the trend over the range $\lambda \in (0,1)$ allowing estimation of $\hat{\pi}_0(\lambda = 1)$. The degrees of freedom are set to 3 in the natural spline to limit curvature in the fit to a quadratic function. Note that $\lambda = 1$ is undefined in 3.11, so $(m-1)/m$ is the closest estimable value of the tuning parameter.

In a preprint of Storey and Tibshirani (2003) [35], a weight of $(1 - \lambda)$ was applied to each observation reducing the variance of the estimate of $\hat{\pi}_0(1)$ since $\hat{\pi}_0(\lambda)$ becomes more accurate as $\lambda \to 0$. However, this weighting procedure was not used in the final version of the paper. In Bioconductor a package called *qvalue*[1] provides programming routines in **R** to calculate an automated estimate of $\hat{\pi}_0(1)$ using natural splines. In the current release (version 1.0), the qvalue function fitting the natural cubic spline does not weight the observations by $(1 - \lambda)$ (Appendix A.4). A simulation was carried out to

[1] Authors: J. D. Storey, G. R. Warnes (maintainer); http://www.bioconductor.org

test the difference in observed estimates of $\hat{\pi}_0(1)$, when including the $1-\lambda$ observation weighting. Exclusion of the $1-\lambda$ weighting argument in the natural cubic spline increases the estimated variance considerably, illustrated in Figure 3.9. Due to the bias variance tradeoff, variation in 95% confidence intervals is larger as $\pi_0 \to 1$. Note that the estimate $\hat{\pi}_0(\lambda)$ is generally unbiased under a normal mixture simulation over the entire range or π_0. When excluding the $1-\lambda$ weighting, there does appear to be a slight downward bias in the observed estimates of $\hat{\pi}_0(1)$ as $\pi_0 \to 1$.

Figure 3.9: Estimation of π_0, comparing the use of $1-\lambda$ natural spline weights in the automated procedure of Storey and Tibshirani (2003). Simulated p-values are from a mixture of normal distributions over a range of π_0 where $\mu = 3$ under H_A in equation 3.9. Black lines are the observed $E[\hat{\pi}_0(1)]$, grey lines are 2.5 and 97.5 percentiles of the distribution of observed $\hat{\pi}_0(1)$ values.



Practitioners must use procedures to estimate π_0 from experimental data if adaptive control of the FDR is desired at the level $\alpha^* = \alpha^*/\hat{\pi}_0$. Estimation of π_0 depends on the nature of the data examined and is not a straight forward calculation. Any variation or bias in the estimate $\hat{\pi}_0$ will cascade into the observed FDR as a function of π_0. Variability in the observed FDR is greatest when $\pi_0 = 1$. Experiments that have low sensitivity will have more variability in desired FDR control.

4
Bayesian Modelling of cDNA Microarrays

Empirical Bayes analysis was first applied to *E. coli* cDNA microarray data [16] in 2001 by Newton et al. [36], improving the statistical inference of gene expression changes on a single analyzed array. The approach models a parametric hierarchy of Gamma distributions, empirically estimating parameters of interest from the data. An alternative model using the same methodology was proposed in 2002 by Kendziorski et al. [37], fitting a hierarchy of Normal distributions on log transformed intensities. In the 2002 paper, both of these modelling approaches were extended to analyze experiments with replication of spot intensities. In 2003, Newton et al. [38] modified the approach further by relaxing the parametric assumption in the target mean layer, fitting a data driven semi-parametric hierarchical model to microarray data.

4.1 Bayesian overview

Bayesian probability theory is named after the 18th century clergyman Reverend Thomas Bayes (1702-1761) who worked on the "doctrine of chances". The underlying philosophy is that the only sensible measure of uncertainty is probability. Probability theory is the body of knowledge that enables us to reason formally about uncertain events. The most common view of probability is the classical frequentist approach defining the probability

P of an uncertain event A, written P(A), by the frequency of that event based on previous observations. Suppose we define the random variable $X = x_1, x_2, ..., x_n$ where x represents the vector of observations from an experiment, and θ represents the unknown parameters of interest. The classical modelling approach would be to assume the data come from a parametric family of distributions and model the likelihood $f(x|\theta)$, a function of the data, x, given the unknown fixed parameters, θ.

Bayesians treat parameters as random variables from a parametric family of distributions, any prior belief about θ is characterised in the prior distribution $\pi(\theta)$. The posterior distribution $\pi(\theta|x)$ represents the updated belief about θ given the actual data observed from the experiment. Bayes Theorem relates the posterior distribution to assumptions made about the prior belief , and the likelihood of the data.

4.1.1 Bayes Theorem for random variables

Let $B_1, B_2, ..., B_m$ form a set of mutually exclusive and exhaustive events in S. Then for any event A, Bayes Theorem states,

$$P(B_i|A) = \frac{P(B_i \cap A)}{P(A)} = \frac{P(A|B_i)P(B_i)}{\sum_{i=1}^{m} P(A|B_i)P(B_i)}.$$

The updated posterior probability of B_i given event A is $P(B_i|A)$, in the light of the observed data $P(A|B_i)$ and the prior probabilities $P(B_i)$ about event B_i. Assuming the distributions have densities, for the random variables X and θ the continuous variable analogue to Bayes Theorem is,

$$\pi(\theta|x) = \frac{f(x|\theta)\pi(\theta)}{\int_\Theta f(x|\theta)\pi(\theta)d\theta},$$

where $\pi(\theta|x)$ are posterior probabilities of the parameters, θ, given the data, $f(x|\theta)$ are likelihoods of the data given parameters θ, and $\pi(\theta)$ is the prior distribution for the unknown parameters θ. In practice the denominator does not need to be calculated, it is a scaling constant ensuring that the sum of all probabilities or area under the posterior distribution equals one. That is,

$$\pi(\theta|x) \propto f(x|\theta)\pi(\theta). \tag{4.1}$$

Simulation is a central part of solving Bayesian problems, due to the relative ease in which samples can be generated from the data and prior distributions. Large samples of simulated data can obtain good estimates of summary statistics about unknown posterior distributions of interest. Certain problems that are mathematically convenient allow empirical calculation of summary statistics of posterior distributions. Problems with the additional property that the posterior distribution follows the same parametric form as the prior distribution are said to be *conjugate.* If $\mathcal{F}$ is a class of likelihood sampling distributions $p(y|\theta)$, and $\mathcal{P}$ is a class of distributions representing prior belief about θ, the class P is *conjugate* for $\mathcal{F}$ if

$$\pi(\theta|y) \in \mathcal{P} \text{ for all } f(.|\theta) \in \mathcal{F} \text{ and } \pi(.) \in \mathcal{P}. \tag{4.2}$$

4.2 Modelling a single microarray

The modelling approach described in this section was published by Newton et al. (hereafter NKRBT) on a series of microarray experiments from the *E. coli* K-12 genome [16]. The entire set of open reading frames was reverse transcribed into cDNA probes and spotted on each microarray slide targeting 4290 genes of interest. Four separate microarray experiments were conducted comparing different cell lines of interest. Within each microarray experiment, there was no technical replication of cDNA spots on the microarrays, that is, each unique cDNA probe was spotted only once on each array.

Each microarray was normalized using a simple total intensity normalization in the *E. coli* datasets before analysis. This assumes that the numbers and mass of mRNA molecules in each sample are similar. First the background intensities of neighbouring unspotted areas were subtracted from spot foreground intensity values to adjust for slide surface background differences, and any resulting negative intensity spots in either channel were removed from further analysis. Defining $\underset{\sim}{x}^{'}$ to be the vector of raw intensities from the cy3 green channel, and $\underset{\sim}{y}^{'}$ to the vector of intensities from the cy5 red channel, the data can be represented as,

$$\underset{\sim}{x}^{'} = \begin{pmatrix} x_1^{'} \\ \vdots \\ x_m^{'} \end{pmatrix} \qquad \underset{\sim}{y}^{'} = \begin{pmatrix} y_1^{'} \\ \vdots \\ y_m^{'} \end{pmatrix}$$

where m equals the total number of spots on the microarray. Each intensity value was divided by the sum of all the intensities in that channel, with a scale adjustment multiplying the resulting intensities by a factor of 10^5,

$$\underset{\sim}{x} = \underset{\sim}{x}^{'} \frac{10^5}{\sum_{i=1}^{m} x_i^{'}} \qquad \underset{\sim}{y} = \underset{\sim}{y}^{'} \frac{10^5}{\sum_{i=1}^{m} y_i^{'}}$$

to avoid computational underflow.

Notation used in NKRBT assigned the random variables R and G to represent measured normalized intensity vectors from the cy5 red and cy3 green fluorescent channels since in this setting dyes are completely confounded with mRNA target treatments. Standard convention relates the red channel to the green channel in graphical exploration of single microarrays by plotting the cy5 red dye on the y axis. The notation $x = G$ and $y = R$ for each cell line treatment reflects this.

In competitive hybridization of single cDNA microarrays, the primary comparison of interest is the relative expression ratio T of the red normalized intensities divided by the green normalized intensities,

$$T_i = \frac{R_i}{G_i} = \frac{y_i}{x_i} = \frac{y_i^{'}/\sum_{i=1}^{m} y_i^{'}}{x_i^{'}/\sum_{i=1}^{m} x_i^{'}} = \frac{y_i^{'}\sum_{i=1}^{m} x_i^{'}}{x_i^{'}\sum_{i=1}^{m} y_i^{'}}.$$

The normalization adjusts each within gene ratio so that the average ratio across all genes is equal to 1. After normalization it is assumed that differences between the two dyes in the target treatments introduced in the experimental process have been removed. The fluorescent labelling reactions of target mRNA may have different efficiencies for each species of mRNA, but within gene competitive hybridizations will not be effected. The naive approach to identifying genes which differentially express is to rank the relative ratios from highest to lowest, calculating a threshold to identify potential differentially expressed genes at the beginning and end of the list, expressing upward and downward.

The Bayesian approach proposed by NKRBT models the entire dataset of 4290 genes represented on the microarray under a parametric hierarchy using an Empirical Bayes approach, allowing estimation of the unknown parameters of interest from the normalized intensity measurements derived from the each treatment. A major advantage of this approach is that it is able to take between gene information into account when estimating variability.

4.2.1 Sampling distribution for measured expression

The histograms of normalized intensity values in each fluorescence channel are extremely right skewed and can be parametrically modelled on the raw scale using independent Gamma distributions for each fluorescent intensity channel. The general form of the probability density function for the Gamma distribution is,

$$f(z \mid a, \theta_z) \;=\; \frac{\theta_z^a}{\Gamma(k)} z^{a-1} e^{-az} \qquad z,\ a,\ \theta_z > 0,$$

with summary statistics,

$$E[z] = \mu_z = \frac{a}{\theta_z}, \quad Var[z] = \sigma_z^2 = \frac{a}{\theta_z^2}, \quad CV[z] = \frac{\mu_z}{\sigma_z^2} = \frac{1}{\sqrt{a}}.$$

The advantage of the Gamma distribution is that it is extremely flexible, and supported on the positive number line, with shape parameter a and scale parameter θ_z.

Taking the approach of NKRBT, let x and y represent the normalized intensities in either treatment channel, from which any experimental effects have been removed. Spots undergoing differential expression are assumed to arise from independent Gamma distributions with common shape parameter, a, but different scale parameters, θ_x and θ_y. That is,

$$x \sim \Gamma(a, \theta_x), \quad y \sim \Gamma(a, \theta_y).$$

The coefficient of variation for each intensity channel depends only on the common shape parameter a, independent of the scale parameters θ_x and θ_y which may be quite different in each intensity channel.

The sampling distribution of measured differential expression $T = y/x$, is derived from the joint independent Gamma distributions of the two intensity channels x and y, with expected target differential expression ratio $\rho = \mu_y/\mu_x = \theta_x/\theta_y$. The sampling distribution of T given ρ and a can be derived from the joint distribution of x and y,

$$f(y,x) = \frac{\theta_y^a \theta_x^a}{\Gamma^2(a)} y^{a-1} x^{a-1} e^{-(\theta_y y + \theta_x x)} \tag{4.3}$$

by specifying an additional dummy variable S, and determining the joint distribution of T and S,

$$\begin{aligned}
& S = \theta_y y + \theta_x x, \quad T = \frac{y}{x} \\
& \Rightarrow \quad y = \frac{ST}{\theta_x + \theta_y T}, \quad x = \frac{S}{\theta_x + \theta_y T} \\
J\left(\frac{S,T}{y,x}\right) &= \begin{vmatrix} \theta_y & \theta_x \\ \frac{1}{x} & -\frac{y}{x^2} \end{vmatrix} = -\frac{\theta_y y + \theta_x x}{x^2} = -\frac{(\theta_x + \theta_y T)^2}{S} \\
dydx &= \frac{S}{(\theta_x + \theta_y T)^2} dSdT
\end{aligned}$$

$$\begin{aligned}
f(S,T|\rho,a) &= \frac{\theta_y^a \theta_x^a}{\Gamma^2(a)} \left(\frac{ST}{\theta_x + \theta_y T}\right)^{a-1} \left(\frac{S}{\theta_x + \theta_y T}\right)^{a-1} \left(\frac{S}{(\theta_x + \theta_y T)^2}\right) e^{-S} \\
&= \frac{\theta_y^a \theta_x^a}{\Gamma^2(a)} \frac{S^{2a-1} T^{a-1}}{(\theta_x + \theta_y T)^{2a}} e^{-S} \\
&= \frac{1}{\Gamma^2(a)} \left(\frac{\theta_y}{\theta_x}\right)^a \frac{S^{2a-1} T^{a-1}}{\left(\frac{1}{\theta_x}\right)^{2a} (\theta_x + \theta_y T)^{2a}} e^{-S} \\
&= \frac{1}{\Gamma^2(a)} \left(\frac{1}{\rho}\right)^a \frac{S^{2a-1} T^{a-1}}{(1 + T/\rho)^{2a}} e^{-S}, \quad \text{where } \rho = \frac{\theta_x}{\theta_y}.
\end{aligned}$$

Integrating out S, the distribution of T is given by,

$$\begin{aligned} f(T|\rho, a) &= \int_0^\infty f(S, T|\rho, a) dS \\ &= \frac{1}{\Gamma^2(a)} \left(\frac{1}{\rho}\right)^a \frac{T^{a-1}}{(1+T/\rho)^{2a}} \int_0^\infty S^{2a-1} e^{-S} dS \\ &= \frac{\Gamma(2a)}{\Gamma^2(a)} \left(\frac{1}{\rho}\right)^a \frac{T^{a-1}}{(1+T/\rho)^{2a}} \\ &= \frac{\Gamma(2a)}{\Gamma^2(a)} \left(\frac{1}{\rho}\right) \frac{(T/\rho)^{a-1}}{(1+T/\rho)^{2a}} \quad \text{for } T > 0. \end{aligned} \tag{4.4}$$

If there is no differential expression then $\rho = 1$, and $f(T|\rho = 1, a)$ will depend on a only. For large values of T the tail of the distribution will be asymptotic to $1/T^{(a+1)}$. The restriction $a > 1$ implies that,

$$\begin{aligned} f(T \mid \rho = 1, a) &\propto \frac{(T)^{a-1}}{(1+T)^{2a}} \\ &\propto \frac{1}{T^{(a-1)}} \text{ for } T \gg 1. \end{aligned}$$

To consider statistical analysis on the sampling distribution $T = y/x$ alone, however, results in a loss of information, as $T = y/x$ contains no information about $S = xy$. Under no differential expression with $\rho = 1$, and a common scale for the sampling distributions of x and y, NKRBT show that the distribution of T given S is proportional to

$$f(T \mid S, \theta, a) \propto \frac{1}{T} e^{-\theta\sqrt{S}\left(\sqrt{T}+\frac{1}{\sqrt{T}}\right)}.$$

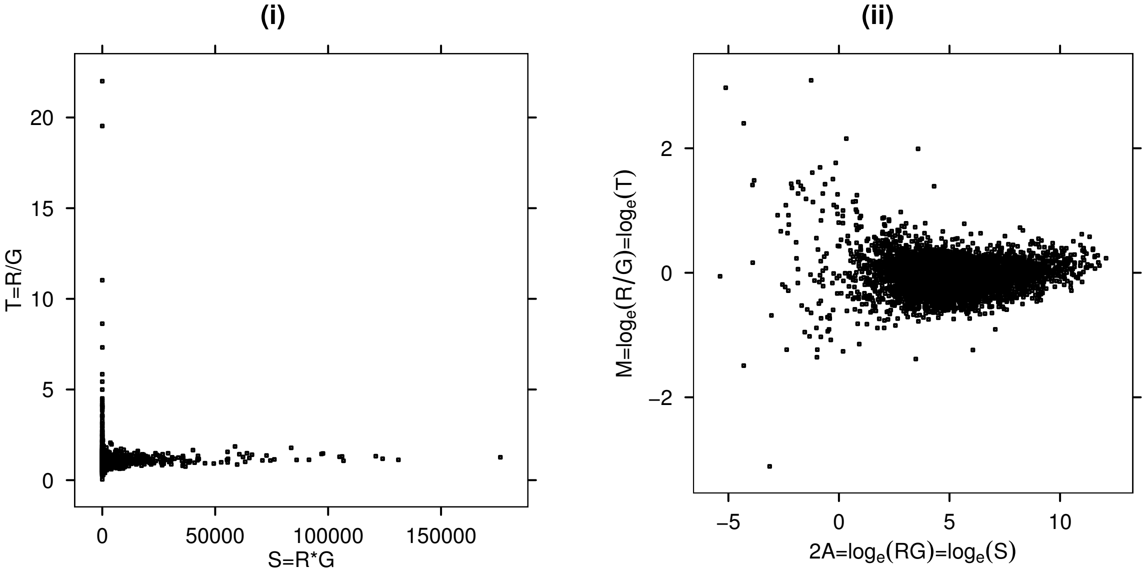
The magnitude of S effects the variation in T, for small S there will be greater variation in T, since suggesting that at low expression levels there are greater amounts of measurement error from multiple sources of variation in the microarray experiment. Actual data from the NKRBT *E. coli* control microarray, illustrates the dependence of T on S, as shown in Figure 4.1. Due to the highly skewed nature of the intensities, $log_e(T)$ versus $\log_e(S)$ is also plotted. As the strength of the signal in S increases, it can be seen that variability in T decreases.

4.2.2 Posterior distribution of differential expression

NKRBT make the prior assumption that the scale parameters θ_y and θ_x under differential expression are themselves sampled from a Gamma distribution, with common shape parameter a_0, and scale parameter λ,

$$\theta_y \sim Gamma(a_o, \lambda), \qquad \theta_x \sim Gamma(a_o, \lambda).$$

Figure 4.1: *E. coli* control dataset [16]. (i) T versus S, As S increases the magnitude of T also decreases. (ii) $log_e(T)$ versus $log_e(S)$, this is equivalent to a plot of M versus $2A$.



Deriving the posterior distributions, $\pi(\theta_z|z)$, for either target intensity channel yields,

$$\begin{aligned}\pi(\theta_z \mid z) &\propto f(z \mid \theta_z)\pi(\theta_z)\\ &\propto \theta_z^{a} e^{-z\theta_z}\theta_z^{a_o-1}e^{-\theta_z\nu}\\ &\propto \theta_z^{a+a_o-1}e^{-\theta_z(z+\nu)}.\end{aligned}$$

The posterior distribution is a conjugate Gamma distribution, so for each intensity channel the distributions of the scale parameters are given by,

$$\begin{aligned}\theta_y \mid y &\sim Gamma(a+a_o, y+\nu)\\ \theta_x \mid x &\sim Gamma(a+a_o, x+\nu).\end{aligned}$$

The expected value of the ratio of posterior *Gamma* distributions is

$$E\left(\frac{\theta_x|x}{\theta_y|y}\right) = \frac{y+\nu}{x+\nu}.$$

The posterior distribution of ρ given the intensity data, and $\eta = (a, a_o, \nu)$ is derived from

4.4, as follows,

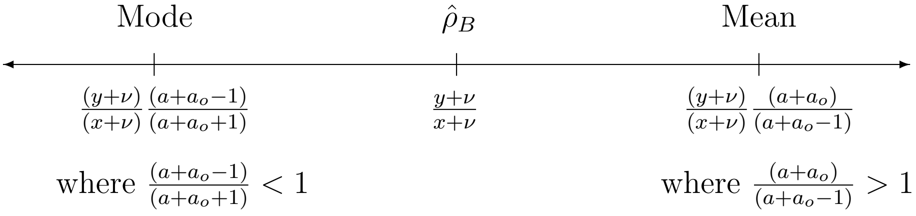
$$\begin{aligned}
\pi(\rho \mid y, x, \eta) &\propto \frac{(y+\nu)}{(x+\nu)}^{-1} \frac{\left(\frac{\rho}{\frac{(y+\nu)}{(x+\nu)}}\right)^{a+a_o-1}}{\left(1+\frac{\rho}{\frac{(y+\nu)}{(x+\nu)}}\right)^{2(a+a_o)}} \frac{\rho^{-2(a+a_o)}}{\rho^{-2(a+a_o)}} \\
&\propto \frac{(y+\nu)}{(x+\nu)}^{-1} \left(\frac{1}{\frac{(y+\nu)}{(x+\nu)}}\right)^{a+a_o-1} \frac{\rho^{(a+a_o-1)}\rho^{-2(a+a_o)}}{\left(\frac{1}{\rho}+\frac{x+\nu}{y+\nu}\right)^{2(a+a_o)}} \\
&\propto \rho^{-(a+a_o+1)} \left(\frac{1}{\rho}+\frac{x+\nu}{y+\nu}\right)^{-2(a+a_o)} . \qquad (4.5)
\end{aligned}$$

The posterior distribution in 4.5 is proportional to the distribution of the ratio of two independent Gamma distributions. NKRBT proposed the following Bayesian posterior summary statistic, $\hat{\rho}_B$, to characterize the posterior distribution of ρ,

$$\hat{\rho}_B = \frac{y+\nu}{x+\nu}.$$

The ratio $\hat{\rho}_B$ depends on ν, the common scale parameter from the target Gamma differential expression distributions. This statistic is a shrinkage estimator. If the signal in both channels is small, the ratio will be attenuated significantly by the parameter ν, whereas if the signal in both channels is large, ν will have less influence and the estimate of $\hat{\rho}_B$ will be closer to $\hat{\rho}_N = y/x$. This choice of summary estimator is for computational simplicity, as $\hat{\rho}_B$ lies between the mode and the mean of the posterior distribution, as illustrated in Figure 4.2.2.

Figure 4.2: Bayes posterior estimate $\hat{\rho}_B$ lies between the Mode and the Mean of the posterior distribution in equation 4.5.


Mode
$\hat{\rho}_B$
Mean
$\frac{(y+\nu)}{(x+\nu)}\frac{(a+a_o-1)}{(a+a_o+1)}$
$\frac{y+\nu}{x+\nu}$
$\frac{(y+\nu)}{(x+\nu)}\frac{(a+a_o)}{(a+a_o-1)}$
where $\frac{(a+a_o-1)}{(a+a_o+1)} < 1$
where $\frac{(a+a_o)}{(a+a_o-1)} > 1$

The approach taken by NKRBT to estimate the unknown parameters $\eta = (a, a_o, \nu)$ uses the maximum likelihood of the observed data to obtain empirical parameter estimates of η. If spot intensities are differentially expressed it is assumed that they have different scale parameters, θ_x and θ_y, and are independently derived from separate Gamma sampling distributions. The joint probability of differential expression (denoted by $\mathcal{DE}$) is,

$$P_{\mathcal{DE}}(x,y) \;=\; P(x)\cdot P(y).$$

The marginal distributions for each intensity channel are derived by integrating out uncertainty in the joint distribution of each channel of intensity data, and the scale parameters for θ_x and θ_y. The marginal distributions of x and y are prior predictive distributions, prior since the distributions are not conditional on previous observations, and predictive of a distribution that is observable. Denoting z as the observed intensity data in either target channel,

$$P(z) \;=\; \int_{\theta_z} f(z,\theta_z)d\theta_z = \int_{\theta_z} f(z|\theta_z)\pi(\theta_z)d\theta_z.$$

The marginal distributions for normalised intensity data in either channel are derived by integrating out uncertainty in the scale parameter θ_z for the observed likelihood and prior distribution,

$$\begin{aligned} P(z) &= \int_{\theta_z} f(z|\theta_z)\pi(\theta_z)d\theta_z \\ &= \int_0^\infty \frac{\theta_z^a}{\Gamma(k)} z^{a-1}e^{-az}\frac{\nu_o^a}{\Gamma(k)}\theta_z^{a_o-1}e^{-a_o\theta_z}d\theta_z \\ &= \frac{\nu^{a_o}z^{a-1}}{\Gamma(a)\Gamma(a_o)}\int_o^\infty \theta_z^{(a+a_o-1)}e^{-\theta_z(z+\nu)}d\theta_z \\ &= \frac{\Gamma(a+a_o)}{\Gamma(a)\Gamma(a_o)}\frac{\nu^{a_o}z^{a-1}}{(z+\nu)^{a+a_o}}, \end{aligned}$$

and the joint distribution of x and y derived from independent Gamma sampling distributions under differential expression is

$$\begin{aligned} P_{\mathcal{DE}}(x,y) &= P(x).P(y) \\ &= \int_{\theta_x} f(x|\theta_x)\pi(\theta_x)d\theta_x \cdot \int_{\theta_y} f(y|\theta_y)\pi(\theta_y)d\theta_y \\ &= \frac{\Gamma(a+a_o)}{\Gamma(a)\Gamma(a_o)}\frac{\nu^{a_o}x^{a-1}}{(x+\nu)^{a+a_o}}\cdot\frac{\Gamma(a+a_o)}{\Gamma(a)\Gamma(a_o)}\frac{\nu^{a_o}y^{a-1}}{(y+\nu)^{a+a_o}} \\ &= \left(\frac{\Gamma(a+a_o)}{\Gamma(a)\Gamma(a_o)}\right)^2\frac{(\nu)^{2a_o}(xy)^{(a-1)}}{[(x+\nu)(y+\nu)]^{(a+a_o)}}. \end{aligned} \tag{4.6}$$

After integrating out uncertainty in the gene specific scale parameters, θ_x, and θ_y, to estimate marginal distributions in each channel, the marginal likelihood, $l(a,a_o,\nu)$, can be maximised to estimate the unknown parameters of interest, $\eta = (a, a_o, \nu)$ as follows,

$$\begin{aligned} l(a, a_o, \nu) &= \sum_{k=1}^{m} \log P_{\mathcal{DE}}(x_k, y_k) \\ &= 2m[\log\Gamma(a + a_o) - \log\Gamma(a) - \log\Gamma(a_o) + a_o \log(\nu)] \\ &+ \sum_{k=1}^{m} [(a-1)(\log(r) + \log(g)) - (a + a_o)((r+\nu)(g+\nu))] \,. \end{aligned}$$

4.2.3 Gamma-Gamma-Bernoulli model

NKRBT use the Gamma-Gamma-Bernoulli (GGB) model to add a third discrete layer. Here each gene arises from either a differentially expressed distribution $P_{\mathcal{DE}}(x, y)$, or an equivalently expressed (denoted by $\mathcal{EE}$) distribution $P_{\mathcal{EE}}(x, y)$, from assumed Bernoulli sampling. Under equivalent expression the scale parameters for each sampling distribution are derived from the same prior distribution,

$$\begin{aligned} P_{\mathcal{EE}}(x, y) &= \int_\theta f(x, y \mid \theta)\pi(\theta) d\theta \\ &= \int_\theta f(x \mid \theta) \cdot f(y \mid \theta)\pi(\theta) d\theta \\ &= \int_\theta \frac{\theta^a}{\Gamma(k)} x^{a-1} e^{-ax} \cdot \frac{\theta^a}{\Gamma(k)} y^{a-1} e^{-ay} \cdot \frac{\nu_o^a}{\Gamma(k)} \theta^{a_o - 1} e^{-a_o\theta} d\theta \\ &= \frac{(xy)^{a-1}\nu^{a_o}}{\Gamma^2(a)\Gamma(a_o)} \int_\theta \theta^{(2a+a_o-1)} e^{-\theta(x+y+\nu)} d\theta \\ &= \left(\frac{\Gamma(2a + a_o)}{\Gamma^2(a)\Gamma(a_o)}\right) \frac{(\nu)^{a_o}(xy)^{(a-1)}}{[(x+y+\nu)]^{(2a+a_o)}}. \end{aligned} \tag{4.7}$$

The marginal likelihood, $l(a, a_o, \nu)$ can be maximised to estimate the unknown parameters of interest $\eta = (a, a_o, \nu)$, as follows,

$$\begin{aligned} l(a, a_o, \nu) &= \sum_{k}^{m} \log P(x_k, y_k) \\ &= m[\log\Gamma(2a + a_o) - 2\log\Gamma(a) - \log\Gamma(a_o) + a_o \log \nu] \\ &+ \sum_{k}^{m} [(a-1)(\log(x) + \log(y)) - (2a + a_o)(x + y + \nu)] \,. \end{aligned}$$

As the identity of the changed spots is unknown, we observe complete data with incomplete mixing of unknown differentially expressed spots.

4.2.4 EM algorithm

The EM algorithm [39] is applicable to a wide range of problems where incomplete data is observed. In the case of the Gamma-Gamma-Bernoulli model, the identity of genes undergoing differential expression is unknown. Assuming each spot, $k = 1, \ldots, m$ represents a single gene on the microarray, complete data expression patterns for x and y are observed from either $P_{\mathcal{E}\mathcal{E}}(x_k, y_k)$ under equivalent expression, or $P_{\mathcal{D}\mathcal{E}}(x_k, y_k)$ under differential expression, according to an underlying unknown Bernoulli random variable z_k. Assuming the z_k's are independent, then $z = \sum z_k$, represents the unknown number of differentially expressed genes. Thus z follows a Binomial distribution with parameters m and $p = P(z_k = 1)$. In this setting the goal is to estimate the probability of differential expression for the completed data (x_k, y_k, z_k). In two channel microarrays the probability of equivalent expression is $\pi_0 = 1 - p$. The marginal density of (x_k, y_k) is

$$\begin{aligned} f(x_k, y_k) &= \sum_i P(x_k, y_k \cap z_i) \\ &= \sum_i P(Z_i) P(x_k, y_k | z_i) \\ &= P(Z = 0) f(x_k, y_k | Z = 0) + P(Z = 1) f(x_k, y_k | Z = 1) \\ &= p.p_A(x_k, y_k) + (1 - p).p_0(x_k, y_k). \end{aligned} \tag{4.8}$$

The completed data density function is

$$\begin{aligned} f(x_k, y_k, z_k) &= (p.p_A(x_k, y_k))^{z_k}((1 - p).p_0(x_k, y_k))^{1-z_k} \\ &= p^{z_k}.(1 - p)^{1-z_k}.p_A(x_k, y_k)^{z_k}.p_0(x_k, y_k)^{1-z_k}, \end{aligned}$$

and the complete data log likelihood is given by,

$$\begin{aligned} l_c(a, a_o, \nu, p) &= \log \prod_{k=1}^{n} f(x_k, y_k, z_k) \\ &= \sum_{k=1}^{m} \{z_k \log p_(x_k, y_k) + (1 - z_k) \log p_0(x_k, y_k) \\ &+ z_k \log(p) + (1 - z_k) \log(1 - p)\}. \end{aligned} \tag{4.9}$$

The EM algorithm is a two step iterative estimation process. In iteration $i+1$, the first step is to find the expected value of the complete data log likelihood given the estimate $p^{(i)}$ of the previous iteration,

$$\begin{aligned} z_k &= E[z_k | x_k, y_k] \\ &= P(z_k = 1 | x_k, y_k) \end{aligned}$$

$$
\begin{aligned}
&= \frac{f(z=1 \cap x_k, y_k)}{f(x_k, y_k)} \\
&= \frac{p.p_A(x_k, y_k)}{p.p_A(x_k, y_k) + (1-p).p_0(x_k, y_k)}. \qquad (4.10)
\end{aligned}
$$

The updated estimate $p^{(i+1)}$ is the value maximising the Binomial component $p^z(1-p)^{n-z}$ from the completed likelihood, where,

$$
\begin{aligned}
p^{(i+1)} &= \frac{z}{m} \\
&= \frac{1}{m}\sum_{k=1}^{m} z_k \\
&= \frac{1}{m}\sum_{k=1}^{m} \frac{p.p_A(x_k, y_k)}{p.p_A(x_k, y_k) + (1-p).p_0(x_k, y_k)}.
\end{aligned}
$$

The second step is to use the updated estimate of $p^{(i+1)}$ to maximise the complete data log likelihood, which maximises the expectation found in the first step to obtain updated parameter estimates for $\eta = (a, a_o, \nu)$. NKRBT further stabilizes the computations by assuming a $Beta(2,2)$ prior distribution for p, and calculating a posterior update for p. The assumption is made the z_k are exchangeable, invariant to permutations of the indices, modelled independently and identically distributed.

4.2.5 Computational considerations in the EM algorithm

NKRBT used the **Splus** (Statistical Sciences, 1993)[40] optimization function **nlminb** to maximize the log likelihood of the Gamma-Gamma-Bernoulli models. These results are reproduced using here the open source software **R** and the equivalent optimization function **optim**. The optimization functions obtain new parameter estimates for η from the maximization step of the EM algorithm by minimizing the negative log likelihood in 4.9. The input arguments required are the negative log likelihood function to be minimized, and initial estimates of η. Results in NKRBT used fixed initial values (FIV) of $\eta = (10, 1, 1)$ at every maximization step of the EM algorithm. This analysis was repeated on the IPTG-a microarray using the current estimates (CE) of η in the optimization function during each iteration of the EM algorithm. The modified EM algorithm produces estimates for η which were slightly different than those of NKRBT. A comparison between the two EM models is provided in Table 4.1. Using FIV during each maximization step affected convergence slightly. To reach convergence, the optimizer functions minimize η to within a certain tolerance which will be obtained in less iterations if the initial values are closer to the optimized solution. If the sum of these gene probabilities is significantly different using FIV versus CE in the EM algorithm, it is possible that the rank order of the individual gene probabilities could also change in position, depending on the optimization

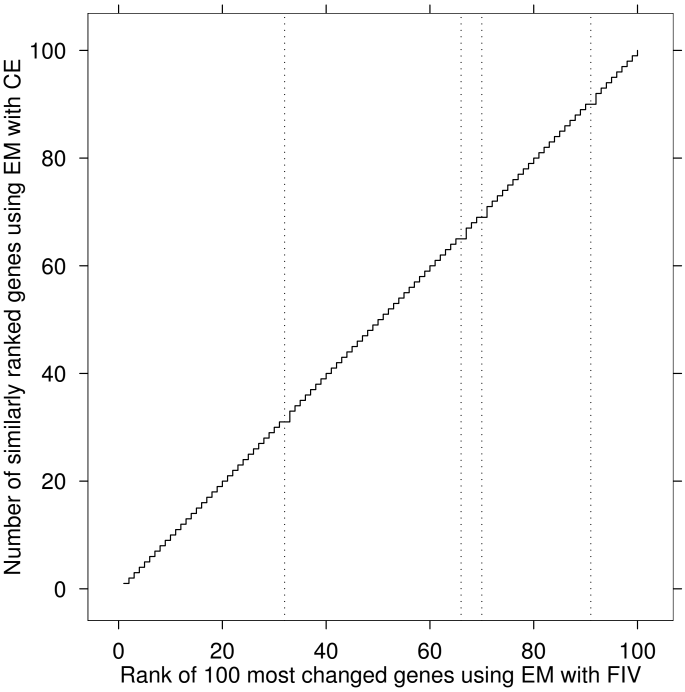
Table 4.1: *E coli.* IPTG-a microarray using different optimization initial values.

	a	a_o	ν	p
FIV	12.535	0.816	0.371	0.00688
CE	12.569	0.816	0.369	0.00695
% difference	0.273	−0.050	−0.380	1.01039

function initial values used during computation.

Figure 4.3 illustrates the ranking changes of the first 100 predicted differentially expressed genes in the Gamma-Gamma-Bernoulli model using CE versus FIV in the IPTG-a microarray. The individual probabilities of differentially expressed genes from the FIV Gamma-Gamma-Bernoulli model were ranked from first to last and plotted on the x axis, while the number of similarly ranked genes from the CE Gamma-Gamma-Bernoulli model was calculated and plotted on the y axis. Lines have been added at points in the step function where the gene ranking has changed across the two methods . If computation of the Gamma-Gamma-Bernoulli model used FIV in the maximization step, a list of differentially expressing genes could potentially exclude a candidate due to the bias in the maximization step. As this problem is associated with numerical approximation, the effect is likely to increase as the number of genes on the microarray increases.

Figure 4.3: Effect of ranking on the first 100 genes, comparing similarly ranked genes under CE to FIV in the EM algorithm. Vertical dotted lines show points where gene ranking changes.



4.3 Modelling replicate microarrays

There are two major ways that cDNA probes can provide multiple within gene intensity measurements on microarrays. Unique cDNA species can be spotted multiple times on a microarray allowing competitive hybridization to take place with equal probability for all technical replicates. Dye swaps are a common design which provide additional within gene information eliminating any dye bias effects. It is assumed that a suitable preprocessing technique has been used to adequately combine and normalize multiple intensity values derived from the replication of cDNA probes or multiple experimental runs of the same microarray information.

Consider a set of microarray experiments, with n_1 replicate measurements in the first treatment cell line, and n_2 replicate measurements in the second treatment cell line. If there are m transcripts spotted on any microarray targeting individual genes, the matrixes X and Y, contain the spot intensity information for both treatment channels,

$$X = \begin{pmatrix} x_{11} & \dots & x_{1n_1} \\ \vdots & \ddots & \vdots \\ x_{m1} & \dots & x_{mn_1} \end{pmatrix} \qquad Y = \begin{pmatrix} y_{11} & \dots & y_{1n_2} \\ \vdots & \ddots & \vdots \\ y_{m1} & \dots & y_{mn_2} \end{pmatrix}.$$

For a particular gene, i there are n_1 within gene replicate measurements for x_i, and n_2 within gene replicate measurements for y_i where $x_i = (x_{i1}, \dots, x_{in_1})$, and $y_i = (y_{i1}, \dots, y_{in_1})$. The number of within gene replicate measurements is assumed to be constant over genes

Under differential expression, Kendziorski et al. [37] show that the sampling distribution for each treatment intensity channel is the product of independent gamma distributions. Defining Z as being the normalised intensities from either channel,

$$\begin{aligned} f_{\mathcal{DE}}(Z) &= \int_{\theta_z} f(Z|\theta_z)\pi(\theta_z)d\theta_z \\ &= \int_{\theta_z} \prod_{j=1}^{n} \left(\frac{\theta_z^a}{\Gamma(k)} z_{\cdot j}^{a-1} e^{-az_{\cdot j}} \right) \frac{\nu_o^a}{\Gamma(k)} \theta_z^{a_o-1} e^{-a_o\theta_z} d\theta_z \\ &= \frac{\prod_{j=1}^{n} z_{\cdot j}^{a-1} \nu^{a_o}}{\Gamma^n(a)\Gamma(a_o)} \int_{\theta_z} \theta_z^{(na+a_o-1)} e^{-(\nu+\sum_{j=1}^{n} z_{\cdot j})\theta_z} d\theta_z \\ &= K_n \cdot \frac{\prod_{j=1}^{n} z_{\cdot j}^{a-1}}{(\nu + \sum_{j=1}^{n} z_{\cdot j})^{na+a_o}}, \end{aligned} \tag{4.11}$$

where

$$K_n = \nu^{a_o} \frac{\Gamma(na+a_o)}{\Gamma^n(a)\Gamma(a_o)}.$$

Under differential expression the joint distribution of X and Y arises independently from

$\pi(\theta)$,

$$\begin{aligned} f_{\mathcal{DE}}(X,Y) &= f(X)\cdot f(Y) \\ &= \int_{\theta_x} f(X|\theta_x)\pi(\theta_x)d\theta_x \int_{\theta_y} f(Y|\theta_y)\pi(\theta_y)d\theta_y \\ &= K_{n_1}K_{n_2}\cdot \frac{\prod_{j=1}^{n_1} x_{\cdot j}^{a-1}\prod_{j=1}^{n_2} y_{\cdot j}^{a-1}}{(\nu+\sum_{j=1}^{n_1} x_{\cdot j})^{n_1a+a_o}(\nu+\sum_{j=1}^{n_2} y_{\cdot j})^{n_2a+a_o}}, \qquad (4.12) \end{aligned}$$

while under equivalent expression the joint distribution of X and Y arises from the same prior distribution $\pi(\theta)$

$$\begin{aligned} f_{\mathcal{EE}}(X,Y) &= \int_\theta f(X|\theta)f(Y|\theta)\pi(\theta)d\theta \\ &= \int_\theta \prod_{j=1}^{n_1}\left(\frac{\theta^a}{\Gamma(k)}x_{\cdot j}^{a-1}e^{-ax_{\cdot j}}\right)\prod_{j'=1}^{n_2}\left(\frac{\theta^a}{\Gamma(k)}y_{\cdot j'}^{a-1}e^{-ay_{\cdot j'}}\right)\frac{\nu_o^a}{\Gamma(k)}\theta^{a_o-1}e^{-a_o\theta}d\theta \\ &= \prod_{j=1}^{n_1} x_{\cdot j}^{a-1}\prod_{j'=1}^{n_1} y_{\cdot j'}^{a-1}\frac{\nu^{a_o}}{\Gamma^{n_1}(a)\Gamma^{n_2}(a_o)\Gamma(a_o)}\cdot \\ &\quad \int_\theta\left(\prod_{j=1}^{n_1}\theta^a e^{-x_{\cdot j}\theta}\right)\left(\prod_{j'=1}^{n_2}\theta^a e^{-x_{\cdot j'}\theta}\right)\theta^{a_o-1}e^{-\theta\nu}d\theta \\ &= K\cdot\frac{\prod_{j=1}^{n_1} x_{\cdot j}^{a-1}\prod_{j'=1}^{n_1} y_{\cdot j'}^{a-1}}{(\nu+\sum_{j=1}^{n_1} x_{\cdot j}\sum_{j'=1}^{n_1} y_{\cdot j'})^{n_1a+n_2a+a_o}}, \qquad (4.13) \end{aligned}$$

where

$$K = \frac{\nu^{a_o}\Gamma(n_1a+n_2a+a_o)}{\Gamma^{n_1}(a)\Gamma^{n_2}(a)\Gamma(a_o)}.$$

Equations 4.12 and 4.13 are multiple replicate extensions of equations 4.6 and 4.7 which can be substituted into the E step (equation 4.10), and the M step (equation 4.9) of the EM algorithm. The sufficient statistics required to estimate the Gamma-Gamma-Bernoulli model are the number of replicates within each channel, and the within gene products and summations of normalized intensity values. Computationally it is easier to obtain these statistics on the log scale, back-transforming intensity values as required.

4.3.1 Bayesian adaptive FDR estimation

Newton et al. [38] provide an approach for controlling the adaptive false discovery rate from gene lists obtained by posterior probability in their Bayesian analysis of microarray data. The goal is to correctly identify a list of genes of size J predicted to be differentially expressed while controlling the rate of false discoveries at probability level α^*, and maximising the size of the list.

Within the Bayesian framework, $P(z_k = 1|x_k, y_k, p)$ is the posterior probability of differential expression for each gene. If a gene has a high posterior probability of being differentially expressed, it has a low posterior probability, $P(z_k = 0|x_k, y_k, p)$, of being a Type I error where,

$$P(z_k = 0|x_k, y_k, p) = \frac{(1-p)P_{EE}(x_k, y_k)}{pP_{DE}(x_k, y_k) + (1-p)P_{EE}(x_k, y_k)}.$$

Let $J(\kappa)$ be the size of the list of predicted differentially expressed genes as a function of a significance threshold, κ, where,

$$J(\kappa) \quad = \quad \{g \in (1, 2, \ldots N) : P(z_k = 0|x_k, y_k, p) \leq \kappa\}\,. \tag{4.14}$$

$J(\kappa)$ is a measure of the observed number of hypotheses declared significant, and thus is an estimate of the observable random variable R in Table 3.1. Under multiple hypothesis testing, the unobservable random variable V is estimated by summation of all the unique gene probabilities contained in the list $J(\kappa)$. This is the expected number of false discoveries, denoted #FD by NKRBT,

$$E\,[\#FD] \quad = \quad \sum_{g \in J(K)} P(z_k = 0|x_k, y_k, p). \tag{4.15}$$

In a list containing all m genes where $J(\kappa = 1)$, the expected number of false discoveries equals m_0. Equations 4.14 and 4.15 are used to estimate the proportion of false discoveries V/R, under adaptive FDR control for a α^* threshold level so that,

$$\frac{E[\#FD|x_g, y_g]}{J(\kappa)} \leq \alpha^*. \tag{4.16}$$

This is similar to control of the positive FDR in [41], except in the Bayesian context the expectations are conditional on the observed data. Note, FDR control in equation 4.16 requires unique gene probabilities in the numerator associated with unique values of $J(\kappa)$, a measure of gene list size based on the unique posterior probability threshold κ. If this is not the case, a situation can occur where the FDR threshold splits posterior probabilities of the same magnitude into two groups of equivalently expressed and differentially expressed genes.

4.4 Simulation of normalized microarrays

Three models simulating microarray datasets (with no experimental effects present) were investigated for statistical comparisons between the analysis approaches outlined in chapter 5. The first two hierarchical models simulate from the Gamma-Gamma-Bernoulli and Log-Normal-Normal models based on the work of Newton et al. [36] and Kendziorski et al. [37]. The third hierarchical model is a Gamma-Uniform-Bernoulli model [38].

Each of these models consists of a three layer hierarchy. The top layer is discrete; with genes coming from a differentially expressed parametric distribution (DE), or an equivalently expressed parametric distribution (EE). In the middle layer, measured spot intensities are assumed to vary around some mean value, modelled by a parametric distribution. The actual measured spot intensities in the bottom layer are derived from another parametric distribution which accounts for measurement error within the arraying process.

In these simulation scenarios, tracking the identity of equivalently expressed and differentially expressed genes, allows validation between statistical models analyzing the simulated datasets.

4.4.1 Gamma-Gamma-Bernoulli model

The actual measured spot intensities are sampled from a $Gamma(a, \theta)$ distribution. The target mean intensities, θ, are sampled from a $Gamma(a_o, \nu)$ distribution. θ is assumed to be the same for each treatment condition under equivalent expression, and different for each treatment channel under differential expression,

$$\begin{aligned}
&X_i \sim \Gamma(a, \theta_x), \quad Y_i \sim \Gamma(a, \theta_y)\\
&\text{Equivalent expression } (\mathcal{EE}) : \theta \sim \Gamma(a_o, \nu), \text{ where } \theta_x = \theta_y = \theta\\
&\text{Differential expression}(\mathcal{DE}) : \theta_x, \theta_y \sim \Gamma(a_o, \nu).
\end{aligned}$$

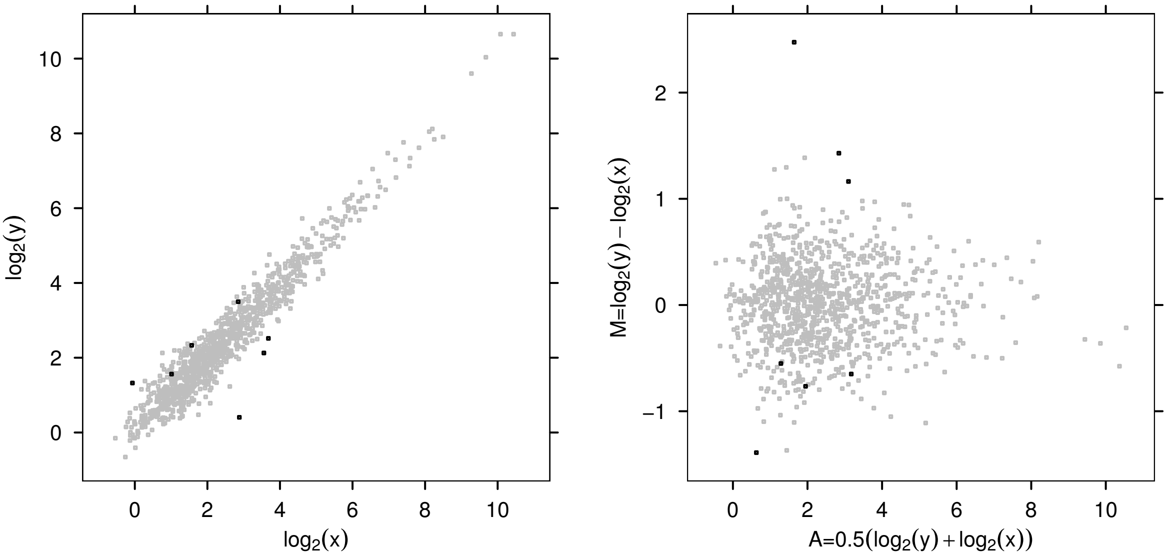
This model adequately describes increasing measurement error as the normalized intensity signal decreases, Gamma models have positive support on the measured intensity range, and are flexible and scaleable distributions. They graphically display realistic structure in M versus A plots. Figure 4.4 provides a simulation example with parameters $\eta = (a = 12.53, a_0 = 0.82, \nu = 0.37, p = 0.007)$. These are based on the IPTG-a *E. coli* experiment [16] fitting a Gamma-Gamma-Bernoulli Empirical Bayes model to the dataset.

4.4.2 Log-Normal-Normal-Bernoulli model

The simulation for the log-normal-normal model is on the log-intensity scale. The actual measured spot log-intensities are sampled from a $N(\mu, \sigma)$ distribution. The target mean intensities μ, are sampled from a $N(\mu, \tau)$ distribution as follows,

$$\begin{aligned}
&X_i \sim N(\mu_x, \sigma), Y_i \sim N(\mu_y, \sigma)\\
&\text{Equivalent expression } (\mathcal{EE}) : \mu \sim N(\mu, \tau), \text{ where } \mu_x = \mu_y = \mu
\end{aligned}$$

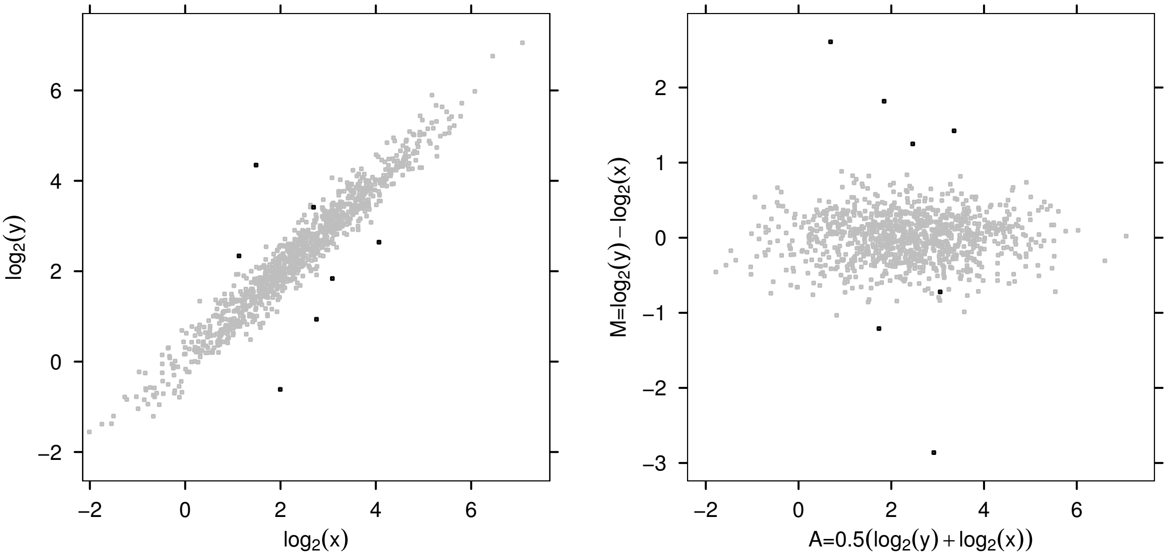
Figure 4.4: Simulated dataset (m=1000 genes) from a Gamma-Gamma-Bernoulli Hierarchy, $\eta = (a = 12.53, a_0 = 0.82, \nu = 0.37, p = 0.007)$. Equivalently expressed spots are coloured grey, differentially expressed spots are black.



$$\text{Differential expression}(\mathcal{DE}) : \mu_x, \mu_y \sim N(\mu, \tau).$$

Figure 4.5 provides a simulation example with parameters $\eta = (\mu = 2.37, \sigma^2 = 0.05, \tau^2 = 1.73, p = 0.007)$. These are based on the IPTG-a *E. coli* experiment [16] fitting a Log-Normal-Normal Empirical Bayes model to the dataset. [37].

Figure 4.5: Simulated dataset (m=1000 genes) from a Log-Normal-Normal-Bernoulli Hierarchy, $\eta = (\mu = 2.37, \sigma^2 = 0.05, \tau^2 = 1.73, p = 0.007)$. Equivalently expressed spots are coloured grey, differentially expressed spots are black. Note the symmetry in the simulation.



Using normal distributions in the middle and bottom layers of the hierarchy creates symmetry in observed simulations from the Log-Normal-Normal model. The mass of the log-intensities lies in the centre of the plots in Figure 4.5. This is not likely to be the case in real microarray datasets, as marginal log-intensities from either intensity channel are likely to be right skewed.

4.4.3 Gamma-Uniform-Bernoulli model

This model is based upon a similar approach by Newton et al. [38]. In the Gamma-Uniform-Bernoulli model, the spot intensities are sampled from a $Gamma(a, \theta)$ distribution. The target mean intensities are derived from uniform distributions on an M versus A transformed scale [10]. The difference in log expression between the two treatments, M, equals 0 in the target mean intensity layer under equivelent expression, while under differential expression, M is sampled from a $U(-3, 3)$ distribution. A is a measurement of abundance of the combined intensity channels, under both equivalent expression and differential expression it is sampled from a $U(-3, 8)$,

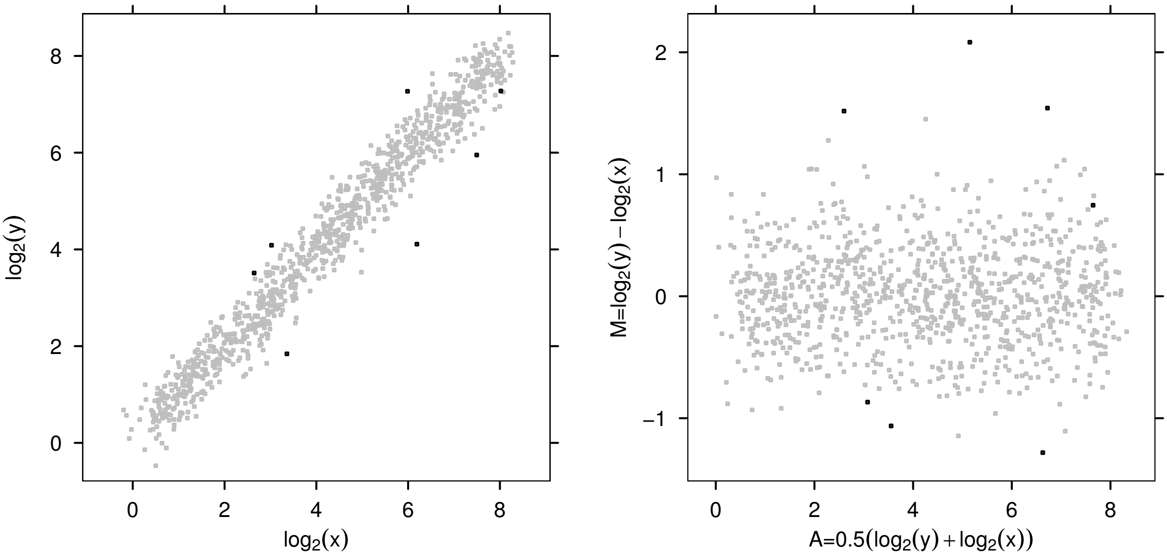
$$\begin{aligned}
&X_i \sim \Gamma(a, 2^{(A-M/2)}), \quad Y_i \sim \Gamma(a, 2^{(A+M/2)}) \\
&\text{Equivalent expression } (\mathcal{EE}) : A \sim U(-8, 3), \quad M = 0 \\
&\text{Differential expression}(\mathcal{DE}) : A \sim U(-8, 3), \quad M \sim U(-2.5, 2.5). \qquad (4.17)
\end{aligned}$$

Newton et al. (2003) [38] chose the uniform layer modelling the mean target intensity layer because it roughly approximates relationships in observed microarray examples.

Figure 4.6 provides a simulation example with the bottom layer Gamma parameter $\eta = (a = 12.53, p = 0.007)$. These are based on the IPTG-a *E. coli* experiment [16] fitting a Log-Normal-Normal Empirical Bayes model to the dataset. [37]. Note the uniform pattern of equivalently expressed spots in the MA-plot of Figure 4.6, which translates up the diagonal of the $log_2(y)$ versus $log_2(x)$ plot.

The ranges of the uniform distributions in the target mean intensity layer are different from the ranges of the observed intensities on the MA scale. The Gamma sampling in the bottom layer shifts the intensities upwards, as seen in the range of observed average intensity A, on the x-axis of the MA plot (Figure 4.6).

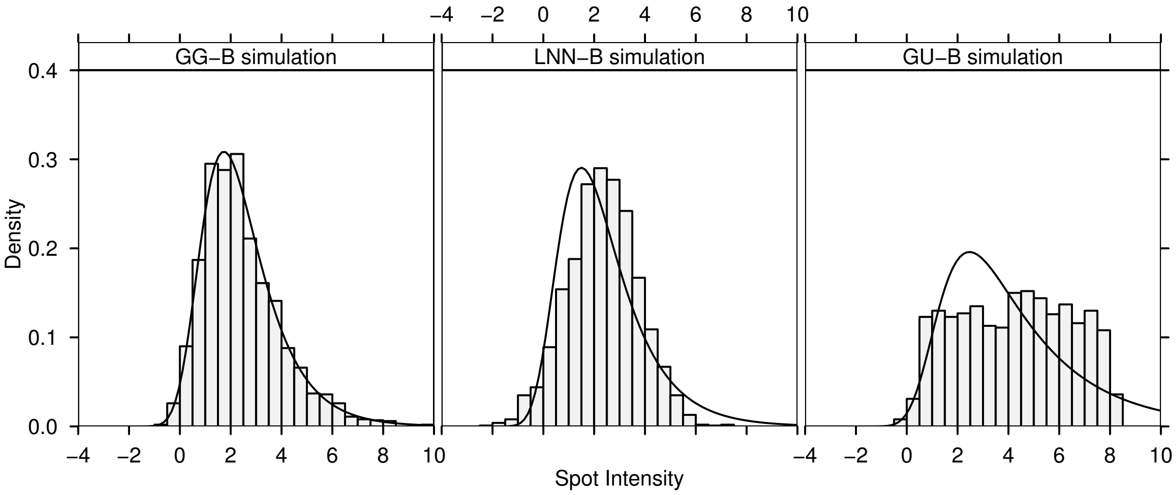
Figure 4.6: Simulated dataset (m=1000 genes) from a Gamma-Uniform-Bernoulli Hierarchy, $\eta = (a = 12.53, p = 0.007)$. The mean intensity layer is sampled from the uniform distribution on the MA log-intensity scale. Equivalent expression $(\mathcal{EE}) : A \sim U(-8, 3),\ M = 0$. Differential expression $(\mathcal{DE}) : A \sim U(-8, 3),\ M \sim U(-2.5, 2.5)$. Equivalently expressed spots are coloured grey, differentially expressed spots are black.



4.4.4 Model validation

Newton et al. [36] test model validation by plotting a histogram of log-intensities for the marginal data obtained from each channel. Using the parameters obtained by the *EM* algorithm, a Gamma-Gamma-Bernoulli log likelihood fit from the Empirical Bayes analysis was superimposed over the histogram to identify serious departures of the model from the observed data. Figure 4.7 provides a typical example of one dataset simulated from each of the scenarios GG-B, LNN-B, and GU-B. Superimposed is the Gamma-Gamma-Bernoulli log likelihood fit to the simulated dataset. The Gamma-Gamma-Bernoulli *EBarrays* model is an adequate fit to the GG-B, and LNN-B simulated data. Data obtained under GU-B simulation displays significant departures from the Gamma-Gamma-Bernoulli model fit.

Figure 4.7: Diagnostic validation plot. Histograms are marginal log-intensities for a single simulated dataset under GG-B, LNN-B, and GU-B scenarios. Solid line is a log-likelihood fit from Empirical Bayes using the Gamma-Gamma-Bernoulli model.



5

Simulation of Microarray Models

5.1 Simulation study

A simulation study was conducted to evaluate the performance of microarray analysis using the parametric Empirical Bayes methodology (denoted *EBarrays*) of chapter 4, against a standard frequentist multiple hypothesis t-test methodology (denoted *Multtest*). These analysis methods are available as add-on package libraries in **R** [18] also named *EBarrays*[1], and *multtest*[2]. Three simulation scenarios were chosen to compare the performance of these modelling methodologies. The parameters chosen for each scenario were based on actual observed Empirical Bayes parameter estimates for the IPTG-a microarray experiment [16] of Newton et al. [36].

I. GG-B scenario: simulating normalized intensities under a Gamma-Gamma-Bernoulli hierarchy (Section 4.4.1) with Gamma distribution parameters $\eta = (a = 12.53, a_0 = 0.82, \nu = 0.37)$ over the range $\pi_0 \in (0,1)$, where π_0 governs the proportion of equivalently expressed genes simulated.

II. LNN-B scenario: simulating normalized intensities under a Log-Normal-Normal-Bernoulli hierarchy (Section 4.4.2) with Normal distribution parameters $\eta = (\mu = 2.37, \sigma^2 = 0.05, \tau^2 = 1.73)$ over the range $\pi_0 \in (0,1)$.

[1] Authors: M. A. Newton & C. M. Kendziorski; http://www.biostat.wisc.edu/∼kendzior

[2] Authors: Y. Ge & S. Dudoit; version: 1.3.3, http://www.bioconductor.org

III. GU-B scenario: simulating normalized intensities under a Gamma-Uniform-Bernoulli hierarchy (Section 4.4.3) with Gamma distribution parameter ($a = 12.53$), and uniform on $-2.5 \leq M \leq 2.5$, $-8 \leq A \leq 3$ over the range $\pi_0 \in (0, 1)$.

Simulations in all 3 scenarios were conducted for $m = 1000$ genes, and $n_1 = n_2 = 5$ replicates per condition over the range $\pi_0 = (0, 0.005, 0.010, \ldots, 1)$, with 1000 microarray datasets generated for each level of π_0. First the underlying hypothesis was simulated for each gene, then suitable adaptive FDR controlling procedures were applied to the statistical analysis of simulated datasets. Sufficient statistics (V, T, m, π_0) were recorded to generate the observable random variables in Table 3.1. Based on this information, observed distributions of false discovery, false non-discovery, sensitivity, and specificity were examined.

The *EBarrays* model fitted a Gamma-Gamma-Bernoulli hierarchy to untransformed intensity data, with no experimental effects present. Prior to simulation it was expected that *EBarrays* would perform well under simulation scenario I, adequately under simulation scenario II, and poorly under simulation scenario III, which violates the parametric mean component.

The multiple hypothesis two sample Welch t-test approach required spot replicate information either within or between normalized microarrays to provide degrees of freedom for error estimation in each hypothesis test. Equal variance was assumed between the two treatment conditions. The two sided t-test was performed for each gene on log_2 transformed normalized intensities, assuming normality of sampled spot intensities, by generating a test statistic for each gene,

$$t_i = \frac{\bar{x}_{1i} + \bar{x}_{2i}}{\sqrt{\frac{\hat{\sigma}^2_{1i}}{n_1} + \frac{\hat{\sigma}^2_{2i}}{n_2}}}$$

Multiple hypothesis testing assumes independence of within gene t-tests, that is, no information sharing between genes. Both models provide predictions of differential expression for each gene; p-values under *Multtest*, and probabilities of differential expression under *EBarrays*.

Adaptive false discovery rate control [28, 29], was chosen as it guarantees that on average the OPFD is maintained at the constant level α^*, over $\pi_0 \in (\alpha^*, 1)$. *Multtest* analysis is naturally suited to non-adaptive FDR control which does not require estimation of π_0, therefore a comparison was made comparing and contrasting the methods *Multtest* and *EBarrays*, by maintaining adaptive FDR control assuming perfect knowledge of π_0, denoted *Multtest-P*.

In real microarray analysis situations the proportion of equivalently expressed genes is unknown, therefore adaptive control of the FDR requires an estimate of π_0. To facilitate calculation of $\alpha = \alpha^*/\hat{\pi}_0$, additional computation of $\hat{\pi}_0$ using natural splines, [34] was incorporated into the analysis (denoted *Multtest-E*), to provide adaptive control of the FDR at $\alpha = \alpha^*/\hat{\pi}_0(1)$. This allowed a further model comparison of *Multtest* (estimating π_0) to *EBarrays*.

Analysis under *EBarrays* used the method of Newton et al. [38] described in 4.3.1 to control the positive FDR [41] adaptively. This incorporates the Bayesian model estimate of π_0 into the process of adaptive positive FDR control. The statistical software package **R** [18] was used to evaluate the performance of the three models; *EBarrays*, *Multtest-P* (assuming perfect knowledge of π_0), and *Multtest-E* (estimating π_0). A large amount of optimization was required to obtain the speed necessary to run more than 200,000 datasets in each of the three data generation scenarios; GG-B, LNN-B, GU-B.

The t-test approach used the *mt.teststat* function available in the Bioconductor package *multtest* to compute t-test statistics for each row of simulated data. This function was chosen since it incorporates underlying C routines, thus providing a considerable gain in speed. The *EBarrays* EM algorithm was written by first principles using the *optim* function for the maximization step. This was considerably faster than using Newton and Kendziorski's **R** library [42] *EBarrays* for Empirical Bayes estimation. Using a Pentium 1.8GHz processor with the Linux Redhat 8.0 operating system, *Multtest* was able to analyze approximately 100 datasets per second, and *EBarrays* was able to analyze approximately two datasets per second. The EBarrays EM algorithm was limited to eight iterations, with a slight trade off in parameter estimation accuracy for greater processing speed (**R** code in Appendix A.6). Results are presented using *lattice*[3] graphics to display observed distribution responses versus π_0. Each plot is conditioned over the statistical method used and the 3 simulation scenarios (GG-B, LNN-B, GU-B).

5.2 π_0 estimation comparison

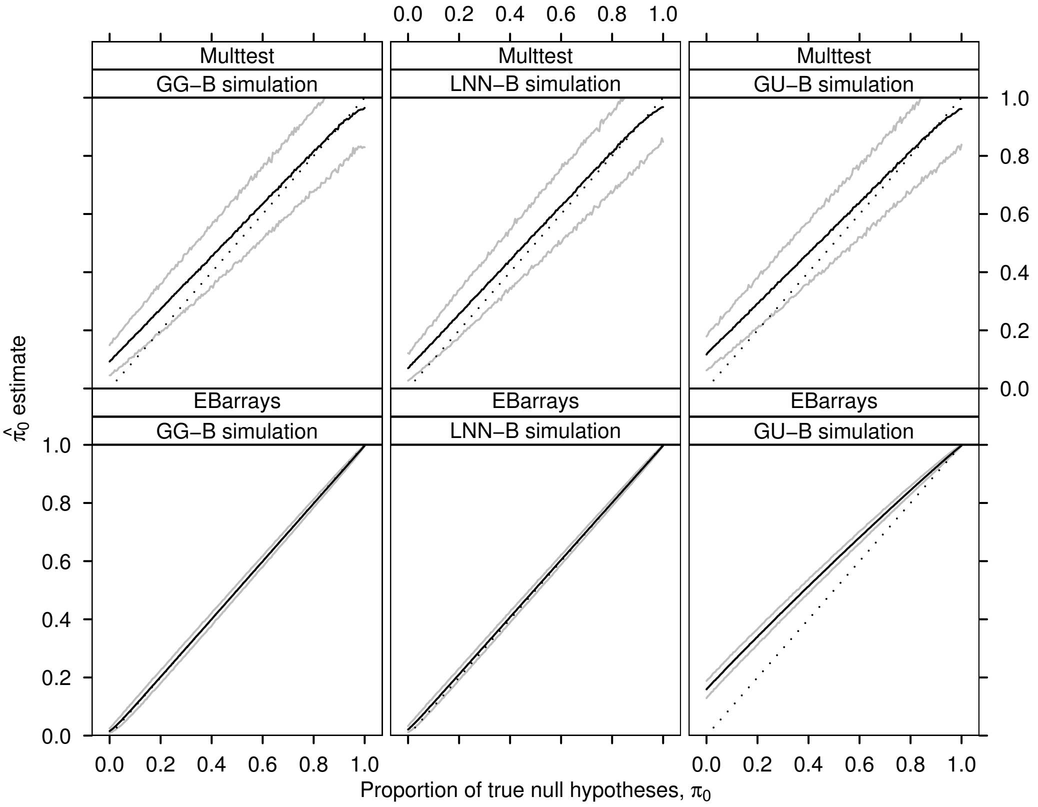
Simulation results for π_0 estimation comparing and contrasting *Multtest* with *EBarrays* are shown in Figure 5.1. The 2.5 and 97.5 percentiles of the observed distribution of $\hat{\pi}_0$ were calculated to illustrate observed confidence intervals at the 95% level. The 95% confidence intervals in the observed estimates of π_0 from the cubic spline procedure in *Multtest* were significantly more variable than *EBarrays* over the entire range of π_0 in all three simulation scenarios.

The EBarrays procedure provided a highly accurate estimate of π_0 under the simulation scenarios GG-B, and LNN-B, although a breakdown in fit is evident for the GU-B simulated data with an upward bias (deviation from the expected dotted line) in the estimation of π_0 as $\pi_0 \to 0$. This bias is caused by the uniform target mean intensity

[3] Implementation of Trellis Graphics, author: Deepayan Sarkar, http://cran.r-project.org

component on the M versus A scale under differential expression, which deviates from the Gamma assumption of the model. Note that when π_0 is close to 1, the estimate $\hat{\pi}_0$ is only slightly biased. The bias in the cubic spline estimate of π_0 from *Multtest* p-values increases between LNN-B, GG-B, and GU-B simulations, where LNN-B is most suited for satisfying the *Multtest* normality assumptions. In all of the *Multtest* analyses, as $\pi_0 \rightarrow 0$ there was a bias variance trade off, where the variance in the 95% confidence intervals decreases and the estimate becomes biased upwards.

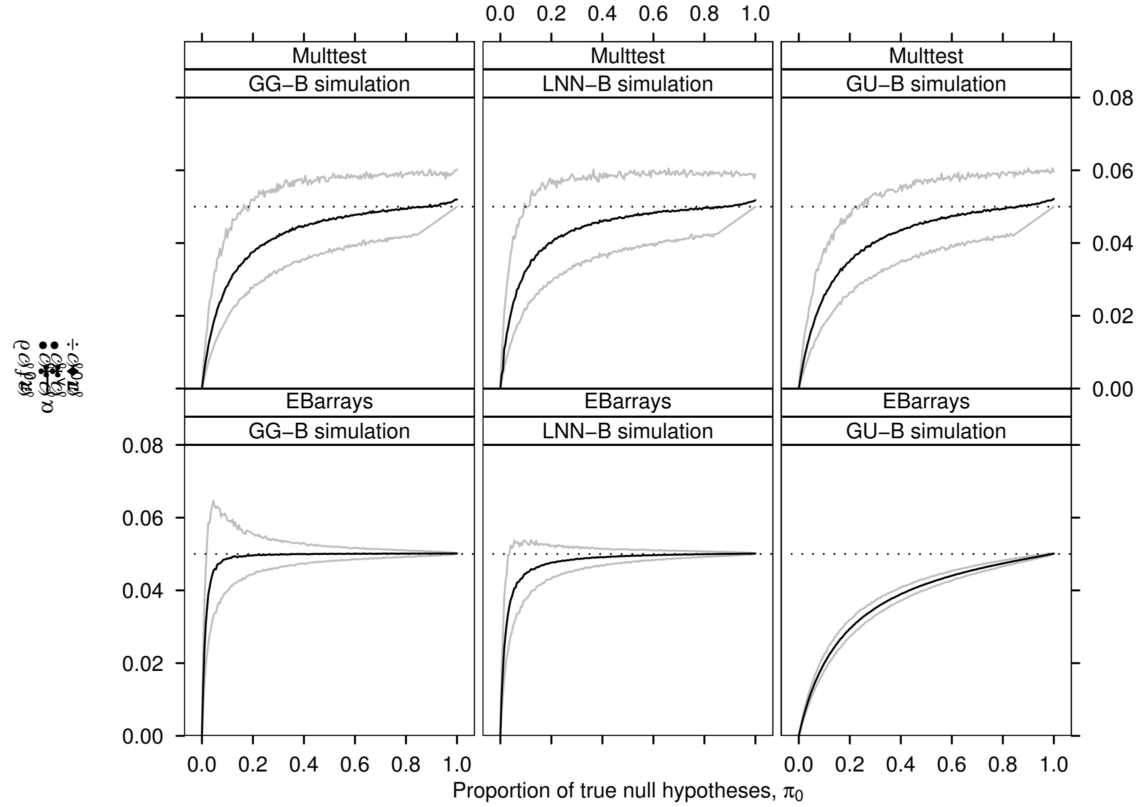
Figure 5.1: Comparison of observed $\hat{\pi}_0$ estimates versus π_0 between EBarrays and Multtest analysis procedures. Panels are conditioned over simulation scenarios GG-B, LNN-B, and GU-B. Black lines are the observed $E[\hat{\pi}_0]$, grey lines are 2.5 and 97.5 percentiles of $\hat{\pi}_0$. Dotted line depicts the expected estimate as a function of π_0.



Adaptive FDR control at $FDR = \alpha\pi_0$ was provided by setting $\alpha = \alpha^*/\hat{\pi}_0$. The quantity $\alpha(\pi_0/\hat{\pi}_0)$ should be constant in α over all values of π_0. Figure 5.2 illustrates the effectiveness of both modelling approaches aiming for constant control at $\alpha = 0.05$. The 95% confidence intervals in the observed estimates of π_0 from the cubic spline procedure in *Multtest* were significantly more variable than *EBarrays* over the entire range of π_0 in all three simulation scenarios. In all cases *Multtest* maintained poor control of the FDR at $\alpha = 0.05$. As $\pi_0 \rightarrow 0$ a large downward bias occured. The observed $E[\alpha(\pi_0/\hat{\pi}_0)]$ was worst under GU-B dataset generation. *EBarrays* under the GG-B and LNN-B simulation scenarios maintained constant control of the FDR at $\alpha = 0.05$ over most of the range of

π_0. Note that the variability in the 95% confidence intervals increased as $\pi_0 \to 0$. Under GU-B simulation, *EBarrays* maintained poor control of the FDR at $\alpha = 0.05$, at least as biased as *Multtest* for the observed $E[\alpha(\pi_0/\hat{\pi}_0)]$. Note that there was still high precision in π_0 estimation under *EBarrays* even though the observed estimates were biased.

Figure 5.2: Comparison of observed $\alpha(\pi_0/\hat{\pi}_0)$ versus π_0 between EBarrays versus Multtest analysis procedures. Panels are conditioned over simulation scenarios GG-B, LNN-B, and GU-B. Black lines are the observed $E[\alpha(\pi_0/\hat{\pi}_0)]$, grey lines are 2.5 and 97.5 percentiles of $\alpha(\pi_0/\hat{\pi}_0)$. Dotted line depicts expected constant control at α^*.



5.3 EBarrays versus Multtest

The nine plot panels in each of Figures 5.5 to 5.8 are presented with statistical procedures *EBarrays*, *Multtest-P* (assuming perfect knowledge of π_0), and *Multtest-E* (estimating π_0) presented across plot rows, and the three simulation scenarios (GG-B, LNN-B, GU-B) down plot columns. Observed distribution results for *Multtest-E*, are more variable than *Multtest-P*, due to the associated variability added.

As $\pi_0 \to 0$, the number of differentially expressing genes, m_1, converges to m (all genes differentially expressing), and as $\pi_0 \to 1$, the number of equivalently expressed genes, m_0, converges to m (no genes differentially expressing). In microarray experiments, the number of differentially expressing genes is usually a small proportion of the total number of genes represented on the array. For this reason the region as $\pi_0 \to 1$ is of particular interest in the observed distributions.

Sensitivity and Specificity

Observed measures of model power, sensitivity, and specificity are presented in Figures 5.3, and 5.4. Sensitivity is related to statistical power, $1 - \beta$, and is defined as the proportion of correctly identified differentially expressing genes,

$$\text{Sens} = \frac{S}{m - m_0}.$$

Specificity is related to $1 - \alpha$, and is defined as the proportion of correctly identified equivalently expressed genes,

$$\text{Spec} = \frac{U}{m_0}.$$

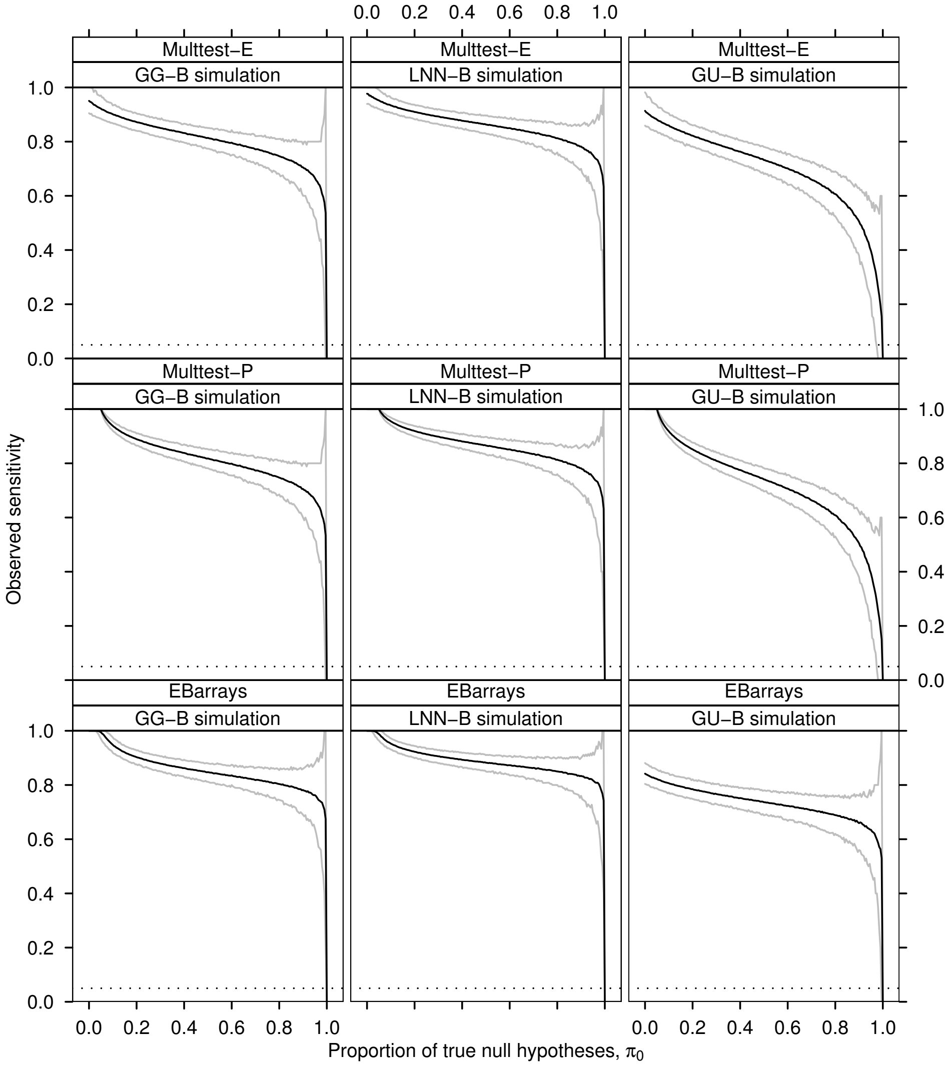
The random variables S, U, and m_0 are observable values s, u and m_0 in each simulation scenario. Figure 5.3 shows the observed sensitivity comparisons of *Multtest* with *EBarrays*. The sensitivity decreases sequentially between the simulations scenarios LNN-B, GG-B, and GU-B, for fixed π_0. Within each plot, as π_0 increases the observed E[Sens] decreases, and the width of the 95% confidence intervals increases. In the LNN-B and GG-B simulation scenarios, the observed E[Sens] is greater for *EBarrays* than *Multtest* over the entire range of π_0. Under GG-B simulation, *EBarrays* outperforms *Multtest* as $\pi_0 \to 1$, while the reverse is the case as $\pi_0 \to 0$.

In Figure 5.4, the specificity is similar between *EBarrays* and *Multtest-P* under the GG-B, and LNN-B microarray simulation scenarios. As π_0 increases, the specificity converges to 1. The specificity is worst in the GU-B simulation scenario analysed using *EBarrays*, where at $\pi_0 = 0$, the observed average specificity is still about 0.9. Note the large variability in 95% confidence intervals under *Multtest-E* as $\pi_0 \to 0$.

5.4 Adaptive FDR comparison

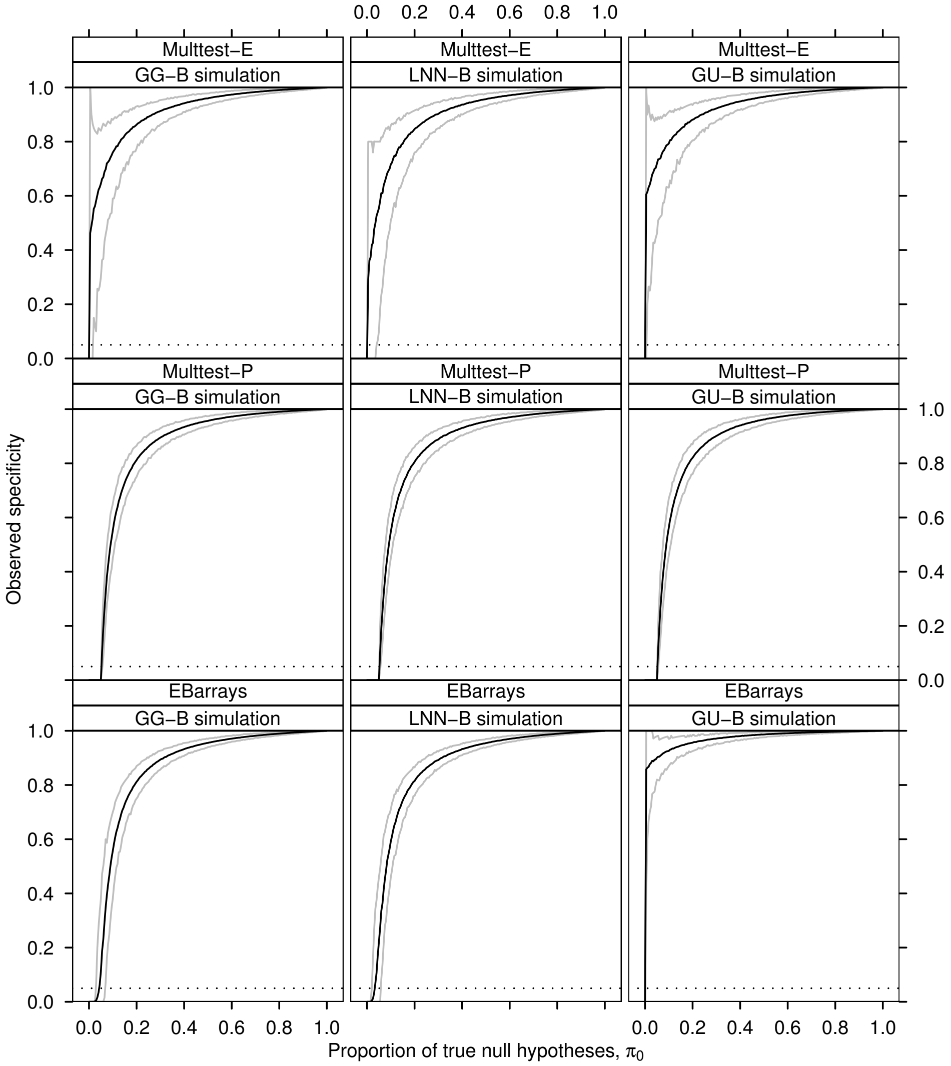
In Figure 5.5, the observed proportion of false discoveries (OPFD) distribution shows that analysis using *EBarrays* controls the FDR with less variability than *Multtest* over almost the entire range of π_0, as illustrated in the widths of the 95% confidence intervals. The differences between the models are most apparent as $\pi_0 \to 1$. In the *EBarrays* model the observed $E[OPFD] \to 0$, whereas with *Multtest* the observed $E[OPFD]$ is

Figure 5.3: Observed sensitivity versus π_0 for the analysis methods *EBarrays*, *Multtest-P* (perfect knowledge of π_0), *Multtest-E* (estimating π_0). Panels are conditioned by the simulation scenario used; GG-B, LNN-B, GU-B. Black lines are the observed E[Sens], grey lines are 2.5 and 97.5 percentiles of the observed Sensitivity. Dotted lines depicts FDR control at $\alpha^* = 0.05$.



still controlled at $\alpha^* = 0.05$. Using *EBarrays* analysis, the observed maximum of the observed 97.5 percentile of GG-B and LNN-B simulations is 0.2 when $\pi_0 = 0.995$, and under GU-B simulation is 0.125 when $\pi_0 = 0.99$. The observed 97.5 percentile is 1 under all simulations using *Multtest*, as $\pi_0 \rightarrow 1$. The observed $E[OPFD]$ is controlled at $\alpha^* = 0.05$ relatively well using *Multtest* under all simulation scenarios. The $E[OPFD]$ is controlled at $\alpha^* < 0.05$ using *EBarrays* under the GU-B simulation scenario, with the lack of fit in the model making the *EBarrays* FDR threshold appear more conservative. In the GG-B and LNN-B simulations, as $\pi_0 \rightarrow 0.05$ there is a small region (under high amounts of simulated differential expression) where the 95% confidence intervals of the *Multtest* are smaller than under *EBarrays*. In this region the upper bound of the 95%

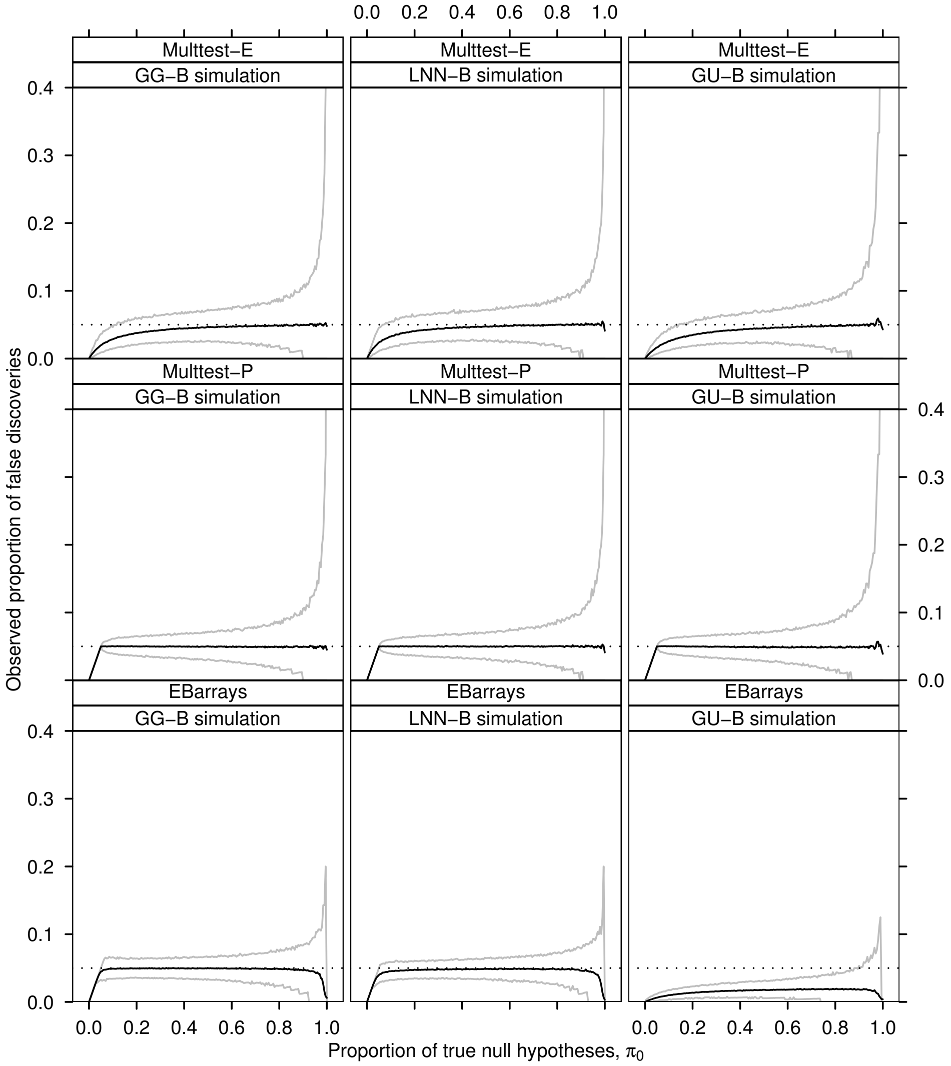
Figure 5.4: Observed specificity versus π_0 for the analysis methods *EBarrays*, *Multtest-P* (perfect knowledge of π_0), *Multtest-E* (estimating π_0). Panels are conditioned by the simulation scenario used; GG-B, LNN-B, GU-B. Black lines are the observed E[Spec], grey lines are 2.5 and 97.5 percentiles of the observed Specificity. Dotted lines depicts FDR control at $\alpha^* = 0.05$.



confidence is largest in *EBarrays* when $\pi_0 = 0.0495$. *Multtest-E* also controls the FDR at less than $\alpha = 0.05$ as $\pi_0 \rightarrow 0$, due to the upward bias in the natural cubic spline estimate of $\hat{\pi}_0$.

The observed proportion of true discoveries (OPTD) is related to the power exhibited by each statistical procedure (Figure 5.6), where TDR $= P(R > 0) -$ FDR in 3.4. The *EBarrays* model exhibits a significant increase in E(OPTD) as $\pi_0 \rightarrow 1$, illustrated by the spike in the mean observed proportion of true discoveries. This is the area of specific interest in microarray experiments, as only a small proportion of genes differentially express. As $\pi_0 \rightarrow 0$, *Multtest-E* analysis displays upward curvature in the E(OPTD), due to the bias in $\hat{\pi}_0$ estimation. Generally there is greater variability in the OPTD using

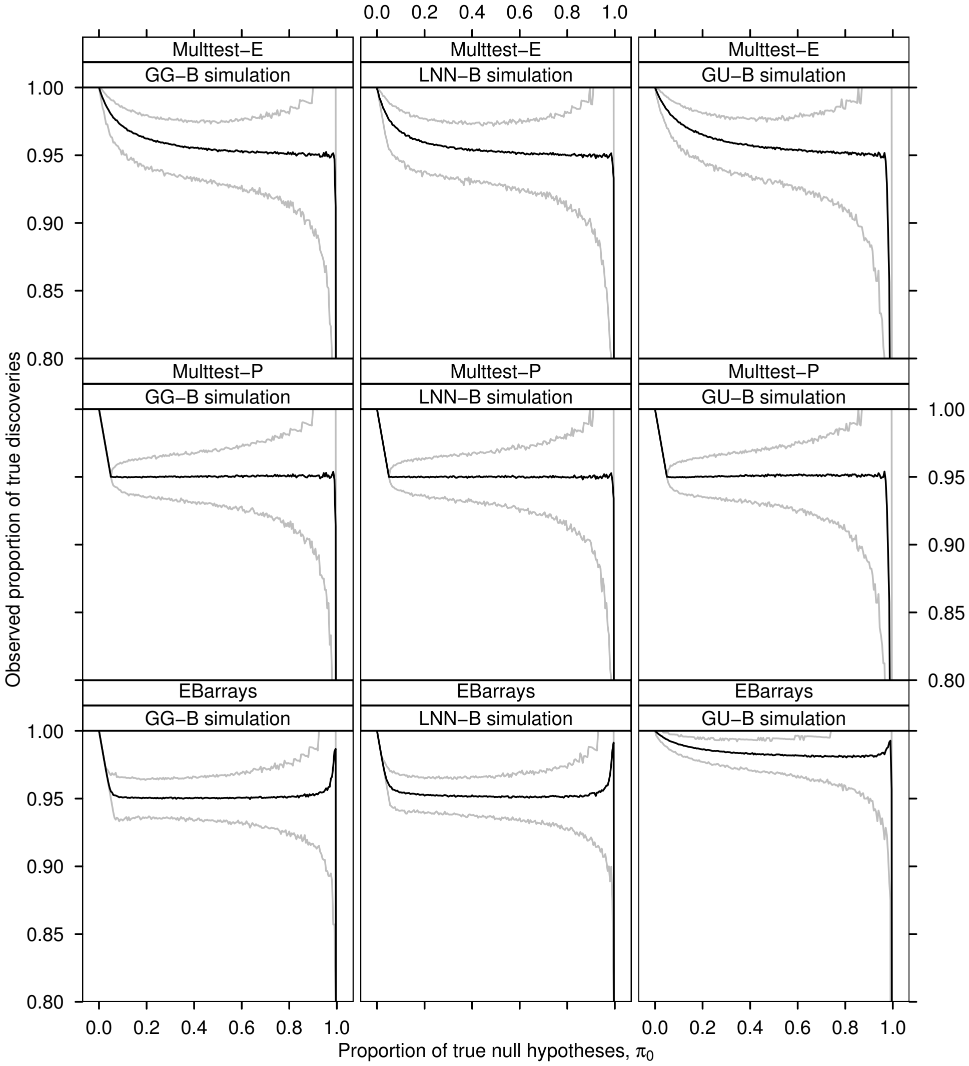
Figure 5.5: OPFD versus π_0 for the analysis methods *EBarrays*, *Multtest-P* (perfect knowledge of π_0), *Multtest-E* (estimating π_0). Panels are conditioned by the simulation scenario used; GG-B, LNN-B, GU-B. Black lines are the observed E[OPFD], grey lines are 2.5 and 97.5 percentiles of the OPFD. Dotted lines depicts FDR control at $\alpha^* = 0.05$.



Multtest-P over almost the entire range of π_0. This is most significant when $\pi_0 = 1$. The E(OPTD) is controlled at $1 - \alpha^* = 0.95$ relatively well using *Multtest* over all simulation scenarios. The E(OPTD) is controlled at $1 - \alpha^* > 0.95$ using *EBarrays* under the GU-B simulation scenario, due to the lack of fit caused by the Uniform target mean intensity component on the M versus A scale under differential expression deviating from the Gamma model fit.

In the simulations examined, when all simulated genes are equivalently expressed the probability that the observed proportion of false discoveries equal to 1 increases dramatically. The observed probabilities are presented in Table 5.1. Observed $P(\text{FDR} = 1)$ is 5 times larger using *Multtest* than *EBarrays*, with at least 4% of observed simulations pre-

Figure 5.6: OPTD versus π_0 for the analysis methods *EBarrays*, *Multtest-P* (perfect knowledge of π_0), *Multtest-E* (estimating π_0). Panels are conditioned by the simulation scenario used; GG-B, LNN-B, GU-B. Black lines are the observed E[OPTD], grey lines are 2.5 and 97.5 percentiles of the OPTD.



dicting differentially expressing gene lists that were completely incorrect using the t-test procedure across all simulation scenarios.

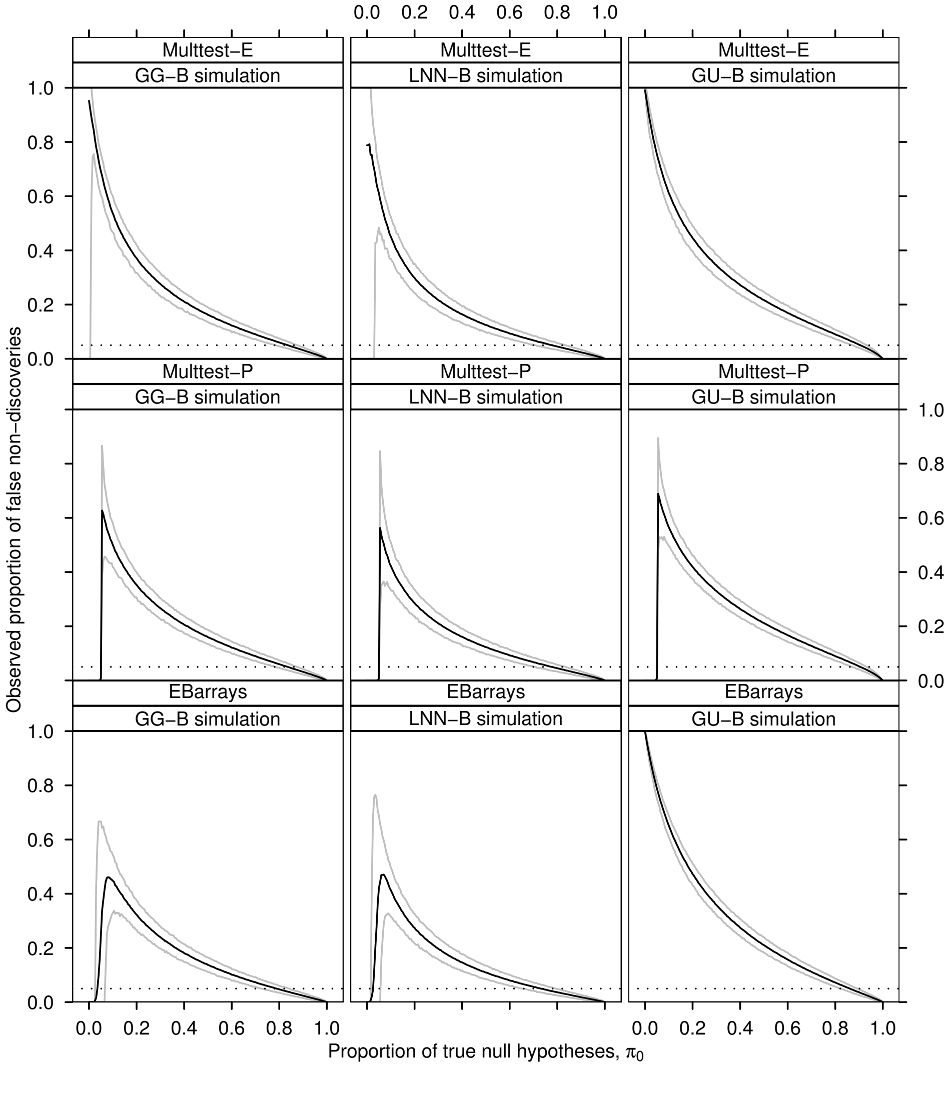
The observed proportion of false non-discoveries (OPFN) generally increases as $\pi_0 \to 0$ (Figure 5.7). In GG-B and LNN-B simulation scenarios, analysis using *EBarrays* and *Multtest* with perfect knowledge of π_0 increases to a maximum at $\pi_0 = 0.05$, with the 95% confidence interval exhibiting most variability at this point. In the GU-B simulation, under *EBarrays* analysis the fit of the model causes the observed E[OPFN] $\to 1$ as $\pi_0 \to 0$. This is also observed under all simulation scenarios using *Multtest* analysis estimating π_0.

Figure 5.8 illustrates the number of hypotheses rejected given that they were differ-

Table 5.1: Probability that the observed proportion of false discoveries equals 1 when $\pi_0 = 1$.

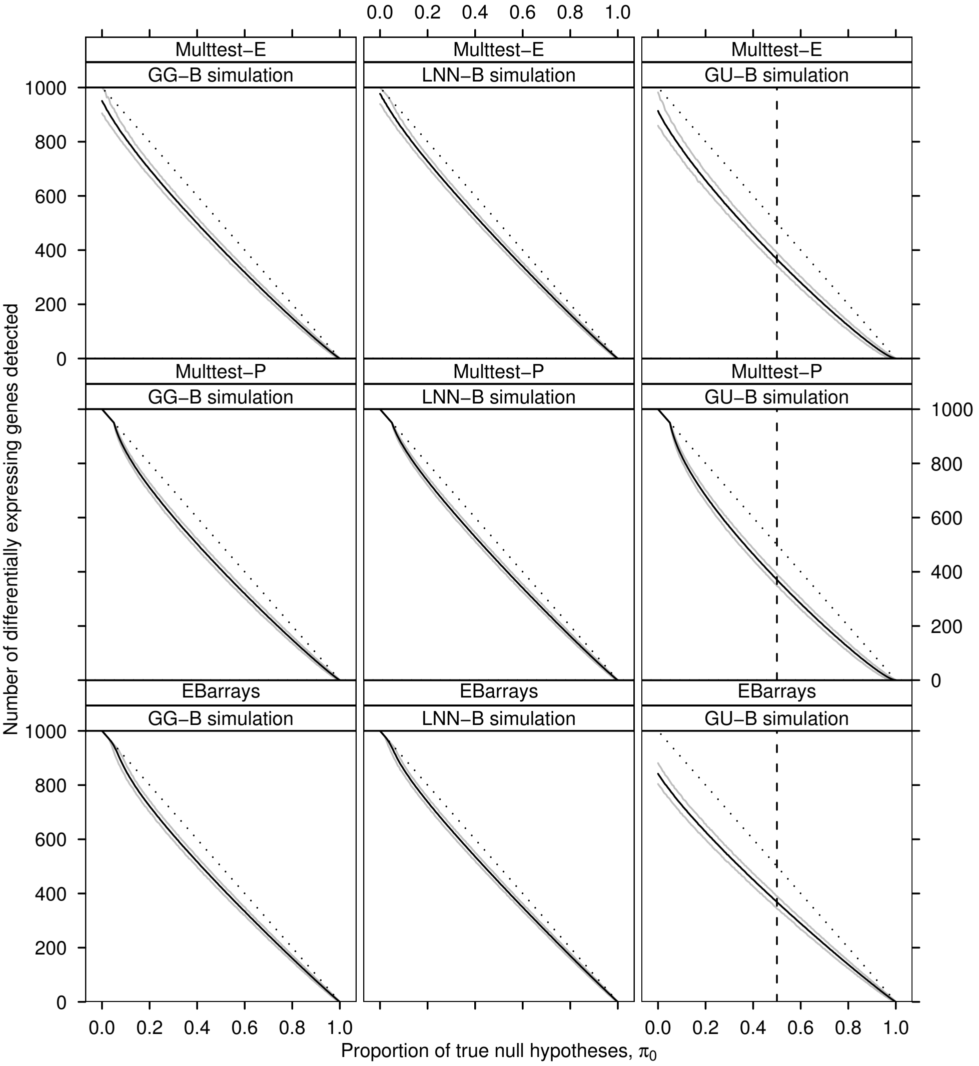
	GG-B	LNN-B	GU-B
Multtest Estimating π_0	0.048	0.041	0.043
Multtest Perfect Control	0.045	0.041	0.039
EBarrays	0.006	0.003	0.004

Figure 5.7: OPFN versus π_0 for the analysis methods *EBarrays*, *Multtest-P* (perfect knowledge of π_0), *Multtest-E* (estimating π_0). Panels are conditioned by the simulation scenario used; GG-B, LNN-B, GU-B. Black lines are the observed E[OPFN], grey lines are 2.5 and 97.5 percentiles of the OPFN. Dotted lines depicts FDR control at $\alpha^* = 0.05$.



entially expressing genes. Under GG-B and LNN-B simulation scenarios, the *EBarrays* procedure detects a slightly larger number of differentially expressed genes across the entire range of π_0. Under GU-B simulation, the *EBarrays* procedure outperforms both forms of *Multtest* adaptive control when $\pi_0 > 0.5$, the reverse is the case when $\pi_0 < 0.5$.

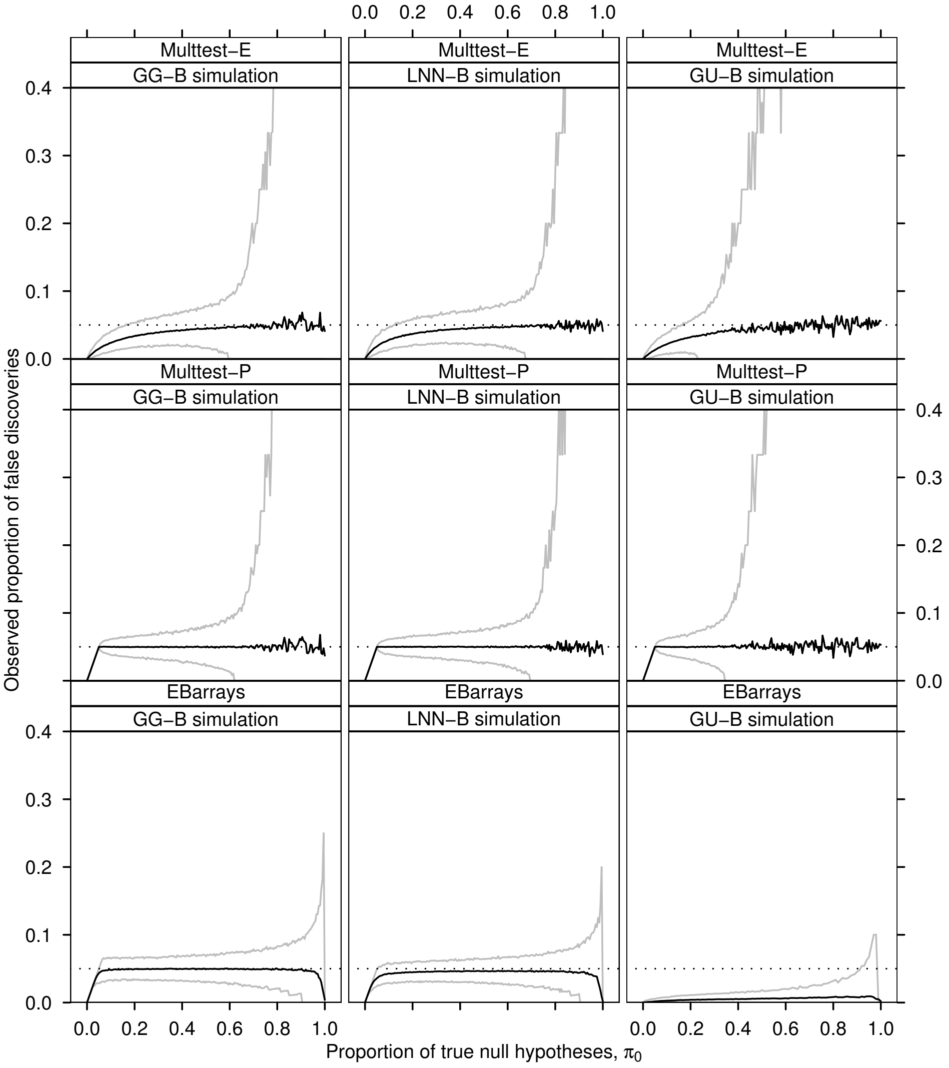
Figure 5.8: Observed number of differentially expressed genes reported versus π_0 for the analysis methods *EBarrays*, *Multtest-P* (perfect knowledge of π_0), *Multtest-E* (estimating π_0). Panels are conditioned by the simulation scenario used; GG-B, LNN-B, GU-B. Black lines are the observed $E[S]$, grey lines are 2.5 and 97.5 percentiles. Dotted line depicts the number of expected differentially expressed genes as a function of π_0. Dashed line is a threshold at $\pi_0 = 0.5$, *EBarrays* predicts more differentially expressed genes as π_0 increases above this threshold.



5.4.1 Minimal replication

The simulation scenario described in Section 5.1, uses $n_1 = n_2 = 5$ replicate spots per condition. Consider minimal spot replication as being the case where $n_1 = n_2 = 2$ for all spots on each array. A question of interest is, "What effect does minimal replication of spots have on false discovery rate control between the analysis methods *EBarrays* and *Multtest*?". The simulations in Section 5.1 were repeated using $m = 1000$ genes, and $n_1 = n_2 = 2$ replicates per condition over the range $\pi_0 = (0, 0.005, 0.010, \ldots, 1)$ to examine this scenario. Observed distributions that significantly differ from those presented in Sections 5.3 and 5.4 are presented here. Figures 5.9 to 5.11 examine the effect of minimal replication under *EBarrays* and *Multtest*.

Figure 5.9: Two replicate spots: OPFD versus π_0 for the analysis methods *EBarrays*, *Multtest-P* (perfect knowledge of π_0), *Multtest-E* (estimating π_0). Panels are conditioned by the simulation scenario used; GG-B, LNN-B, GU-B. Black lines are the observed E[OPFD], grey lines are 2.5 and 97.5 percentiles of the OPFD. Dotted lines depict FDR control at $\alpha^* = 0.05$.



The observed proportion of false discoveries (OPFD) distribution (Figure 5.9) shows significantly greater variability using *Multtest* than with *EBarrays*. Under GG-B and LNN-B simulation scenarios, when $\pi_0 > 0.7$, the 95% confidence intervals are extremely variable using *Multtest* for either form of adaptive FDR control. Under the GU-B simulation scenario, when $\pi_0 > 0.4$ the 95% confidence intervals in the OPFD are even more variable using *Multtest* for either form of adaptive FDR control, assuming perfect knowledge of π_0, or estimation of $\hat{\pi}_0$. The instability in E[OPFD] under *Multtest* as $\pi_0 \to 1$ is caused by OPFD to bouncing between 0, and a high proportion of false discoveries as $\pi_0 \to 1$.

In Figure 5.10, the OPTD is now extremely poor under both forms of *Multtest* adaptive control. The E[OPTD] decreases quickly at the same thresholds of increasing variability in the 95% confidence intervals of Figure 5.9. As π_0 increases the observed sensitivity (Figure 5.11) drops away rapidly under *Multtest* for either form of adaptive FDR control. The sensitivity decreases between LNN-B, GG-B, and GU-B, for fixed π_0 across all simulation scenarios. There is still a threshold on π_0, where *Multtest* outperforms *EBarrays* as $\pi_0 \to 0$.

Figure 5.12 illustrates the number of hypotheses rejected given that they were differentially expressing genes for $n_1 = n_2 = 2$ replicates. Under GG-B and LNN-B simulation scenarios, the *EBarrays* procedure detects significantly higher differentially expressed genes across the entire range of π_0. Under GU-B simulation, the *EBarrays* procedure significantly outperforms both forms of *Multtest* adaptive control when $\pi_0 > 0.2$, the reverse is the case when $\pi_0 < 0.2$.

When there is minimal replication of spots, the amount of information available to make predictions of genes undergoing differential expression decreases substantially. Figure 5.11 shows that *EBarrays* maintains higher sensitivity than *Multtest* by taking advantage of between gene information sharing in variance estimation. Sensitivity was drastically reduced in *Multtest* analysis compared to when the number of spot replicates $n_1 = n_2 = 5$. Under *Multtest* analysis, as π_0 increases, there is an observed threshold on π_0 in each simulation scenario where the observed sensitivity drops below α^*, the level of FDR control. When this happens the OPFD distribution becomes extremely variable (Figure 5.9), and the E[OPFD] in Figure 5.9 displays increased variability. Under *Multtest-P* analysis, sensitivity below α^* affects the E[OPTD] in Figure 5.10, which starts to drop away from the constant control at $1 - \alpha^*$ maintained for small values of π_0.

Figure 5.10: Two replicate spots: OPTD versus π_0 for the analysis methods *EBarrays*, *Multtest-P* (perfect knowledge of π_0), *Multtest-E* (estimating π_0). Panels are conditioned by the simulation scenario used; GG-B, LNN-B, GU-B. Black lines are the observed $E[\text{OPTD}]$, grey lines are 2.5 and 97.5 percentiles of the OPTD.

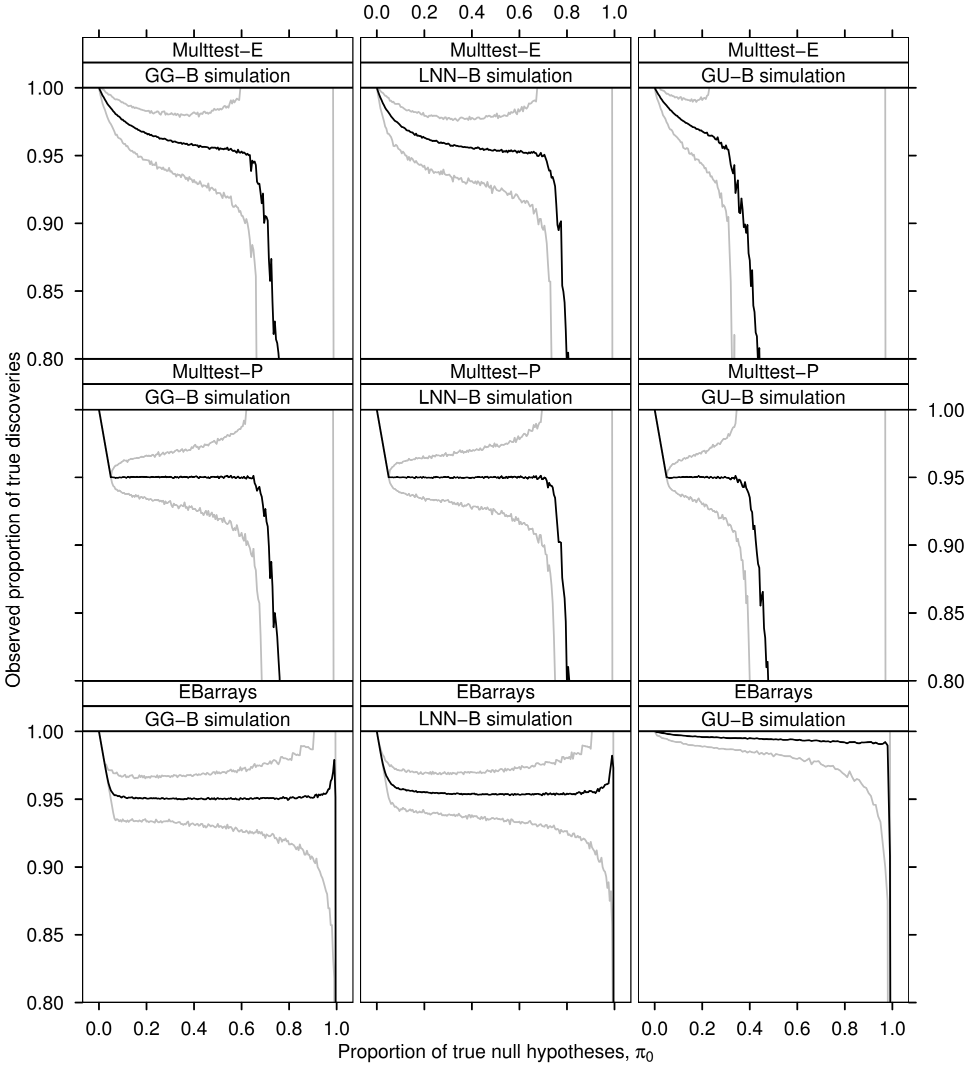


Figure 5.11: Two replicate spots: Observed sensitivity versus π_0 for the analysis methods *EBarrays*, *Multtest-P* (perfect knowledge of π_0), *Multtest-E* (estimating π_0). Panels are conditioned by the simulation scenario used; GG-B, LNN-B, GU-B. Black line is the observed $E[\text{Sens}]$, grey lines are 2.5 and 97.5 percentiles of the observed Sensitivity. Dotted lines depicts FDR control at $\alpha^* = 0.05$.

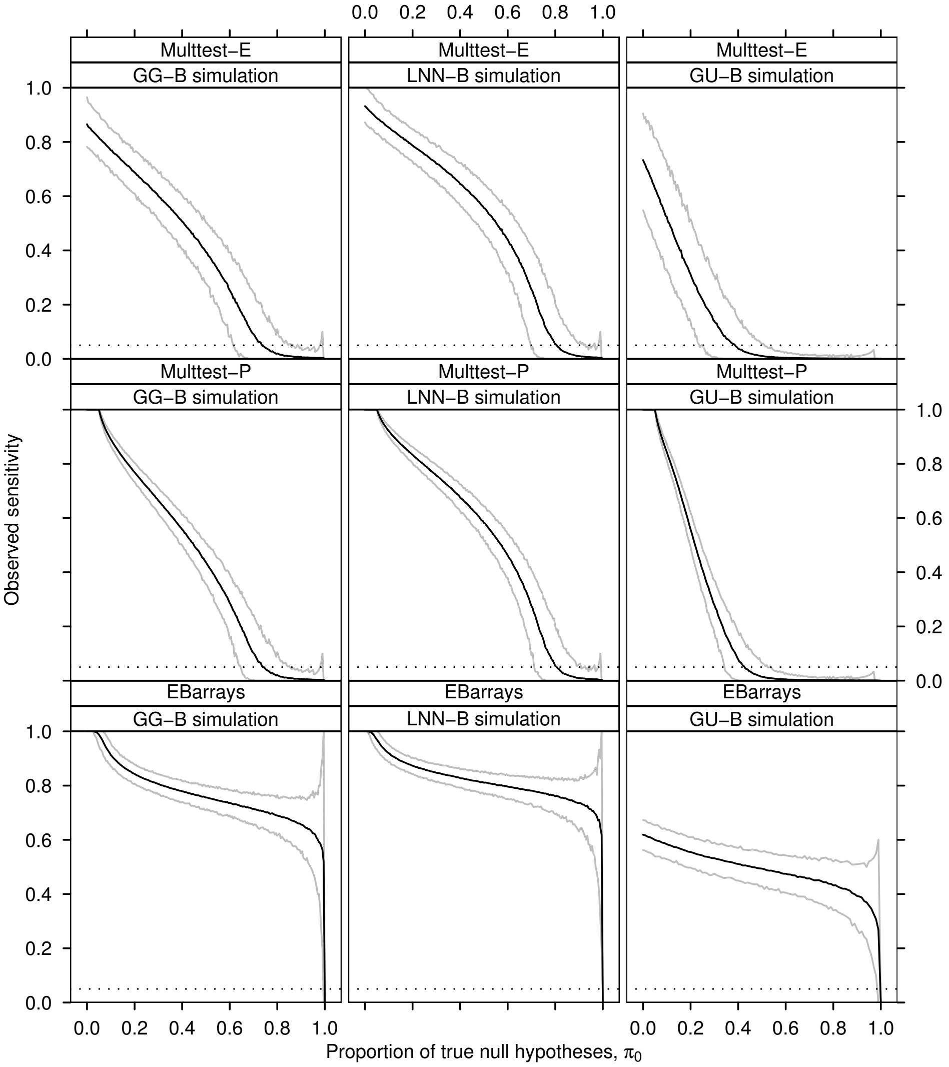
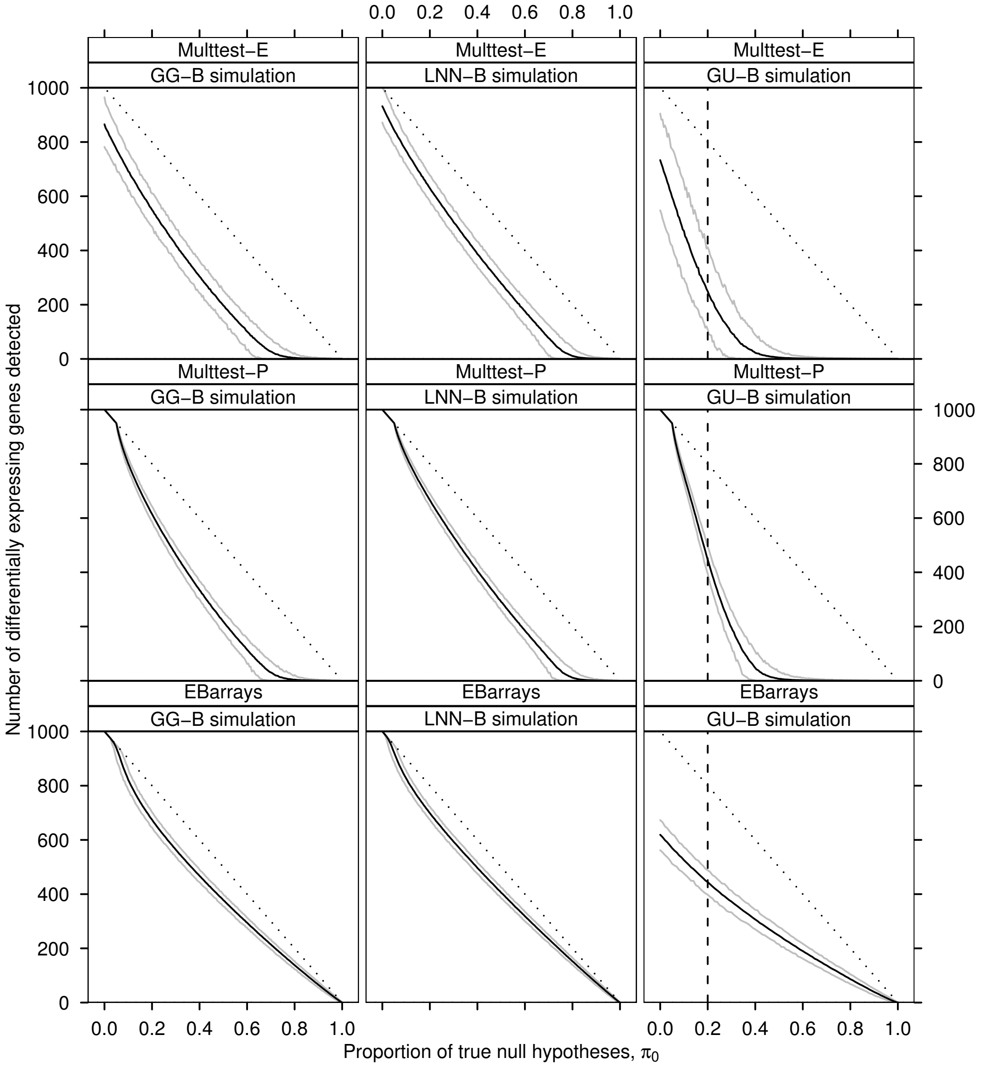


Figure 5.12: Two replicate spots: Observed number of differentially expressed genes reported versus π_0 for the analysis methods *EBarrays*, *Multtest-P* (perfect knowledge of π_0), *Multtest-E* (estimating π_0). Panels are conditioned by the simulation scenario used; GG-B, LNN-B, GU-B. Black lines are the observed $E[S]$, grey lines are 2.5 and 97.5 percentiles of the observed $E[S]$. Dotted line depicts the number of expected differentially expressed genes as a function of π_0. Dashed line is a threshold at $\pi_0 = 0.2$, *EBarrays* predicts more differentially expressed genes as π_0 increases above this threshold.



What you get out depends on what you put in; and as the grandest mill in the world will not extract wheat-flour from peascods, so pages of formulae will not get a definite result out of loose data.
-Thomas Henry Huxley, biologist and writer (1825-1895)

6 Discussion

The main focus of this work was to investigate general characteristics of false discovery rate controlling procedures in the context of microarray experimentation. Distributional properties of the observed proportion of false discoveries were critically examined for the microarray analysis approaches *EBarrays* and *Multtest* using a comparable FDR controlling procedure.

6.1 Results

In chapter 3, the multiple comparison step-up procedure of Benjamini and Hochberg [21] was used to apply non-adaptive [21] and adaptive [28, 29] FDR control to simulated mixtures of normal distributions generated as a function of the mixing proportion π_0. Under non-adaptive FDR control, the $E[\text{OPFD}] = \alpha\pi_0$. This was in agreement with the result of Finner and Roters [27], illustrating that the OPFD is dependant on the magnitude of π_0. Under adaptive control, the $E[\text{OPFD}]$ maintained constant control at α^* for values of π_0 ranging between α^* and 1. In both multiple comparison step-up procedures controlling the FDR, increased variability was observed in the OPFD as π_0 increased in magnitude (Figures 3.3, 3.6). As the desired level of control α^* was increased, or μ (the location of the alternative hypothesis) decreased, the OPFD increased in variability. In Figures 3.4, and 3.7, the $E[\text{OPTD}]$ decreased as $\pi_0 \to 1$. This was most evident for $\mu = 1$, in the alternative hypothesis, when the sensitivity in hypothesis testing was lowest. The ratio of the confidence interval size between perfect adaptive control, and non-adaptive FDR control (Figure 3.2) showed an interesting relationship between adaptive and non-adaptive variability. The observed ratio was largest at $\pi_0 = 0.1$, estimated to be 2.25 from the natural spline fit. The ratio decreased as expected to 1, as $\pi_0 \to 1$ exhibiting smooth curvature. The underlying relationship of variation between adaptive and non-adaptive control for fixed π_0 requires further examination.

The Empirical Bayes approach to modelling microarray datasets [36, 37, 38] was described in chapter 4. In Section 4.2.5 it was shown that some care is required when using the EM algorithm for *EBarrays* analysis. Further investigation is required to examine the properties effecting parameter estimation when using *fixed initial values* in the maximization step of the *EM* algorithm.

Simulations of microarray experiments in which no experimental effects were present were generated in chapter 5 using GG-B, LNN-B, and GU-B simulation models. These were chosen to examine the strengths and weaknesses when using *EBarrays* and *Multtest* approaches analyzing microarray data. Adaptive control of the false discovery rate was used to compare both analysis methods directly over the entire range of π_0. *Multtest* analysis was controlled under two different scenarios; perfect knowledge of π_0, and estimation of π_0. By comparing *Multtest* using adaptive error rate control (with perfect knowledge of π_0), to *EBarrays*, the performance of these two approaches was able to be contrasted. Using the natural spline estimation procedure of Storey & Tibshirani (2003) to estimate π_0, the real world situation of adaptive FDR control in *Multtest* analysis was investigated. Higher variability was observed in this estimation procedure than the estimates of π_0 using *EBarrays.* A significant bias was also observed as π_0 decreased in magnitude (Figures 5.1, 5.2). The increased variability and bias (as π_0 decreased) were evident in the observed characteristics of simulated distributions using $\hat{\pi}_0$ estimates for adaptive control in *Multtest* analysis.

The results in chapter 5 provide strong evidence that *EBarrays* is a powerful modelling approach for analyzing microarrays, and is ideally suited to controlling the adaptive false discovery rate. In simulations where the number of spot replicates $n_1 = n_2 = 5$, the observed sensitivity comparisons between methods were similar (Figure 5.3), likewise the number of genes rejected given truly differentially expressed were similar (Figure 5.8). The main differences were in the OPFD and OPTD distributions, between the analysis methods. In *EBarrays*, the range of 95 % confidence intervals (as $\pi_0 \to 1$) in the OPFD were substantially smaller, whereas with *Multtest*, the 95% confidence interval included situations where the entire gene list was incorrect (Figure 5.5). *EBarrays* analysis detected more true discoveries as $\pi_0 \to 1$, illustrated by the spike in the E[OPTD] in Figure 5.6.

Comparisons between the two techniques were repeated under low replicate conditions where $n_1 = n_2 = 2$. FDR control in *EBarrays* was found to be far more powerful than

Multtest in all simulation scenarios tested for moderate to large values of π_0 (Figure 5.9). For *Multtest*, these results suggest that maintaining control at $\alpha = 0.05$ when the experimental power is poor, as in analysis of microarray data with only $n_1 = n_2 = 2$ spot replicates, the probability of making false discoveries in the predicted list of differentially expressing genes is extremely high. This infers that microarray experiments with low numbers of spot replicates should utilize variance estimation information between genes to counteract the effects of low replicates. The observed sensitivity of the *EBarrays* procedure was also substantially higher in low replicate conditions for moderate to large values of π_0 (Figure 5.11).

Due to the lack of fit to the GU-B simulation scenario, *EBarrays* analysis maintained adaptive FDR control at $\alpha < 0.05$, implying that it is more conservative when structural features in the data are not adequately captured. The Uniform target mean intensity layer on the MA scale in GU-B simulations mainly effected the differential expression component of the *EBarrays* model as illustrated by the bias in π_0 estimation (Figure 5.1). As π_0 increases in magnitude, the uniform target mean intensity layer simulates a higher proportion of M value components equal to 0. Target mean intensity component back transformations in 2.5 and 2.6 will have less effect on the *EBarrays* model estimation when π_0 is close to 1.

Estimation of the proportion of true null hypotheses, π_0, is an important factor if adaptive error rate control is desired in experimental analysis. The *EM* algorithm framework within *EBarrays* obtained extremely accurate estimates of π_0. Even when modelling a severely biased simulation scenario such as the Gamma-Uniform-Bernoulli situation, the estimate of π_0 was reasonably unbiased when π_0 was large. The poor fit due to the Uniform mean intensity layer only seriously effected model prediction as π_0 decreased in magnitude. The procedure of Storey & Tibshirani (2003) [34] which estimated π_0 by fitting natural splines to p-values was found to be far less accurate than the *EBarrays* estimate of π_0. The bias associated with the natural spline estimate as $\pi_0 \rightarrow 0$ shows that there are serious departures in the fit to hierarchal microarray data simulations. Given that estimation of π_0 is not straightforward, analysis methods of microarray experiments that are not naturally suited to adaptive FDR control should probably maintain non-adaptive control of the FDR until an improved π_0 estimation technique can be utilized. Under non-adaptive control, the unknown proportion of equivalently expressing genes is likely to range between 0.8 and 1, providing an expected level of control between 0.04 and 0.05 for $\alpha^* = 0.05$. This is a conservative level of FDR control in microarray experimentation.

6.2 Future work

Significance analysis of Microarrays (SAM) [43] incorporates a between gene variance component into the denominator of the pseudo t-statistic during analysis. A comparison of *EBarrays*, and *Multtest* to *SAM* methodology would be useful to determine what benefit the between gene variance component is adding, and how well *SAM* performs on simulated microarray data. The semiparametric *EBarrays* procedure [38] models the mean intensity layer nonparametrically. The rest of the hierarchal model is identical to the parametric Gamma-Gamma-Bernoulli model; measured intensities are fitted using flexible Gamma distributions (which are numerically and analytically convenient), discrete Bernoulli mixing of equivalently expressed, and differentially expressed genes are modelled. Although computationally more intensive, this model should also be compared with the models already examined. It is expected that semiparametric *EBarrays* will fit to GU-B simulated data well, and generally outperform parametric *EBarrays* analysis. The EM algorithm approach of Empirical Bayes methods provides a highly accurate estimate of π_0 when assumptions in the model are reasonably well satisfied. Further investigation is required to establish if an EM approach can be used to estimate π_0 from t-statistics calculated within gene, in microarray analysis of normalized data.

A
Appendix

A.1 Bioconductor source code

The following **R** language code determines the class input argument. If the bioconductor object is of class "marrayRaw", no calculation is necessary, and $log_2(R)$, and $log_2(G)$ values are returned. If the bioconductor object is of class "marrayNorm", backtransformed $log_2(R)$, and $log_2(G)$ values are returned.

```
maLRp <- function(object)
  {
    if(class(object) == "marrayRaw")
      {
        return(maLR(object))
      }
    else if(class(object) == "marrayNorm")
       {
         return(maA(object) + maM(object)/2)
       }
    else
      {
        stop(paste("object of wrong class:", class(object)))
      }
  }

maLGp <- function(object)
  {
    if(class(object) == "marrayRaw")
      {
        return(maLG(object))
      }
    else if(class(object) == "marrayNorm")
       {
         return(maA(object) - maM(object)/2)
       }
    else
      {
        stop(paste("object of wrong class:", class(object)))
      }
  }
```

A.2 Package marrayInput source code modifications

This **R** language code is modified from the marrayInput bioconductor library. The function *read.GenePix* calls *read.marrayRaw*, which scans in multiple Spot of GPR files. Additional code has been added to upload the description information; Name and ID. This is automatically synchronized with intensity information contained with GPR files. Additional code chunks are contained between head and cut tags

```
read.GenePix <-
function (fnames = NULL, path = ".", name.Gf = "F532 Mean", name.Gb = "B532 Median",
          name.Rf = "F635 Mean", name.Rb = "B635 Median", name.W = NULL,
          layout = NULL, gnames = NULL, targets = NULL, notes = NULL, name.NAME="Name",
          name.ID="ID", skip = 0, sep = "\t", quote = "", ...)
{
  if (is.null(fnames))
    fnames <- dir(path = path, pattern = paste("*", "gpr",
                                  sep = "."))
  y <- readLines(file.path(path, fnames[1]), n = 100)
  skip <- grep(name.Gf, y)[1] - 1
  if (is.null(notes))
    notes <- "GenePix Data"
  print(skip)
  mraw <- read.marrayRaw(fnames = fnames, path = path, name.Gf = name.Gf,
                         name.Gb = name.Gb, name.Rf = name.Rf, name.Rb = name.Rb,
                         name.W = name.W, layout = layout, gnames = gnames,
                         targets = targets, notes = notes, skip = skip, sep = sep,
                         quote = quote, name.NAME = name.NAME, name.ID = name.ID,
                         ...)
  return(mraw)
}

read.marrayRaw <-
function (fnames, path = ".", name.Gf, name.Gb = NULL, name.Rf,
            name.Rb = NULL, name.W = NULL, layout = NULL, gnames = NULL,
            targets = NULL, notes = NULL, skip = 0, sep = "\t", quote = "",
            name.NAME=NULL, name.ID = NULL,
            ...)
{
  if (is.null(path))
    fullfnames <- fnames
  else fullfnames <- file.path(path, fnames)
#  fname <- fullfnames[1]
  Gf <- Gb <- Rf <- Rb <- W <- Name <- ID <- NULL
  if (is.null(name.Gb))
    Gb <- matrix(0, 0, 0)
  if (is.null(name.Rb))
    Rb <- matrix(0, 0, 0)
  for (f in fullfnames) {
    print(paste("Reading", f))
# head
    h <- scan(f, quiet=TRUE, what=character(1), sep=sep, skip = skip, quote=quote, nlines=1)
    names(h) <- gsub("\"","",h)
    h <- lapply(h,as.null)
    cols <- c(name.Gf, name.Gb, name.Rf, name.Rb, name.W, name.NAME, name.ID)
    h[charmatch(cols,names(h))]  <- character(1) #Ignores columns that are spelt incorrectly
# cut
    dat <- scan(f, quiet = TRUE, what = h, sep = sep, skip = skip +
                1, quote = quote, ...)
    Gf <- cbind(Gf, as.numeric(dat[[name.Gf]]))
    if (!is.null(name.Gb))
      Gb <- cbind(Gb, as.numeric(dat[[name.Gb]]))
    Rf <- cbind(Rf, as.numeric(dat[[name.Rf]]))
    if (!is.null(name.Rb))
      Rb <- cbind(Rb, as.numeric(dat[[name.Rb]]))
    if (!is.null(name.W))
      W <- cbind(W, as.numeric(dat[[name.W]]))
# head
    if (!is.null(name.NAME))
      Name <- cbind(Name, gsub("\"", "", dat[[name.NAME]]))
    if (!is.null(name.ID))
      ID <- cbind(ID, gsub("\"", "", dat[[name.ID]]))
# cut
  }
# head
  if(!is.null(name.NAME))# check that multiple cols for ID and Name are identical
    {
      Names <- apply(Name[,1]==Name,2,all)
      if(any(Names==F))
        {
          print(paste("Warning: The ’Names’ column in the gpr file(s):",  fnames[!Names], ",
          differ from the first:",  fnames[1], sep=" "))
        }
    }
  if(!is.null(name.ID))
    {
      IDs  <- apply(ID==ID[,1],2,all)
      if(any(IDs==F))
        {
          warning(paste("The ’ID’ column in the gpr file(s):",  fnames[!IDs],
                  ", differ from the first:",  fnames[1], sep=" "))
        }
    }
# cut

  if (!is.null(name.W))
    colnames(W) <- fnames
  if (!is.null(name.Gb))
    colnames(Gb) <- fnames
  if (!is.null(name.Rb))
    colnames(Rb) <- fnames
  colnames(Gf) <- colnames(Rf) <- fnames
  if (is.null(notes))
    notes <- ""
  mraw <- new("marrayRaw", maRf = Rf, maRb = Rb, maGf = Gf,
              maGb = Gb, maNotes = notes)
#head
  if(!is.null(name.NAME))
    {
      Info <-  new("marrayInfo", maLabels=Name[,1], maInfo= data.frame(cbind(ID=ID[,1],
                   Name=Name[,1])), maNotes = notes)
      maGnames(mraw) <- Info
    }
#cut
  if (!is.null(layout))
    maLayout(mraw) <- layout
  if (!is.null(gnames))
    maGnames(mraw) <- gnames
  if (!is.null(targets))
    maTargets(mraw) <- targets
  if (!is.null(W))
    maW(mraw) <- W
  return(mraw)
}
```

A.3 Package marrayinput speed code chunk

This code modification is for the marrayRaw function in the bioconductor marrayInput library. It obtains significant speed increases in uploading time. The following code only scans in the columns of interest into memory ignoring columns which are not of interest.

```
    h <- scan(f, quiet=TRUE, what=character(1), sep=sep, skip = skip, quote=quote, nlines=1)
    names(h) <- gsub("\"","",h)
    h <- lapply(h,as.null)
    cols <- c(name.Gf, name.Gb, name.Rf, name.Rb, name.W, name.NAME, name.ID)
    h[charmatch(cols,names(h))]  <- character(1) #Ignores columns that are spelt incorrectly
```

A.4 Package qvalue π_0 estimation R code

This **R** language code is from the qvalue bioconductor library. The function qvalue, is an automated procedure to calculate an estimate of π_0 from a list of p-values. The code omits the use of observation weighting by $(1-\lambda)$. Note that the estimate of π_0 is slightly biased, as it is estimated at $\hat{\pi}_0(\lambda = 0.95)$,

```
qvalue <-
function (p, lambda = seq(0, 0.95, 0.05), pi0.meth = "smoother",
    fdr.level = NULL, robust = FALSE){
...
 if (pi0.meth == "smoother") {
            spi0 <- smooth.spline(lambda, pi0, df = 3)
            pi0 <- predict(spi0, x = max(lambda))$y
            pi0 <- min(pi0, 1)
        }
...
}.
```

To reduce variance and eliminate bias by estimating π_0 at $\hat{\pi}_0(\lambda = 1)$, the following changes to the qvalue function can be made,

```
qvalue <-
function (p, lambda = seq(0, 0.95, 0.05), pi0.meth = "smoother",
    fdr.level = NULL, robust = FALSE){
...
 if (pi0.meth == "smoother") {
            spi0 <- smooth.spline(lambda, pi0, w = 1-lambda, df = 3)
            pi0 <- predict(spi0, x = 1)$y
            pi0 <- min(pi0, 1)
        }
...
}.
```

A.5 Normal simulation R code

The function *FDRnormalsim* was written to generate observations from a mixture of normal distributions to examine adaptive and non-adaptive FDR control [21, 29]

```
FDRnormalsim <- function(outfile = "tmp", m=1000, iiter=100, seqpi0=1, alpha = 0.05, adaptive=F,
mu0=0, mu1=3, sigma=1, seed=1, Olddir=F, Estpi0 = F)#, nreps=5, generator="GG")
{
############################################
# Number of genes to simulate
# m <- 1000
# Number of iterations
# iiter <- 1
# seqpi0 is the pi[0] values to test at
# seq(0.05,1, length=ceiling(1001*0.95))
# alpha typeI probability test cutoff
# alpha = 0.05
# Finner adaptive FDR control
# adaptive <- F
############################################
library(multtest)
set.seed(seed)
starttime <- proc.time()[3]
# Set output Data directory (removing /Code directory if R is RUN in it)
if(Olddir){
datadir <- gsub("/Code","/Data/Old", getwd())
}else{
datadir <-  gsub("/Code","/Data",getwd())
}
# False discovery threshold
alphastar <- alpha
#pcols  <- m # Number of observations
FD <- rep(NA, iiter)
FND <- rep(NA, iiter)
FDlist <- list(m=m, alpha=alpha, parameters=list(mu0=mu0,mu1=mu1), V=list(), T=list(), p0=list(),p0NW=list())
lambdaseq <- seq(0,(m-1)/m, length=100)
```

```
for(pi0 in seqpi0)
  {
   if(adaptive && !Estpi0)
      {
        alphastar<- alpha/pi0
      }
    m1 <- round(m * (1-pi0))
    m0 <- m - m1
    DEmeans <- c(rep(mu1,m1), rep(mu0,m0))
    DEflag <- as.logical(DEmeans)
    for(i in seq(iiter)){
      X <- rnorm(m, DEmeans, sigma)
      pvalues <- 2*(1-pnorm(abs(X),0,1))
if(Estpi0){
# pi0 estimation
# Storey uses seq(0,0.95, by=0.05) for lambda, estimates at pi(0.95)
# Maximum value of lambda needs to be less than 1 estimated by (m-1)/m
      pi0lambda <- tapply(lambdaseq, lambdaseq, function(x){sum(pvalues>x)/(m*(1-x))})
      # a) weighting by 1-lambda
      pi0hat <- min(smooth.spline(lambdaseq, pi0lambda,  w=1-lambdaseq, df=3)$y[length(lambdaseq)],1)
      FDlist[["p0"]][[deparse(pi0)]][i] <- round(pi0hat,8) #(pi0hat,8)
      # B) no 1-lambda weighting
      pi0hat <- min(smooth.spline(lambdaseq, pi0lambda, df=3)$y[length(lambdaseq)],1)
      FDlist[["p0NW"]][[deparse(pi0)]][i] <- round(pi0hat,8)
      if(adaptive){
        alphastar <- alpha/pi0hat
      }
}
      # STEP DOWN
      Porder <- order(pvalues)
      cutoff <- pvalues[Porder] > (seq(m) * alphastar)/m # Sorted pvalues
      DEorder <- DEflag[Porder] #Reordered DEflag
      rejected <- DEorder[!cutoff]
      accepted <- DEorder[cutoff]
      FD[i] <- sum(!rejected)
      FND[i] <-  sum(accepted)
    }   # i in seq(iiter)
    # Writing to FDlist every iteration of p
    FDlist[["V"]][[deparse(pi0)]] <- FD
    FDlist[["T"]][[deparse(pi0)]] <- FND
    dput(FDlist, paste(datadir, outfile, sep="/"))
  } # pi0 in seqpi0
endtime <- proc.time()[3] - starttime
return(endtime)
}
```

A.6 Multtest and EBarrays simulation R code

The functions *EBarrays* and *Multtest* were written to simulate microarray datasets under 3 scenarios, GG-B, LNN-B, and GU-B. Adaptive FDR control [21, 29] in statistical analysis is used to calculate summary statistics.

```
EBarrays <- function(outfile = "EB.GGtmp", m=1000, iiter=10,
seqpi0= 1, alpha = 0.05, adaptive=T, nreps=5, generator="GG"){
###########################################
```

```
# Number of genes to simulate
# m <- 1000
# Number of iterations
# iiter <- 1
# Flag to cleanup the .RData temporary files
# debug    <- F
# seqpi0 is the pi[0] values to test at
# seq(0.05,1, length=ceiling(1001*0.95))
# alpha typeI probability test cutoff
# alpha = 0.05
# Finner adaptive FDR control
# adaptive <- F
# Number of reps
# nreps  <- 5
############################################
set.seed(1)
starttime <- proc.time()[3]
# Set output Data directory (removing /Code directory if R is RUN in it)
if(.Platform$OS.type=="windows"){
  datadir <- "C://Documents and Settings//Administrator//My Documents//Masters Thesis//FDRpaper//Data//"
}else{
  datadir <- gsub("/Code","", paste(getwd(),"/Data/", sep=""))
#datadir <- "/home/hramwd/MICROARRAY/Analysis/Simulation/Data/"
}
# False discovery threshold
alphastar <- alpha
n1 <- n2 <- nreps
pcols  <- 2*nreps
FD <- rep(NA, iiter)
FND <- rep(NA, iiter)
pprior  <- 2
maxiter <- 5
model <- "IPTG-a"
parameters <- list(HS = list(
       GG = c(2.74886, 1.36546, 4.12844),
       LNN = c(2.32501, sqrt(0.38106), sqrt(1.11561))
       ),
     "IPTG-a" = list(
       GG = c(12.534793153, 0.816305808,   0.370668871),
       LNN = c( 2.36878, sqrt(0.04726), sqrt(1.72682))
       )
     )
nploglikrep <- function(theta, sumx, sumlogx, n1, sumy, sumlogy, n2, z, m)
  {
    a   <- theta[1]
    a0  <- theta[2]
    eta <- theta[3]
    sumz <- sum(z)
         # xx,yy are intensities in the two channels; zz=P(b!=c|xx,yy)
         # theta=(a,a0,eta)
         # (I'll separately optimize pp=P(zz=1); hence npl.. for partial loglik
    ll <- m*(-n1*lgamma(a) - n2*lgamma(a)) + (a-1)*sum((sumlogx + sumlogy)) +
          (n2*a + a0)*sum(z*log(eta+sumy))   +
          (m + sumz)*(a0*log(eta) - lgamma(a0)) +
          (m - sumz)*lgamma(n1*a + n2*a + a0) - (n1*a + n2*a + a0)*sum((1 - z)*log(eta + (sumx + sumy)))
    return(-ll)
  }
#sumlog <- function(x){sum(log(x))}
```

```
 # Gamma parameters and EM initial values
if(generator=="GG"){
    a.shape  <-   parameters[[model]][[generator]][1]#12.534793153
    a0.shape <-   parameters[[model]][[generator]][2]# 0.816305808
    scale    <-   parameters[[model]][[generator]][3]# 0.370668871
    FDlist <- list(m=m, alpha=alpha,
                   GG = c(alpha=a.shape, alpha0=a0.shape, neta=scale),
                   V=list(),
                   T=list(),
                   p0=list()
                   )
  init.theta <- c(a.shape, a0.shape, scale)
  }
if(generator=="LNN") # LNN parameters
  {
    mu    <-       parameters[[model]][[generator]][1]#2.36839
    sigma <-       parameters[[model]][[generator]][2]# sqrt(0.04730)
    tau   <-       parameters[[model]][[generator]][3]#1.72850
    FDlist <- list(m=m, alpha=alpha,
                   LNN = c(mu=mu, sigma=sigma, tau=tau),
                   V=list(),
                   T=list(),
                   p0=list())
  init.theta <- c(mu, sigma, tau)
  }
if(generator=="GU")
  {
    a.shape  <- parameters[[model]][["GG"]][1] #  12.534793153
    Mrange <- c(-2.5,2.5)
    Arange <- c(-8,3)
    FDlist <- list(m=m, alpha=alpha,
                   GU = c(alpha=a.shape),
                   V=list(),
                   T=list(),
                   p0=list())
  init.theta <- c(a.shape, 0.816305808, 0.370668871)
  }
theta <- c(init.theta, NA)
for(pi0 in seqpi0)
  {
    theta[4]<- (1-pi0)
    if(pi0==0) # In Expect of EM log(0) = Inf
      {
        pi0 <-  0.0000001
      }
    if(pi0==1) # In Expect of EM log(1-1) = Inf
      {
        pi0 <-  0.9999999
      }

    if(!adaptive)
      {
        alphastar<- alpha*pi0
      }
    m1 <- round(m * (1-pi0)) # Opposite from ttest model
    m0 <- m - m1
    DEflag <- c(rep(1,m1), rep(0,m0))
# 2) Setting up the dataset (On raw scale)
```

```
    for(i in seq(iiter)){
      if(generator=="GG"){
        DEscales <- rgamma(2*m1, shape=a0.shape, rate=scale)
        EEscales <- rgamma(m0 , shape=a0.shape, rate=scale)
        scales   <- c(rep(DEscales, each=nreps), rep(EEscales, each=2*nreps))
        X <- rgamma(m*pcols, a.shape, rate=scales)
        dim(X) <- c(pcols, m)
        X <- t(X)
      }
      if(generator=="LNN"){
        DEmeans <- rnorm(2*m1, mu, tau)
        EEmeans <- rnorm(m0 , mu, tau)
        means   <- c(rep(DEmeans, each=nreps),rep(EEmeans, each=2*nreps))
        X <- exp(rnorm(m*pcols, means, sigma))
        dim(X) <- c(pcols, m)
        X <- t(X)
      }
      if(generator=="GU"){
        Mmeans <- c(runif(m1, Mrange[1], Mrange[2]), rep(0,m0))
        Ameans <- runif(m, Arange[1], Arange[2])
        Ymeans <- (2^(Ameans + Mmeans/2))
        Xmeans <- (2^(Ameans - Mmeans/2))
        means <-  rep(c(t(cbind(Xmeans, Ymeans))), each=nreps)
        X <- rgamma(m*pcols, 12.53, rate = means)
        dim(X) <- c(pcols, m)
        X <- t(X)
      }
# 3) Parameters for DE (mu1, mu2) and EE (mu)
##################################################################
############################ EM Bayes ############################
##################################################################
      # Objects needed for em (m is # genes)
      x <- X[,1:nreps]
      y <- X[,(nreps+1):pcols]
# rowSums much faster than apply
      sumx <- rowSums(x)
      sumy <- rowSums(y)
      logx <- log(x)
      logy <- log(y)
      sumlogx <- rowSums(logx)
      sumlogy <- rowSums(logy)
      iter <- 1
      notdone <- T
      while( notdone )
        {
          a   <- theta[1]
          a0  <- theta[2]
          eta <- theta[3]
          p   <- theta[4]
# E-step
          tmp <- log(p) - log(1-p)                                                      +
            a0*log(eta) + lgamma(n1*a + a0) + lgamma(n2*a + a0) - lgamma(a0)            -
              (n1*a + a0)*log(eta + sumx) - (n2*a + a0)*log(eta + sumy)                 +
                (n1*a + n2*a + a0)*log(eta+(sumx+sumy))                                 -
                  lgamma(n1*a + n2*a + a0)
          z <- 1/( 1 + exp(-tmp) )
          if(any(is.nan(z))){
            print(paste("NaNs found at, pi[0] = ",pi0, sep=""))
            dput(Error.seed, .Random.seed)
          }
# M-step
          fit <- optim(par=theta[1:3], fn=nploglikrep, method="L-BFGS-B",
                       lower=c(1,0.001,0.001), sumx=sumx, sumlogx=sumlogx, n1=n1,
                       sumy=sumy,sumlogy=sumlogy, n2=n2, z=z, m=m)
          theta <- c( fit$par[1:3],  ( pprior + sum( z ) )/(2*pprior+m ) )
#          print(c(iter, round(theta,4)))
          notdone <- (iter<maxiter)
          iter <- iter + 1
        }
##################################################################
############################ FDR calc ############################
##################################################################
            P0 <- 1-z
        P0order <- order(P0)
        P0sort <- P0[P0order]
        EP0 <- cumsum(P0sort)
        cutoff <- (EP0/seq(EP0))<alphastar
 ######### J(R) calculation ########
 #       P0sort <- P0[P0order]
 #       JR <- rep(NA, m)
 #       for(ind in seq(m)){
 #       theoretically sum should be in here too
 #       JR[ind] <- sum(P0sort<=P0sort[ind])
 #     }
 #       EP0 <- cumsum(P0sort)
 #       cutoff <-  (EP0/JR)<alphastar
 ######### J(R) calculation ########
      DEorder <- DEflag[P0order] #Reordered DEflag
      rejected <- DEorder[cutoff]
      accepted <- DEorder[!cutoff]
      FD[i] <- sum(!rejected)
      FND[i] <-  sum(accepted)
      FDlist[["p0"]][[deparse(pi0)]][i] <-  round(1-theta[4],8)
    }   # i in seq(iiter)
    FDlist[["V"]][[deparse(pi0)]] <- FD
    FDlist[["T"]][[deparse(pi0)]] <- FND
    dput(FDlist, paste(datadir, outfile ,sep=""))
  } # p in seqp
endtime <- proc.time()[3] - starttime
return(endtime)
}

Multtest <- function(outfile = "tmp", m=1000, iiter=10, seqpi0= 0.9,
alpha = 0.05, adaptive=F, nreps=5, generator="GG",estpi0=F, seed=1)
{
############################################
# Number of genes to simulate
# m <- 1000
# Number of iterations
# iiter <- 1
# Flag to cleanup the .RData temporary files
# debug    <- F
# seqpi0 is the pi[0] values to test at
# seq(0.05,1, length=ceiling(1001*0.95))
# alpha typeI probability test cutoff
# alpha = 0.05
# Finner adaptive FDR control
# adaptive <- F
# Number of reps
# nreps  <- 5
############################################
library(multtest)
set.seed(seed)
starttime <- proc.time()[3]
# Set output Data directory (removing /Code directory if R is RUN in it)
#datadir <- "/home/hramwd/MICROARRAY/Analysis/Simulation/Data/"
datadir <- gsub("/Code","", paste(getwd(),"/Data/", sep=""))
# False discovery threshold
alphastar <- alpha
pcols  <- 2*nreps
FD <- rep(NA, iiter)
FND <- rep(NA, iiter)
# repfile
replicates <- rep(0:1, rep(nreps,2))
# Model parameters
model <- "IPTG-a"
parameters <- list(HS = list(
       GG = c(2.74886, 1.36546, 4.12844),
       LNN = c(2.32501, sqrt(0.38106), sqrt(1.11561))
       ),
     "IPTG-a" = list(
       GG = c(12.534793153, 0.816305808,   0.370668871),
       LNN = c( 2.36878, sqrt(0.04726), sqrt(1.72682))
       )
     )
# HEAT SHOCK
# GG  Model params.theta.ests
#[1] 2.74886 1.36546 4.12844
# LNN Model params.theta.ests
#[1] 2.32501 0.38106 1.11561
# IPTG-a
# GG  Model params.theta.ests
#[1] 12.50769  0.81749  0.37262
# LNN Model params.theta.ests
#[1] 2.36878 0.04726 1.72682
if(generator=="GG")
  {
 # Gamma parameters
    a.shape  <-  parameters[[model]][[generator]][1]
    a0.shape <-  parameters[[model]][[generator]][2]
    scale    <-  parameters[[model]][[generator]][3]
    FDlist <- list(m=m, alpha=alpha,
                  GG = c(alpha=a.shape, alpha0=a0.shape, neta=scale),
                  V=list(), T=list(), p0=list())
  }
if(generator=="LNN")
  {
# LNN parameters  IPTG-a     HeatShock
    mu    <- parameters[[model]][[generator]][1]
    sigma <- parameters[[model]][[generator]][2]
    tau   <- parameters[[model]][[generator]][3]
    FDlist <- list(m=m, alpha=alpha,
              LNN = c(mu=mu, sigma=sigma, tau=tau),
              V=list(), T=list(), p0=list())
  }
if(generator=="GU")
  {
    a.shape  <- parameters[[model]][["GG"]][1]
    Mrange <- c(-2.5,2.5)
    Arange <- c(-8,3)
    FDlist <- list(m=m, alpha=alpha,
              GU = c(alpha=a.shape),
              V=list(), T=list(), p0=list())
  }
lambdaseq <- seq(0,(m-1)/m, length=100)
for(pi0 in seqpi0)
  {
    if(pi0==0)
      {
        pi0 <-  0.0000001
      }
    if(pi0==1)
      {
        pi0 <-  0.9999999
      }
    if(adaptive && !estpi0)
      {
        alphastar<- alpha/pi0
      }
    m1 <- round(m * (1-pi0))
    m0 <- m - m1
    DEflag <- c(rep(1,m1), rep(0,m0))
#    p.init <- c(p,1-p)
    for(i in seq(iiter)){
      if(generator=="GG")
        {
          DEscales <- rgamma(2*m1, shape=a0.shape, rate=scale)
          EEscales <- rgamma(m0 , shape=a0.shape, rate=scale)
          scales   <- c(rep(DEscales, each=nreps), rep(EEscales, each=2*nreps))
          X <- log2(rgamma(m*pcols, a.shape, rate=scales))
          dim(X) <- c(pcols, m)
          X <- t(X)
        }
      if(generator=="LNN")
        {
          DEmeans <- rnorm(2*m1, mu, tau)
          EEmeans <- rnorm(m0 , mu, tau)
          means   <- c(rep(DEmeans, each=nreps),rep(EEmeans, each=2*nreps))
          X <-rnorm(m*pcols, means, sigma) # Data already on log2 scale
          dim(X) <- c(pcols, m)
          X <- t(X)
        }
       if(generator=="GU")
        {
          Mmeans <- c(runif(m1, Mrange[1], Mrange[2]), rep(0,m0))
          Ameans <- runif(m, Arange[1], Arange[2])
          Ymeans <- (2^(Ameans + Mmeans/2))
          Xmeans <- (2^(Ameans - Mmeans/2))
          means <-  rep(c(t(cbind(Xmeans, Ymeans))), each=nreps)
          X <- rgamma(m*pcols, 12.53, rate = means)
          dim(X) <- c(pcols, m)
          X <- t(X)
        }
############## t tests ##############
      tscores <- mt.teststat(X, classlabel=replicates, test="t.equalvar")
      dfs <- pcols - 2
      pvalues <- 2*(1-pt(abs(tscores), df=dfs))
# Storey uses seq(0,0.95, by=0.05) for lambda, estimates at pi(0.95)
# Maximum value of lambda needs to be less than 1 estimated by (m-1)/m
      pi0lambda <- tapply(lambdaseq, lambdaseq, function(x){sum(pvalues>x)/(m*(1-x))})
      pi0hat <- min(smooth.spline(lambdaseq, pi0lambda,  w=1-lambdaseq, df=3)$y[length(lambdaseq)],1)
      # Must turn alphastar back into alpha before calculating estimate
      if(estpi0 && adaptive){
        alphastar <- alpha/pi0hat
      }
      # STEP DOWN
      Porder <- order(pvalues)
      cutoff <- pvalues[Porder] > (seq(m) * alphastar)/m
      DEorder <- DEflag[Porder] #Reordered DEflag
      rejected <- DEorder[!cutoff]
      accepted <- DEorder[cutoff]
      FD[i] <- sum(!rejected)
      FND[i] <-  sum(accepted)
      #print(paste("Not Rejected=",sum(!rejected), "Accepted=",sum(accepted)))
      FDlist[["p0"]][[deparse(pi0)]][i] <- round(pi0hat,8)
    }   # i in seq(iiter)
    # Writing to FDlist every iteration of p
    FDlist[["V"]][[deparse(pi0)]] <- FD
    FDlist[["T"]][[deparse(pi0)]] <- FND
    dput(FDlist, paste(datadir, outfile, sep=""))
  } # pi0 in seqpi0
endtime <- proc.time()[3] - starttime
return(endtime)
}
```

Bibliography

[1] Avery O.T. MacLeod C.M. & McCarty M. Studies of the chemical nature of the substance inducing transformation of pneumococcal types. induction of transformation by a deoxyribonucleic acid fraction isolated from pneumococcus type iii. *Journal of Experimental Medicine*, 79:137–158, 1944.

[2] Watson J.D. and Crick F.H.C. Molecular structure of nucleic acids. *Nature*, 171:737–738, 1953.

[3] Crick F.H.C. On protein synthesis. *Symposium of the society for experimental biology XII*, page 153, 1958.

[4] Baltimore D. RNA-dependent DNA polymerase in virions of RNA tumour viruses. *Nature*, 226(252):1209–1211, 1970.

[5] Temin H. and Mizutani S. RNA-dependent DNA polymerase in virions of Rous sarcoma virus. *Nature*, 226(252):1211–1213, 1970.

[6] Nathans D. and Smith H. O. Restriction endonuclease in analysis and restructuring of DNA molecules. *Ann. Rev. Biochem*, 44:272–293, 1975.

[7] Southern E. M. Detection of specific sequences among DNA fragments separated by gel electrophoresis. *Journal of Molecular Biology*, 3(8):503–517, 1975.

[8] Alwine J. C. Kemp D. J. and Stark G. R. Method for detection of specific RNAs in agars by transfer to diazobenzyloxymethyl-paper and hybridization with DNA probes. *Proceedings of the National Acadamy of Sciences*, 74(8):5350–5354, 1977.

[9] Basdevant J. L. and Dalibard J. The Stokes Shift §1.1, The Quantum Mechanics Solver: How to Apply Quantum Theory to Modern Physics. *Berlin: Springer-Verlag*, pages 4–5, 2000.

[10] Yang Y. H. Dudoit S. Luu P. and Speed T. P. Normalization for cDNA microarray data. *Microarrays: Optical Technologies and Informatics,volume 4266 of Proceedings of SPIE*, 2001.

[11] Cleveland W. S. Robust locally weighted regression and smoothing scatterplots. *J. Amer. Stat. Assoc.*, 74:829–836, 1979.

[12] Smyth G. K. and Speed T. P. Normalization of cDNA microarray data. *D. Carter (ed), METHODS: Selecting Candidate Genes from DNA Array Screens: Application to Neuroscience*, 2003.

[13] Kerr M. K. Churchill G. A. Experimental design for gene expression microarrays. *Biostatistics*, 2(2):183–201, 2001.

[14] *GenePix Pro microarray and analysis software*. 1001 Chess Drive, Foster City, CA 94404 USA, http://www.axon.com.

[15] Dudoit S. and Yang Y. H. Bioconductor R packages for exploratory analysis and normalization of cDNA microarray data. In G. Parmigiani E. S. Garrett R. A. Irizarry and S. L. Zeger (eds), editors, *The Analysis of Gene Expression Data: Methods and Software*. Springer, New York., 2002.

[16] Richmond C.S. Glasner J.D. Mau R. Jin H. and Blattner F.R. Genome-wide expression profiling in *Escherichia coli* K-12. *Nucl. Acids Res.*, 27(19):3821–3835, 1999.

[17] Gentleman R. Bates D. Bolstad B. Carey V. Dettling M. Dudoit S. Ellis B. Gautier L. Gentrey J. Hornik K. Hothorn T. Huber W. Iacus S. Irizarry R. Leish F. C. Maechler M. Rossini A. J. Sawitski G. Smyth G. K. Tierney L. Yang J. Y. H. Zhang J. Bioconductor: a software development project. Technical report, Department of Biostatistics, Harvard School of Public Health, Boston, 2003.

[18] Ihaka R. and Gentleman R. R: A language for data analysis and graphics. *Journal of Computational and Graphical Statistics*, 5(3):299–314, 1996.

[19] Leisch F. Dynamic generation of statistical reports using literate data analysis. *Compstat 2002 Proceedings in Computational Statistics*, page 575 580, 2002.

[20] Buckley M. J. The Spot user's guide. *CSIRO Mathematical and Information Sciences, http://www.cmis.csiro.au/IAP/spotinfo.htm*, 2000.

[21] Benjamini Y. and Hochberg Y. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society*, 57(1):289–300, 1995.

[22] Reiner A. Yekutieli D. and Benjamini Y. Using resampling−based FDR controlling multiple test procedures for analyzing microarray gene expression data, 2001.

[23] Shaffer J. P. Multiple hypothesis testing. *Biometrika*, 46:561–584, 1995.

[24] Dudoit S. Yang Y. H. Callow M. J. and Speed T. P. Statistical methods for identifying differnetially expressed genes in replicated cDNA microarray experiments. Technical Report #578, Stanford University, Department of Biochemistry,Stanford University School of Medicine,Beckman Center, B400, Stanford, CA 94305-5307, August 2000.

[25] Benjamini Y. and Liu W. A step-down multiple hypothesis testing procedure that controls the false discovery rate under independence. *Journal of Statistical Planning and Inference*, 1(82):163–170, 1999.

[26] Dudoit S. Shaffer J.P. and Boldick J.C. Multiple hypothesis testing in microarray experiments. *Statistical Science*, 18(1):71–103, 2003.

[27] Finner H. and Roters M. On the false discovery rate and expected Type I errors. *Biometrical Journal*, 43(8):985–1005, 2001.

[28] Benjamini Y. and Hochberg Y. The adaptive control of the False Discovery Rate in multiple hypothesis testing with independent statistics. *Journal of Educational and Behavioural Statistics*, 25:60–83, 2000.

[29] Benjamini Y. Krieger A. and Yekutieli D. Adaptive linear step-up false discovery rate controlling procedures. Technical Report RP-SOR-01-03, Department of Statistics Tel Aviv University, Israel, 2001.

[30] Genovese C. and Wasserman L. Operating characteristics and extentions of the false discovery rate procedure. *Journal of the Royal Statistical Society, Series B*, 64:499–517, 2002.

[31] Black M. A. *Statistical issues in the design and analysis of spotted microarray experiments.* PhD thesis, Department of Statistics, Purdue University, 2002.

[32] Storey J. D. A direct approach to false discovery rates. *Journal of the Royal Statistical Society, Series B*, 64:479–498, 2002.

[33] Black M. A. A note on the adaptive control of false discovery rates. *JRSS B, in press*, 2004.

[34] Storey J. D. and Tibshirani R. J. Statistical significance for genome-wide experiments. *Proceedings of the National Academy of Sciences*, 100:9440–9445, 2003.

[35] Storey J. D. and Tibshirani R. J. Statistical significance for genome-wide experiments. Preprint, January 2003.

[36] Newton M.A. Kendziorski C.M. Richmond C.S. Blattner F.R. and Tsui K.W. On differential variability of expression ratios: Improving statistical inference about gene

expression changes from microarray data. *Journal of Computational Biology*, 8(1):37–52, 2001.

[37] Kendziorski C. M. Newton M. A. Lan H. and Gould M. N. On parametric empirical Bayes methods for comparing multiple groups using replicated gene expression profiles. Technical Report #166, Department of Biostatistics and Medical Informatics, University of Wisconsin, K6 466 Clinical Science Center 600 Highland Avenue Madison, WI 53792-4675 608-263-1706, February 2002.

[38] Newton M.A. Noueiry A. Sarkar D. and Ahlquist P. Detecting differential gene expression with a semiparametric hierarchical method. Technical Report #1074, Department of Statistics, UW Madison, January 2003.

[39] Dempster et al. Maximum likelihood from incomplete data via the em algorithm. *JRSS*, B(39):1–38, 1977.

[40] Statistical Sciences. *S-PLUS Guide to Statistical and Mathematical Analysis. Version 3.2*. StatSci, a division of MathSoft, Inc., Seattle, 1993.

[41] Storey J. D. The positive false discovery rate: A Bayesian interpretation and the q-value. Technical report, Stanford Department of Statistics, 2001.

[42] Newton M. A. Kendziorski C. M. Parametric Empirical Bayes Methods for Microarrays. In G. Parmigiani E. S. Garrett R. A. Irizarry and S. L. Zeger (eds), editors, *The Analysis of Gene Expression Data: Methods and Software*. Springer, New York., 2002.

[43] Tusher V. Tibshirani R. and Chu G. Significance analysis of microarrays applied to the ionizing radiation response. *Proceedings of the National Acadamy of Sciences*, 98(9):5116–5121, April 2001.