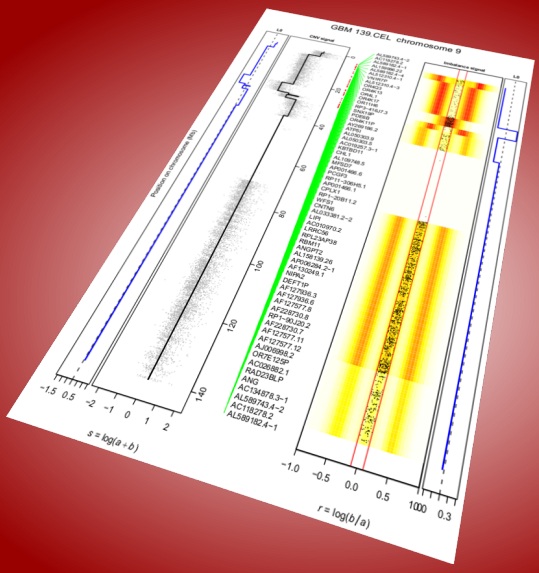
Advanced statistical tools for SNP arrays : signal calibration, copy number estimation and single array genotyping

Rippe, R.C.A.

ADVANCED STATISTICAL TOOLS FOR SNP ARRAYS

Signal Calibration, Copy Number Estimation and Single Array Genotyping



Ralph C.A. Rippe

Advanced Statistical Tools for SNP Arrays

Signal Calibration, Copy Number Estimation

and Single Array Genotyping

Ralph C.A. Rippe

Leiden University
Leiden 2012

This book was formatted using the LaTeX class memoir,
with custom chapter style definitions.

Printed by Mostert en Van Onderen, Leiden.

Advanced Statistical Tools for SNP Arrays
Signal Calibration, Copy Number Estimation and Single Array Genotyping
PhD Thesis, Leiden University

ISBN/EAN: 978-94-90858-14-8

Proefschrift

ter verkrijging van
de graad van Doctor aan de Universiteit Leiden
op gezag van Rector Magnificus prof. dr. mr. P.F. van der Heijden,
volgens besluit van het College voor Promoties
te verdedigen op dinsdag 13 november 2012
klokke 16:15 uur
door

Ralph C.A. Rippe

Geboren te Delft
in 1982

Promotiecommissie

Promotores
Prof. Dr. J.J. Meulman
Prof. Dr. ing. P.H.C. Eilers (Erasmus Medisch Centrum)

Overige leden
Prof. Dr. D.I. Boomsma (Vrije Universiteit Amsterdam)
Prof. Dr. F.A. van Eeuwijk (Wageningen Universiteit)
Prof. Dr. W.J. Heiser (Universiteit Leiden)
Prof. Dr. P. Slagboom (Leids Universitair Medisch Centrum)

Contents

List of Figures

List of Tables

Introduction

1

In this chapter some background to the data used in this thesis is described, in both biological and methodological sense. First, a basic discussion on DNA and the genetics of tumor tissue is given, followed by details on commonly used SNP measurement technology, to close with different applications of the same signals. Most of the descriptions are a strongly simplified version of reality, but this is needed to understand most of the concepts and ideas described in this thesis.

1.1 Human DNA and disease

Recently a book called "The Emperor of all Maladies: A Biography of Cancer" (Mukherjee, 2010) was published. Its message is clear: cancer is a large problem. In general, increasing amounts of evidence have been gathered that each case of cancer or tumor development is related to genetics at least to some extent. More specifically, it has to do with genetic mutations which can be caused by heritable susceptibility or simply by external factors (mutagens) like chemicals or radiation. The human body consists of numerous cells and each of them contains a full copy of our complete DNA. Therefore, there are a lot of opportunities for problems to occur. Small scale changes (mutations) in DNA regularly occur and are not harmful per se, since the structure of DNA has several recovery methods for successful replication. However, if despite the backups a cell with problematic DNA has reproduced, that DNA is also copied. Since cell division is a continuous process in order to replace damaged cells, genetic problems can spread quickly. Mutations occur infrequently, while we see the result of the mutations in the form of so-called polymorphisms, the different DNA variants that can arise from mutations.

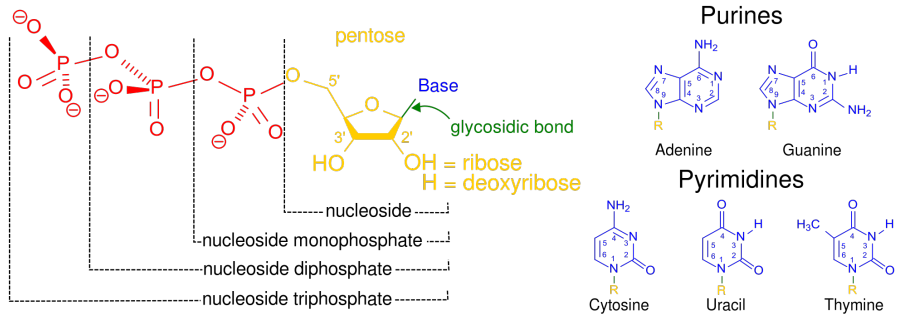


Figure 1.1: Nucleotides and nucleosides: molecules and bases ©Scientific Commons.

Therefore, to understand this problem better, a more detailed description of DNA and polymorphisms is given below.

DNA was first discovered in 1869 by the Swiss biochemist J. F. Miescher. He performed chemical tests on tissue obtained from hospital waste. However, it was not before 1909 that Ph. Levene formulated the first, but incorrect, theory about the chemical structure of DNA. He suggested that it was a large structure of 4 building blocks, the nucleotides. The actual and correct structure was published in 1953 by Watson & Crick in Nature, to be followed later by a paper on DNA replication, by the same authors. They based their publications on X-ray diffraction data (1952) from Rosalind Franklin and colleagues. Their combined efforts taught us some valuable lessons.

Healthy (human) DNA is contained in pairs of chromosomes which are two (long) chains of nucleotides which are referred to using an “alphabet" of four letters, representing molecules (nucleotides) with the bases (nucleosides) adenine, cytosine, guanine and thymine. We distinguish purines and pyrimidines, to which sugar and phosphate groups are attached and together make up the whole nucleotide. See Figure 1.1 for an illustration. A nucleotide with a purine base pairs to one with a pyrimidine base. Human DNA holds about 3 billion of these pairs. Without going into more detail, the chemical and physical properties of all these bonds cause the polymer to coil up into a

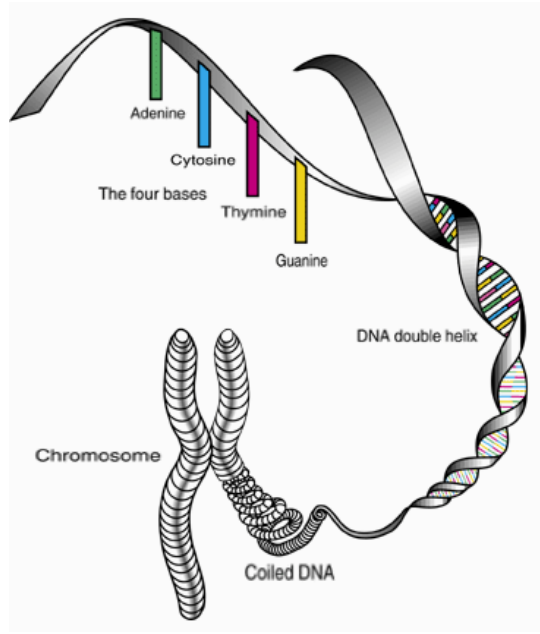


Figure 1.2: DNA: a double helix structure.
Image ©retrieved from https://www.llnl.gov/str/June03/Stubbs.html.

so-called double helix structure, which makes storage of genetic information very compact and safe. This widely known property is described for example in Brown (1999) and Griffiths et al. (2002), and is depicted in Figure 1.2. We can define any (nucleotide) position on a chromosome by counting the number of nucleotides from a unique end.

Genes are specific sets of nucleotides, which can have very different purposes. Some genes code for expression of certain physical features, like eye color, hip width or shape of the nose. Others are involved in regulating the expression of genetic information or serve to simply hold protein structures together. For organisms other than human the same principles hold, but with a few differences. For example, in non-human species the number of chromosome pairs is different from that in humans. Furthermore, where hu-

man DNA holds two chromosome copies - two homologues, called diploid - different numbers of homologues or ploidy also occurs. For example, DNA from some potato species has sets of four homologues - tetraploid - and the Drosophila melanogaster (a type of fruit fly) has 4 pairs of chromosomes (hence is diploid), in which about 75% of (known) human disease genes can be matched to the fruit fly genome (Reiter et al., 2001). Therefore the latter is often used as a genetic model for human disease(s) and to study biological processes such as aging.

The remainder of this introduction and thesis is however restricted to diploid human DNA and a very specific problem therein: nucleotide polymorphisms.

A nucleotide on one of the two chromosome is referred to as an allele (not to be mistaken for a gene allele, which is a large set of nucleotide base pairs). At most positions on our chromosomes we will always find the same nucleotide on both chromosomes. But there are millions of places (about 1 in every 1000 nucleotides) where we see different allele variations (polymorphic occurence) due to a mutation on one of the chromosomes. If the allele for one chromosome is called A and B for the other chromosome and if the chromosomes are indeed identical, then only pairs AA or BB would occur. Small changes can also be AB or BA, which cannot be distinguished. When for a SNP one particular pair of alleles in one person is different from the population, it is called a mutation. If this mutation occurs in more than 1% of the population it is called a variation. These variations are known to occur on specific chromosomal locations, which means that only a small selection of the whole genome sequence (all base pairs in a row), is of interest. They are called Single Nucleotide Polymorphisms (SNPs, pronounced as “snips") and they are being studied extensively in biology and medicine (Adorjan et al., 2002; Altshuler et al., 2005).

These mutations can have (possibly strong) implications, for example breast cancer (Chin et al., 2007). Two general classes of mutations in human DNA are changes in allele copy number (CNV) and imbalance in the allelic ratio. Further mutational distinctions can be made between duplications, inversions and deletions (Conrad et al., 2006). Duplication means that more than two allele copies were found, where two were expected. In

contrast to duplication, deletions of specific sections of chromosomes chromosomal sections can occur, in two ways. In the first, a section is lost in one of the cell reproduction stages. An inversion occurs when a broken chromosome is repaired, but (some of) the parts are reversed before actual repair. Translocations involve 'reordering' of chromosomal sections.

1.2 SNP chips and recent developments in DNA measurement

As described before, each nucleotide with base A, C, G or T on one chromosome is complementary to another. This complementarity is used in a process called hybridization, in which a sequence-specific oligonucleotide (a sequence of 50 bases or less) binds to the DNA strand under treatment. The DNA strand contains the SNP of interest. The SNP measurements are obtained from a photo(n) sensor that captures photons with a given wavelength from one channel, that are emitted after the hybrid DNA is targeted by a laser. By laser illumination the molecules change energetic state and when they fall back to their original state, they emit energy. The result from the photo sensor is a high-resolution image, which is analyzed and translated to numeric values. These numbers are then used in downstream analyses. In the end we have, for one sample, a fluorescence value for each allele in each SNP. Below, two major platforms are discussed, but more exist.

The first major manufacturer is Affymetrix Affmetrix, 2006), known for its trademark 'GeneChip' products. In these single-channel fluorescence chips four probes with oligonucleotides are used (Figure 1.3), to reduce the influence of possible miss-matches. The first and second probe interrogate one direction for the A and B target allele, while the third and fourth interrogate the opposite direction of the the same alleles. Through the years different enzymes were used to split the DNA prior to hybridizing in a particular direction. Measurements are performed on solid surfaces, usually glass or silicon.

The second manufacturer, started in 2001, is Illumina (Fan et al., 2006). One implementation that got Illumina to the front of SNP research was the

Figure 1.3: An Affymetrix SNP chip. ©Affymetrix.

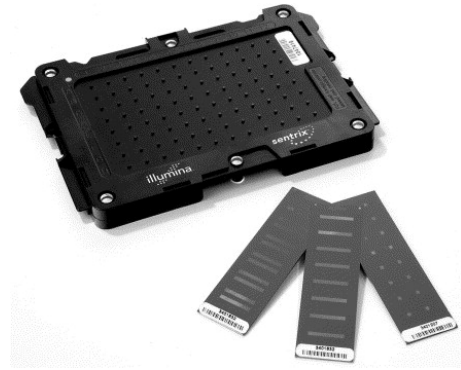


Figure 1.4: An Illumina 8x12 cell well-plate. ©Illumina.

so-called bead array (top device in Figure 1.4, known as the 'GoldenGate Genotyping' technology. One of the trademarks was the use of two-color fluorescence signals (two-photon excitation fluorometry) using bead arrays, which are also used in methylation applications (Bibikova et al., 2006). Common dyes are Cy3 (with a fluorescence emission of 570nm; green light) and Cy5 (670nm: red light). The main plate consisted of 96 (8 by 12) wells for the same number of bead arrays, where in each of the wells a different tissue sample or blood sample was analyzed for about 1600 SNPs. These SNPs were probed on individual beads instead of a solid surface. Later, the SNP resolution increased to 550.000 in more recent technologies like 'Infinium'.

Initial SNP technology for human DNA could target about 1600 SNPs in the whole genome set. In subsequent years the number of targeted SNPs, the SNP resolution, gradually increased through 50.000, 100.000, 500.000 to currently up to 1.200.000 SNPs (full resolution probing) in the most recent platforms.

Apart from the two platforms discussed above, a number of other platforms exist, e.g. Sequenom, Perlegen, Mip, FP-TDI and InVader. However, due to their absence in this thesis, their technical properties are not discussed in further detail.

The latest trend away from SNPs, although not addressed specifically in the following chapters, does not probe at a high number of SNP positions but simply analyzes the whole genomic sequence, including all allele pairs in between currently probed SNP positions. This technique is, unsurprisingly, called 'whole genome sequencing'. Accompanying challenges for this new technique come in terms of data storage (several TBs per individual) and model efficiency (think of memory capacity and computation times).

1.3 Applications of SNP fluorescence signals

The bottom line for the remainder of this thesis, deriving from the above descriptions, is that, per SNP, we have two fluorescence signals that are proportional to the amount of the respective A and B alleles.

One major application of these DNA measurements is to determine which specific combination of alleles is found at individual SNP positions. Assume two channels a and b for the two different alleles, one can measure the double signal for allele A (AA), indicated by double fluorescence strength in channel a and low in b, the double strength signal for B (BB, strong channel b and low a) or equivalent signal strength for each allele (AB, channel a and b in equal proportions) This process is generally referred to as *genotyping* or *genotype calling*. Finding an AA or BB genotype is called homozygous; the alleles are the same for both chromosomes. The AB genotype is called heterozygous. Genotyping of normal (non-tumor) DNA is common practice in e.g. epidemiology (e.g. Huebner et al., 2007).

It comes as no surprise that the genotyping problem was addressed before. For example, companies that manufacture chemical platforms to measure DNA composition (described in section 1.2 in more detail) generally provide their own software to determine genotypes. Third party solutions also exist, like CRLMM (Carvalho et al., 2007) and BirdSeed (Korn et al., 2008). Of course many other methods exist. All of these methods rely one way or another on combined information of multiple reference arrays. Large-scale efforts have been made to catalog known SNP positions for different sets of publicly available arrays, in order to create a 'gold standard' database with genotypes: HapMap (The International HapMap Consortium, 2003; 2007). On the other end, there is also a database that contains annotations for each of these SNPs: BioMart (www.biomart.org).

A second application is to determine deviations from the common composition of two alleles in healthy tissue (e.g. Lips et al., 2005). For example, due to a variety of causes, one (or both) of the alleles may be lost and replaced by either 'empty' DNA, a so-called null-allele, or copied back from the remaining allele. Alternatively, erroneous extra alleles copies may have been created during cell division. Checking for changes in the number of alleles in both chromosome is generally referred to as Copy Number assessment. For example, each chromosome has the same deviation from normal tissue. A more general case is allelic imbalance, where the number of alleles is no longer the same for the two chromosomes. The special case of loss of one allele (out of two) and replacement by the remaining one, hence losing the possibility of finding a heterozygous genotype, is called allelic imbalance (as discussed in McCarroll et al., 2006). From section 1.1 it is immediately clear what the consequences of these problems can be. From this point of view it is a challenge to accurately estimate breakpoints between chromosomal regions with different numbers of alleles. A related challenge is that of finding how many allele copies are found in a specific region. A lot of methods have devised for this specific purpose, like FLasso, CGHseg, DNACopy, VEGA and cumSeg. Several extensive comparisons have been made between subsets of these methods (Lai et al., (2005); Winchester et al., (2009); Tsuang et al., (2010); Muggeo et al., (2011) and Morganella et al. (2010)), all concluding that there is not one procedure that serves all purposes. They collectively

suggest to use multiple methods in conjunction.

For all purposes of SNP signals described above, specialized models exist. One property found in almost all algorithms, is that decisions are made by relating allele intensities between multiple arrays, one SNP at the time (e.g. genotype calls) or by using a set of healthy tissue reference arrays (e.g. CNV profiling). In contrast to this 'school of thinking', the methods described in this thesis use a single array for signal calibration, genotype calling or CNV profiling. This approach can be very useful in situations where new chips have to be designed and a large pool of testing and/or reference material is not available. An example can be found in ALCHEMY by Wright et al. (2010).

1.4 Data format and signal properties

If measurements are performed for an individual, they are performed using a chip or array. Therefore, from here on we use the term "array" for a biological sample, instead of the common use as a multidimensional matrix.

A typical collection of arrays can be seen as a data box, containing only ones and zeros, with three modes: SNPs, arrays and genotypes. Each allele is associated with only one genotype, and hence has a value of 1 in only one of the three genotype layers. This implies that 2/3 of the data box depicted in Figure 1.5 contains no information, or in more statistical terms, is missing. The missingness is a result of this way of structuring the data. Aggregation of the three incomplete genotype layers results in a full matrix with measurements for the alleles on all arrays, without genotype information. This suggests to develop a multi-way model (Kroonenberg, 2008; Smilde et al., 2004) that can cope with large and structural amounts of missing observations. For practical reasons we will not. However, the three-dimensional structure can be used to see and evaluate other interesting properties.

If the technology we described worked perfectly, our story would end here. In practice we observe a number of interesting and relevant patterns in normal DNA:

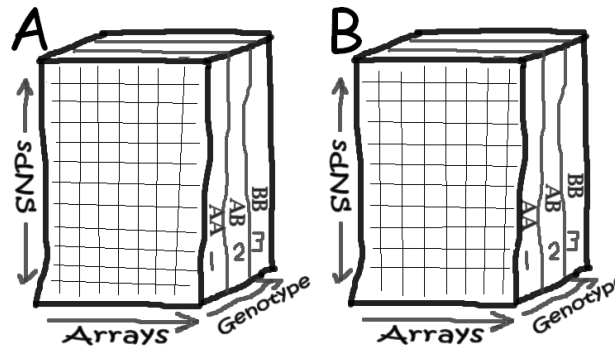


Figure 1.5: SNP data with three modes: SNPs in the rows, samples in the columns and genotypes in the layers.

1. The strength of the fluorescence signals varies systematically between SNPs. If a SNP signal is strong, it is strong in all arrays. A weak SNP signal is weak in all arrays.
2. The strength of the signals varies between biological samples (which is unavoidable; it is caused by differences in the quality of the biological material and the efficiency of DNA extraction) and hybridization.
3. Systematic deviations from the theoretical genotype factors 1 and 2 occur (1 or 2 times the allele).
4. Noise and background signals are present.

The effects described under 1 and 2 are depicted in Figure 1.6. After sorting it is clear that SNPs that have weaker fluorescence signal compared to other SNPs, also are relatively weaker in all arrays. There is a similar effect for arrays, but it is less strong.

1.5 Contributions in this thesis

As valuable as fundamental research is, it is useless without the proper field of application. By coincidence, a prospective user (group) in the Erasmus

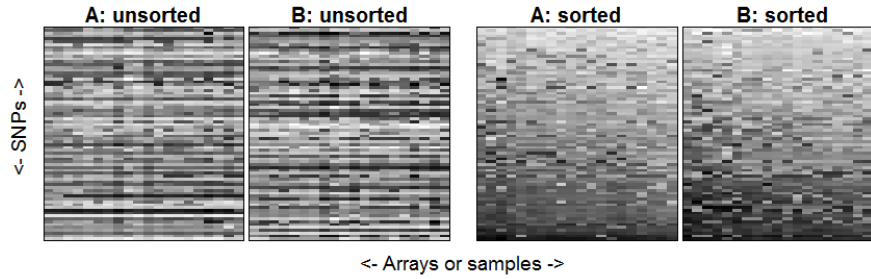


Figure 1.6: Selection of raw SNPs and arrays, based on a small two-color Illumina set. Left: signals unsorted. Right: both signals sorted for rows and columns of A allele intensities.

Medical Center became involved in the development of a tailor-made statistical solution. A short discussion evolved into a more extensive project, from which all parties learned a lot. Detailed communications resulted in more questions as well as theoretical and procedural suggestions.

This thesis introduces a number of new statistical models and algorithms:

- The SCALA model that contains parameters for estimating the systematic effects of SNPs, arrays and genotypes. This model is applied to both alleles.
- The model leads to an extremely large linear regression problem with millions of observations and possibly a million parameters or more. However, it has an extremely sparse structure. A specialized semi-symbolic algorithm allows exact estimation in very short time.
- Once the parameters of the model have been estimated, they are used to eliminate the systematic effects, thereby greatly enhancing the quality of the fluorescence signals. This is called calibration. Further analysis, like genotyping, or copy number estimation is improved.
- It is common practice to perform genotyping SNP by SNP, using relatively large sets of arrays. In this monograph we break with this tradition and perform genotyping for all SNPs on individual arrays. A

semi-parametric mixture model is fitted, with three component densities, one for each of the AA, AB and BB genotypes. Comparison to results of established SNP by SNP algorithms, as found on the HapMap archive shows equal or better performance.

- An interesting part of the SNP practice are so-called waves. When fluorescence signals representing copy numbers are plotted along chromosomes and smoothed, a systematic wavelike pattern becomes visible. A common misunderstanding is that this is a manifestation of a real, slowly changing, wavelike spatial structure. It is shown that the latter is not the case: after calibration the waves disappear.

- Copy number variation (CNV) generally occurs in a segment-wise manner. There is a large literature on smoothing and segmentation of CNV signals, in order to obtain the boundaries of the segments and their levels. A new smoothing algorithm is presented that uses a so-called L_0 penalty on jumps between smoothed values. The result is an extremely sharp segmentation, with extremely smooth segments in between: the segments are constant.

- It is not possible to apply the same smoother to allelic imbalancesignals, because several parallel data bands occur. An existing scatterplot smoother was modified to get sharp segmentation here too.

- The basis of the genotyping method proposed in this monograph is a display of the ratio of the A and B signals versus their sum (on logarithmic scales). Low signals on the sum scale as well as unclear separation between the three genotype groups on the ratio scale indicate low(er) chip quality. The proposed approach to individual arrays is very useful to exploit this knowledge to select only the SNP observations of the highest quality, by a user-defined threshold.

- All models and algorithms are written in R, and combined in a software suite, called SCALA. SCALA also provides both command-line functions (for estimation and calibration, as well as genotpying) and a graphical user interface for interactive (simultaneous) smoothing and plotting of

CNV and allelic imbalance. It can convert DNA array files from different platforms. By user request evaluative and interactive graphs can be created, as well as customized numerical output.

1.6 Thesis outline

After this introduction follow six additional chapters. Chapter 2 to 7 have been written as individual papers, and chapter 8 is a discussion chapter. The software is extensively described in a special section that contains two more individual papers. These address the implementation of most of the concepts and models from chapters 2 to 6.

Chapter 2 describes the linear SCALA models that estimate the calibration parameters. We exploit the structural properties of SNP-specific intensities over arrays in a model that models an overall genotype parameter (over all SNPs in the whole set of arrays) for the three genotypes as well as a model that estimates a set of three genotype parameters for each SNP. Signal calibrations using both types of model are derived and illustrated.

Chapter 3 frames genotyping using a single array against common procedures that call genotypes using a single SNP from multiple arrays. A semi -parametric model is derived that fits three log-concave densities on a two-dimensional histogram.

Chapter 4 discusses the possibilities of removing SNP signals of low quality. Performance for genotyping is assessed in terms of sample call-rate, while performance for CNV calling is illustrated visually, rather than numerically. The proposed low signal filter show improvements for both applications.

Chapter 5 shows that so-called in a copy number signal can be removed effectively using the signal calibration from chapter 2. SNP signals are transformed into aCGH-resembling signals, so that performance by SCALA can be compared with a specialized method for aCGH signals, NoWaves.

Chapter 6 discusses a single-array L_0 penalty signal smoother, ZEN, for CNV signals. Existing methods either need a set of reference arrays, or in single array models, small (random) fluctuations in the signal are modelled,

while ideally only significant signal jumps should be modelled. Performance of the L_0 norm is compared to a multi-array method, VEGA and applications in CNV estimation, estimations of allelic imbalance and scatterplot smoothing are shown.

Chapter 7 showcases the SCALA software suite that was developed based on the ideas and concepts in Chapter 2 to 6. It provides a short theoretical introduction on the implemented models and functions, then discusses details and options for each function, and concludes with an extensive example.

Chapter 8 consists of three parts. It provides a short summary, an overall discussion and suggestions for future research.

Research Articles

Correction of Fluorescence Bias on Affymetrix Genotyping Microarrays

2

Fluorescence signals obtained from microarrays for SNP genotyping show systematic strong variations in the levels for SNPs and arrays as well as genotypes. Linear models that take all three effects into account fit very well. Once the model parameters have been estimated for a set of reference arrays, they can be used to calibrate new arrays in a simple way, thereby improving genotyping and analysis of copy number variations and allelic imbalance.

2.1 Introduction

Probably the largest scale application of fluorescence these days is the use of microarrays for gene expression, or for genotyping of single nucleotide polymorphisms (SNPs, pronounced as “snip"). A modern SNP microarray contains millions of spots or small beads, called probes, that are covered with small strings of the four nucleotides A, C, G and T that are the building blocks of DNA. Each string is constructed to be the complement (A to T, C to G, and vice versa) of the specific sections of the (human) genome on which SNPs occur.

In a preliminary step, DNA is fragmented by a specific enzyme. The fragments selectively bind (hybridize) to the complementary probes. The amount that hybridizes is, within certain limits, proportional to the concentration of the DNA segments. By preparation with biotin before hybridization, and by

This chapter was published as the article:
Rippe, R.C.A., Eilers, P.H.C. and Meulman, J.J. (2012). Correction of Fluorescence Bias on Affymetrix Genotyping Microarrays, *Journal of Chemometrics*, **26**: 191–196. doi: 10.1002/cem.2436.

attaching a fluorophore after, it becomes possible to quantify concentrations by measuring fluorescence intensities. A high-resolution image is formed by scanning the surface of the array with a laser, and the intensities at the probe spots are quantified.

SNPs generally have two variants, called alleles, and the probes are selective to each of the alleles. If we indicate alleles of one SNP by A and B, there will also be two fluorescence signals for each SNP, which we indicate by a and b. The DNA of humans (which we consider here), but also that of many other organisms, is contained in two chromosomes. There are three possible combinations of alleles, AA, AB and BB. It is not possible to discern BA from AB, so there is no fourth combination. These combinations are called genotypes.

SNP arrays have two main applications: 1) genotyping of normal DNA, and 2) detection of aberrations, so-called copy number variations (CNV), in tumor DNA. In the first case the result is either AA, AB or BB. Copy number variations allow, in principle, a combination of any number (from zero to five or more) of As and Bs. Usually, if these aberrations occur, they occur in many adjacent positions on the chromosome: whole regions show aberrations in copy number. Franke et al. (2008) describe CNV and its origin in more detail.

We expect the signal a to be proportional to the concentration of the A allele, so only two levels, say $a = a'$ (genotype AB) and $a = 2a'$ (genotype AA) should occur (and a very small background signal in case of genotype BB). For the b signal we similarly expect $b = b'$ (genotype AB) and $b = 2b'$ (genotype BB). Under ideal circumstances, a' and b' should be the same for all SNPs. While working with the fluorescence signals of several types of SNP arrays, we discovered that this is not the case; a strong SNP-dependent bias exists. However, the size of this bias, which is characteristic for each SNP, can be estimated reliably, with a linear regression model, applied to a training set of microarrays. Once the parameters of the model have been estimated, they can be used to correct the bias in new arrays, a procedure we call calibration.

We present two models, one which leads to parameters that can be used

to calibrate a new array without knowledge of the SNP genotypes of a new biological sample. An extended model uses this information and allows for somewhat better calibration. However, it can only be used when genotypes are available, which limits its usefulness to special situations, for instance as a building block that iteratively combines calibration with genotype estimation.

The main purpose of this paper is to introduce the model, to show the effect of the calibration procedure, and to illustrate its potential for more precise copy number estimation. Because the regression model is huge (a million parameters or more, derived from approximately 50 million data points), we pay extra attention to efficient calculation.

The organization of the paper is as follows. In the next section we introduce our models, estimation algorithms, model fit and the resulting calibration. We close the paper with a Discussion.

2.2 Methods

Data

We use Affymetrix microarrays. The source of our data is the HapMap (www.hapmap.org) archive (The International HapMap Consortium, 2003, 2007). It provides three types of Affymetrix data files, which are mainly distinguished by the number of SNPs they measure. The oldest platform is the 100k chip, which measures 50.000 SNPs in two different sub-chips: one using the Hind enzyme to cut the DNA into fragments, the other using the Xba enzyme. A newer generation is the 500k chip, which measures 2 x 250.000 SNPs using the NSP or STY enzyme respectively. The final and most recent chip is called SNP6.0, which measures 1.000.000 SNPs in one sample.

We only describe results for the SNP6.0 array, because this is the most recent one. We also analyzed other types of arrays (100k Hind and Xba, 500k NSP and STY) and the results were essentially the same.

Procedures

The image that is obtained by laser scanning is summarized by averaging the fluorescence intensity over all pixels that belong to one probe. The numbers are collected in a so-called CEL file. Although, to simplify the presentation, we described the technology as if there is one probe per SNP allele, in reality there are four on SNP 6.0 arrays. We simply average the intensities of the four probes to get one number for each of the two alleles of each SNP. This gives us two vectors, each of length p.

2.3 Models and estimation

In this section we describe the data in more detail. We first explain prior transformation of the raw data. Then we develop two models. One we call "global" because it summarizes the effect of the genotype by just three parameters (per allele) for all SNPs. The second model is called "local" because it has three parameters (per allele) for each SNP. We have to fit the models to very large data matrices (a million SNPs and 90 arrays). We present an efficient semi-symbolic algorithm.

Two linear models

We denote the number of SNPs by p and the number of arrays (each based on one biological sample) by n. The raw fluorescence measurements are contained in two $p \times n$ matrices $\boldsymbol{A} = [a_{ij}]$ and $\boldsymbol{B} = [b_{ij}]$, one for each allele. A careful study of images of these matrices shows three things:

- Some rows are systematically brighter than others, so each SNP appears to have its own level of brightness.
- Some columns are brighter than others; this is related to the quality of the DNA and its handling in the laboratory. Thus, each array has its own level of brightness.
- Brightness is modulated by the number of alleles (0, 1 or 2).

As a first approximation, it is reasonable to assume multiplicative effects of SNP level, array level and number of alleles. Hence a linear model for the logarithms of the fluorescence intensity is expected to work well.

Let $t_{ij} = \log(a_{ij})$, where the logarithms are to base 10. Let the genotypes be coded in the 3-way indicator matrix $\boldsymbol{H} = [h_{ijk}]$, where $k \in \{1, 2, 3\}$ codes for the genotype; $h_{ijk} = 1$ if SNP i on array j has genotype k, otherwise $h_{ijk} = 0$. The model is written as

$$t_{ij} = \mu + \alpha_i + \beta_j + \sum_{k=1}^{3} \gamma_k h_{ijk} + e_{ij}, \tag{2.1}$$

where μ is the grand mean, α_i the effect of SNP i, β_j the effect of array j, and γ_k the effect of genotype k. For identifiability, we introduce the constraints $\sum_i \alpha_i = 0$ and $\sum_j \beta_j = 0$. The error $e = [e_{ij}]$ is assumed to have constant variance. This is a simplifying assumption, but it cannot do much harm. We are only interested in point estimates of the model parameters, not in their standard errors.

We call the model in (2.1) the global model, since it has one set of genotype parameters (γ) for all SNPs. A refinement is to have separate genotype parameters for each SNP: $\boldsymbol{\Gamma} = [\gamma_{ik}]$. We call this the local model, which is specified as

$$t_{ij} = \mu + \beta_j + \sum_{k=1}^{3} \gamma_{ik} h_{ijk} + e_{ij}, \tag{2.2}$$

where we again require that $\sum_j \beta_j = 0$.

Identical models are used for the B allele, with $t_{ij} = \log(b_{ij})$. As said, we are interested in standard errors per se. Nevertheless, it is good to have a rough estimate. If the estimated α is unreliable, using it for bias correction might introduce additional variance, with detrimental effects. Let us assume that we use 90 arrays, obtained from the HapMap site (www.hapmap.org) to calibrate the model. With many thousands of SNPs, the degrees of freedom consumed by estimating β and γ are negligible, so $\hat{\alpha}$ for an individual SNP is roughly determined by averaging over 90 arrays. Its variance will thus be approximately 1/90th of the variance of the noise on an individual array. Hence we conclude that we do not have to worry about introducing extra variance.

Parameter estimation

The arrays we are analyzing here cover up to a million SNPs each. To get parameter estimates, we apply the model to an available set of 90 arrays. Hence we have millions of data points and a huge number of parameters: approximately one million for the global model and triple that number for the local model. Our models can be written as regression models, but explicit construction of the design matrix and invoking a regression procedure is not a good idea: the design matrix would have many billions of elements. However, it is very sparse, so a better solution would be to use sparse matrix software. We have not tried this approach, so we cannot report on its effectiveness. Instead, we have explored block relaxation and symbolic solutions of the regression equations.

In both models (2.1) and (2.2) it is easy to compute one set of parameters if the rest is available. One simply has to average residuals, over SNPs, arrays or genotypes, dependent on the type of parameters. Departing from reasonable starting values (averages over SNPs for $\boldsymbol{\alpha} = [\alpha_i]_{i=1}^{p}$, averages over arrays for $\boldsymbol{\beta} = [\beta_j]_{j=1}^{n}$), one iteratively updates each set of parameters. In the numerical analysis literature this is known as block relaxation.

Alternatively one can build and solve the normal equations symbolically. We illustrate this for the local model (2.2). With appropriate $\boldsymbol{C}$ and $\boldsymbol{D}$, we can write

$$\boldsymbol{t} = \boldsymbol{C}\boldsymbol{\beta} + \boldsymbol{D}\boldsymbol{\gamma} + e \tag{2.3}$$

where $\boldsymbol{\beta}$ contains the n β_j parameters in (2.2) and γ = vec($\boldsymbol{\Gamma}$), i.e. the columns of $\boldsymbol{\Gamma} = [\gamma_{ik}]$ stacked below each other, and $\boldsymbol{t} = \text{vec}(\boldsymbol{T})$. The structure of $\boldsymbol{C}$ is simple, it can be written as $\boldsymbol{C} = \boldsymbol{I}_n \otimes \boldsymbol{1}_p$, where $\boldsymbol{I}_n$ is the $n \times n$ identity matrix and $\boldsymbol{1}_p$ is a vector of ones, of length p. The structure of $\boldsymbol{D}$ is more complex; it consists of n blocks of diagonal matrices. Each block has three diagonal matrices $\boldsymbol{D}_{jk}$, one for each layer of $\boldsymbol{H}$, and each matrix $\boldsymbol{D}_{jk}$ contains the elements of the jth vector in the kth layer of the 3-way matrix $\boldsymbol{H}$ on its diagonal. Thus, $\boldsymbol{D}$ has dimensions $(n \times p) \times 3p$.

We don't form $\boldsymbol{C}$ and $\boldsymbol{D}$ explicitly. Instead we study the normal equations

$$\begin{bmatrix} \boldsymbol{C}'\boldsymbol{C} & \boldsymbol{C}'\boldsymbol{D} \\ \boldsymbol{D}'\boldsymbol{C} & \boldsymbol{D}'\boldsymbol{D} \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{\beta}} \\ \hat{\gamma} \end{bmatrix} = \begin{bmatrix} \boldsymbol{C}'\boldsymbol{t} \\ \boldsymbol{D}'\boldsymbol{t} \end{bmatrix}, \tag{2.4}$$

or

$$\begin{bmatrix} \boldsymbol{V_{11}} & \boldsymbol{V_{12}} \\ \boldsymbol{V_{21}} & \boldsymbol{V_{22}} \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{\beta}} \\ \hat{\gamma} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f_1} \\ \boldsymbol{f_2} \end{bmatrix}, \tag{2.5}$$

where $\boldsymbol{V}_{11} = \boldsymbol{C}'\boldsymbol{C}$, $\boldsymbol{V}_{12} = \boldsymbol{C}'\boldsymbol{D}$, $\boldsymbol{V}_{21} = \boldsymbol{D}'\boldsymbol{C}$, $\boldsymbol{V}_{22} = \boldsymbol{D}'\boldsymbol{D}$, $\boldsymbol{f}_1 = \boldsymbol{C}'\boldsymbol{t}$ and $\boldsymbol{f}_2 = \boldsymbol{C}'\boldsymbol{t}$. One can prove that $\boldsymbol{C}'\boldsymbol{C} = p\boldsymbol{I_n}$, $\boldsymbol{D}' = \tilde{\boldsymbol{H}}$ and $\boldsymbol{D}'\boldsymbol{D} = \boldsymbol{F}$, where $\tilde{\boldsymbol{H}}$ is a matrix formed by placing the three layers of $\boldsymbol{H}$ below each other. $\boldsymbol{F}$ is a $3p$ by $3p$ diagonal matrix; its first (second, third) p diagonal elements gives, for each SNP, the number of times genotype 1 (2, 3) occurs. Furthermore, $\boldsymbol{C}'\boldsymbol{t}$ contains the sums of the columns of $\boldsymbol{T}$, while $\boldsymbol{D}'\boldsymbol{t}$ is a stack of three vectors; the first (second, third) vectors contain the sum, per SNP of the elements of $\boldsymbol{t}$ corresponding to genotype 1 (2, 3).

From (2.5) follows:

$$\hat{\gamma} = \boldsymbol{V}_{22}^{-1}(\boldsymbol{d}_2 - \boldsymbol{V}_{21}\hat{\boldsymbol{\beta}}) \tag{2.6}$$

and hence

$$(\boldsymbol{V}_{11} - \boldsymbol{V}_{12}\boldsymbol{V}_{22}^{-1}\boldsymbol{V}_{21})\hat{\boldsymbol{\beta}} = \boldsymbol{d}_1 - \boldsymbol{V}_{12}\boldsymbol{V}_{22}^{-1}\boldsymbol{d}_2. \tag{2.7}$$

Because $\boldsymbol{V}_{22}$ is a diagonal matrix, multiplication by $\boldsymbol{V}_{22}^{-1}$ boils down to dividing the elements of a vector or the rows of a matrix by the corresponding diagonal elements of $\boldsymbol{V}_{22}$. Hence, it is not hard to compute $\boldsymbol{V}_{11} - \boldsymbol{V}_{11}\boldsymbol{V}_{12}^{-1}\boldsymbol{V}_{21}$ and to solve for $\hat{\boldsymbol{\beta}}$, a vector of moderate length. Additional efficiency can be realized by exploiting the way $\boldsymbol{V}_{21}$ is formed. Details on the latter suggestion are considered outside the scope of the current paper.

In this analysis we have ignored the fact that the system in (2.5) is singular, because the condition $\sum_j \beta_j = 0$ is not applied. One way to handle this restriction is to introduce a Lagrange multiplier, λ and extend the objective function of the model (the sum of squares of differences between observed and fitted values) with $\lambda \sum_j \beta_j$. The system of equations (2.8) becomes

$$\begin{bmatrix} \boldsymbol{C}'\boldsymbol{C} & \boldsymbol{C}'\boldsymbol{D} & \boldsymbol{e} \\ \boldsymbol{D}'\boldsymbol{C} & \boldsymbol{D}'\boldsymbol{D} & \boldsymbol{0} \\ \boldsymbol{e'} & \boldsymbol{0} & 0 \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{\beta}} \\ \hat{\gamma} \\ \lambda \end{bmatrix} = \begin{bmatrix} \boldsymbol{C}'\boldsymbol{t} \\ \boldsymbol{D}'\boldsymbol{t} \\ 0 \end{bmatrix}, \tag{2.8}$$

where $\boldsymbol{e}$ is a vector of ones and $\mathbf{0}$ a matrix of zeros.

An somewhat easier solution is to demand the minimum-norm solution for $\boldsymbol{\beta}$, by replacing $\boldsymbol{C}'\boldsymbol{C}$ in (2.5) by $\boldsymbol{C}'\boldsymbol{C} + \kappa\boldsymbol{I}$ with κ a small number. This is the approach we have chosen, using $\kappa = 10^{-6}$.

2.4 Results

In this section we describe model fit and effect of possible model-based calibration using parameters from the two models described above. All computations were done with R scripts (R Development Core Team, 2012).

Model fit

In our experience the speed of convergence is quite good: from 10 to 30 iterations generally suffice to find changes in the updates in the order of 10^{-6} (relative size). The constraints on $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are applied in each iteration.

Running the implementation of the aforementioned symbolic model on an Intel Core2 Duo 1.4 GHz processor takes about 50 seconds for 90 Affymetrix 100k Hind arrays (10^5 SNPs). A larger dataset (63 arrays with 10^6 SNPs from SNP6) takes 220 seconds.

A typical model fit is shown in Figure 2.1. The standard deviations of the residuals are approximately 0.063 for the global model and 0.051 for the local model. These results are for a set of Affymetrix SNP6.0 arrays. Similar results were obtained for the Affymetrix 100k and 500k arrays. To reduce visual clutter by too many data points in the scatterplots, results for only one chromosome are shown.

The random variation around the fitted line is larger for the BB genotype than for others. The graphs show the fit to the logarithm of the fluorescence signal for the A allele, which is small if the the genotype is BB, as can be seen from the positions of the centers of the clouds of observations. It is well known that constant addtive noise errors appear as increasing relative to low signal values. From the graphs we can deduct that the assumption of

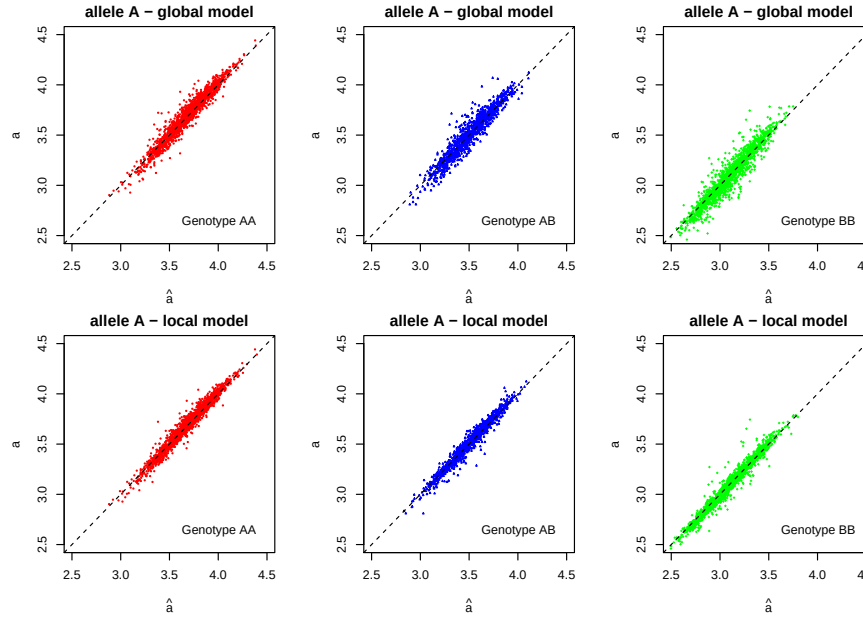


Figure 2.1: Results for a selected SNP6.0 array on chromosome 1. Model fit for one allele (A). Top panels show the fitted versus original signals in the global model. Bottom panels show the results for the local model. For allele B, the results are similar.

constant variance of the errors is a simplification, but not an extreme one. We will return to more advanced error models in the Discussion.

Model-based calibration

From the model we obtain, for each color, either a vector $\hat{\boldsymbol{\alpha}}$ (global model) or a matrix $\hat{\boldsymbol{\Gamma}}$ (local model). We can use them to calibrate the signals of new arrays. Assume that we add one or more columns to our data matrix, representing new SNP arrays, which have not been used for model fitting. Let l indicate one of these columns. For the global model, we compute $t^*_{il} = t_{il} - \hat{\alpha}_i$, to correct for the SNP effects. If we wish to correct for the array effect, we can compute $\hat{\beta}_l$ such that $\sum_i (t^*_{il} - \hat{\beta}_l - \hat{\mu}) = 0$. In our applications we do not need

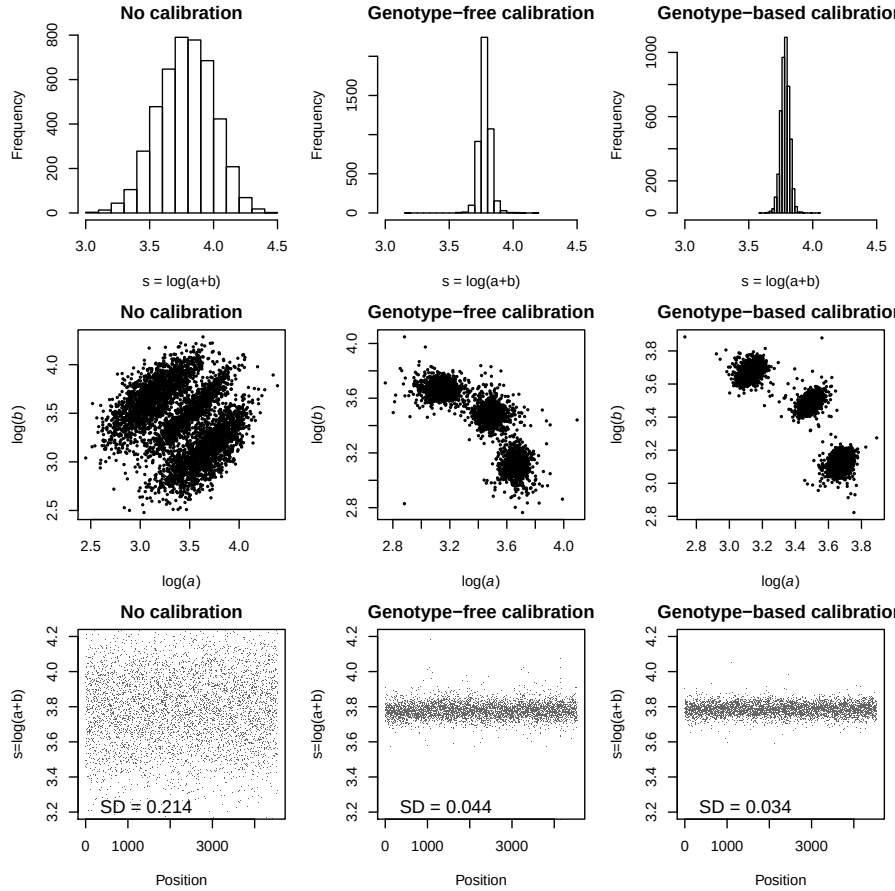


Figure 2.2: Top row: histograms of $log(a+b)$. middle row: scatterplots of $log(b)$ against $log(a)$. Bottom row: $log(a+b)$ against chromosomal positions. Standard deviations for raw signal (left), signal after genotype-free calibration (middle) and genotype-based calibration (right). Genotype-free calibration reduces noise considerably, genotype-based calibration provides a small further improvement.

this calibration, because we study single arrays, but this might not be true in other applications.

When we do not use the genotypes of the new array, we call this *genotype-free* calibration. If the genotypes are available we can use the results of the local model, by computing $t^*_{il} = t_{il} - \sum_k h_{ilk}\gamma_{ik}$. We call this *genotype-based* calibration. To calibrate for the array effect, one computes $\hat{\beta}_l$ such that $\sum_i (t^*_{il} - \hat{\beta}_l - \sum_k h_{ilk}\hat{\gamma}_{ik}\hat{\mu}) = 0$.

In Figure 2.2 we show how calibration improves the bandwidth (standard deviation) of the SNP signal. Histograms of $\log(a+b)$ show reduced standard deviations, the middle scatterplots show more condensed (genotype) clusters, and the bottom scatterplots now show $\log(a+b)$ against the position on the chromosome. The improvement from uncalibrated to genotype-free calibrated signal is major, while genotype-based calibration provides a smaller additional improvement.

Genotype-free calibration is less precise, but it can be used for new samples, for which genotypes generally are not available. We propose that the model parameters are estimated once for a set of high-quality DNA samples. The parameters so obtained can be used to calibrate all future arrays.

Genotype-based calibration is less generally useful, but we can envisage a multi-step procedure in the context of genotyping. The first step for any new array is to perform genotype-free calibration. The next step is to determine genotypes using the calibrated signals. Given these genotypes, genotype-based calibration can be performed, followed by a second round of genotyping. Of course, there is the danger of a self-fulfilling prophecy, so only careful testing can show the performance of this recipe. We consider the latter outside the scope of the current paper.

In the next section we show how genotype-free calibration might improve detection of CNV and allelic imbalance along chromosomes.

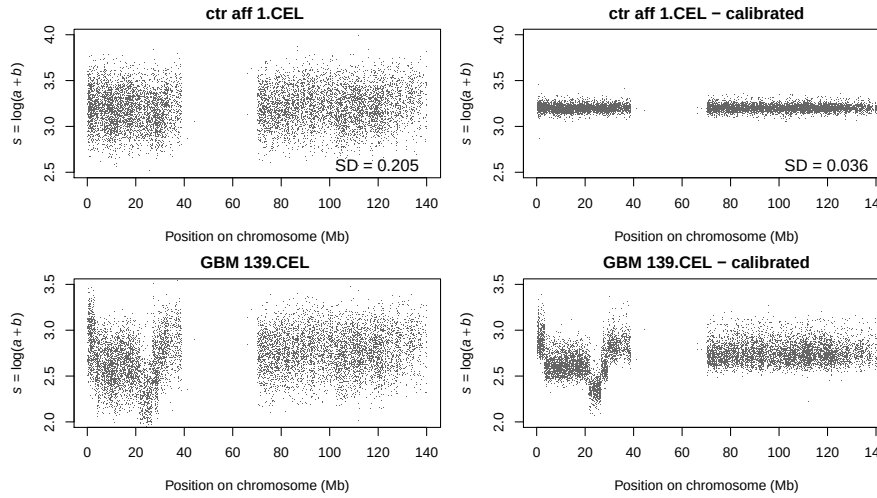


Figure 2.3: CNV plot for chromosome 9, based on an Affymetrix 500k NSP array. Top panels: normal tissue; bottom panels: brain tumor. The dots show the (calibrated) signal s.

2.5 Application: CNV and imbalance maps

An application of SNP signals in tumors is to graph signal levels against their position on the chromosome. Most interesting are copy numbers (the sum of the a and b signals) and allelic imbalance(their ratio). We show possible improvements using genotype-free calibration with our model.

Figure 2.3 shows CNV data for chromosome 9, for normal tissue and for a brain tumor, obtained from the Rotterdam Erasmus Medical Center (Bralten et al., 2010). Figure 2.4 shows allelic imbalance, $\log(b/a)$, before and after calibration. The improvement is evident.

The correction is more effective when the overall signal is strong, because then the systematic SNP effects are strong. In low-quality arrays noise is more dominant, and we cannot correct that with calibration. This can be seen in the figures 2.3 and 2.4. The sample from Affymetrix is of better quality than the tumor sample.

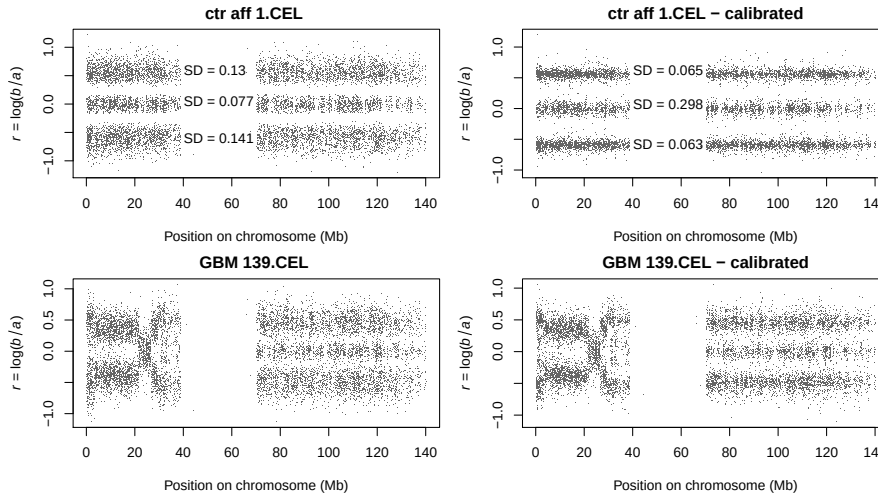


Figure 2.4: Data on chromosome 9. Illustration of improvement in signals for allelic imbalance after calibration (right panels). In the top panels, we see three bands (one for each genotype) in a control sample, whereas in the bottom panels we see just two bands in the left side signal (P-arm). Deviations to the three-band signal indicate problematic DNA.

2.6 Conclusions and Discussion

We have described two models for systematic effects of SNPs, arrays and genotypes in fluorescence signals on microarrays. The first model contains overall genotype effects and the second contains SNP-specific genotype effects. The parameter estimates following from these models were used to calibrate the raw fluorescence signals. Calibration removes apparent noise in signal maps.

The calibration we propose is simple, fast, and effective. Once parameters for the global model have been estimated, based on a set of high-quality reference arrays, calibration entails only the correction of probe summaries by a single number, per allele of each SNP. This has to be done only once, whether one in interested in genotypes, copy number variations, or both. Each array can be calibrated in isolation, in less than a second.

The idea to model systematic effects and using these to calibrate signals, for improved downstream processing, is not new, see RLMM (Rabbee and Speed, 2006), BRLMM (Affymetrix, 2006) and CRLMM (Carvalho et al., 2007, 2010). These procedures were developed for genotyping and they have in common that they demand a relatively large set of arrays to work reliably and only correct signals for that set. A similar procedure has been developed for CNV (Bengtsson et al., 2008).

We developed our calibration model for, and applied it to, Affymetrix microarrays. It will be interesting to see how it will perform for Illumina arrays. This will be not too hard, because only summary data (per allele, per SNP) are needed.

One of the assumptions of the model is a constant error variance. As Figure 2.1 shows, this is only approximately true: the variance appears to increase for weaker fluorescence signals. This quite common when taking logarithms. A more advanced approach would be to model the relationship between expected value of the model fit and the variance and the error, like in the error model of Rocke and Durbin (2003). Although this would improve the model, we expect little change in the estimated model parameters if estimation is based on a relatively large number of arrays (like the 90 we use here).

Our approach is completely pragmatic: we postulate a model, estimate its parameters, observe a good fit and use the results for calibration. But there should be a fundamental explanation of the very stable systematic patterns we observe. Further research is needed for better understanding.

A software package, named SCALA, is presently available from the corresponding author. It was written for the R system (R Development Core Team, 2012), and has been used for all the analyses mentioned in this paper. We plan to turn it into a Bioconductor package. The software contains a module to convert Affymetrix CEL file data to the aggregated signals that were used in this paper, and a module to estimate the calibration parameters. A module that creates the copy number and allelic imbalance maps including a signal smoother is available. There also is a module for genotyping of single arrays.

Single chip genotyping with semi-parametric log-concave mixtures

3

The common approach to SNP genotyping is to use (model-based) clustering per individual SNP, on a set of arrays. Genotyping all SNPs on a single array is much more attractive, in terms of flexibility, stability and applicability when developing new chips. A new semi-parametric method, named SCALA, is proposed. It is based on a mixture model using semi-parametric log-concave densities. Instead of using the raw data, the mixture is fitted on a two-dimensional histogram, hence making computation time almost independent on the numbers of SNPs. Furthermore, the algorithm is effective in low MAF situations.

Comparisons between SCALA and CRLMM with HapMap genotypes show very reliable calling of single arrays. Some heterozygous genotypes from HapMap are called homozygous by SCALA and to lesser extent by CRLMM too. Furthermore, HapMap's NoCalls (NN) could be genotyped by SCALA, mostly with high probability.

3.1 Introduction

Genotyping algorithms for SNP chips can be partitioned roughly into two classes: 1) those that call genotypes for individual SNPs for a set of arrays and 2) those that call all SNPs for a single array.

The first approach is the common one: for each SNP it collects the pairs of fluorescence intensities for all arrays and applies a clustering algorithm. This is known as multi-array genotyping. However, one major disadvantage

This chapter is an adapted version of the article:
Rippe, R.C.A., Meulman, J.J. and Eilers, P.H.C. (2012). Reliable single chip genotyping with semi-parametric log-concave mixtures, *PLoS ONE*, to appear.

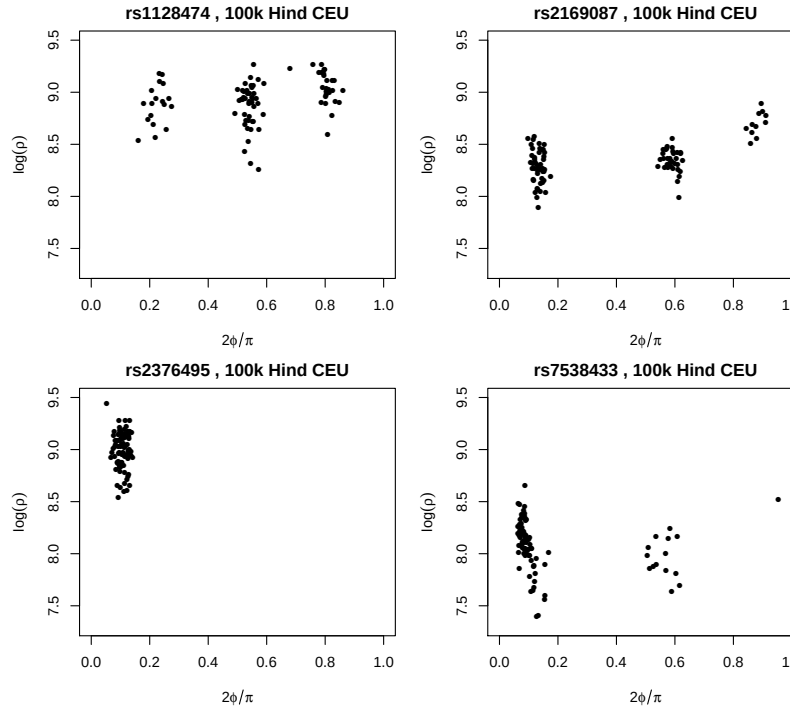


Figure 3.1: Multi-array genotyping for four separate SNPs in a sample set from the CEU HapMap population. Top row: a clear three genotype division without minor allele frequency problem. Bottom row: genotype clusters with minor allele frequency problems.

is that the number of available data points is limited to the number of samples: fewer data generally yield less reliable results. The latter problem is especially troubling if the SNP has a very low minor allele frequency (MAF), the allele that has the lowest frequency in a given population. Low MAFs are known to have a detrimental effect on downstream analyses. Tabangin et al. (2009) describe the latter in the genome-wide association scans, but the results extend to other areas as well. Therefore, HapMap only targets MAFs with a population occurrence of 5% or higher.

In cases of low MAF, there are very few or even no observations in a given

Table 3.1: Frequencies of total number of different genotypes, for the set of HapMap arrays from the CEU population. Genotypes are obtained from HapMap and from the CRLMM algorithm. 13% shows only one, 25% two, and 62% three different genotypes.

# of different genotypes	1	2	3	Total
HapMap Calls (raw)	119186	223413	564001	906600
HapMap Calls (%)	13.2	24.6	62.2	100.0
CRLMM Calls (raw)	119666	195061	591873	906600
CRLMM Calls (%)	13.2	21.5	65.3	100.0

cluster. Figure 3.1 compares four SNPs. In the top row we see SNPs that have a very clear three-genotype structure, while in the bottom row we encounter genotyping problems. The panel at the left shows just a single cluster, while the third cluster in the right panel contains only one observation. A data transformation similar to that used in Illumina Beadstudio was applied. In this transformation the two signals for the two alleles are first transformed to polar coordinates (ϕ, ρ) and displayed on modified scales: $2\phi/\pi$ and $\log_{10}(\rho)$.

It is clear that based on these (90) samples from the Central European (CEU) population, genotype calls for some SNPs can hardly be made effectively without the use of reference samples. It is this problem that causes a 'No Call' for some SNPs due to high uncertainty (where the 'No Call' threshold is set by the software that is used to obtain the calls). For these reasons, common calling algorithms like BirdSeed (Korn et al., 2008) require 100 or more samples with known genotypes to train the model, while BRLMM-P and CRLMM (Carvalho et al., 2007, Rabbee & Speed, 2006) require both a large number of samples as well as presence of all three genotypes AA, AB and BB. Table 3.1 shows that for genotypes obtained from HapMap and from CRLMM in a set of HapMap CEU arrays, a large proportion of SNPs only have one or two different genotypes: around 35% of the SNPs lack observations in all three clusters. In the current CEU arrays low MAFs follow a distribution described in Table 3.1 , which indicates the extreme and discrete MAF cases; the first column shows monomorphic MAF.

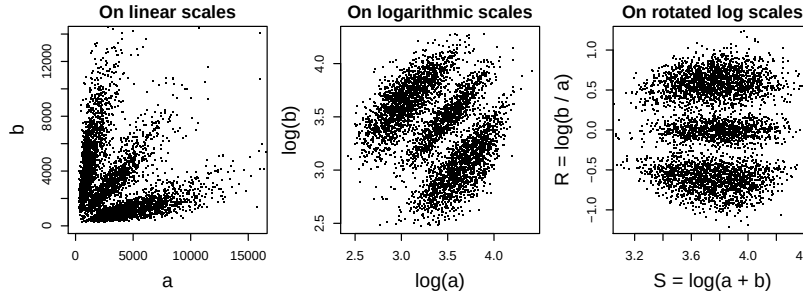


Figure 3.2: Illustration of signal transformation. Signal a (b) represents allele A (B). The left panel shows the signals on linear scales. The middle panel shows the same signals on logarithmic scale. The right panel shows transformed signals to $s = \log(a + b)$ on the x-axis and $r = \log(b/a)$ on the y-axis.

To overcome the lack of observations for some SNPs, CRLMM has the option to include prior information in the model, in case of low MAF: small genotype clusters are estimated using prior cluster locations. However, in Figure 3.1 is is clear that clusters for the same genotype in different SNPs are not in the same location.

The second approach to genotype calling is to determine genotypes based on the two allele signals for all SNPs on a single array: single-array genotyping. We find it convenient to transform the allele channel signals to $s = \log(a + b)$ and $r = \log(a/b)$ where a and b are fluorescence signals for allele A and B respectively (logs are to base 10). After this transformation (see Figure 3.2), three horizontal clusters are present, which correspond to the three possible genotypes. In Figure 3.3 results of the transformation are shown for two typical Affymetrix (HapMap) arrays and two typical Illumina arrays (source: department of Epidemiology, Erasmus Medical Center, Rotterdam, The Netherlands).

Mixtures have been explored by other researchers. Wright et al. (2010) describe a procedure called ALCHEMY which does *de novo* calling for small sets of samples. For each allele they introduce one-dimensional mixtures of normal distributions, one component for noise (when the allele is absent) and the other for the signal (when the allele is present). Wright et al. work

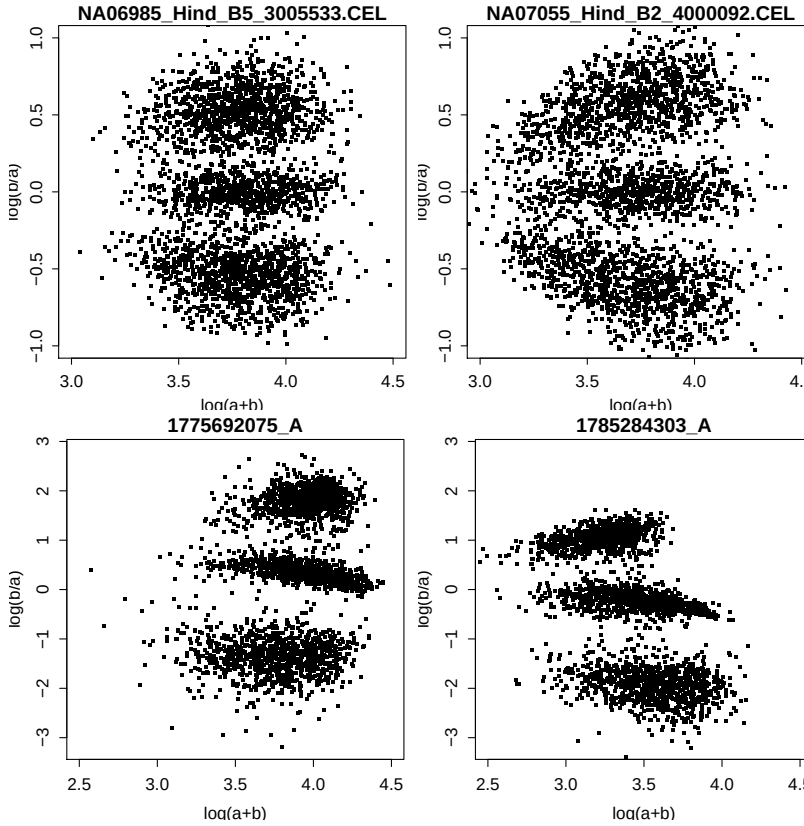


Figure 3.3: Single HapMap Affymetrix 100k Hind samples (NA06985, NA07055 from left to right) in top panels, typical Illumina arrays in bottom panels. SNPs are shown for chromosome 1.

in the context of rice genotyping. They give an instructive overview of the problems connected to per SNP genotyping, one of them being the absence of heterozygous genotypes, due to inbreeding.

Along similar lines, Xiao et al. (2007) introduce an approach that combines multi-SNP and multi-array genotyping, called MAMS. Their first step performs model-based clustering on all SNPs in a single array and the second step applies multi-array refinement of selected SNPs with unique hy-

bridization properties (different from most SNPs). They fit mixtures of two-dimensional normal distributions. This is a time-consuming process, so they have to rely on sampling to get acceptable processing times. Giannoulatou et al. (2009) describe a single-array genotyping algorithm GenoSNP, but their procedure and implementation are limited to Illumina chips only.

We will show that excellent platform-independent genotyping can be obtained from single arrays by fitting a mixture of three nonparametric two-dimensional distributions. We describe a fast algorithm and show its performance on HapMap data.

In the next section we describe the theoretical basis of our approach. In the Appendix we describe how we obtained and pre-processed HapMap data to be able to measure performance. Section 3.3 presents the results. We finish with a short Discussion.

3.2 Semi-parametric single-array genotyping

In this section we describe how we fit a mixture of three two-dimensional semi-parametric log-concave densities to transformed fluorescence signals, as illustrated in Figure 3.3. In the case of an Affymetrix array the signals are summaries of probe sets, so we do not try to exploit any patterns in the signals from the individual probes. The reason is simple: we have no need for it. To avoid scatter plots becoming almost completely black, we use data from one chromosome. This is only for illustrational purposes; it should be understood that all SNPs on one array are genotyped at the same time. Figure 3.4 illustrates the genotype cluster shapes for a selection of chromosomes as well the shapes for the complete array. As can be seen, they are very similar.

We describe in some detail how to fit a mixture of log-concave densities in one dimension, borrowing from Eilers & Borgdorff (2007). Then we sketch the procedure in two dimensions.

To compute a smooth density for a one-dimensional data set, we first construct a histogram with many bins, say $n = 100$. Let y_i denote the count in bin i of the histogram and let u_i be the bin midpoint, with $i = 1, \ldots, n$.

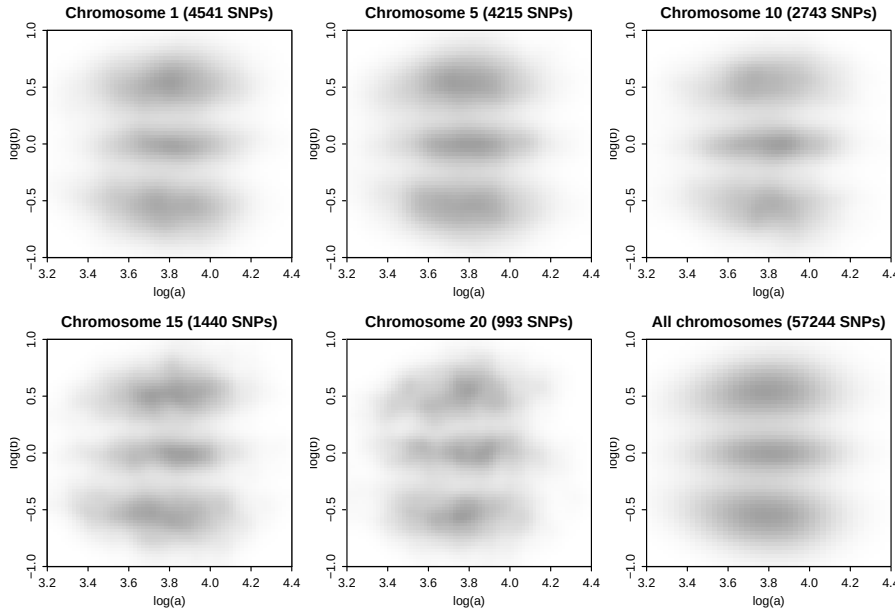


Figure 3.4: Genotype clusters in HapMap sample NA06985 for five individual chromosomes and genotype clusters over all chromosomes (bottom right panel). There is only a difference in SNP density, but not in scale or cluster separation.

The vector of counts is denoted by $\boldsymbol{y} = \{y_i\}$. We write the expected count in bin i as μ_i, and assume that the counts have a Poisson distribution. To be sure that only positive expectations can occur, we work with $\boldsymbol{\eta} = \log(\boldsymbol{\mu})$. The vector $\boldsymbol{\eta}$ is constructed as a sum of B-splines:

$$\eta_i = log(\mu_i) = \sum_{j=1}^{c} b_j(u_i)\theta_j \quad \text{or} \quad \boldsymbol{\eta} = \boldsymbol{B\theta}, \tag{3.1}$$

where $\boldsymbol{B} = [b_{ij}] = [b_j(u_i)]$ is an $(n \times c)$ B-spline basis, with c relatively large, say 20.

Assuming a Poisson distribution for the counts, we maximize the penal-

ized log-likelihood

$$l^* = \sum_{i=1}^{n}(y_i \log \mu_i - \mu_i) - \lambda \sum_{j=1}^{c}(\Delta^3 \theta_j)^2/2. \tag{3.2}$$

The second term is a penalty on the third-order differences of the coefficients. The parameter λ is used to tune smoothness. The larger λ, the stronger the influence of the penalty and the smoother the estimated density. This is the P-spline approach, advocated by Eilers & Marx (1996). They also show that, with third-order differences in the penalty, $\sum_i y_i i^k = \sum_i \mu_i i^k$, for $k = 0$, 1, and 2. This so-called conservation of moments means that, for all values of λ, $\sum_i \mu_i = \sum_i y_i$, and that means and variance computed from $\boldsymbol{\mu}$ are equal to those computed from $\boldsymbol{y}$. The latter property is very important, because it prevents the non-parametric density estimate $\boldsymbol{\mu}$ to deviate much from the observations. Most smoothers do not have this property; the variance of the estimated density increases with the smoothness. For components of mixtures this is an undesirable property.

Smoothness is tuned with the parameter λ. There are ways to optimize it in a data-driven way, using AIC, but in our application we trust our carpenter's eye. The third order differences also have the effect that for larger values of λ the vector $\boldsymbol{\theta}$ tends towards a quadratic series, because for such a series third order differences vanish and the penalty is zero. Unless the series of counts $\boldsymbol{y}$ has a manifest J, U, or L shape, $\boldsymbol{\theta}$ will approach a mountain parabola and the estimated density will show a unimodal log-concave shape. This is a desirable property for components of the mixtures we consider.

Setting the derivative of l^* equal to zero gives

$$\boldsymbol{B}'(\boldsymbol{y} - \boldsymbol{\mu}) = \lambda \boldsymbol{D}'\boldsymbol{D}\boldsymbol{\theta}, \tag{3.3}$$

where $\boldsymbol{D}$ is a matrix of contrasts such that $\boldsymbol{D}\boldsymbol{\theta} = \Delta^3\boldsymbol{\theta}$. Linearization of (3.3) leads to

$$(\boldsymbol{B}'\tilde{\boldsymbol{W}}\boldsymbol{B} + \lambda \boldsymbol{D}'\boldsymbol{D})\boldsymbol{\theta} = \boldsymbol{B}'\tilde{\boldsymbol{W}}\tilde{\boldsymbol{z}}, \tag{3.4}$$

where $\boldsymbol{z} = \boldsymbol{\eta} + \tilde{\boldsymbol{W}}^{-1}(\boldsymbol{y} - \boldsymbol{\mu})$ is the working variable, $\boldsymbol{\eta} = \boldsymbol{B}\boldsymbol{\theta}$, and $\boldsymbol{W} = \mathrm{diag}(\boldsymbol{\mu})$; $\tilde{\boldsymbol{\theta}}$, $\tilde{\boldsymbol{\mu}}$ are approximations to the solution of (3.4). This system is iteratively solved until convergence, which usually is quick (less than ten iterations).

To estimate a mixture with three smooth components, we use the familiar EM (expectation-maximization) algorithm. Two steps are repeated until convergence: 1) split the counts y into three vectors of pseudo-counts, proportional to the current estimate of the mixture components; 2) apply smoothing to the pseudo-counts. Decent starting estimates for the components are needed. We will describe them for our application in what follows.

In two dimensions we use the same idea as described above, but now a two-dimensional histogram is formed, and the log of a density is formed by a sum of tensor products of B-splines. We sketch the adaptations that have to be made. Let $\boldsymbol{Y} = \{y_{ih}\}$ be an $n_1 \times n_2$ matrix of counts in a two-dimensional $n_1 \times n_2$ histogram. The center of bin (i, h) is given by (u_i, v_h). The expected values are modeled by sums of tensor product B-splines. Two bases are computed, $\boldsymbol{B_1}$, with c_1 columns, based on $\boldsymbol{u}$ and $\boldsymbol{B_2}$, with c_2 columns, based on $\boldsymbol{v}$. The bases are combined with a $c_1 \times c_2$ matrix $\boldsymbol{\Theta}$ of coefficients, and the matrix of expected values is computed as

$$\boldsymbol{M} = \exp(\boldsymbol{B_1}\boldsymbol{\Theta}\boldsymbol{B_2}'). \tag{3.5}$$

Like in the one-dimensional case, a penalized Poisson log-likelihood is optimized. The penalty is more complex, because both rows and columns of $\boldsymbol{\Theta}$ are penalized. If $||\boldsymbol{X}||_F$ indicates the Frobenius norm of the matrix $\boldsymbol{X}$, i.e. the sum of the squares of its elements, the penalty is

$$\text{Pen} = \lambda_1 ||\boldsymbol{D_1}\boldsymbol{\Theta}||_F/2 + \lambda_2 ||\boldsymbol{\Theta}\boldsymbol{D_2}'||_F/2, \tag{3.6}$$

where $\boldsymbol{D_1}$ and $\boldsymbol{D_2}$ are matrices of the proper dimensions ($c_1 - 3 \times c_1$ and $c_2 - 3 \times c_2$) that form third differences.

One could vectorize $\boldsymbol{Y}$, $\boldsymbol{M}$ and $\boldsymbol{\Theta}$ and form the Kronecker product of $\boldsymbol{B_2}$ and $\boldsymbol{B_1}$ to mold the equations into a matrix-vector shape. It is however very inefficient to do this. Instead, we use the fast GLAM (generalized linear array model) algorithm (Currie et al., 2006), leading to enormous savings in time and memory use. The details are a bit involved, so we skip them here.

Our model is flexible enough to adapt to the quite different cluster shapes of different microarray platforms. Figure 3.5 shows results for an Affymetrix and for an Illumina array. Left panels show the raw observations, middle

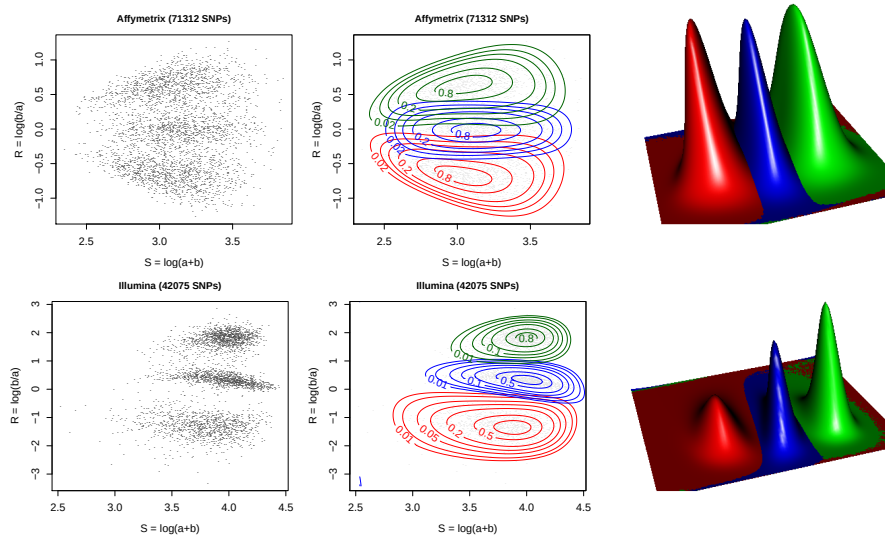


Figure 3.5: Top row: a typical Affymetrix SNP6.0 array. Bottom row: a typical Illumina HumanHap550 array. Left panels : a random selection of 3500 SNPs om chromosome 1 plotted as dots. Middle panels: observations and contour lines of semi-parametric mixture components. Normalized contours (mode set to 1) are shown at [0.01, 0.02, 0.05, 0.1, 0.2, 0.5, 0.8]. Right panels: a 3D perspective of the smoothed densities.

panels shows the density contours after estimation. The cluster contours represent the data well. The right panels show the smooth histograms in a 3D representation. Note how in the Illumina panel the density between the clusters is zero, while in the Affymetrix panel it is not. This can be seen in the genotyping probabilities as well, as discussed below.

The mixture components give three expected values for bin (i, h) of the histogram: μ_{ih1}, μ_{ih2} and μ_{ih3}. From these numbers follow, after division by their sum, three membership probabilities. The largest of the three, which we indicate by $\hat{p}_{ih}$ points to which cluster all the observations in the bin should be assigned. The distribution of $\hat{p}$ over all bins is a good indicator of classification confidence. Ideally all $\hat{p}$ should be very close to one. Of course, strong confidence does not automatically mean good precision; that can only

be assessed by comparison to a standard, as is done in Section 3.3.

Figure 3.6 shows the cumulative distributions of $\hat{p}$ for the two arrays that we used as examples in Figure 3.5. Apparently the Illumina array generates more confidence. Keeping in mind the concentrated clusters in Figure 3.5 this is not a surprise.

The semi-parametric mixture model has a number of parameters that can be chosen by the user. For the histogram we advise a 100 by 100 grid. The domain of the histogram is covered by bases of 10+3 cubic B-splines (the additional three are for extra boundary splines). For the smoothing parameter we choose $\lambda = 10$. Our tests indicate that larger numbers of either bins or basis functions only increase computing time, but do not provide different calls.

To start the EM algorithm, we split the data in three groups by a very simple procedure. In the plot of $\log(a/b)$ vs $\log(a+b)$ two horizontal lines are used to create three sectors (AA, AB and BB). This gives the pseudo-counts for the first round of density estimation. The positions of the separating lines are not very critical.

On an average PC, it takes around 20 seconds to call genotypes for a single Affymetrix SNP6.0 CEL file. Approximately the same time is needed for other arrays, almost independent of the number of SNPs, because the data are first summarized by a two-dimensional histogram.

Our genotyping algorithm has been implemented in R (R Development Core Team, 2012) as part of a larger software system, called SCALA.

3.3 Comparisons

In this section we compare called genotypes from SCALA and CRLMM with the consensus genotypes from HapMap. We explore call differences and evaluate SNPs that are not called by CRLMM and HapMap in terms of SCALA calls.

We use probe set averages of the Affymetrix SNP6.0 CEL-files from the CEU population, CUPID set. To start the EM algorithm the data are split on

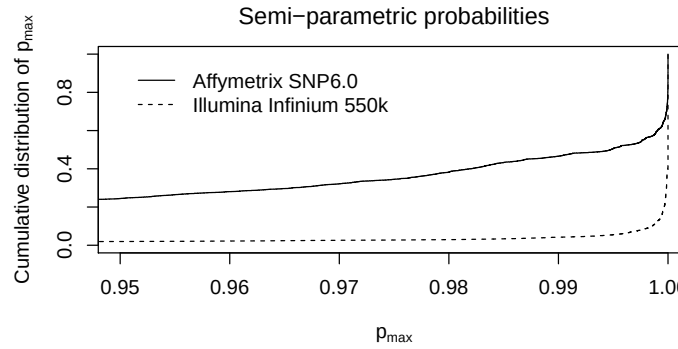


Figure 3.6: Comparison of semi-parametric probability distributions: symmetric Affymetrix (left) vs. asymmetric Illumina (right).

the basis of $\log(b/a)$. The splitting levels can be inferred visually from one (representative) array and kept fixed. We used -0.2 and 0.2, but these values are not critical.

Call agreements

Here we compare genotype calls from SCALA to those from HapMap. Table 3.2 shows, as an example, the cross-table for chromosome 1 on array NA06985. Note that SCALA does not produce NN (NoCalls). SCALA and HapMap completely agree on the AA and BB genotypes, but not on the heterozygotes: 8.4% (633 + 911 divided by the total of column AB) are different; this is 2.7% of all the SNPs called by HapMap.

HapMap is the best reference to judge genotype calling algorithms, but it is not a gold standard. To put this in perspective, we consider a small example, summarized in Table 3.3 and Figure 3.7. The data are for chromosome 1 on an Affymetrix 100k Hind array (NA06991). The left panel of Figure 3.7 shows all SNPs as a gray cloud and the disagreements between SCALA and HapMap. Almost all of them lie in the valleys between a homozygous and the heterozygous clusters, either below (AA) or above (BB) it. Classification is not reliable in these regions. We suspect that we cannot trust HapMap too

Table 3.2: Cross-tabulation of SCALA genotype calls (rows) and HapMap genotypes (columns) for chromosome 1 on array NA06989 (CUPID_p_HapMapPT06_GenomeWideSNP_6_A01_183598.CEL).

	AA	AB	BB	NN
AA	19029	633	0	97
AB	0	16820	0	139
BB	0	911	19326	110

Table 3.3: Cross-tabulation of SCALA genotype calls (rows) and HapMap genotypes (columns) for chromosome 1 in Affymetrix 100k Hind: NA06991.

	AA	AB	BB	NN
AA	837	**12**	0	3
AB	0	731	0	9
BB	0	**13**	826	5

Table 3.4: Call agreement between SCALA (rows) and HapMap (columns), aggregated over all chromosomes in 70 arrays from the HapMap SNP6.0 CUPID set. Numbers in percentages of HapMap genotypes; columns add up to 100%.

	AA	AB	BB
AA	99.97	4.99	0.00
AB	0.03	90.11	0.00
BB	0.00	4.90	100.00

much here. Anyway, we don't see disagreeing AA or BB calls by SCALA that obviously belong to the AB cluster.

To provide a more general indication, we have calculated cross-tables as in Table 3.2 for all chromosomes on all 70 arrays in the SNP6.0 CUPID set for SCALA (Table 3.4) and for CRLMM (Table 3.5). Both tables are normalized to make column totals equal to 100%.

Table 3.5: Call agreement between CRLMM (rows) and HapMap (columns), aggregated over all chromosomes in 70 arrays from the HapMap SNP6.0 CUPID set. Numbers in percentages of HapMap genotypes; columns add up to 100%.

	AA	AB	BB
AA	100.00	2.85	0.00
AB	0.00	94.52	0.00
BB	0.00	2.59	100.00

Table 3.6: Call agreement between SCALA (rows) and GenoSNP (columns), aggregated over all chromosomes in 20 arrays from the Erasmus Medical Center. Numbers in percentages of GenoSNP genotypes; columns add up to 100%.

	AA	AB	BB
AA	99.96	0.86	0.00
AB	0.04	98.52	0.02
BB	0.00	0.62	99.98

We have also compared SCALA performance to GenoSNP, that is dedicated to Illumina arrays. The results on previously mentioned arrays from the Erasmus Medical Center, provided in Table 3.6, illustrate the power of the universal genotyping approach in SCALA; it's performance for asymmetric arrays compared to a dedicated algorithm is even more favorable than for symmetric arrays. Equivalent performance is obtained using Illumina arrays from Staaf et al. (2008).

Furthermore, it is of interest to study the SCALA genotype for those SNPs that HapMap cannot call. We refer to Table 3.3 and to Figure 3.7 in which the transformed measurements are depicted. SCALA can confidently assign them to genotypes (with $p_{max} > 0.95$), because only a few points lie at the boundaries of clusters. We present here only one small example, but it is representative for the general pattern. Figure 3.8 shows, using denstrip (Jackson, 2008), how p_{max} is related to (low) MAF; we see mostly (very) high probabil-

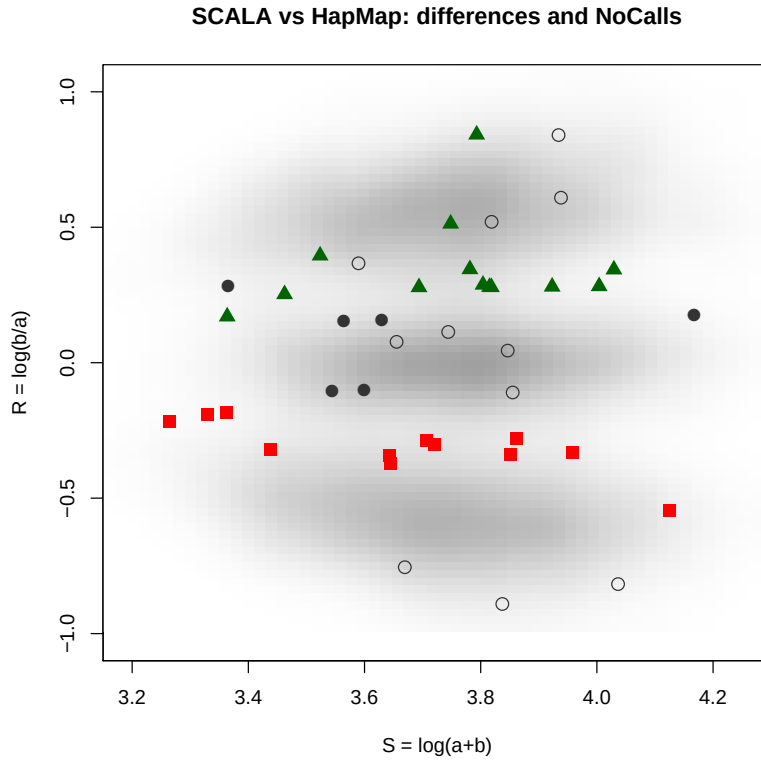


Figure 3.7: Example of SCALA call disagreements with HapMap for chromosome 1 on Affymetrix 100k Hind array NA06991. Some Hapmap AB genotypes called as AA (red squares) or BB (green triangles) by SCALA. HapMap NN calls (circles) can be genotyped with high (open) or low (filled) probability.

ities (dark colored areas) for SNPs with low to very low MAFs.

In summary we found that overall agreement between SCALA and HapMap is comparable to those from CRLMM. However, for the AB calls from HapMap we see differences in the direction of both AA and BB labels, for both SCALA and CRLMM, where the differences for SCALA were about twice as large, up to 4.99% of all HapMap ABs. However, after visual inspection of their lo-

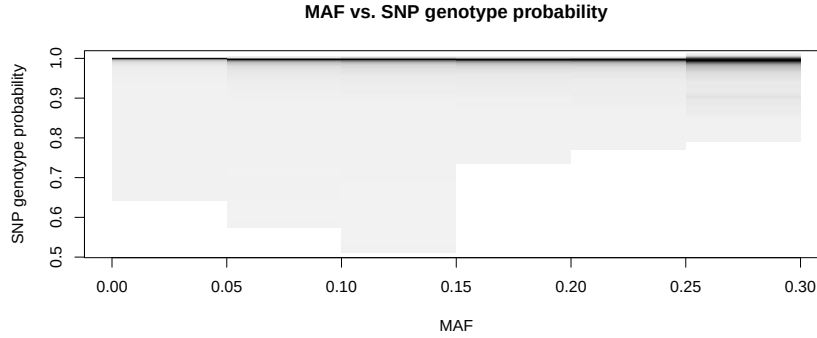


Figure 3.8: Dark color: high density, light color: low density. In data for all chromosomes, p_{max} is still high in low to moderate MAF situations, albeit with somewhat higher variance for lowest MAF.

cation in their single array genotype clustering, for a large number of these differences it seems almost strange that they were called AB by HapMap: they lie in or close to the AA or BB cluster in the single array. In addition we found that for many genotypes that were not called in HapMap, probably due to problems with minor allele frequencies or low call probabilities, we could call those SNPs with a probability larger than 0.95 in most cases. Further visual inspection revealed that those SNPs lie close to the center of one of the three clusters in a single array setting.

3.4 Conclusion and discussion

We presented a fast novel approach to call SNP genotypes in individual arrays using semi-parametric log-concave mixtures.

To assess performance we compared genotype calls from a multi-array method (CRLMM) and from our single-array method (SCALA) to a set of consensus genotypes from HapMap. The number of agreements and differences in terms of homo- and heterozygous calls showed that SCALA can be used to call genotypes efficiently and effectively. Even SNPs that were not genotyped

in HapMap can be genotyped with reasonable certainty using a single chip. We also evaluated performance against the single-array algorithm GenoSNP, dedicated to Illumina chips. Strong agreement was found.

The proposed single chip genotyping approach is therefore very universal in terms of platforms and cluster shapes, for existing (human DNA) chips, but also for new technology, since the algorithm can be applied to the first available chip immediately.

Optimal use of low quality SNP samples by the single array approach

4

SNP samples from any platform can be of low quality due to many reasons. We propose to select the highest quality part of the sample using a probability-based signal threshold. Genotype call rates, call quality as well as visualization and detection of CNV and allele imbalance can be strongly improved using this method. For example, call rates can be increased from around 60% to around 90% or higher. Reprocessing these low quality arrays may therefore be unnecessary, hence improving research efficiency and reducing sampling costs. Since the method approaches single arrays, it is also possible to threshold (parts of) individual chromosomes.

4.1 Introduction

In cancer research, one of two primary goals is to determine the according genotype from obtained Single Nucleotide Polymorphism (SNP) signals (Mao et al., 2007). To quantify SNPs, the strength of fluorescence in two channels (one for each chromosome in a pair) is measured. The ratio of the signals for the two alleles determines the genotypes. The sum of these signals can be used for CNV estimations.

SNPs are measured through different platforms and with different methodologies. Major companies like Affymetrix provide platforms as well as their own software. The major difference between their technologies is found in the way the fluorescence signals for each allele are produced. Platforms of

This chapter is an adapted version of the article:
Rippe, R.C.A., Eilers, P.H.C. and Meulman, J.J. (2012). Optimal use of low quality SNP arrays by the single array approach, *manuscript*.

varying SNP density are available, ranging from 50.000 (Nicolae et al., 2006) to 250.000 (Dunbar et al., 2008) to 1.000.000 SNPs (Nishida et al., 2008). Of course, other platforms exist (e.g. Illumina), but this paper focuses on the Affymetrix platforms. Throughout the current work we use signals a and b that are averaged over all probes for the A and B allele.

Genotyping

The most procedure to genotype SNPs is to do clustering per SNP, using multiple arrays. From a practical point of view this is very unattractive, because it is almost impossible to monitor the influence of individual arrays on the results. Rippe et al. (2010) developed a single array genotyping algorithm with excellent performance on normal tissue arrays (e.g. as used in epidemiology) and as provided by HapMap. Single arrays are easy to judge in terms of quality, and along the same lines, are easier to have the low quality part of the signal removed.

Commercial software generally calls genotypes for one SNP, using information from multiple arrays. However, taking an alternative perspective by calling genotypes for all SNPs in an individual array might be more fruitful than single SNP - multi-array calling (Tikhomirov, Konkashbaev & Nicolae, 2008; Rippe et al., 2010). One advantage is that there are enough data points available to perform very stable model fitting to obtain the genotype calls (e.g. 4600 SNPs on chromosome 1 on a 100k Hind array). Problems with low minor allele frequencies do not occur. Another major advantage is that besides genotyping, this provides an indication of array quality. For the left sample in Figure 4.1, genotyping is a clear matter, but for the right one it isn't. The SNPs are mostly cluttered in the low signal area, restricting any genotype separation.

In the described single array approach, a SNP genotype generally follows from the ratio of two signals ($r = \log_{10}(b/a)$) with a the signal for allele A and b for allele B. Since the ratio r of the two fluorescence intensities (one for each allele) determines the genotype, signal noise can deteriorate the quality of the genotype calls. It is based on a two-dimensional mixture of log-concave densities (along the lines of Eilers & Borgdorff, 2007), fitted on smoothed 2-

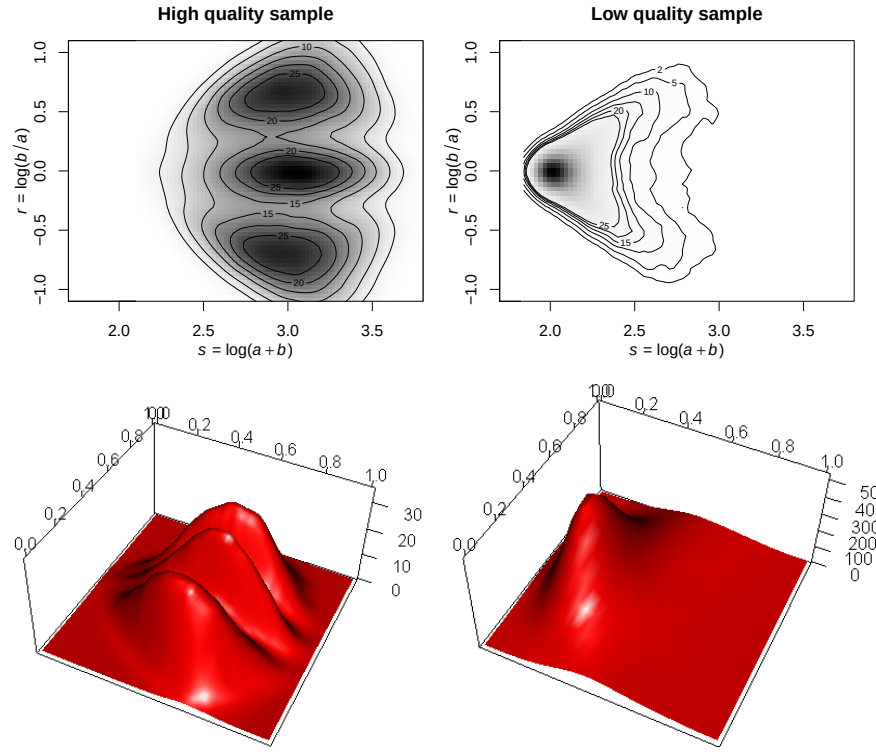


Figure 4.1: Top row: left panel shows a high quality Affymetrix SNP6.0 control sample. Right panel shows a low-quality control sample. On the horizontal axis the sum $s = \log_{10}(a + b)$ is shown; higher values (total signal) indicate higher quality. On the y-axis, the ratio of the signals for the two alleles $r = \log_{10}(b/a)$ is shown; this indicates the relative concentration of fluorescence signals for the two alleles. Genotype cluster contours are superimposed to illustrate cluster separation; better separation results in higher call rates. Bottom row: 3-dimensional replication of the same data. The left panel shows three clear peaks in an otherwise flat landscape, while the right panel shows one peak in a hilly environment: noise.

dimensional histograms (Eilers & Marx, 2007).

To estimate a mixture with three smooth components in two dimensions, we use the familiar EM (expectation-maximization) algorithm. Two steps are

repeated until convergence: 1) split the counts y into three vectors of pseudo-counts, proportional to the current estimate of the mixture components; 2) apply smoothing to the pseudo-counts. Decent starting estimates for the components are needed.

Let $\boldsymbol{Y} = \{y_{ih}\}$ be an $n_1 \times n_2$ matrix of counts in a two-dimensional $n_1 \times n_2$ histogram. The center of bin (i, h) is given by (u_i, v_h). The expected values are modeled by sums of tensor product B-splines. Two bases are computed, $\boldsymbol{B_1}$, with c_1 columns, based on $\boldsymbol{u}$ and $\boldsymbol{B_2}$, with c_2 columns, based on $\boldsymbol{v}$. The bases are combined with a $c_1 \times c_2$ matrix $\boldsymbol{\Theta}$ of coefficients, and the matrix of expected values is computed as

$$\boldsymbol{M} = \exp(\boldsymbol{B_1}\boldsymbol{\Theta}\boldsymbol{B_2}'). \tag{4.1}$$

A penalized Poisson log-likelihood is then optimized. The penalty is complex, because both rows and columns of $\boldsymbol{\Theta}$ are penalized. If $||\boldsymbol{X}||_F$ indicates the Frobenius norm of the matrix $\boldsymbol{X}$, i.e. the sum of the squares of its elements, the penalty is

$$\text{Pen} = \lambda_1 ||\boldsymbol{D_1}\boldsymbol{\Theta}||_F/2 + \lambda_2 ||\boldsymbol{\Theta}\boldsymbol{D_2}'||_F/2, \tag{4.2}$$

where $\boldsymbol{D_1}$ and $\boldsymbol{D_2}$ are matrices of the proper dimensions ($c_1 - 3 \times c_1$ and $c_2 - 3 \times c_2$) that form third differences. We use third order differences because for larger λ, the values in Θ move towards a quadratic trend, since in such a trend third differences no longer exists and hence the penalty becomes zero. Given the right circumstances we end up with a mixture of log-concave unimodal shapes.

The final mixture components give three expected values for bin (i, h) of the histogram: μ_{ih1}, μ_{ih2} and μ_{ih3}. From these numbers follow, after division by their sum, three membership probabilities. The largest of the three, which we indicate by $\hat{p}_{ih}$ points to which cluster all the observations in the bin should be assigned.

CNV and Allelic Imbalance

If we plot SNP signals according to their chromosomal position, it is easy to visualize any aberrations in high-quality samples. The s-signal can illustrate

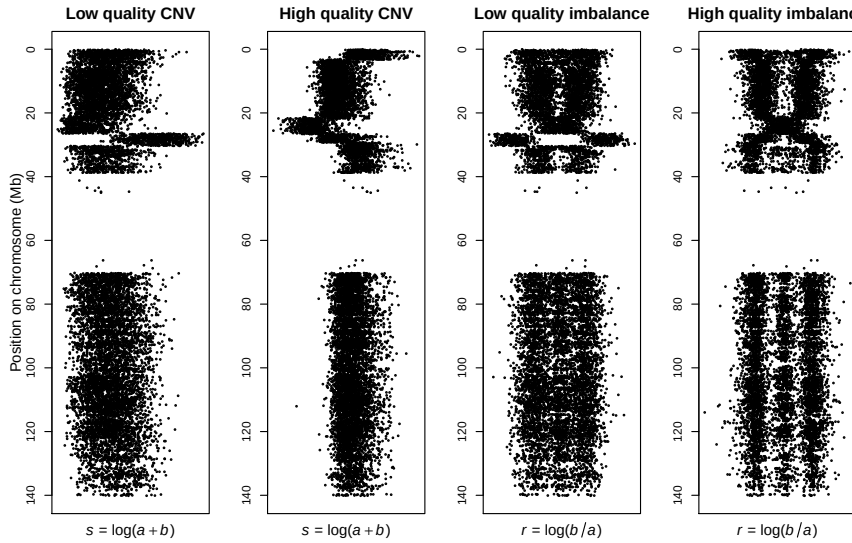


Figure 4.2: High and poor quality examples from Affymetrix 250k NSP tumor arrays. The two left panels show CNV by plotting the *s*-signal from Figure 4.1 against chromosomal position. Right two panels show the *r*-signal against its chromosomal position. Higher quality samples show clearer patterns. Here, higher quality was induced using signal calibration.

copy number variations (CNV) (Sebat et al., 2004; Taylor et al., 2008; McCarrol et al., 2008), while the *r*-signal can indicate allelic imbalance or loss of heterozygosity (LOH) (Beroukhim et al., 2006). Again, having low quality SNP measurements included, besides inducing low call rates (in normal tissue), they can distort any clear pattern in either CNV or LOH. Figure 4.2 illustrates the latter effect. From left to right it shows CNV and allelic imbalancefor a high and low quality array.

Sample quality and signal calibration

With increases in the number of simultaneous SNP probes (currently up to 10^6 SNPs) in recent platforms, the absolute number of errors has also increased. In lower-quality samples this effect is boosted even more. This

problem is illustrated in Figure 4.1 and Figure 4.2, where the left panel shows a high quality Affymetrix SNP6.0 sample compared to a low quality one on the right.

Many researchers that work with SNP samples have come across low quality samples coming in from the lab. Lower sample quality makes it harder to call genotypes and increase the probability (proportion) of wrong calls. In commercial software SNPs below a certain probability threshold would not be called at all (No Call). Individual arrays with too many 'No Calls' may be rejected for further analysis. Unfortunately, processing these samples and their processing still costed money and time.

Rippe et al. (2010, 2012a) described a procedure called SCALA in which single array genotyping and array calibration are implemented. From their linear models follow, for each allele, either a parameter vector $\boldsymbol{\alpha} = [\alpha_i]$ or a three-column array $\boldsymbol{\Gamma} = [\gamma_{ij}]$. See Rippe et al. (2012a) for more information on how the calibration parameters are obtained. We can use them to compute corrected signals and - if wanted - repeat the estimation of the mixture model using these 'new' signals. We call this calibration. In the first case this translates to $x^*_{ij} = x_{ij}/10^{\alpha_i}$. It does not use the genotypes of a new sample, so we call it *genotype-free calibration*. In the second case we have $x^*_{ij} = x_{ij}/10^{\psi}$, where $\boldsymbol{\psi} = \sum_k h_{ijk}\gamma_{ik}$, which we call *genotype-based calibration*. Analogous formulas hold for signal b.

The result of genotype-free calibration for an illustrative array is shown in Figure 4.3. After calibration the cluster are more condensed.

They also discussed a quality control measure similar to the QC criterion used in e.g. Affymetrix Genotyping Console. Our approach is also suitable to save as much information as possible from low quality arrays. In this paper we propose to select only a part of a low quality array and to perform genotyping and calibration methods within the SCALA framework on the remaining data.

Samples can be selected as high or low quality based on their initial call rate. However, since some call rates are either not provided or not obtained from the different data sources, we use the single-array method described above to get an initial call for all samples based on a single method.

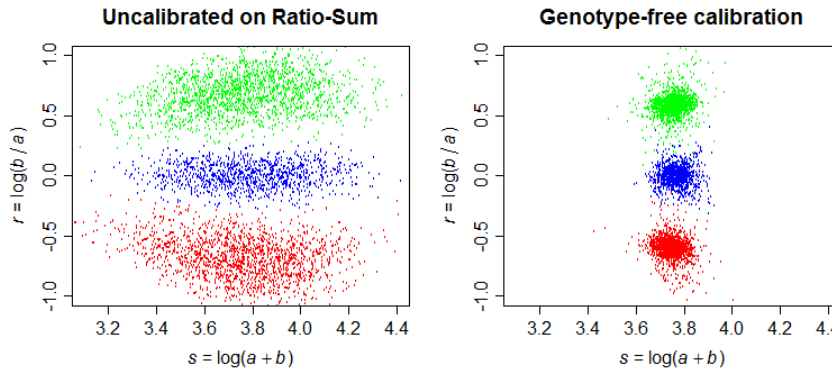


Figure 4.3: Effective genotype-free calibration in a high quality Affymetrix 250k NSP array.

Threshold SNP selection

To tackle the problem of analyzing low overall quality arrays, we propose a signal threshold for non-amplified low quality samples (Mead et al., 2008; Ziegler et al., 2009) to select only those SNPs for which the combination ($s = \log_{10}(a + b)$) of two allelic signals both exceed a certain threshold value. This part of the data shows a much clearer genotype separation. This implies that sample quality is inversely related to the number of SNPs with a low s-value. The suggested method implies that only a part of the probed SNPs are analyzed in depth, but still we obtain more information that nothing at all. Furthermore, in such a sample it is still possible to see larger regions of aberrations on chromosomal regions. Platforms with a higher SNP density benefit more from this approach, since after threshold selection the remaining SNP density is still high enough to perform analyses on.

It is recommended to threshold-select SNPs for one individual array at a time, since observations removed from one array do not match those selected in other array, for a very large part.

We will illustrate our approach using samples obtained from GeoDB, HapMap and the Erasmus Medical Center.

4.2 Methods

Data

In this work we use data from three different platforms. The Affymetrix 100k (Hind and XBA) samples are obtained from GeoDB and include samples of both high and low quality. The Affymetrix 250k NSP (Bralten et al., 2010) and SNP6.0 (Gravendeel et al., 2009) clinical samples of varying quality were provided by the Erasmus Medical Center. Poor quality samples were omitted in these studies, but are included in the current study. The 500k NSP set also contains 8 high quality reference samples. SNP6.0 reference samples were obtained from Affymetrix. 500k STY samples are not evaluated due to the simple fact that no low quality array were avaible.

Genotyping

For the genotyping procedure we chose not to use commercial software, since this software can contain differences in signal processing (Rabbee & Speed, 2006; Xiao et al., 2007) as well as internal SNP and sample comparison. To overcome the problem of platform comparability, we use our own genotyping implementation that is applied on a single array as described in section 4.1.

Call probability as a sample quality measure

Low quality samples can be identified simply by counting the number of SNPs with an s-value below a certain threshold value, even without genotype call rates. However, this requires an objective threshold, which is hard to determine since array base levels may differ for each batch.

Second, the quality of a sample can be measured by the proportion of called genotypes with a large probability. In commercially available software, low-probability calls are rejected and hence result in NC (No Call). We will use an equivalent measure, based on the genotyping results $Q = \max(p_1, p_2, p_3)$ from the SCALA software. This measure is formulated as the proportion of called genotypes with a call probability lower than 0.90 ($Q90$);

the more low-probability calls are obtained, the higher $Q90$ and the lower the sample quality. If a platform collection doesn't contain such samples, 6 samples with the lowest call rate for that platform were selected. High quality arrays were available for all platforms.

Thresholding

To perform threshold signal selection, we use the $\log_{10}$ of the sum of the two allelic signals ($s = log_{10}(a + b)$). This is the x-axis in Figure 4.1. In practice, this means that any SNP with an s-value below a given threshold is completely excluded from further analysis. This can be done for each chromosome separately or for all array observations at once. We choose to perform the computations per chromosome, to have better control.

There are two ways to go from here. First, we can use a straightforward 'hard threshold', which is intuitively very effective. Since the signal quality is specific for a sample, there is no control on the amount of data that is removed: in one sample a threshold value of $s = 3.0$ may exclude 30% of the data, whereas in a very low quality sample the same threshold may remove 100% of the signals.

A different way to apply thresholding is to use the empirical distribution function (edf, which is trivial to determine: for a random sample it is the cumulative distribution function of the values obtained in the sample. It is a staircase function that is equal to 0 for $s < s_1$ and is equal to 1 for $s \geq s_n$ or

$$\hat{F}_n(s) = \frac{\mathrm{z} \leq s}{n} = \frac{1}{n} \sum_{i=1}^{n} I(S_i \leq s) \tag{4.3}$$

with z the number of measured elements in the sample.

Suppose we apply a certain threshold value for the s-signal on a given chromosome. We can restrict the SNP removal to just the one-sided $p = 0.90$ part of the data. If according to the threshold value all SNPs have to be removed, we can force the best $p = 0.10$ below that threshold to always be kept, so that for allele A we have $a^* = a_{s>p}$ where a^* and a are of different lengths (analogous for allele B). This way, there is always some available data

to provide information about genotypes and/or aberrations. We will use the latter approach for further analyses.

Calibration

We will assess the effect of threshold selection on calibration (and downstream call rates).

Furthermore we will look at calibration effects *after* we removed low-signal SNPs and *then* apply precomputed calibration parameters for the remaining SNPs. From these calibrated signals we determine the genotypes. Given these genotypes, we obtain calibration parameters based on the new data. Finally, for illustrational purpose, we apply a genotype-based calibration method to obtain the best possible separation and call rates, using estimated genotypes obtained after genotype-free calibration.

4.3 Results

Call rates for different threshold values

The results for the call rates on uncalibrated chromosome 4 in each of the platforms, for each percentile in $[0, 10, ..., 80, 90]$ are given in Figure 4.4. For high quality SNP6 arrays, it is clear that removal of the lower signals is detrimental to the overall call rate. This pattern holds for all chromosomes (not shown). For the other platforms however, call rates remain more or less constant after removal of the lower signals.

In the left panel (high-quality samples) it can be seen that for all platforms the call rates are high at all threshold levels, but the rates for SNP6 slightly decrease if too much signal is removed. In the right panel (low quality samples) we see lower call rates in all platforms. The SNP6.0 platform seems to benefit the most. This may be due to the fact that in this platform the last 10% still contains 100.000 SNPs. However, both of the Affymetrix 100k (Hind and XBA) call rates do not seem to suffer very much: they are still very high, though lower than for their high-quality counterparts at the same threshold

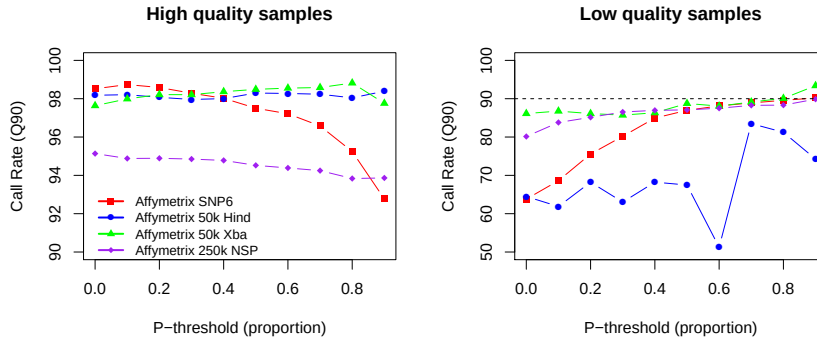


Figure 4.4: Call rates on chromosome 4 using four different platforms. Left: high quality samples. Right: low(er) quality samples. Note that call rates in the left panel ranges from 90 to 100, whereas the right panel ranges from 60 to 100. The dotted line in the left panel indicates the lowerbound of the y-axis in the left panel.

level. Affymetrix 250k NSP and Affymetrix SNP6.0 have very low call rates. All three benefit strongly from threshold selection, which seems to be most effective for the SNP6.0 data. However, the best low-quality SNP6.0 call rate after thresholding (at 90% cut-off) is still below 90%, and this is lower than the call rate for a non-thresholded high-quality sample.

Calibration effects

The effects of thresholding on the same low quality samples as used in section 4.3, but now after genotype-free calibration, are summarized in Figure 4.5. In Table 4.1 these call rates are compared with their uncalibrated counterparts. All platforms show (strong) improvements in call rates using calibration. The Affymetrix improvements are as expected from the calibrated clusters. An example for a 250k NSP sample is shown Figure 4.3.

Furthermore, the 'jump' in call rates at $p = 0.30$ for low quality Affymetrix SNP6 samples indicates after removal of the worst 20% the remaining signal contains much clearer genotype information, which is however still viable for

Table 4.1: Results (median error rates) for low-quality samples after genotype-free calibration. P = (P)roportion of data removed, B = (B)efore calibration, A = (A)fter calibration.

	SNP6		100k Hind		100k Xba		250k NSP	
P	B	A	B	A	B	A	B	A
.0	36.2	19.7	35.5	25.0	13.9	20.7	19.9	5.9
.1	31.4	18.3	38.1	31.3	13.2	17.6	16.2	5.1
.2	24.4	17.0	31.8	28.1	13.9	19.0	14.9	4.7
.3	19.8	4.3	36.9	26.5	14.3	18.1	13.5	4.4
.4	15.1	3.9	31.8	25.4	13.7	15.8	13.0	4.2
.5	13.0	3.7	32.5	24.5	11.2	16.5	12.9	3.9
.6	11.9	3.6	48.8	22.9	11.9	13.9	12.5	3.5
.7	11.1	3.6	16.5	22.9	10.9	12.2	11.7	3.1
.8	10.4	3.6	18.8	17.7	9.9	6.4	11.7	2.8
.9	9.8	3.6	25.8	15.4	6.6	6.1	10.1	1.9

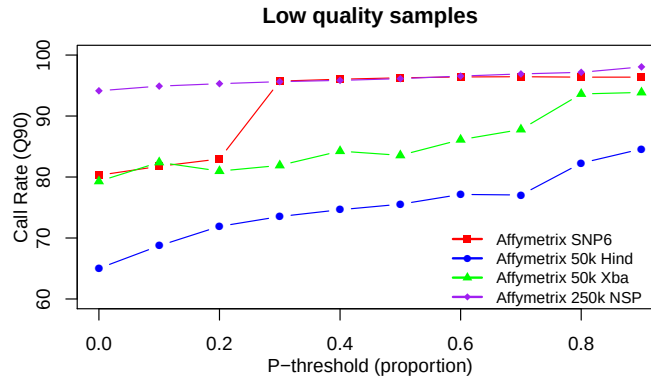


Figure 4.5: Call rates for low quality samples *after* genotype-free calibration. Same samples on chromosome 4.

further improvements. The best call rate is still below the lowest call rate for high quality samples.

Some further investigation shows that the effective combination of calibration and thresholding on the call rates for the 100k Hind, 100k Xba and 250k NSP samples is due to a higher overall quality of the selected low quality samples, compared to the low quality samples from SNP6.0. High quality samples however benefit strongly (not shown), with median call rates consistently approaching 99% for all five platforms.

The effects in Table 4.1 also support the assumption that genotype-based calibration (see section 4.1) is unwise to do: genotype-based calibration requires called genotypes, but these calls can only be made using the original, poor quality signals. This may introduce erroneous calibration due to erroneously called genotypes.

Inspecting aberration profiles

Figure 4.6 shows the effect of threshold selection on signals for profiles of CNV and allelic imbalance. In this particular section we used data on chromosome 9, since we know that aberrations occur there. The left panels show CNV detection, and imbalancepatterns are represented in the right panel. At first sight, in the complete sample the noise in both the left and right panels clearly dominate the data and no patterns can be distinguished at all. However, with an increase in threshold level, we can see deviations patterns (like in Figure 4.2) arise. It may seem like we are introducing bias in the CNV panel. However, matching the remaining signal to the imbalance panel, we can see that we are in fact removing signal indicating "complete loss" or noise. From $p = 0.70$ and above, aberrational patterns on the ratio-signal r are visible: in the right panel in which we look at possible imbalance patterns the left (P) arm clearly lacks a heterozygous genotype and ends with a complete loss of information (looks like a vertical bar). The right arm (Q) has the expected three genotype bands (homozygous on top and bottom, heterozygous in the middle, but it also seems to end with complete loss. These patterns were not (clearly) visible. Since the thresholding is performed on the s-signal, this is just signal that is cut off starting from the bottom and there-

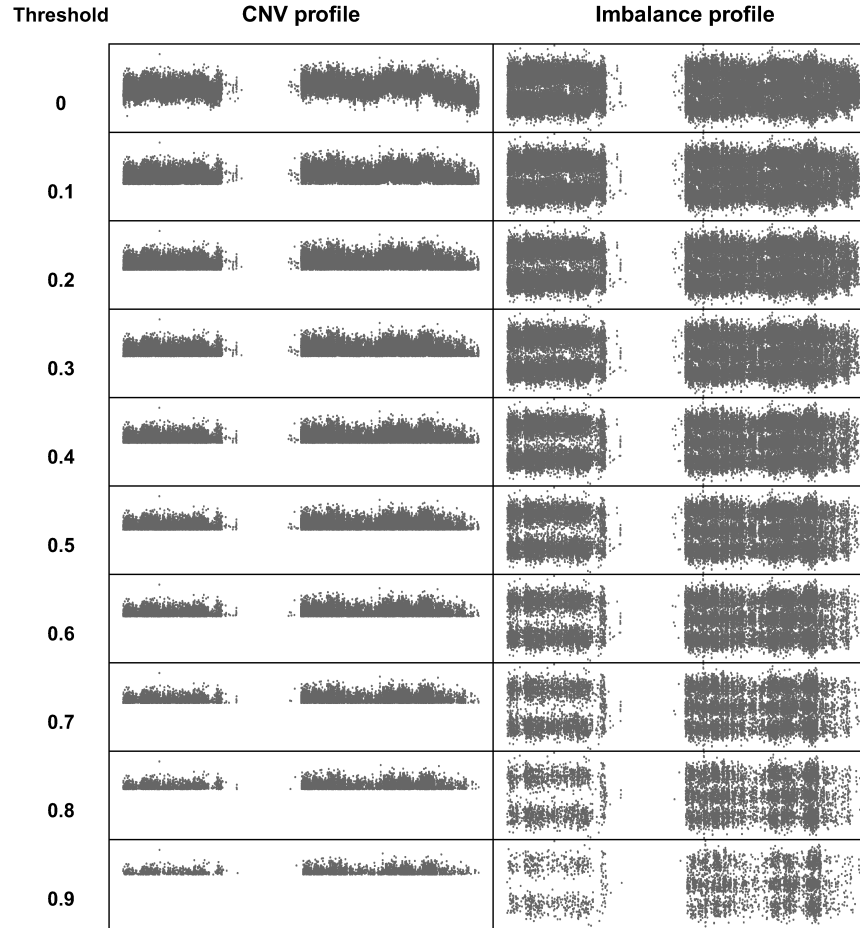


Figure 4.6: Aberration detection. Visual (and statistical) detection in a very low quality sample improves only after removing 80% of the originally low *s*-signal. Left: thresholded CNV signal. Right: thresholded imbalance signal. For CNV profiles, signal selection is not useful, while it is for allelic imbalance.

fore it doesn't benefit from threshold selection. Figure 4.7 shows that when a lower CNV signal (a deletion) is removed, this does not delete information

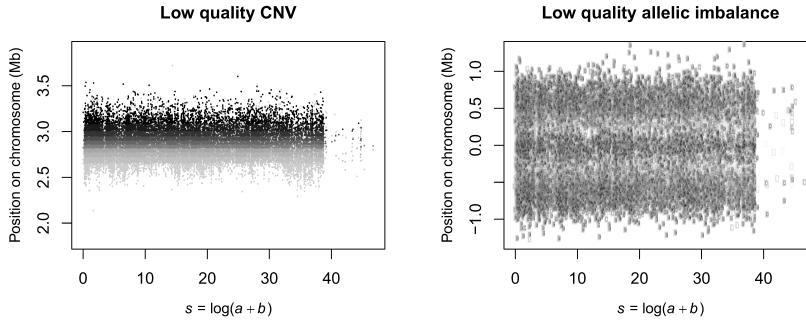


Figure 4.7: Gradient representation of the threshold levels. Lighter dots are removed in low(er) threshold levels. Left: CNV signal threshold. Right: imbalance signal thresholds. This illustrates that when a lower CNV signal (a deletion) is removed, this does not delete information about allele loss: it can still be retrieved from the imbalance panel.

about allele loss: it can still be retrieved from the LOH panel.

4.4 Discussion

We proposed a signal threshold approach to low-quality SNP samples for any given platform. A distribution-based approach provides a wayto ensure that one can never accidentally remove all SNPs; the best $(1 - p)$ % of the SNPs remain available. Applying such a threshold selection on low quality samples (strongly) improves array quality, defined by call rate. Threshold selection on high-quality samples doesn't have the same strong effect, but fortunately these samples do not *need* thresholding. The same effect holds for the effects of both genotype-free and genotype-based calibration; if sequentially applied the sample size is reduced, but the call rate and structural quality improve (strongly). The same holds for the detection of aberrational CNV andLOH patterns in low quality data; with increased threshold, the detection also improves, hopefully improving to a high enough quality level so that patterns are detectable. However, to provide more insight, numerical analyses should

be performed.

With respect to the above results and the fact that low-quality samples still have to be paid for, we suggest to make the best of the situation. The proposed threshold recovery procedure should be applied and evaluated before re-sending samples to the lab again. This approach saves money, time and effort and can be sufficient in many situations.

In this chapter we didn't include Illumina (Infinium) samples, since these arrays have asymmetric homozygous clusters, which is probably due to a dye saturation effect. We are currently looking into this; these results will be discussed elsewhere.

Genomic waves: where they come from, and how to eliminate them

5

Genomic waves are an undesirable distortion in copy number variation. It is generally assumed that they have real physical existence. We show that this is not true. Fluorescence signal on SNP arrays have a systematic bias, that varies strongly from SNP to SNP, giving the appearance of noise. Smoothing removes high frequency variations and so gives the impression of waves. The bias, and hence the waves, can be estimated and removed by a procedure called SCALA.

5.1 Introduction

Although SNP arrays were originally developed for genotyping of (normal) DNA, they are also a popular tool for studying copy number variations (CNV) and allelic imbalancein tumor samples. When studying CNV a persistent nuisance is the occurrence of “genomic waves", which compromise estimation accuracy due to unclear segment breakpoints. They become visible when the raw signal, the sum of the fluorescence intensities of the two alleles is smoothed sufficiently, as shown in Figure 5.2 for four different arrays.

The existence of waves has been reported frequently in both SNP arrays and aCGH profiles. Several remedies have been proposed. The wave phenomenon was first reported in aCGH profiles by Cardoso et al. (2004) and subsequently by Nannya et al., (2005) and Marioni et al. (2007). Cardoso et

This chapter is submitted as the article:
Rippe, R.C.A. and Eilers, P.H.C. (2012). Genomic Waves: where they come from, and how to eliminate them, *submitted for publication*.

al. referred to waves as a spatial bias, which they thought was due to non-constant specificity in the DNA amplification process. However, this idea was countered when the same pattern was seen in HapMap data. Nannya et al. introduced an algorithm that accounts for GC content (the percentage of nitrogenous bases that are either guanine or cytosine), which was extended in Lepretre et al. (2010). They proposed WACA (waves aCGH correction algorithm) that uses both GC content and size of the DNA fragments to correct for wave bias. However, Marioni et al. concluded after thorough evaluation that fitting a lowess curve through the profile was an improvement over GC correction. Also recently a procedure called NoWaves was proposed (Van de Wiel et al., 2010) to correct for wave bias in tumor profiles without using GC content, using ridge regression on (smoothed) normal profiles.

Genomic waves are also found in CNV profiles from SNP arrays, which are fundamentally different from aCGH profiles, because SNP arrays provide information on the (genotypes of the) two individual alleles. Komura et al. (2006) described genomic waves for this type of array and proposed the Genomic Imbalance Map algorithm that reduces signal noise by accounting for sequence characteristics of both probes and targets. The aCGH model from Nannya et al. proved effective for SNP arrays, too. Diskin et al. (2008) describe an algorithm that first quantifies the genomic waves in terms of GC content and uses this quantification as a predictor in a regression model. They also noted that, although commonly observed, genomic waves are not well understood. Marioni et al. thought it should be seen as spatial autocorrelation.

In reality autocorrelation does not exist, but is created by smoothing. In this paper we show that the cause of these waves is the existence of a systematic bias, characteristic for each allele of each SNP. Without smoothing it appears as noise but in fact it is reproducible, see Figure 5.1, which shows highly similar noise in four different arrays. The bias can be estimated as parameters in a linear model called SCALA (Rippe, Meulman & Eilers, 2012a). The model parameters can be estimated using an initial set of (high quality) arrays and the corresponding genotypes. Once the parameters have been estimated they can be used to correct these arrays and any new array that will become available.

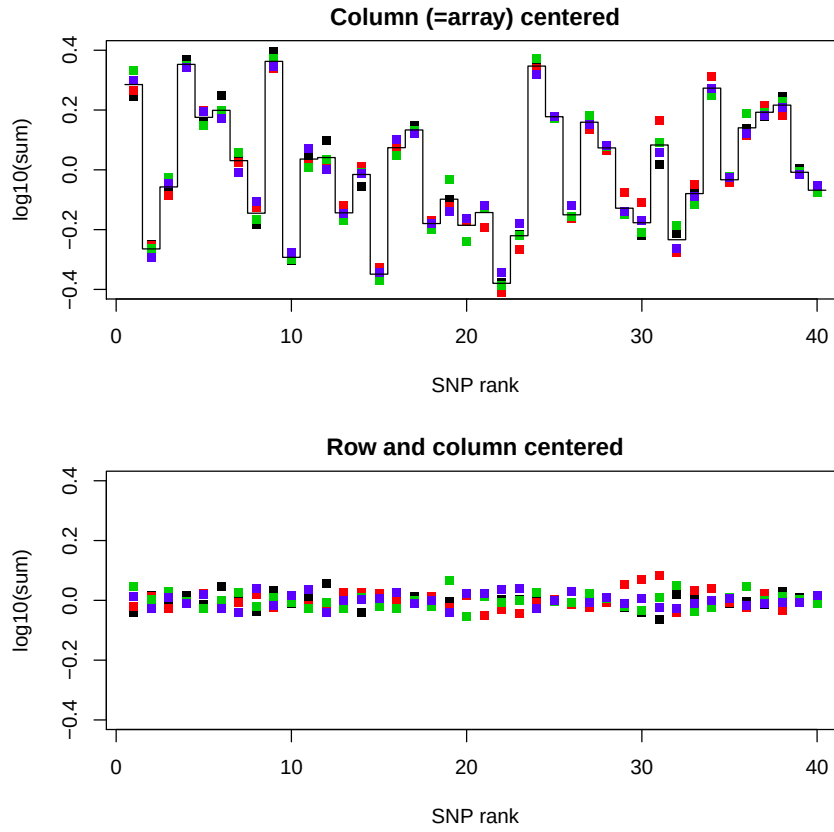


Figure 5.1: The source of the "waves" is systematic bias in the fluorescence signals. Shown are four different arrays, with highly similar noise. Subtracting the mean for each SNP (over arrays) for each SNP essentially eliminates the variation. This only works for normal DNA.

This procedure is easily applicable and therefore we feel it can and should always be applied.

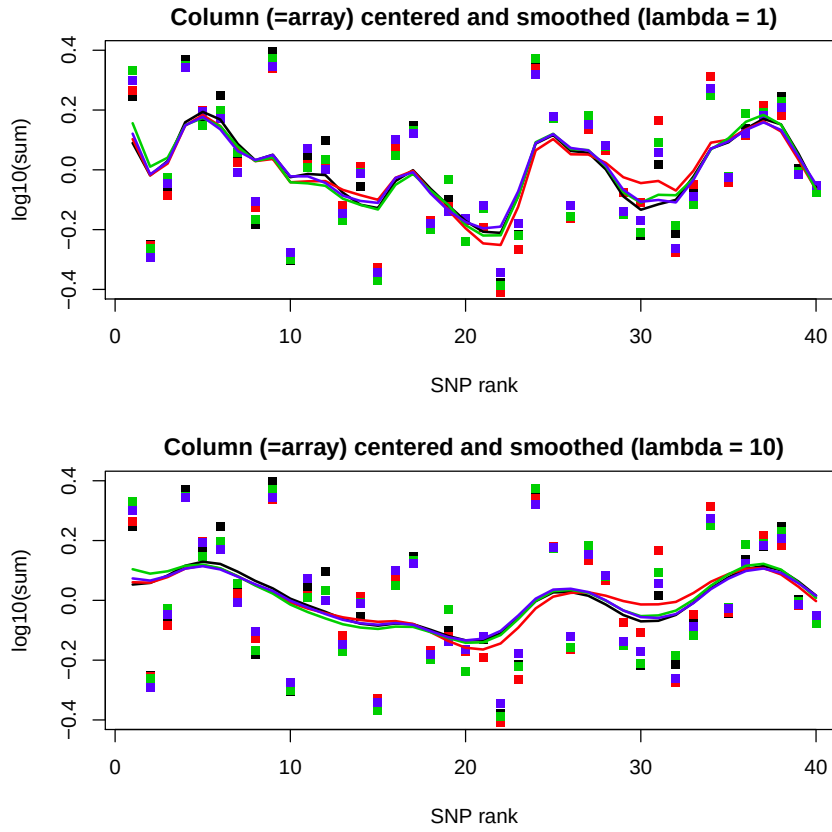


Figure 5.2: An illustration of how smoothing produces "waves", although the raw signals are unstructured. Shown are the first 40 SNPs on chromosome 9. For a clearer display, the positions of the SNPS in the graphs are their ranks, not their physical positions. The Whittaker smoother is used, with two values of the parameter λ.

5.2 Methods

Data and preprocessing

We use Affymetrix 250k NSP tumor profiles from the Erasmus Medical Center (Bralten et al., 2010) and high quality reference profiles from the same

Affymetrix platform. Chromosomes 1 to 22 are analyzed. The non-autosomal chromosomes X and Y are neglected due to the fact that the calibration approach in SCALA requires signals for two alleles, which is impossible to obtain in the Y chromosome. We start from averages of fluorescence intensity over probe sets; information on the individual probes is not used. We transform the signals for the two alleles, a and b, to a single profile $s = log_2(a + b)$.

The origin of the waves

A simple illustration of our claim that each SNP shows a reproducible bias is presented in Figure 5.1. It shows s for the first 40 SNPs (as determined by their position) of chromosome 9. Four high-quality arrays, to which normal DNA was hybridized, were used. Each array was centered by subtracting the mean of s (over the 40 SNPs). To make it easier to see the data for the individual SNPs, their ranks are used for the horizontal coordinate, and not their physical position on the chromosome. From the top panel it is clear that the levels vary strongly from SNP to SNP, but that they are similar within each individual SNP. If we subtract the means per SNP the lower panel is obtained which show much smaller variation and almost no systematic patterns.

This would be a good method to correct data from normal DNA, but copy number variations are not much studied for normal DNA. However, one can compute means per SNP for a set of “normal" arrays and use these values for correcting any other array. In what follows we will present a more advanced allele-specific correction method.

Smoothing

We use the Whittaker smoother (Whittaker, 1923; Eilers, 2003), assuming equally spaced pseudo-positions. This is a simplification, but as we only use the smoothing for illustration, it can do little harm. Results are shown in Figure 5.2, for two values of the smoothing parameter λ. The Whittaker

smoother minimizes the penalized sum of squares

$$Q = \sum_i (s_i - z_i)^2 + \lambda \sum_i (\Delta^2 z_i)^2,$$

where z represents the smooth series and Δ is the operator that forms second order differences: $\Delta^2 z_i = (z_i - z_{i-1}) - (z_{i-1} - z_{i-2})$.

As Figure 5.2 shows, smoothing leads to “waves", even though the unsmoothed data make large jumps from SNP to SNP. Because the “waves" are very similar for the four arrays, it is easy to mistake them for a real spatial pattern, but actually they are an artifact.

The SCALA model

Let Y be a matrix with logarithms of fluorescence intensities for one allele. The rows, indexed by i, represent the SNPs and the columns, indexed by j, the arrays. The SCALA model is defined for any allele signal y_{ij} as

$$y_{ij} = \mu + \alpha_i + \beta_j + \sum_{k=1}^{3} \gamma_k h_{ijk} + e_{ij} \tag{5.1}$$

where μ is the grand mean, α_i describes the overall level of SNP i, β_j describes the overall intensity level of array j, k is the genotype code with $1 = AA$, $2 = AB$, $3 = BB$ (we work with normal DNA) and γ_k is a parameter for genotype k. The genotypes are coded in $H = \{h_{ijk}\}$. H is a 3-dimensional indicator array; for each combination of i and j we have a 1 in layer that is indicated by the genotype, and 0 in the other layers. To make the model identifiable we introduce the constraints $\sum_i \alpha_i = 0$ and $\sum_j \beta_j = 0$. Details on the estimation procedure are described in Rippe et al. (2012a).

In **correction by SCALA**, the model is fitted for each of the two alleles separately. After fitting, we obtain the parameter vectors $\boldsymbol{\alpha} = [\alpha_i]$. These obtain corrected signals by

$$Y_j^c = Y_j / 10^{\alpha}. \tag{5.2}$$

Correction by NoWaves , which finds regression coefficients for each SNP i by

$$Y = \sum_{i=1}^{n} \beta_i Z_{ij} + \epsilon_j \tag{5.3}$$

with Z_j the smoothed (normal) reference profile.

The coefficients β are estimated using ridge shrinkage on the reference profile parameters, through

$$\beta^* = \text{argmin} \left(\sum_{j=1}^{s} \left(Y_j - \sum_{i=1}^{n} \beta_i Z_{ij} \right)^2 + \delta \sum_{i=1}^{n} \beta_i^2 \right) \tag{5.4}$$

with δ the coefficient shrinkage parameter, which is determined through leave-one-out crossvalidation and hence is sample-dependent. Signal correction is then ensured by:

$$Y_j^c = Y_j - \sum_{i=1}^{n} \beta_i^* Z_{ij}. \tag{5.5}$$

Correction performance

To find a smooth estimate for the CNV profile we use the L_2 norm smoother, as proposed by Whittaker (1923) which minimizes

$$L_2 = \sum_{i=1}^{m} (s_i - z_i)^2 + \lambda \sum_{i=2}^{m} (z_i - z_{i-1})^2, \tag{5.6}$$

where the original signal s is of length m and z is the approximate smooth series of s. The smoothness is determined by λ. Larger λ provides a smoother series z, but has a worse fit to the data y. It is common practice to find the optimal amount of smoothing, but here we do not aim to find an optimal value for λ. We use the P-spline implementation by Eilers & Marx (1996).

To quantify the effect of wave removal we compute the normalized difference $d = \sum_i (s_i - z_i)$ between the raw signal s and the smooth profile z (for each SNP i) on a given chromosome. Formally, we write

$$d = \frac{\sum |s_i - z_i|}{n}. \tag{5.7}$$

For the smooth series z we fix $\lambda = 100$. An increase of d indicates more scatter in the SNP signals, re-lative to the smooth estimate. For detection of constant segments between sharp breakpoints de-dicated (and better) algorithms are available, but here we aim for the removal of waves with gradient properties.

5.3 Empirical results

First we visually illustrate the origin of waves and then numerically compare the models discussed above.

Wave origins

In the left panels in Figure 5.3 the uncorrected signals are shown. Each panel contains two parts: on the left of the dashed line a healthy chromosome 1 is shown, while to the right of the dashed line a tumor chromosome 9 is shown. We display only a small selection of observation from one profile because different tumor patterns in different arrays would clutter the image. The top and middle row show the profiles for allele a and b separately, the bottom row shows the actual copy number signal $s = \log(a + b)$. The right column illustrates SCALA correction by α_i. It can be seen from Figure 5.3 that SCALA calibration with just the SNP parameter α_i is not effective for signals from a single allele a or b, but it is for the (logarithm of the) sum. Also note that all corrections do not remove copy number segments (as seen in the right part of each panel).

Numerical evaluation

We first visually inspect the results for SCALA and NoWaves. The top panels in Figure 5.4 show waves in an Affymetrix 250k tumor sample for two selected chromosomes (1 and 9). It is clear that the wave patterns occur on both healthy (1) and tumor (9) tissue. All panels in Figure 5.4 have the same scales on both the x and y axes. First, Figure 5.4 shows that after SCALA calibration (bottom row), the profiles hardly show any waves. The results for

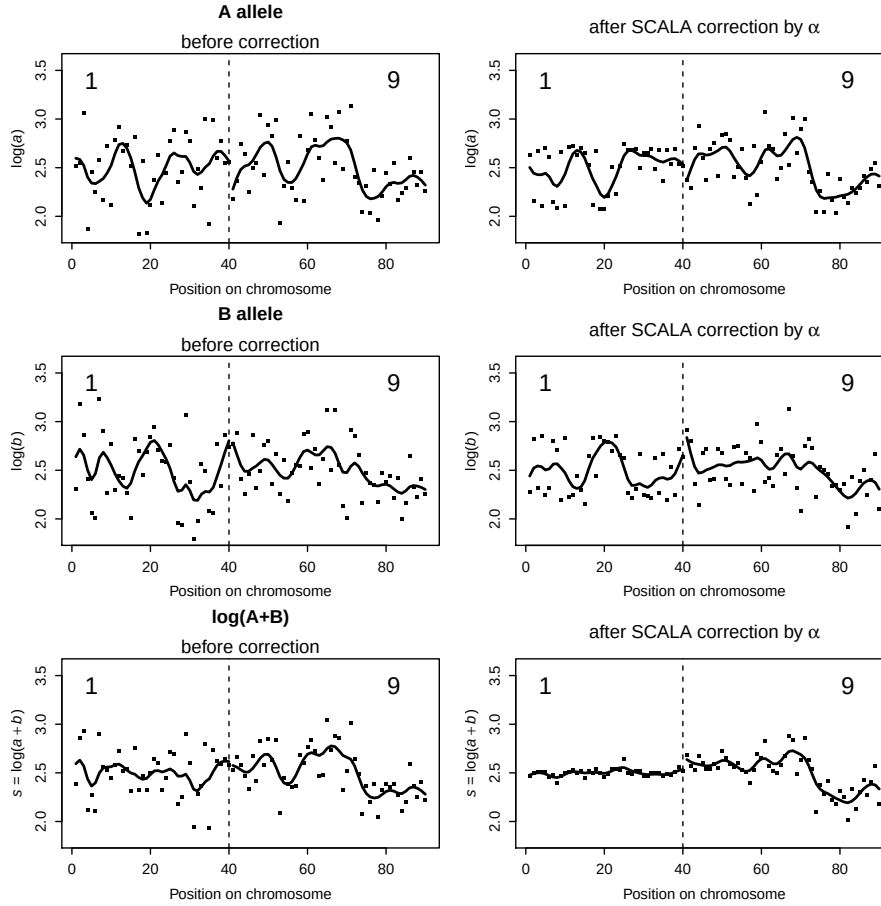


Figure 5.3: Wave patterns in real data. The horizontal axis shows the position of each observation in the sequence. The vertical axis shows either $\log(y)$ with y the allele signal for a or b, or $\log(a+b)$. Left column: uncalibrated signals. Right column: signals after SCALA calibration with α. Top panels: A allele, middle panels: B allele, bottom panels: CNV signal. Left parts of each panel show a healthy chromosome 1; right parts show an unhealthy chromosome 9. Smooth profiles are obtained with the Whittaker smoother ($\lambda = 2000$).

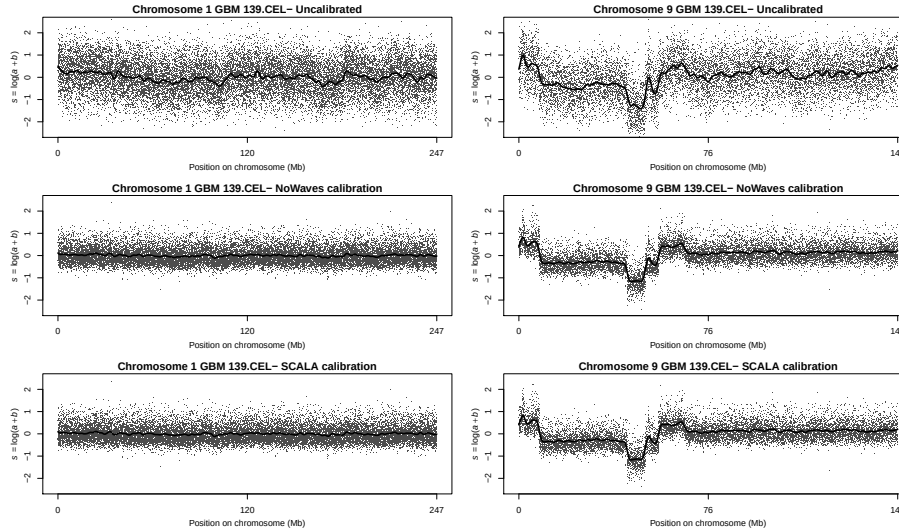


Figure 5.4: Profiles before (top) and after NoWaves (middle) and SCALA (bottom) calibration.

NoWaves (middle row) are similar to equivalent. Removing the waves from the signals clearly keeps CNV segments intact and quantifiable. In fact, the aberrations are the *only* deviations from the reference level $2n$ (2 alleles) that are still visible/detectable.

For the crude data, we find a benchmark d value of 0.60 (0.000-2.662) for chromosome 1 and 0.59 (0.000 -2.922) for chromosome 9. However, applying signal calibration based on SCALA, we find d values for both SCALA and NoWaves of 0.293 with the first ranging (0.000-2.297) and the second (0.000-2.328) for chromosome 1 and 0.295 (0.000-1.786) against 0.297 (0.000-1.745) for chromosome 9. Note that any differences between the latter methods are in the order of 10^{-3}.

Detailed results for chromosome 1 to 22 in several samples, for four different levels of smoothing ($\lambda \in 1, 10, 100, 1000$) are provided in Appendix A. Here too, differences between calibration methods are very small, but large compared to uncalibrated signals.

5.4 Discussion

We have illustrated that the cause of waves in CNV profiles based on SNP fluorescence signals is not spatial autocorrelation. Visual and numerical comparisons between two signal calibration methods, NoWaves and SCALA were made. The first method was developed specifically for single aCGH signals, whereas the second method was developed for two allele channels. The results for the two methods show almost equal improvements. The fact that model-based calibration is effective can be explained by the fact that SNP variation is larger than genotype variation, given that the calibration parameters were computed with only 8 profiles. Therefore, the maximum amount of genotype variation is low by definition.

It can be argued that after transformation to $s = \log_2(a + b)$, the NoWaves correction is already effective, so there is no need for a SCALA correction. However, NoWaves aims solely at wave removal for single channel profiles, while SCALA aims for allele-level correction, which is impossible for NoWaves. Another major advantage of SCALA over NoWaves is that the first calibrates signals with a set of parameters that is calculated only once and can be re-used in later instances, while the latter method needs to recompute the projection for every analysis. The smoothed references profiles can of course be re-used here, too. The SCALA calibration has a very simple nature, subtracting a vector of parameters. Therefore, we argue that it should always be applied, because it require hardly any time, removes waves and leaves segmentation intact.

One of the differences between SCALA and other methods is that for better correction, instead of GC content it exploits genotypes of the reference samples from which the calibration parameters are obtained. This introduces an extra step and thus an extra level of error-proneness. However, since calibration parameters are estimated using high quality reference samples and these data the genotype calls can be made very accurately, this does not pose a threat to the procedure.

It might also be argued that calibration is not necessary when a large amount of smoothing is applied on the uncalibrated data, since this already

removes most of the waves. However, in the right panels (Figure 5.4) we also see that within remaining CNV segments waves still distort the patterns. This problem is absent in the calibrated signals. Furthermore, applying too much smoothing on the raw data will in the end smooth out CNV segments.

In the current work we used a smoother based on the L_2 norm, but in Eilers & DeMenezes (2005) the L_1 norm is illustrated to be more effective in CNV detection. A further refinement to the L_0 norm was proposed by Rippe et al (2012b). The latter norms are much more suitable to detect aberrated regions, since it does not tend to round segment breakpoints (and the L_2 does, true to its quadratic nature). However, both the L_1 and L_0 norm do not respect the wave curvature and hence are not effective in the specific application described here.

Acknowledgements

We acknowledge Mark van de Wiel (VUMC, Amsterdam, The Netherlands) for providing assistance with the NoWaves software.

Visualizing genomic changes by segmented smoothing using an L_0 penalty

6

Copy number variations (CNV) and allelic imbalance in tumor tissue can show strong segmentation. Their graphical presentation can be enhanced by appropriate smoothing. Existing signal and scatterplot smoothers do not respect segmentation well. We present novel algorithms that use a penalty on the L_0 norm of differences of neighboring values. Visualization is our main goal, but we compare classification performance to that of VEGA.

6.1 Introduction

Copy number variations (CNV) and allelic imbalance are common in tumor tissue, reflecting local deviations from diploidy and heterozygosity. When they occur, they typically form segments of widely varying length. As a first step in their analysis, many researchers prefer to have a graphical presentation of genomic changes, as a kind of map along positions on chromosomes. Modern high-density SNP arrays make this possible for hundreds of thousands of positions on the (human) genome.

An array delivers two fluorescence signals for each SNP, one, say a, proportional to the dose of one allele, indicated by A, the other, say b, proportional to the dose of the other allele, indicated by B. This is only true in principle, because noise and differences between fluorophores of different color can distort the picture to a certain amount. If we ignore these facts for

This chapter is an adapted version of the article:
Rippe, R.C.A., Meulman, J.J. & Eilers, P.H.C. (2012). Visualization of Genomic Changes by Segmented Smoothing Using an L_0 Penalty. *PLoS ONE*, **7(6)**: e38230. doi:10.1371/journal.pone.0038230.

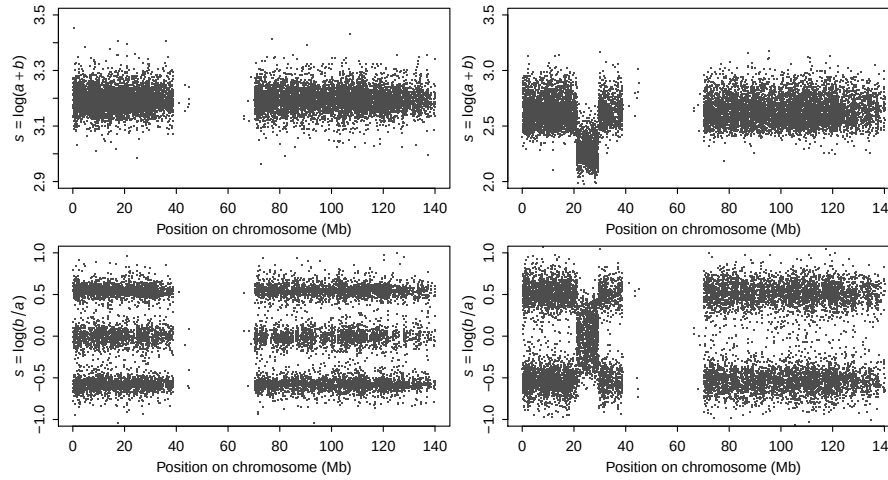


Figure 6.1: Illustrations of copy numbers and allelic ratio, expressed as logarithms, for healthy and tumor tissue. Left panels: healthy tissue. Right panels: tumor tissue. Top row: copy numbers. Bottom row: allelic imbalance.

the moment, and consider normal DNA, then the sum of the doses, the copy number, is 2, for any of the genotypes AA, AB or BB. Hence the sum $a + b$ should be almost constant. Similarly the ratio $b/(a + b)$ is either 0, 1 or 2; it is called the B allele frequency (BAF). Because in tumor DNA many types of changes can occur, leading to any number of A or B alleles from zero to many, a variety of deviations in CNV and BAF can be found.

We prefer to work with somewhat different combinations of the fluorescence signal. One is the log (to base 10) of their sum, $\log(a + b)$, which we abbreviate as LAS (log allelic sum). The reason for working with the logarithm is that usually a quite large range of values of $a + b$ is observed. The other combination is the logarithm of the allelic ratio, $\log(b/a)$, which we will abbreviate as LAR (log allelic ratio). Compared to BAF, LAR strongly expands the scale near 0 and 1, which is crucial when fitting (mixtures of) normal distributions, as we will do in one stage of our data analysis. Figure 6.1 shows examples of maps of the proposed quantities along chromosome 9 of a normal and a tumor sample.

Copy number analysis has received attention from many investigators; a short overview will follow later in this Introduction. In most cases the aim is to determine, with a solid statistical basis, segment boundaries and copy numbers and allelic doses within the segments. A variety of free and commercial products is available. Yet we believe that there is room for enhanced visualization tools, that allow us to inspect data in some depth before embarking on more formal models. Visualization tools for CNV are widely known, while such tools for allelic imbalance are rare. Therefore, we feel that it is most effective to introduce our new idea in the well-explored field of CNV (LAS) and assess its behavior in depth. Once we have obtained an understanding of its performance, we extend its application to a new setting (LAR), for which there are no "gold standard" comparisons available.

In this paper we present a new approach to copy number smoothing, extending the work of Eilers & De Menezes (2005). The main modification is to use a roughness penalty on the number of jumps, instead of on the sum of absolute values of jumps (the L_1 norm). We implement it with an L_0 norm, the sum of absolute values of differences raised to the power zero. The result is much sharper segmentation.

Copy number smoothing is relatively simple, because, as the top panels of Figure 6.1 show, we can interpret the data as one (segmented) trend plus noise. For the allelic ratio the situation is more complicated, because, as the bottom panels show, we can have one, two or three noisy parallel bands. Our solution is to adapt the scatterplot smoother of Eilers & Goeman (2004). In its standard form it computes a histogram on a large two-dimensional grid and applies a smoother on both axes, thus smearing out the counts in both directions. The smoother is based on a penalty on the sum of squares (the L_2 norm) of differences. We apply the same idea, but replace the penalty in the direction along the chromosome with one using the L_0 norm.

After segmentation with the modified scatterplot smoother, we present the distribution of LAR, separately for each segment, using histograms and Gaussian mixtures.

The literature on segmentation of copy number variations is large. It is a fascinating subject for statistical analysis and it has led to a variety of

modeling strategies. We present a short overview of recent work, without claiming completeness.

The hidden Markov model (HMM) is a natural candidate. Liu et al. (2010) propose a model with many hidden states, covering copy numbers from zero to seven. They claim improvements compared to older candidates like PennCNV (Wang et al., 2007) and QuantiSNP (Colella et al., 2007).

Other models use explicit parameters for the positions of jumps and the levels of the segments between them. VEGA (Morganella et al., 2010) uses dynamic programming, while Muggeo & Adelfio (2011) fit a piecewise linear model by maximum likelihood.

Non-parametric smoothing goes in the opposite direction, by modifying smoothing algorithms in such a way that they favor a piece-wise constant fit. MSMAD (Budinska, Gelnarova & Schimek, 2009) is an improvement on the work of Eilers and De Menezes (2005). The fused LASSO works in a similar way (Tibshirani & Wang, 2008).

Systematic comparisons of a number of models are available. We mention Lai et al. (2005), Marenne et al. (2011), Winchester et al. (2009), Tsuang et al. (2010), and Zhang et al. (2011). Large-scale assessments over platforms, lab sites and algorithms were made in Bengtsson et al. (2009). The rest of the paper is organized as follows: in Section 2 we present the algorithms, using real data to illustrate them. In Section 3 we compare our segmentation, obtained after automatic selection of the smoothing parameter, with the segmentation from VEGA. In Section 3 we also present applications to clinical samples, including a comparison with segment calls from external software, CNAG (Nannya et al., 2005).

As an acronym for our smoother we use ZEN, derived from Zero Exponent Norm, because the L_0 norm in the penalty is crucial to its success.

6.2 Statistical methods

In this section we first discuss LAS smoothing with penalized least squares, based on several types of norms in the difference penalty. We present a

procedure to automatically find a good value for the penalty parameter, using cross-validation. Then we extend the discussion to segmented scatterplot smoothing of LAR. In contrast to smoothing methods that use the sum of squares of absolute values in the norm of the penalty, the objective function of the ZEN smoother is not convex. There is no guarantee that a (unique) global minimum will be reached. Yet in practice we see excellent performance. To increase the confidence of potential users of our methods, we present a short study of convergence behavior.

Segmented CNV smoothing

Let the data be m data pairs (x_i, y_i), where x_i gives the position of SNP i ($x_i < x_{i+1}$ for all i) and y_i is the copy number signal LAS, $\log(a + b)$. We are going to compute a smooth series z.

Our starting point is a variant of the Whittaker smoother (see also Eilers, 2003). The objective function is

$$S_2 = \sum_{i=1}^{m}(y_i - z_i)^2 + \lambda \sum_{i=2}^{m}(z_i - z_{i-1})^2. \tag{6.1}$$

The first term measures fidelity of z to y, while the second term is a penalty on roughness of z. The balance between the two is set by the parameter λ; the larger λ is chosen, the smoother z will be. This smoother rounds off edges as is illustrated in the top panel in Figure 6.2. This is fine in many applications, but not here.

Quantile smoothing replaces the sum of squares (the L_2 norm) by sums of absolute values (the L_1 norm). The objective function is

$$S_1 = \sum_{i=1}^{m}|y_i - z_i| + \lambda \sum_{i=2}^{m}|z_i - z_{i-1}|. \tag{6.2}$$

Notice that now fidelity to the data is measured by the sum of the absolute values of $y - z$ (median smoothing), not by their squares. This modification is necessary because a linear programming algorithm is used to compute $\hat{z}$. This increases robustness, but decreases sensitivity to the data, compared to the L_2 norm. Robustness is hardly an issue in CNV studies.

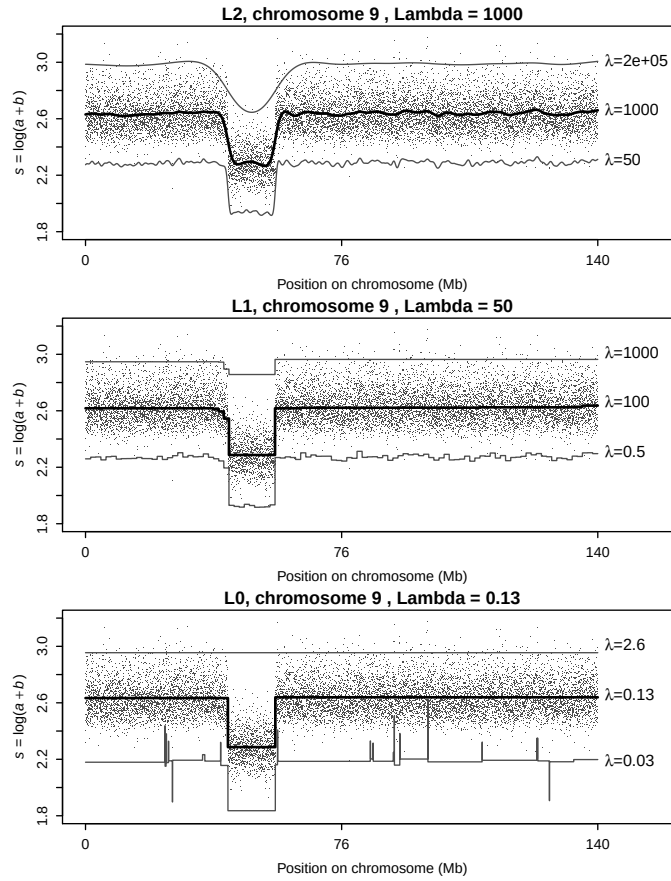


Figure 6.2: Illustration of smoothing with different norms (2,1,0) in the roughness penalty. Top panel: L_2 norm, the Whittaker smoother. Middle panel: L_1 norm. Bottom panel: L_0 norm. Thinner lines drawn with positive and negative offsets illustrate the effect non-optimal λ. Top line: λ too large. Bottom line: λ too small.

As can be seen from the middle panel of Figure 6.2, this modification goes in the right direction. Segments become more clearly visible, but a number of undesirable small jumps occur.

We propose the following modification:

$$S_q = \sum_{i=1}^{m} (y_i - z_i)^2 + \lambda \sum_{i=2}^{m} |z_i - z_{i-1}|^q \tag{6.3}$$

where q is a number between 0 and 1. Actually we will concentrate on $q = 0$, the L_0 norm. Essentially this is a penalty on the number of non-zero differences between neighboring elements of z. Any positive number raised to the power 0 gives 1, while by convention $0^0 = 0$. So only non-zero differences add to the penalty, and all by the same amount, independent of their size. Our numerical algorithm approximates this behavior. The lower panel of Figure 6.2 shows results obtained with the proposed smoother.

Computational details

It is easy to find the solution for the Whittaker smoother, using matrix-vector operations. If D is a matrix that forms first differences, so that if $u = Dz = \Delta z$, $u_i = z_i - z_{i-1}$, the objective function can be written as $S_2 = ||y - z||^2 + \lambda ||Dz||^2$, with an explicit solution that follows from the linear system $(I + \lambda D'D)\hat{z} = y$. The system is very sparse, which can be exploited in Matlab or R (we use the package spam), leading to computation times that increase linearly with the length of the data series.

We propose a simple, but effective, algorithm to minimize S_q, using iterated weights in an adapted Whittaker smoother, borrowing from Schlossmacher (1973). It is clear that $|a|^q = a^2|a|^{q-2}$, for any number a. If we do not know a itself, but an approximation $\tilde{a}$, then $|a|^q \approx a^2|\tilde{a}|^{q-2}$. Using this relation, we approximate $|z_i - z_{i-1}|^q$ by $v_i(z_i - z_{i-1})^2$, with $v_i = |\tilde{z}_i - \tilde{z}_{i-1}|^{q-2}$. If $V = \text{diag}(v)$, the system to be solved becomes $(I + \lambda D'VD)\hat{z} = y$. This gives a new approximation to the solution from which new weights are computed. These steps are iterated until convergence.

The function we try to optimize is non-convex, but with decent starting values optimization is effective. However, to improve numerical stability and reduce the number of iterations, we modify the weights somewhat: $v_i = [(\tilde{z}_i - \tilde{z}_{i-1})^2 + \beta^2]^{(q-2)/2}$, where β is a small number, of the order of 1/10000th

of the expected size of the jumps. If β is set not small enough, rounding will occur near the jumps.

Cross-validation for a good λ

A useful property of the smoother is that it automatically interpolates values for missing observations if we introduce proper weights. The objective function is modified to

$$S_q = \sum_{i=1}^{m} w_i (y_i - z_i)^2 + \lambda \sum_{i=2}^{m} |z_i - z_{i-1}|^q \tag{6.4}$$

For a missing, or left-out, observation we set $w_i = 0$; all other weights are set to 1. Smoothly interpolated values for z will be computed automatically. The system to be solved in each iteration becomes

$$(W + \lambda D'VD)\hat{z} = Wy,$$

with $W = \mathrm{diag}(w)$.

We exploit this property in cross-validation (CV) to find the optimal smoothing parameter λ. We leave out the even observations, by setting their weights to zero. We then compute

$$\mathrm{CV} = \sqrt{\sum_i (1 - w_i)(y_i - \hat{z}_i)^2}$$

for a series of values of λ (a linear sequence for $\log \lambda$) and search for the minimum of CV. This simple cross-validation scheme works well in practice.

Notice that the value of λ that minimizes CV should be doubled when smoothing the complete data. The value of $\sum_{i=1}^{m} w_i (y_i - z_i)^2$ is close to half that of $\sum_{i=1}^{m} (y_i - z_i)^2$, while the penalty contains all elements of z and so will have approximately the same value, whatever the weights.

Applying odd/even cross-validation is effective, as is illustrated in Figure 6.3. For the cross-validated fit values we observe a clear minimum (top panel), while the smoothed result (bottom panel) looks adequate too, when judged visually.

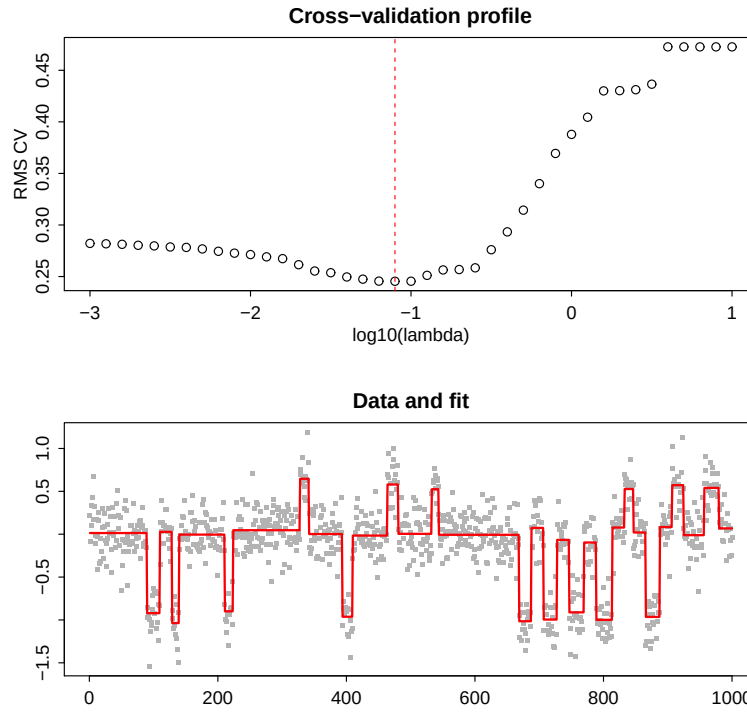


Figure 6.3: Odd-even cross-validation for finding an optimal λ. The selected λ is indicated in the top panel by the vertical broken line. The bottom panel shows data using (double) the selected λ against the raw data. The doubling is needed to compensate for leaving out half of the data.

We don't want to overstate the importance of cross-validation and optimal smoothing in the present application. Our primary goal is visualization and we expect that the user will play with λ when exploring data. The "optimal" value of λ should only be considered an advice. Because the necessary computations take little time on a modern PC, interactive use is possible with attractive speed.

In Section 3 we compare the classification performance of our smoother

with that of VEGA, using cross-validation to select λ.

Convergence behavior

The objective function of the smoother is non-convex, because of the L_0 norm in the penalty. Hence there is no guarantee that local minima do not exist, nor that we will always reach a global minimum. Yet in our experience the results make a lot of sense when inspected visually. So even if a solution might not be optimal — and we have no practical means to decide on that — it can be very useful. In this section we present some details on convergence behavior, following the iterations of smoothing with the adaptive weights in the penalty.

Figure 6.4 presents results for a data set with relatively little noise. They were obtained from the VEGA website (Morganella et al., 2010). We smooth with $\lambda = 0.2$ and show the current estimate of the solution z at five iteration steps. In the first iteration, all weights, v, in the penalty are equal to 1. So effectively we have a light Whittaker smoother. After the first iteration the adaptive weights take effect. As can be seen, after five iterations the final result has almost been reached. The (logarithms) of the change in the solution from one iteration to the next are shown in the lower right panel. The changes are computed as the maximum of the absolute values of the differences.

In this example sufficient convergence has been reached quickly, certainly for visualization purposes. In our experience 20 to 40 iterations is typical. Figure 6.5 shows a noisier data set (also from VEGA), where $\lambda = 0.5$. Convergence is slower there.

Segmented scatterplot smoothing

A fast smoother for scatterplots was introduced in Eilers & Goeman (2004). The principle is to first compute a two-dimensional histogram on a large grid (say 100 by 100 bins) and to smooth first the columns and then the rows with a Whittaker smoother having a slightly changed roughness penalty. In

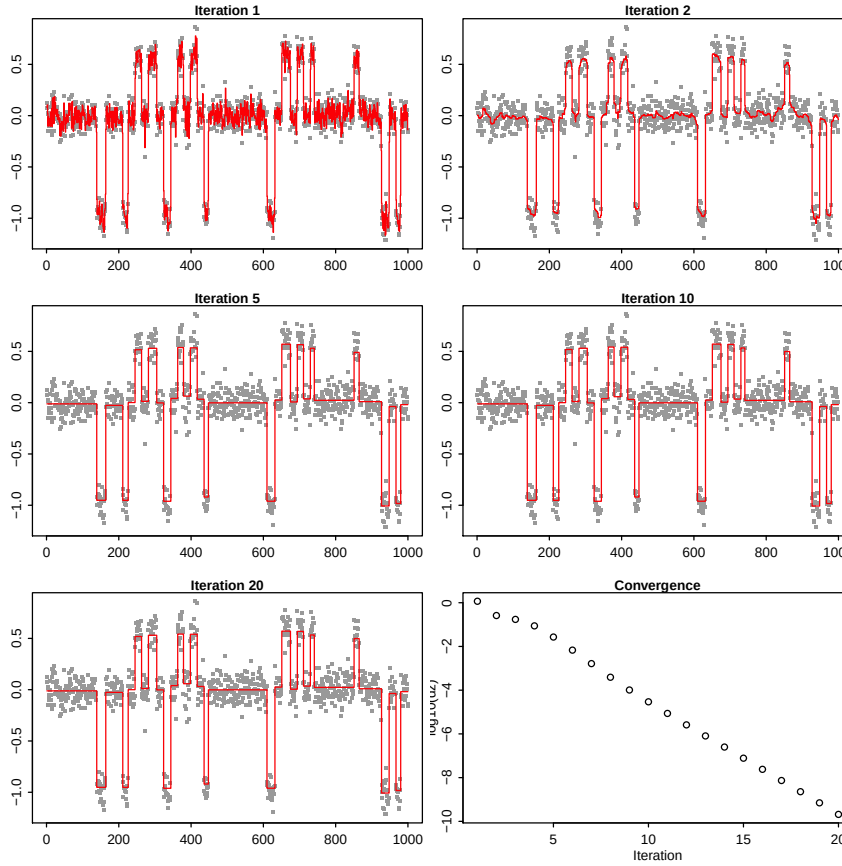


Figure 6.4: Illustration of convergence behavior in zero-norm smoothing with little noise. The data are simulated (VEGA package) and contain relatively little noise. All panels, except the lower-right one, show intermediate solutions, at the iteration numbers as indicated in the titles of the panels. The lower right panel shows the largest absolute change in the solution at each iteration. The smoothing parameter is set to $\lambda = 0.2$.

order to ensure positive values in the histogram, a combination of a first and second-order penalty is used. If y represents one column of the histogram,

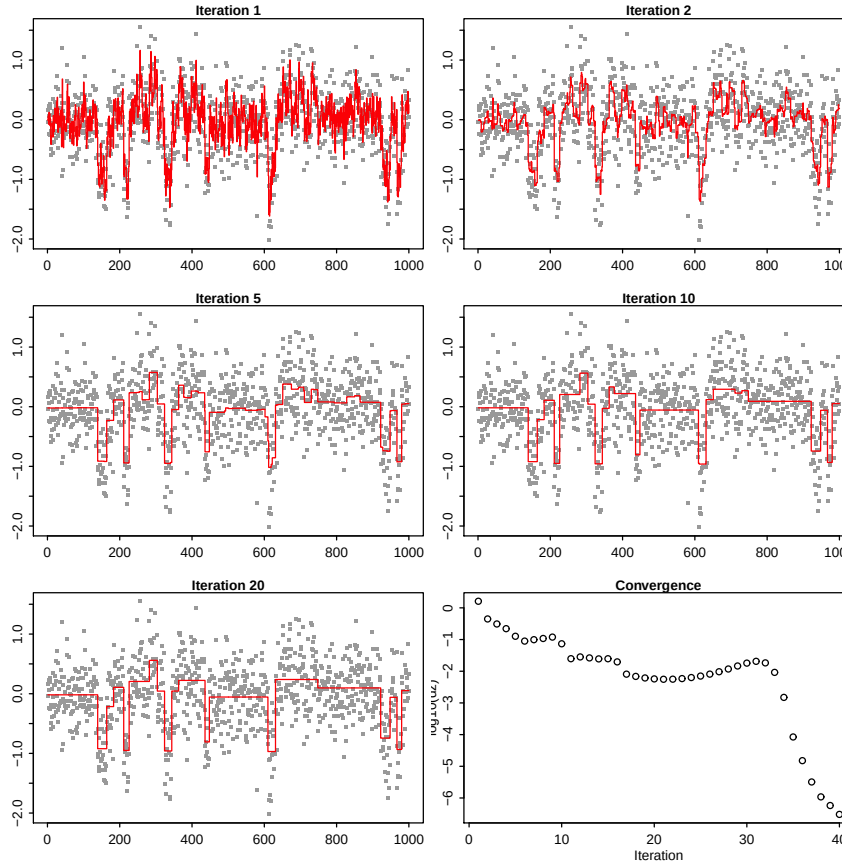


Figure 6.5: Illustration of convergence behavior in zero-norm smoothing with moderate noise. Illustration of convergence behavior. The data are simulated (VEGA package) and contain relatively much noise. All panels, except the lower-right one show intermediate solutions, at the iteration numbers as indicated in the titles of the panels. The lower right panel shows the largest absolute change in the solution at each iteration. The smoothing parameter is set to $\lambda = 0.5$.

that will be smoothed to get z, the objective function is:

$$Q = |y - z|^2 + \lambda^2 |D_2 z|^2 + 2\lambda |D_1 z|^2. \tag{6.5}$$

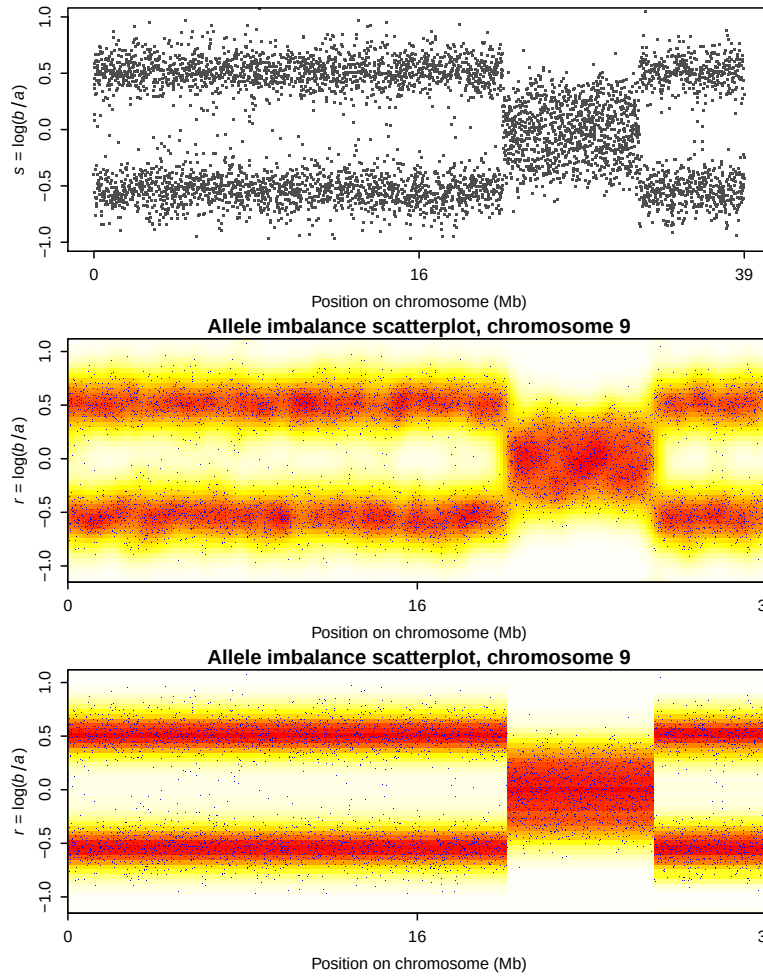


Figure 6.6: Comparing normal and segmented scatterplot smoothing. Top panel shows the raw observations. Middle panel shows straightforward smoothing: no segmentation. Bottom panel shows segmented smoothing: clear segments.

Notice the combinations of a first (D_1) and second order (D_2) difference penalties. A (banded) linear system of equations results:

$$(I + \lambda^2 D_2' D_2 + 2\lambda D_1' D_1)\hat{z} = y. \tag{6.6}$$

The lower panel of Figure 6.6 shows results obtained with this smoother, when applied to a scatterplot of (log) allelic ratio against chromosomal position. The raw observations are shown in the top panel. This would be a useful display if it showed sharp segment edges like those we obtained for copy numbers, while maintaining smoothness in the other direction.

For the segmented scatterplot smoother, we keep the original penalty for the allelic ratio, but for the position we use a penalty based on the L_0 norm of first differences. It will not work to just use that penalty for each row of the histogram: we get segments, but they will generally be in different places for different rows. To avoid it we use the same weight matrix V in the penalty $\lambda|D_1' V D_1|$, but now compute it as the summary of all rows:

$$1/v_j = \sum_i (z_{ij} - z_{i,j-1})^2/m + \beta^2,$$

with m the number of rows and β again a small number to increase stability and speed of convergence. Figure 6.6 (bottom panel) shows a result obtained in this way. Now we get sharp segment boundaries.

A typical vector v consists mostly of large numbers and a few small ones. The latter indicate the segment boundaries and these values have been used to enhance the figure with vertical broken lines at the boundaries.

Once the segment boundaries have been found, it makes sense to plot histograms of the (log) allelic ratio for each segment separately. They are shown in Figure 6.6 and 6.8. In addition we fit gaussian mixtures using the package mclust (Fraley & Raftery, 2007). The centers of the mixture components can be used to summarize results and to help the user in interpreting the observed genomic changes. We do not discuss that here, because we feel that that would stray us to far away from our primary goal, visualization.

Like the scatterplot smoother of Eilers & Goeman (2004), we see the segmented scatterplot smoother only as a visual aid. We did not try to develop

an algorithm for automatic choice of the amount of smoothing, nor did we try to simulate realistic allelic imbalance scenarios to evaluate performance.

6.3 Simulations

A method for visual segmentation is less useful when it remains unclear whether a correct segmentation is found. In this section we compare performance of our smoother with that of VEGA on CNV segment detection.

We use again the simulated data that are provided by Morganella et al. (2010). It contains simulated CNV data for 22 chromosomes, for each of which there are 1000 data points generated. For each chromosome random mutations were generated with a segment length varying between 11 and 25 points. Gain or loss properties for each segment were also randomly selected. Additionally, these data are provided with 10 levels of noise ($\sigma \in \{0.0, 0.1, \ldots, 0.9, 1.0\}$), where $\sigma = 0$ indicates perfect data. We will use these as a reference for segment recovery.

Comparisons between the VEGA method and the proposed L_0 norm smoother are made in terms of *precision*, *recall* and associated *F*-scores. All of these require True Positive Rate (TPR), False Positive Rate (FPR), True Negative Rate (TNR) and the False Negative Rate (FNR). Hits compared to the noise-free data are assessed per individual data point. We define a deviation as at least 1% of the largest difference between the smoothed signal and the baseline normal signal (here: 0). A match is defined as a single observation for which such a deviation from zero (0) was found in both VEGA and ZEN.

Precision (positive predictive value) is defined as

$$P = \frac{TPR}{TPR + FPR}.$$

Recall (sensitivity) is defined as

$$R = \frac{TPR}{TPR + FNR}.$$

F-scores (harmonic mean, interpreted as a weighted average of precision

Table 6.1: Comparing ZEN (L_0) and VEGA on (P)recision, (R)ecall and (F)-value, using simulated data.

	ZEN			VEGA		
σ	P	R	F	P	R	F
0.0	1.000	1.000	1.000	1.000	1.000	1.000
0.1	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
0.2	0.999	**1.000**	0.999	**1.000**	**1.000**	**1.000**
0.3	0.976	0.992	0.984	**0.989**	**0.993**	**0.991**
0.4	0.808	0.938	0.864	**0.911**	**0.953**	**0.931**
0.5	0.797	0.912	0.848	**0.867**	**0.916**	**0.888**
0.6	0.635	**0.821**	**0.709**	**0.675**	0.770	0.706
0.7	0.619	**0.797**	0.687	**0.669**	0.794	**0.721**
0.8	0.601	**0.818**	**0.687**	**0.630**	0.785	0.685
0.9	**0.530**	0.614	0.536	0.469	**0.741**	**0.565**
1.0	**0.485**	0.593	0.514	0.465	**0.752**	**0.559**

and recall) are given by the combination of P and R:

$$F = 2\frac{P \times R}{P + R}.$$

We present results for method comparison on the simulation data, cross-validation effectiveness and convergence. They are summarized in Table 6.1. Note that for the F-scores, 1 = best performance and 0 = worst performance. The best performing method is indicated in bold font. It can be seen that for no and very little amount of noise (0.1), performance for the L_0 norm and VEGA are equivalent. Increasing the noise levels VEGA seems to perform slightly better. For noise level 0.6, VEGA wins for precision, but not for Recall and F-score. For even higher levels of noise, there is no clear winner. However, these levels of noise are not very interesting, since real-life data of this quality would not be analyzed.

6.4 Applications

In this section we discuss two applications: smoothing of CNV signals (as in the above study) and scatterplot smoothing combined with segmented mixture estimation. The data were obtained in the Erasmus University Medical Center and concern several types of brain tumors (Bralten et al., 2010). In the examples below, we use tumor samples named GBM 139.CEL, GBM 180.CEL, GBM 203-2.CEL and GBM 254.CEL. Since this research focuses largely on chromosome 9, we only use signals on this chromosome in our illustrations.

Figure 6.7 shows smoothing of copy number variations in GBM 139.CEL, using odd-even cross-validation to select a good λ. There is not much to say about this result: the segmentation conforms to our visual impression of what the data tell us. Remarkable is the rather narrow segment at 28 MB that is detected.

ZEN smoothing of the allelic ratio in GBM 139.CEL is shown in Figure 6.8. Most segment boundaries, but not all, correspond to those found in the copy number signal.

Although ZEN performance was already addressed, we also compared our copy number results to results from dedicated copy number software, CNAG (Nannya et al., 2005). In Figure 6.9 we show copy number maps for selected interesting regions on chromosome 9, and we show the corresponding segmented allelic imbalancemap for the four samples mentioned above. In Figure 6.10 it shows that CNAG provides equivalent results on the same selected regions, but with less noise in the smoother. Therefore, we argue that ZEN outperforms VEGA.

The adaptive weights in the penalty are small where jumps occur, and so they indicate segment boundaries. This was done to produce Figure 6.11, where histograms and estimated normal mixtures are shown. The package mclust was used to estimate the mixtures. It chooses the number of components (which we limited to maximally four) based on BIC. Apparently the two components of the mixture in the top-right panel have longer tails than a normal distribution, and mclust uses the sum of a narrow and a wide normal distribution to approximate them.

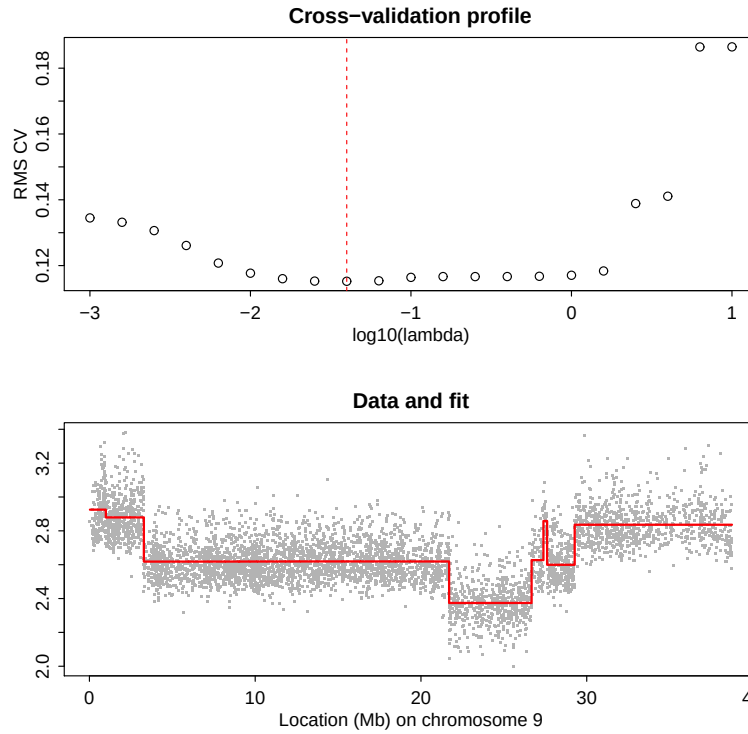


Figure 6.7: ZEN smoothing of CNV in tumor data (sample GBM139.CEL). Top panel: cross-validation profile and location of minimum (at broken vertical line). Bottom panel: data and fit, using $\lambda = 0.08$ (double the value indicated by cross-validation, to correct for leaving out half of the data).

6.5 Discussion

Smoothing algorithms generally have two components: one to measure the fidelity to the data, the other a penalty on roughness of the result. For the first term typically a sum of squares or of absolute values of residuals (i.e. data minus fit) is being used. To measure roughness, the size of the differences between adjacent fitted values is an effective and attractive choice. The

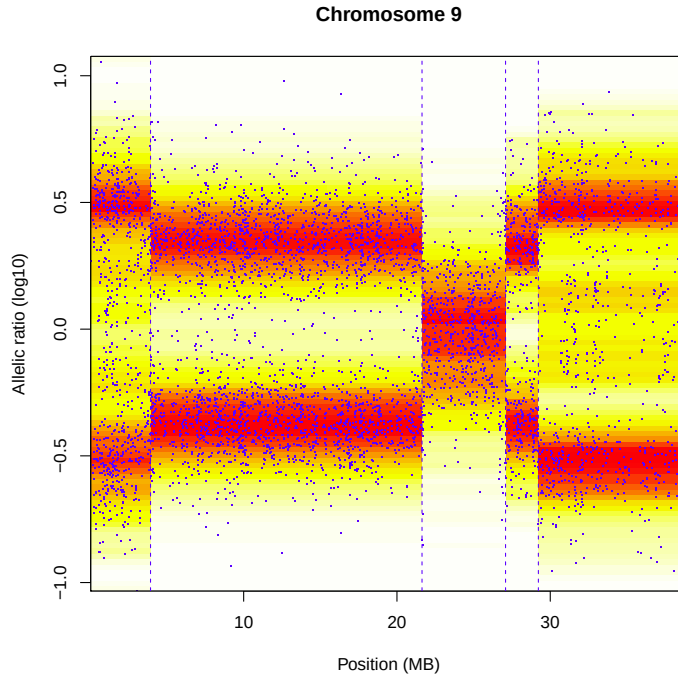


Figure 6.8: ZEN smoothing of log allelic ratio (sample GBM 139.CEL). The vertical broken lines indicate the segment boundaries, as computed from the adaptive weights in the penalty. The smoothing parameters (λ) are 0.01 for position and 0.5 for log allelic ratio.

way these differences are being expressed has a large influence on the shape of the fitted curve. Eilers & DeMenezes (2005) showed that a variant of the Whittaker smoother, using the L_1 norm in the penalty on differences, is attractive for copy number smoothing, because it can deliver constant segments with relatively sharp jumps in between.

We propose to use the L_0 norm, essentially the count of the number of jumps. To make computation practical, we also present an algorithm based on iteratively re-computed weights in a sum-of-squares penalty. This turns

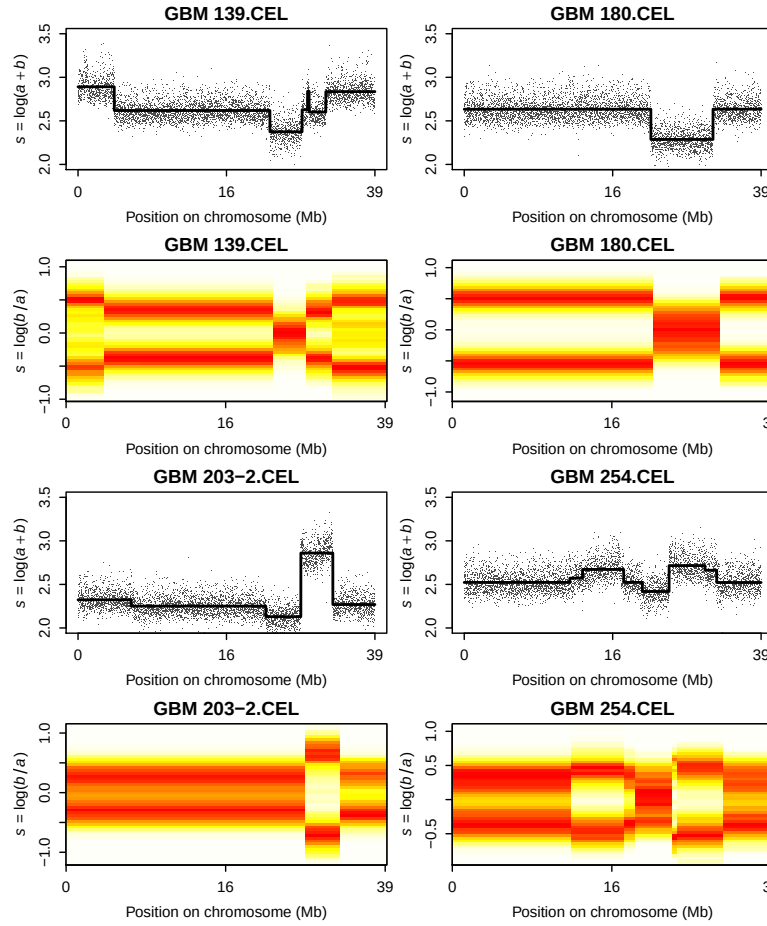


Figure 6.9: Examples of smoothed CNV and allelic imbance in clinical samples, using ZEN. First and third row show CNV profiles, second and fourth rows show the matching segmented allelic imbalance plots.

out to be effective: very sharp jumps between segments are obtained.

Because our algorithm can automatically interpolate missing data, it is possible to use a simple odd-even scheme for cross-validation, to automatically choose the amount of smoothing. However, we propose cross-validation

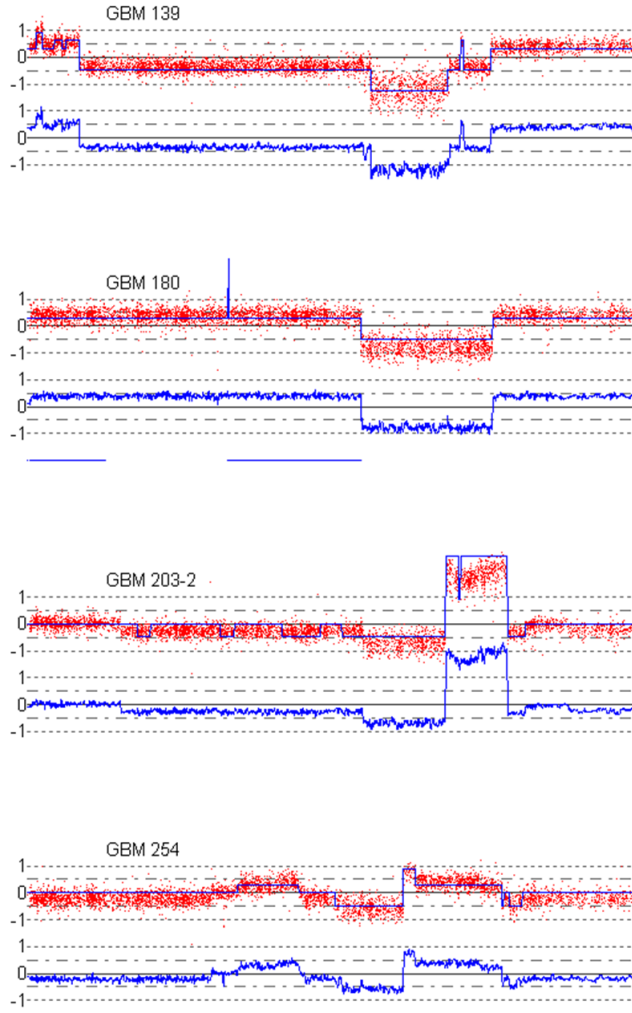


Figure 6.10: Examples of smoothed CNV in clinical samples, using CNAG software. Panels show CNV profiles for the samples mentioned in the panel titles. The smoothed signals show unexepected jumps (GBM180) and unclear level overestimations (GBM203-2).

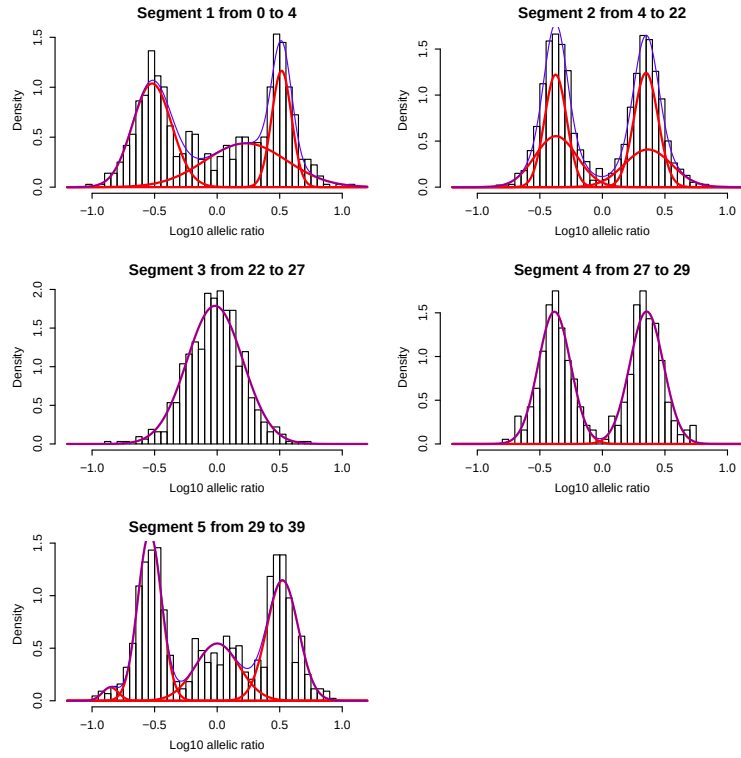


Figure 6.11: Histograms and estimated normal mixtures for the log allelic ratio. Estimations are separate for each of the five segments that were derived from the scatterplot smoother in Figure 6.8.

only as a guide to find a good ball park for the penalty parameter, because fast and easy visualization is our main goal.

We use cross-validation-based smoothing to compare classification performance in a little contest with VEGA, using the simulated data that come with that software. The performance of our smoother is quite close to that of VEGA. This should give users the confidence that the segments they get are realistic ones.

The objective function of the proposed smoother is non-convex. In principle this is a cause for worries: we can never be sure that the global minimum was found. In practice we have seen that we always get very good results, as judged by visual inspection. To give some insight, we presented a few illustrations of how intermediate results converge towards the final solution.

A plot of copy numbers along a chromosome contains only one “curve" as a noisy band with jumps. A plot of allelic imbalanceis different: at any position from one to three bands can be present. Jumps are present too and there the number of bands as well as their positions can change. The smoothing algorithm for copy numbers will not work on such data. Instead we modified the scatterplot smoother of Eilers & Goeman (2004), which is based on smoothing rows and columns of a two-dimensional histogram by penalized least squares. One of the penalties was changed, to accept iteratively recomputed weights, like in the copy number smoother. The weights are based on summaries of the columns of the histogram, to have the same segment boundaries in all rows. The approach is rather ad-hoc, as there is no explicit objective function to minimize, but the results look attractive and computation is fast, allowing interactive use.

Segmented smoothing of allelic imbalance can indicate boundaries that are not visible in copy numbers. An example is copy number-neutral loss of heterozygosity. It makes sense to study histograms of the (log of the) allelic ratio for each separate segment in the plot. In addition to histograms we also propose fitting of mixtures of normal distributions. The package mclust gives good results.

In summary, we believe that we have extended the toolbox for exploration of copy number variation and allelic imbalance with attractive new instruments. All computation was done in R (R Development Core Team, 2012) and the programs are available from the first author on request (Rippe et al., submitted).

The SCALA software suite

7

The SCALA Suite for Single SNP chip analysis provides functions to convert raw CEL-files to Rdata objects, one for each chip. Furthermore, given a set of high quality arrays, universal calibration parameters can be computed and applied to new arrays. Genotypes are called on a single chip with a dedicated function. Maps of copy numbers and allelic imbalance are also implemented for single arrays.

7.1 Introduction

SNP (Single Nucleotide Polymorphisms) arrays have two major applications: genotyping of DNA and studying copy number variations (CNV) and allelic imbalance. Here we describe integrated R software called SCALA designed for this purpose. It has a unique combination of properties: it can perform CEL file conversion, genotyping, copy number mapping and signal calibration. After using SCALA no further software is needed. In the remainder of this Introduction we describe the main components of SCALA in more detail.

Genotyping

To estimate all SNP genotypes in a single chip, semi-parametric log-concave mixtures were proposed in Rippe, Eilers & Meulman (2010). One reason is that the latter only works effectively on array sets of reasonable size, in order

This chapter is an adapted version of the submitted article:
Rippe, R.C.A., Eilers, P.H.C. and Meulman, J.J. (2012). SCALA: a software suite for single chip SNP calibration, genotyping and copy number mapping, *submitted for publication*.

to obtain stable estimates. Furthermore, low minor allele frequencies pose additional problems. The above is circumvented when genotypes are called for a single chip. Proposals for small sample sets have been made, also using mixtures (ALCHEMY by Wright et al., 2010), as well as a combination of single and multi-array analysis (MAMS by Xiao et al., 2007). Building on their arguments we have developed an algorithm that performs single array genotyping. It is based on a two-dimensional mixture of log-concave densities (along the lines of Eilers & Borgdorff, 2007), fitted on 2-dimensional histograms (Eilers & Marx, 2007). To estimate a mixture with three smooth components, we use the familiar EM (expectation-maximization) algorithm. Two steps are repeated until convergence: 1) split the counts y into three vectors of pseudo-counts, proportional to the current estimate of the mixture components; 2) apply smoothing to the pseudo-counts. Decent starting estimates for the components are needed. In Rippe et al. (2010) genotype calls from a multi-array method (CRLMM) and from our single-array method (SCALA) are compared to a set of consensus genotypes from HapMap. The number of agreements and differences in terms of homo- and heterozygous calls showed that SCALA can be used to call genotypes efficiently and effectively. Even SNPs that were not genotyped in HapMap can be genotyped with reasonable certainty using a single chip. The above model is implemented in the `SCALA.genotype` function.

Visualization of copy numbers and allelic imbalance

DNA in tumors can show a variety of deviations like allele copy number variation (CNV) and allelic imbalance. SNP arrays provide a fluorescence signal for each allele, both of which are assumed to be proportional to the number of both alleles. Sums ($\log(a+b)$) and ratios ($\log(b/a)$) of these signals can be plotted, on logarithmic scales, versus positions on chromosomes, to give a useful graphical representation (like in DNACopy, 2010; Golden Helix, 2011). These plots can be enhanced in several ways. Here we present an R program, called SCALA.Map, for this purpose. The program offers smoothing of CNV and allelic imbalancesignals.

SNP signal intensity calibration

SNP fluorescence signals are not perfect: they contain "noise". This noise appears not to be random, implying that it can be modeled in order to correct for it and so calibrate the intensity signals. A remarkable and useful property of fluorescence signals from all types of platforms is that they contain a specific structure. First, there is the (trivial) difference between arrays, which most readers are familiar with. However, a similar pattern also holds for individual SNPs over sets of arrays, and this is what we exploit here. Given a set of called genotypes for the current array, one can obtain calibration parameters for arrays and SNPs. These parameters need to be estimated only once for a given chip type, based on a set of (high quality) arrays. Once a set of calibration parameters has been estimated, it can be used to calibrate the signals in any new individual array, without the need of knowing the genotypes. We estimate α using a set of high quality samples of normal tissue. Estimation of these parameters is implemented in the function `SCALA.calibrate`. The calibration is very effective in copy number mapping, but it can also be used in genotyping. The latter is only offered as an experimental function; assessment has yet to be performed.

Functions and a graphical user interface

The modules for file conversion, genotyping and calibration are accessed via regular function calls. Copy number mapping is done through a custom graphical user interface, which controls the plot settings as well as export options.

7.2 Functions and implementation

In this section we describe the main function in the software suite. We start with CEL files conversion, then discuss calibration, genotyping, and finish with a graphical interface for copy number estimation. Supporting data files and subfunctions are placed in the relative folder locations `'../Maps'` and

'../Calibration' (for file conversion), and '../Support Files' (for genotyping and mapping).

File conversion: SCALA.convert

The software is built around an object of class SCALA, which is a conversion of a raw CEL file to aggregated fluorescence signals for each allele. The conversion function is specific for each chip type, but the result is generic. Currently supported platforms are Affymetrix (100k Hind and Xba, 250k NSP and STY, and SNP6.0) and Illumina (Infinium). For each Affymetrix chip type, probe maps from the corresponding BioConductor packages are used to match the probes for signal aggregation. A call to the function converts all CEL files of the same type (Affymetrix 250k NSP) in the current working folder. It is used as

```
> SCALA.convert(datatype=[type], calibrate=[T/F]),
                readfolder=getwd(), savefolder=getwd() )
```

Possible data types that are currently implemented are `Affy50kHind`, `Affy50kXba`, `Affy250kNSP`, `Affy250kSTY` and `AffySNP6.0`, which are self-explanatory. For successful conversion, if `SCALA.convert` is located in folder `X:/`, then the conversion maps should be placed in `X:/Maps`. Note that these files need to be of the same chip type; here all files are Affymetrix 250k NSP chips. Including other chips will provide errors.

For Illumina arrays we have a function that takes the X-raw and Y-raw columns. To use these arrays, the X and Y components in the SCALA object described below should be replaced with the X-raw and Y-raw columns from an Illumina data file. The chromosome allocation and position can be replaced similarly.

Genotype calls are all set by default to `NA` after file conversion but they can be added from any other source like HapMap, CRLMM or BirdSeed. Genotypes from HapMap have to be matched by SNP ids. This is because not all SNP ids in a sample are genotyped in HapMap with an identical id. Therefore, adding HapMap genotypes is also not (yet) automated. The required

format is a vector having AA=1, AB=2 and BB=3 for the genotype for each SNP, with SNPs and genotypes in the same order.

Precomputed calibration sets are also provided for a number of platforms, so that calibration can already be performed at the file conversion stage. However, it is also possible to convert a custom set of arrays, add genotypes, and then compute the calibration parameters from this new set with the function in section 7.2 and correct the signals manually.

Estimating SNP genotypes: SCALA.genotype

After file conversion genotype information is not available (NA), but can be obtained with the provided calling function. It requires a number of parameters. First, the number of bins (in both horizontal (`xbin`) and vertical (`ybin`) direction) for the histograms smoother has to be chosen. Second, the smoothing parameter λ has to be set, to determine the amount of smoothing in the histogram. Third, the initial vertical split levels for the three mixture components have to be set. These levels are defined in terms of the ratio of the number of vertical bins. If in Figure 7.3 the number of vertical bins is set to 100, the split levels for the left panel could be [0.45, 0.55] leading to splits at bin 45 and 55. Similarly, for the right panel initial split levels might be [0.50, 0.70]. The number of iterations is limited by `nit` and the convergence criterion is set by `crit`. After invoking the function, the `NA`s in `calls` are replaced with real genotypes by

```
> SCALA = SCALA.call(data=[name-of-CELfile], model=[],
                     plot=[T/F], save=[T/F], xbins=[val],
                     ybins=[val], lambda=[val], split1=[val],
                     savefolder=[])
```

As stated before, genotypes can also be added from another source (e.g. HapMap), but currently no function is provided to do so. If done manually, make sure that the ordering of the external genotypes matches the SNP ordering in the object. The column `SCALA$rsid` can be used for this purpose.

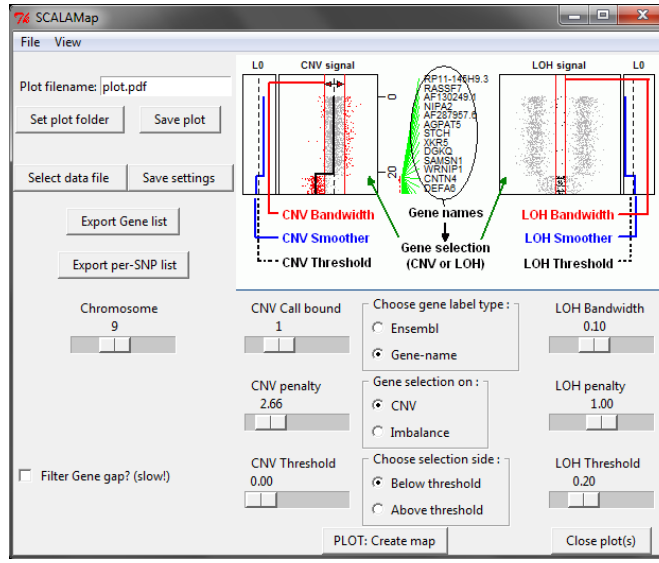


Figure 7.1: Graphical User Interface for SCALA.map.

Copy number mapping: SCALA.map

The mapping function set provides a novel way to combine visualization as well as analysis of both CNV and allelic imbalanceat the same time. The program is controlled solely by a graphical user interface (Figure 7.1) based on RPanel (Bowman et al, 2007). The GUI provides easy access to like chromosome selection, threshold signal filtering, the amount of L_0 smoothing, the type of gene labeling (e.g. taken from Ensembl or BiomaRt) to be used and some L_0 detection bandwidths and thresholds. These options can be saved to a(n `.Rdata`) file. It also provides the option to save the "tuned" image to PDF with a filename chosen by the user. Furthermore, it exports signals and detection results for all SNPs and genes located on the detected chromosome to a *comma-separated* table.

Input and output format

The mapping program takes a SCALA object as input. The main components of the object are shown below:

```
 $ meta :List of 6
  ..$ fname      : chr "name-of-CELfile"
  ..$ readpath   : chr "path-to-readfiles"
  ..$ savepath   : chr "path-to-savefiles"
  ..$ convertDate: chr "yyyy-mm-dd hh:mm:ss"
  ..$ calibrated : logi FALSE
  ..$ callDate   : logi NA
$ chr  : chr [1:numberofSNPs] "1" "1" "2" "3" ...
$ pos  : int [1:numberofSNPs] 101 102 103 104 ...
$ rsid : chr [1:numberofSNPs] "rsid1" "rsid2" "rsid3" ...
$ X    : int [1:numberofSNPs] val1 val2 val3 val4 ...
$ Y    : int [1:numberofSNPs] val1 val2 val3 val4 ...
$ calls: logi [1:numberofSNPs] NA NA NA NA NA NA NA ...
 - attr(*, "class")= chr "SCALA"
```

Settings are stored in an .controls object can be saved to an .Rdata file. These settings can be loaded upon GUI startup to recreate exactly the same plot as before, by using

```
> SCALA.map(controls = [settings_file])
```

The exported results contain gene-name, chromosome, start and stop location, a detection indicator (indicated by 0 or 1) that shows whether or not the gene was detected (by either the CNV or imbalance smoother), the accompanying L_0 values for the CNV and imbalance signal for each SNP as well as the smoothed CNV and imbalance values for each gene. Filenames for exported results as well as for the created PDF plot are user-controlled. A resulting plot window is shown in the example in section 7.3.

Graphical representation

At GUI startup, the plot window is not created immediately; the GUI starts with either the default settings or previous settings as specified by the user. The `PLOT: Create map` button is used to create and update the plot window after changing settings. The title contains the name of the selected sample and current chromosome by default, but it is highly adjustable via a separate window that is called from the menu. An partial plot example is included in the GUI, in which the main controls and their effect are illustrated. From left to right, the first panel shows the L_0 values for the CNV signal based on the data shown in the adjacent panel. In the CNV signal panel, the full CNV signal is given along with the L_0 smoother results. In the middle panel, selected gene names are shown for the chromosomal region(s) that show(s) abnormalities. The names can be shown for either the CNV or imbalance signal. To the right of the column with gene names, the imbalance signal is given (with an L_0 detection band), followed by the accompanying smoother based on the selected data points. Regions of aberrations are detected relative to a threshold value (dotted line) that is set by the user, as well as the level of smoothing and penalty norm power (default: $p = 0$) in the L_0 computations.

Estimating calibration parameters: SCALA.calibrate

The sets of calibration parameters based on a chosen number of arrays (located in the current working folder) are obtained (after file conversion) using

```
> params = SCALA.global(filefolder=getwd(), savefolder=getwd(),
                        filename=[nameofsavefile], kappa = 1e-8))
```

with `kappa` a small additive term to avoid singularity. The α (and β) vectors can be used to calibrate the original fluorescence signals.

7.3 Illustrative examples

In this example we use 8 high quality reference arrays from Affymetrix and one brain tumor file from the Erasmus Medical Center, Rotterdam, The

Netherlands (Bralten et al., 2010). The first are used to obtain calibration parameters, to be applied to the second. We start by setting the R working directory: `> setwd("D:/SCALASuite")`.

Obtaining calibration parameters

Place the Affymetrix 250k NSP control files in the folder `"D:/SCALASuite/01 raw/"` and create the folder `"D:/SCALASuite/02 raw/"`. Next we convert all CEL files in the first folder.

File conversion

We specify the chip type, read and save folder, as well as that *no* calibration should be applied; at this stage, calibration parameters are not yet available.

```
> source("SCALA.convert.r")
> SCALA.convert(datatype = 'Affy250kNSP', calibrate = F,
                readfolder = paste(getwd(),'/01 raw',sep = ""),
                savefolder = paste(getwd(),'/02 arrays',sep = ""))
Converting ctr aff 1.CEL
⋮
Converting ctr aff 8.CEL
```

Now that these files are converted and in stored R format, the next step is to call the genotypes for each array. After genotyping, it is possible to estimate the calibration parameters.

Genotype calling

A list of all converted files is obtained from the save directory defined above. Next the genotyping function is invoked. The semi-parametric estimation procedure is used, mixture plots for each chip are not requested, and the histogram is built from 100 by 100 bins.

```
> source("SCALA.genotype.r")
> fnames = list.files(path=paste(getwd(),"02 arrays",sep=""),
                      full.names=T)
> for (i in 1:length(fnames)) {
      load(fnames[i])
      scala = SCALA.genotype(scala=scala, model="s", plot=F, save=T,
                             xbins = 100, ybins = 100, lambda = 10,
                             nit=50, crit=1e-4, savefolder =
                             paste(getwd(),"/02 arrays",sep=""))
}
Calling ctr aff 1.CEL
⋮
Calling ctr aff 8.CEL
```

The (partial) result for the first of the 8 Affymetrix arrays shows that indeed the genotypes (and their cluster probabilities) have been added to the object and file, as well as the genotyping date.

```
> str(scala)
List of 8
 $ meta :List of 7
  ..$ fname      : chr "ctr aff 1.CEL"
  ..
  ..$ callDate   : chr "2011-09-12 22:16:46"
..
 $ calls: num [1:262264] 3 2 3 1 2 2 3 3 3 1 ...
 $ W    : num [1:262264, 1:3] 1.96e-10 2.15e-04 2.57e-08 1.00 ...
 - attr(*, "class")= chr "SCALA"
```

Calibration function

Now that genotypes are available, we estimate the calibration parameters for this chip type by

```
> source("SCALA.calibrate.r")
```

```
> params = SCALA.calibrate(
                              filefolder=paste(getwd(),"/02 arrays",sep=""),
                              savefolder=getwd(),
                              filename="scala.global.Rdata",
                              kappa=1e-8)
```

The generic result (object) has the following structure:

```
> str(params)

List of 7
 $ celfiles: chr [1:numberofFiles] "name-of-CELfile" ...
 $ alphaX  : num [1:numberofSNPs] num1 num2 num3 ...
 $ alphaY  : num [1:numberofSNPs] num1 num2 num3 ...
 $ betaX   : num [1:numberofFiles] num1 num2 num3 ...
 $ betaY   : num [1:numberofFiles] num1 num2 num3 ...
 $ gammaX  : num [1:3] num1 num2 num3
 $ gammaY  : num [1:3] num1 num2 num3
```

We then extract and store the calibration parameters for later use:

```
> alphaX = params$alphaX; alphaY = params$alphaY
> save(alphaX, alphaY, file = paste(getwd(),"/Calibration/",
                                    "Affy250kNSP.Rdata",sep="")
```

Copy numbers in a new glioblastoma array

The vectors just saved will be applied to a tumor tissue chip.

```
> SCALA.convert(datatype = 'Affy250kNSP', calibrate = T,
                readfolder = paste(getwd(),'/01 raw',sep = ""),
                savefolder = paste(getwd(),'/02 arrays',sep = ""))
Converting GBM 139.CEL
```

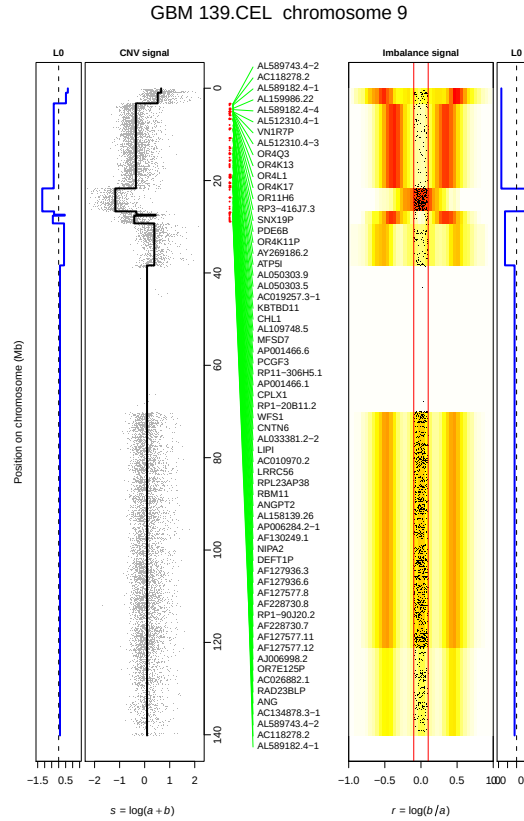


Figure 7.2: Example of a SCALA.map image. On the far left, it shows the CNV L_0 signal smoother and to its right the raw CNV signal. The middle part shows the selected gene names. The rightmost parts show the signal for allelic imbalance and its L_0 signal smoother. Here, SCALA.Map is set here to detect aberrated regions on the CNV signal, identifying one problematic region and the genes it contains.

After file conversion, we start the GUI with predefined settings (in `SCALA.map-controlsGBM139.Rdata'`). To obtain the plot window (and implicitly perform all computations), click the appropriate button.

```
> source("SCALA.map.r")
```

```
> SCALA.map(controls='SCALA.map-controlsGBM139.Rdata')
Selections:  ..  done!
Plotting:  ..  done!
Computations:  ..  done!
```

The resulting plot for chromosome 9 is given in Figure 7.2. It shows a small CNV region that has less than 2 alleles present. 27 genes are contained in that region of chromosome 9. The white space is the centromere. A similar detection can be performed on the imbalance signal by simply changing the GUI options to this purpose. Saving the settings used to obtain the current plot can be done by clicking the "save settings" button. Exporting the detection show in the plot can be either per gene or per SNP, depending on the selected button. For button location, see Figure 7.1.

7.4 Technical model details

Semi-parametric genotyping

Let $Y = \{y_{ih}\}$ be an $n_1 \times n_2$ matrix of counts in a two-dimensional $n_1 \times n_2$ histogram. The center of bin (i, h) is given by (u_i, v_h). The expected values are modeled by sums of tensor product B-splines. Two bases are computed, B_1, with c_1 columns, based on u and B_2, with c_2 columns, based on v. The bases are combined with a $c_1 \times c_2$ matrix Θ of coefficients, and the matrix of expected values is computed as

$$M = \exp(B_1 \Theta B_2'). \tag{7.1}$$

A penalized Poisson log-likelihood is then optimized. The penalty is complex, because both rows and columns of Θ are penalized. If $||X||_F$ indicates the Frobenius norm of the matrix X, i.e. the sum of the squares of its elements, the penalty is

$$\text{Pen} = \lambda_1 ||D_1 \Theta||_F / 2 + \lambda_2 ||\Theta D_2'||_F / 2, \tag{7.2}$$

where D_1 and D_2 are matrices of the proper dimensions ($c_1 - 3 \times c_1$ and $c_2 - 3 \times c_2$) that form third differences.

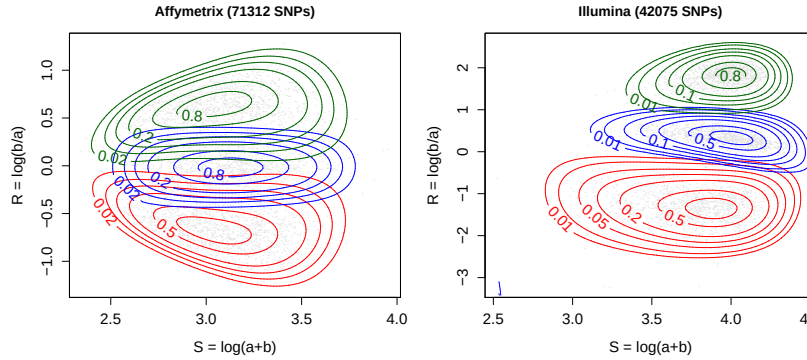


Figure 7.3: Raw data with estimated smooth densities. Left panel: a typical (symmetric) Affymetrix array. Right: a typical (asymmetric) Illumina array. Contours, normalized to 1, are overlayed for [0.02,0.05,0.1,0.2,0.5,0.8] (left) and [0.01,0.02,0.05,0.1,0.2,0.5,0.8] (right).

The mixture components give three expected values for bin (i, h) of the histogram: μ_{ih1}, μ_{ih2} and μ_{ih3}. From these numbers follow, after division by their sum, three membership probabilities. The largest of the three, which we indicate by $\hat{p}_{ih}$ points to which cluster all the observations in the bin should be assigned. The result of this algorithm is depicted in Figure 7.3, where contours of the fitted densities are shown for an Affymetrix (left) and Illumina (right) array.

In Rippe et al. (2010) genotype calls from a multi-array method (CRLMM) and from our single-array method (SCALA) are compared to a set of consensus genotypes from HapMap. The number of agreements and differences in terms of homo- and heterozygous calls showed that SCALA can be used to call genotypes efficiently and effectively. Even SNPs that were not genotyped in HapMap can be genotyped with reasonable certainty using a single chip. The above model is implemented in the SCALA.call function.

Copy number signal smoother

To obtain smooth estimates of the data, a method derived from Eilers and DeMenezes (2005) is applied. In Rippe et al. (2012b), the algorithm of Eilers and DeMenezes has been improved in at least two ways: 1) a least squares measure of fit to increase sensitivities and 2) and an L_0 norm penalty to reduce the number of jumps. The formal model is given in (7.3).

$$S_q = \sum_{i=1}^{m} (y_i - z_i)^2 + \lambda \sum_{i=2}^{m} |z_i - z_{i-1}|^q \tag{7.3}$$

with y the original data and z the smoothed signal. λ is again a tuning parameter that controls amount of smoothness. Furthermore, q can be any number between 0 and 2. Here, $q = 0$, essentially a penalty on the number of non-zero difference between neighboring elements of z, while in Eilers & DeMenezes (2005), $q = 1$.

For CNV signal smoothing all observations are used. For allelic imbalance a selection of observations within a user-defined bandwidth is used.

SNP signal intensity calibration

Consider one allele. Assuming that SNP i has a specific intensity level a_i, and that array j has a normalization factor b_j, a reasonable model for the intensity fluorescence signal is given by $x_{ij} = a_i b_j u_{ij} + \epsilon_{ij}$, with $i = 1, ..., m$ and $j = 1, ..., n$ and where u_{ij} represents the number of copies of the allele (0, 1 or 2) and ϵ_{ij} represents the error. We do not specify a distribution of ϵ. Instead we will use signals on a logarithmic scale (base 10), with $y_{ij} = \log x_{ij}$. A similar model holds for the other allele.

For the moment we assume the genotypes to be given, so we can formulate a linear model:

$$y_{ij} = \mu + \alpha_i + \beta_j + \sum_{k=1}^{3} \gamma_k h_{ijk} + e_{ij}. \tag{7.4}$$

where μ is the grand mean, α_i the level of SNP i, and β_j the level of array j; k indexes the genotype ($1 = AA$, $2 = AB$, $3 = BB$) and γ_k is a parameter for

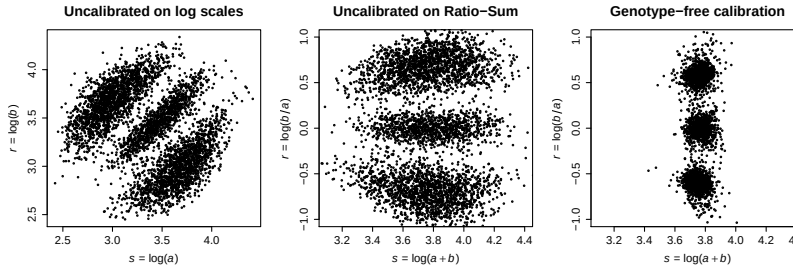


Figure 7.4: Calibration results for chromosome 1 in array NA06985 from the HapMap CEU population. Left: signals before calibration. Right: signals after genotype-free calibration. In the right panel, note the change in x-axis range, after calibration.

genotype k. The genotypes are coded with the indicator array $H = \{h_{ijk}\}$; for each combination of i and j we have a 1 in the array cell that corresponds to the genotype k, and 0 in the other cells. To make the model identifiable we introduce the constraints $\sum_i \alpha_i = 0$ and $\sum_j \beta_j = 0$. We call this the global model, since all SNPs share the same genotype parameters γ.

Further theoretical and technical details on the implementation of model (7.4) are discussed in Rippe et al., (2012a).

The α parameters that represent the levels of the SNPs are used to calibrate the intensities on a new array, by computing $x^*_{ij} = x_{ij}/10^{\alpha}_i$. This is done for each allele separately. Note that genotypes are not used, hence we call it 'genotype-free calibration'.

Figure 7.4 illustrates the results of calibration for chromosome 1 in an Affymetrix 100k Hind sample (NA06985) from the complete CEU population, retrieved from the HapMap database (HapMap Consortium, 2007). We show the signal combination $\log(a + b)$ against $\log(b/a)$, for all SNPs in a single chip. It is clear that a strong reduction of the 'noise' is obtained.

7.5 Discussion

We have described a set of programs that perform signal calibration, genotyping and copy number mapping for individual SNP chips. In situations with novel species and chips (Wright et al., 2010) this approach can be very useful: the first available chip can be genotyped or inspected for copy numbers immediately.

When genotyping, the software uses a sum-and-ratio transformation ($x = \log(a+b)$ and $y = \log(b/a)$) of the raw signals, before computing the smoothed 2D histogram. These transformations are hard coded. It can however be argued that other transformations can or should be used. With respect to the horizontal axis, $x = \log(a \times b)$ can be used, for example. We state that this alternative doesn't influence the genotype calls, since it only stretches the observations in horizontal direction. Furthermore, this algorithm calls genotypes for a whole chip (all chromosomes) at once. A discussion of genotyping individual chromosomes versus the whole genome is presented in Rippe, Eilers & Meulman (2010).

SCALA.Map provides a method to map SNPs on their chromosomal position both visually and statistically, using signals that indicate CNV and allelic imbalance. It integrates visual analysis with L_0 signal smoothers, which are all user-controlled and very easy to use. Quantification of the CNV or allelic imbalance regions is implicitly done via the L_0 norm. Furthermore, the software has several customization and export options. Intended additions and extensions are probability-based signal thresholding (i.e. to remove 'rubbish' signal for low quality samples by taking only signals above a user-defined threshold value into account). Cross-validation to determine the optimal amount of smoothing as well as a segmented scatterplot (both described in Rippe et al., 2012b)). are candidates for implementation. Furthermore, in the same paper a method for segmented imbalance estimation using segment-wise mixtures is proposed. We feel that this idea still deserves further attention before final implementation.

Discussion

8

This chapter provides a short review of the preceding chapters and restates why single array genotyping should be applied more widely. Furthermore, it addresses some open problems, illustrates new ideas in extension of the current chapters and points out benefits of the proposed methods.

8.1 Advantages of single array analysis

One of the main themes in this thesis was the propagation to switch to single array genotyping, as opposed to single SNP, multi-array genotyping, which is the current common practice. The approach has a number of advantages, which are summarized below.

Single array genotyping is fast and flexible, due to its semi-parametric approach. It is insensitive to differences in sample size, and depends only on user-chosen dimensions of the underlying histogram. The process is very easy to monitor, since it requires tracking only one sample at a time.

Along the same lines, it also allows for better quality control, because the overall level of the signals is an indication of data quality. Because quality control is easy, the procedure is also highly suitable for use in development of small series of chips, for example when devising new layout to research "new" organisms. Furthermore, the procedure is readily available in open source software, in the SCALA software suite.

8.2 A short review

The theme of this thesis can be summarized in a few words: "better data analysis for SNP arrays". The five main chapters present efficient and effective solutions to many problems that are encountered in practice. They are reviewed concisely.

SNP platforms provide a variety of opportunities as well as challenges. Fluorescence signals from these platforms have structural properties: overall fluorescence levels differ not just between arrays, but also between SNPs within one array. The SCALA model, discussed in Chapter 2, contains parameters for estimating the systematic effects of SNPs, arrays and genotypes. This large regression model is applied to both alleles separately, and delivers a million parameters or more. However, due to its extremely sparse structure, a specialized semi-symbolic algorithm allows exact estimation in a very short time. Model fit is highly adequate in terms of (standard deviation) of residuals. Once the parameters of the model have been estimated, they are used to eliminate the systematic effects, thereby greatly enhancing the quality of the fluorescence signals. We call this calibration and apply it in a later chapter.

In Chapter 5 it is shown that the signal calibration is also useful for correction of genomic waves, visible as a systematic pattern when plotting fluorescence signals along chromosomes and smoothed. Calibration removes these waves. Because the model used to obtain calibration parameters does not model spatial autocorrelation, the results of calibration imply that wave patterns in reality are not caused by not spatial autocorrelation. Furthermore, noise in the signals is reduced. When compared to a dedicated wave correction model, NoWaves, performance is equal, but the proposed calibration is more efficient. NoWaves requires reference samples for each array subject to correction, while SCALA applies calibration parameters that were estimated at some prior point in time.

One application of SNP fluorescence signals is to determine SNP genotypes. In Chapter 3 we break with common practice and perform genotyping for all SNPs on individual arrays. A semi-parametric mixture model is

estimated, with three component densities, one for each of the AA, AB and BB genotypes. Comparison to results of SNP by SNP algorithms (CRLMM) as well as a de-facto standard, as found on the HapMap archives, show equal or better performance. Furthermore, where traditional methods do not provide reliable estimates for all scenarios, i.e. low probabilities, for low Minor Allele Frequencies (MAF) due to small or missing components, the estimates from the single array model have higher probabilities and additionally provide genotypes for SNPs that were not called by HapMap. The current model is suitable for different platforms as well as chips with different densities.

Throughout the chapters, genotyping is based on a display of the ratio of the A and B signals versus their sum (on logarithmic scales). Low signals on the sum scale, as well as unclear separation between the three genotype groups on the ratio scale indicate low(er) chip quality. Applying calibration before single array genotyping, as described in Chapter 4, allows us to exploit this knowledge to select only the SNP observations of the highest quality, by a user-defined threshold. This results in higher genotyping probabilities for the selected high-quality observations on low(er) quality arrays.

Another application of the fluorescence signals is the estimation of profiles of copy number changes. These changes generally occur in a segment-wise manner along chromosomes. There is a large literature on smoothing and segmentation of CNV signals, all with the goal to obtain the boundaries of the segments and their levels. A new smoothing algorithm was presented in Chapter 6. The model uses a so-called L_0 penalty on jumps between smoothed values and is therefore referred to as the Zero Exponent Norm, ZEN. The result is an extremely sharp segmentation. A similar segmentation also holds for allelic imbalance signals. However, it is not possible to apply the same smoother to allelic imbalance signals, because several parallel data bands occur. Therefore, we modified an existing scatterplot smoother to use the L_0 penalty in one direction and the L_2 norm in the other, in order to get sharp segmentation here too.

All models and algorithm are written in R, and are combined in a software suite, The SCALA suite (Chapter 7). It provides both command-line functions (for estimation and calibration, as well as genotpying) and a graphical user interface for interactive (simultaneous) smoothing and plotting of CNV and

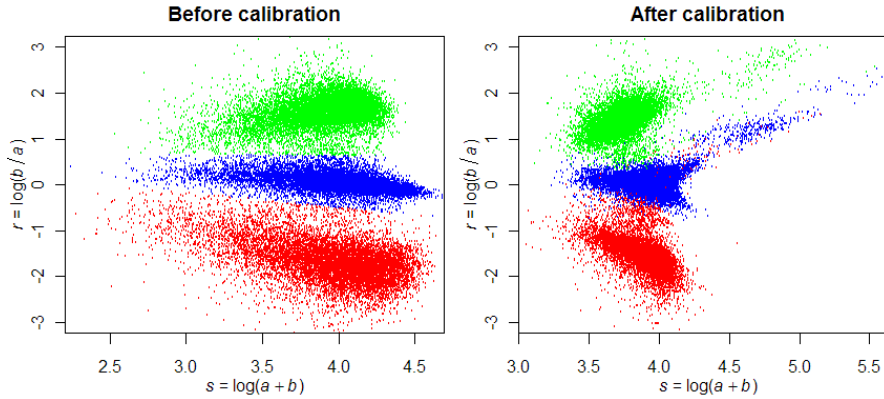


Figure 8.1: An Illumina array with asymmetric signals before and after calibration. Clusters are not condensed, but tails appear.

allelic imbalance.

8.3 Ideas for future research

Although the previous chapters have addressed specific questions and problems, there are still open questions and new directions to be explored. Below, a few are discussed.

Calibration of asymmetric fluorescences signals

In Chapter 3, a method was developed for single array signal calibration. This method was tested extensively for Affymetrix arrays, which have strong symmetric properties when looking at a single array. However, it was also mentioned that e.g. the asymmetric signals from the Illumina *Infinium* platform was not evaluated due to problems with signal calibration. Unfortunately, no explanation was provided as for why the calibration doesn't work, except for the fact that Illumina (but others too) uses two-color fluorescence, where Affymetrix uses just one. It seems that the resulting wavelength differences are at the heart of the asymmetry. In the near future we aim to provide more

insight into the positive and negative aspects of asymmetric signals and propose a solution for the less desirable ones. After calibration, the clusters are not condensed like for Affymetrix, but seem to obtain a swallow-like shape (Figure 8.1). The tails appear after calibration and compromise quality of the genotype calls.

Staaf et al. (2008) used quantile normalization using reference arrays to overcome the asymmetry. However, in practice their approach is not effective in a single ratio-sum transformation since they use a set of arrays to find a symmetric transformation within the given set (Bolstad, Irizarry & Speed, 2003). Still, a part of the solution for asymmetry in fluorescence signals before calibration may be found here.

Extended models

A possible model extension is to perform simultaneously modeling of genotypes, copy number profiles and calibration parameters. An example in which independent estimations for genotypes and CNV have been combined in a single representation is shown for chromosome 9 in Figure 8.2. We refer to the model as the Michelin model, because this representation of the data has similarities to the profile on a (car) tire. However, calling all genotypes at once for such a sample will induce errors. For better clarification, the complete chromosome is split into the tumorous P-arm and healthy Q-arm in Figure 8.3. The top panel shows the healthy tissue with constant CNV and full allelic balance, and has clear genotypes. The three separate views are shown in the left panels in Figure 8.4. The bottom panel however shows the tumor tissue, showing CNV and allelic imbalance. These are shown in the right panels in Figure 8.4. Genotyping this arm at once will be largely incorrect, because one number of genotype clusters is estimated, while a different number of clusters for each segment in this arm would be more appropriate.

The core principle behind this idea is that it is possible to have a different genotype component mixture for each copy number or allelic imbalance segment. Mixtures of 1, 2 or 3 components can occur in different segments. It then is possible to fit a log-concave component mixture per segment, as a fundamental approach to “interactions” between CNV and genotypes. An

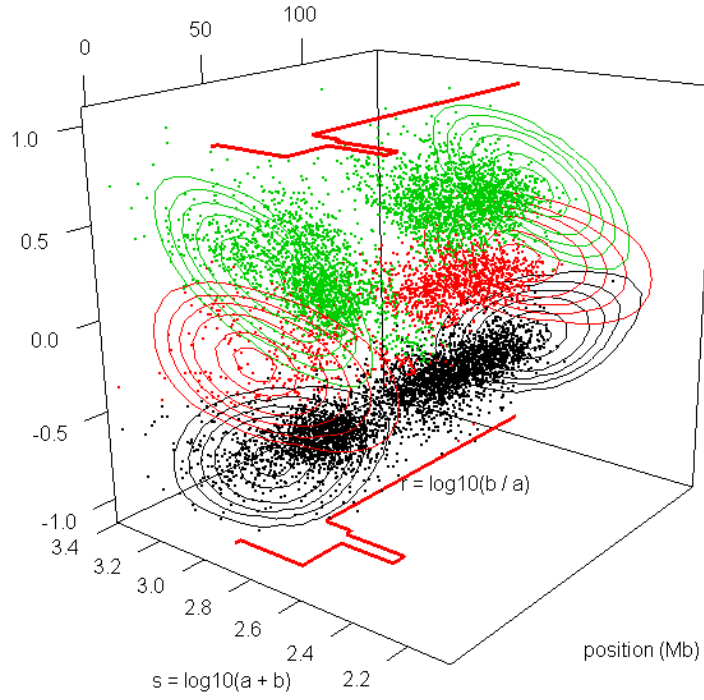


Figure 8.2: Three models combined in one SNP signal representation: 1) CNV profiles (top view), 2) Allelic imbalance (right side view) and 3) Genotyping (front side view).

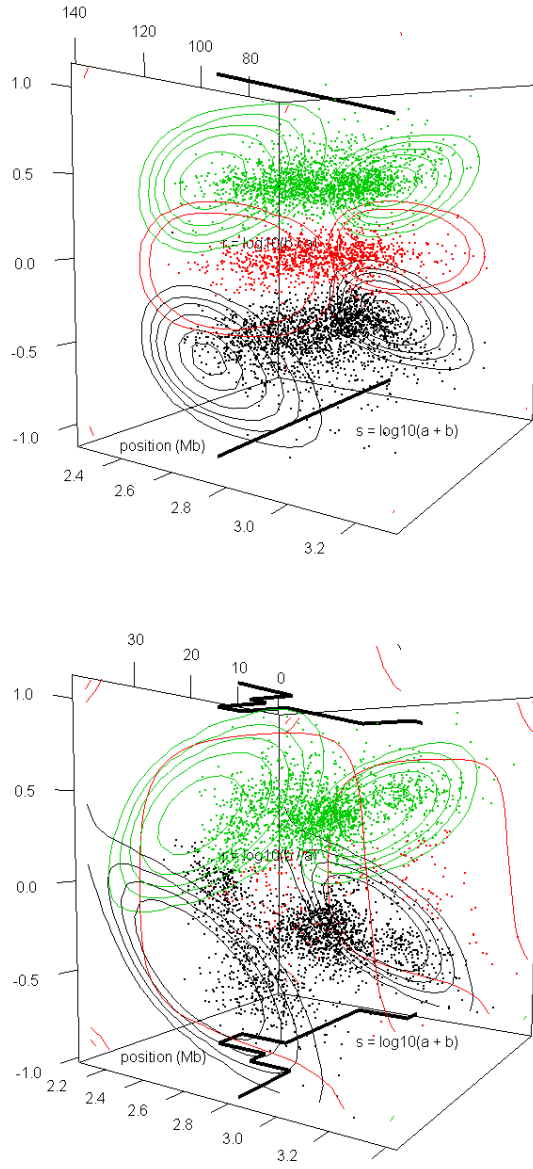


Figure 8.3: Combined SNP model for normal (top) and diseased (bottom) tissue.

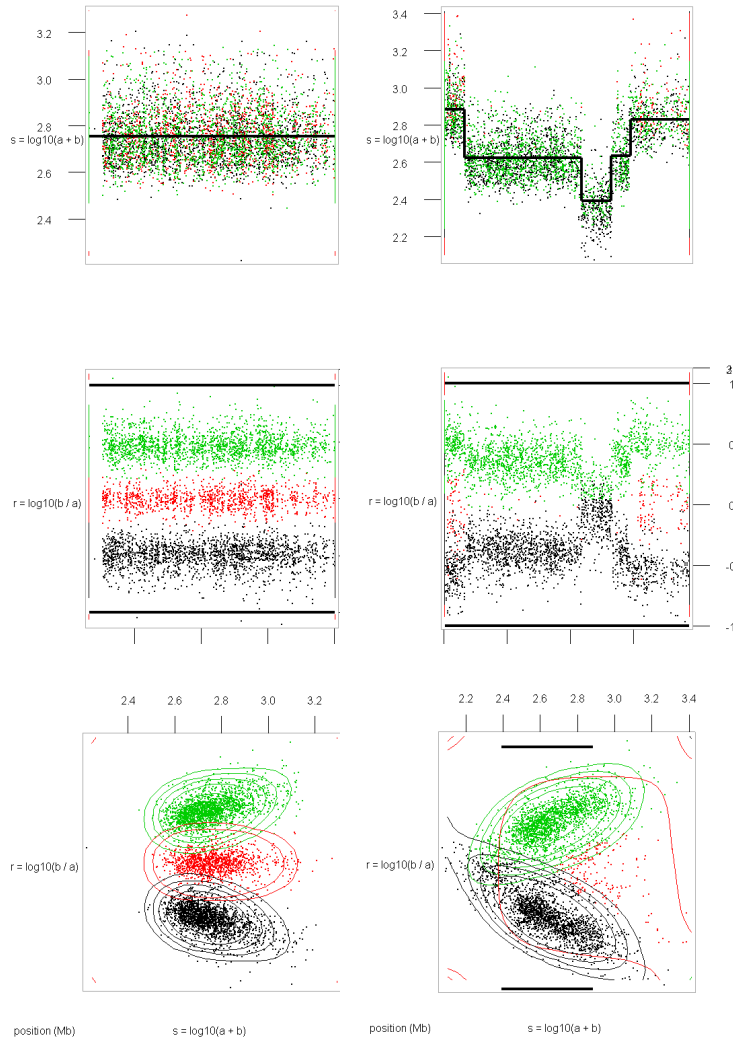


Figure 8.4: Combined SNP model in single view orientations. Left column shows healthy tissue; right column shows tumor tissue. The top panel shows a CNV profile, the middel panel shows allelic imbalance, and the bottom panel shows genotypes.

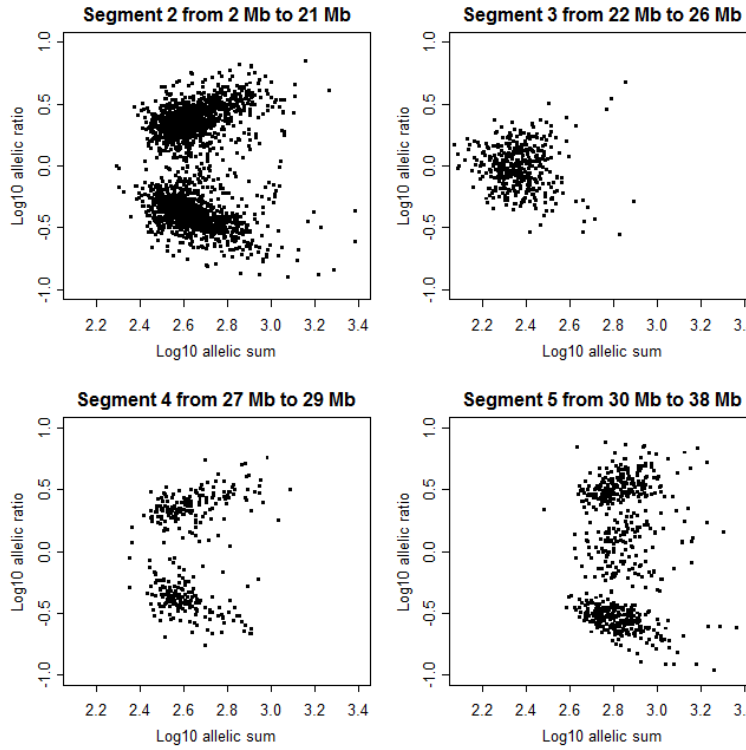


Figure 8.5: Genotype representation of the data within segments. Mixtures of one (top right), two (top left and bottom left) and three (bottom right) components can be distinguished.

illustration is given in Figure 8.5. This approach will provide some mathematical challenges in terms of overfitting or overparametrization.

Figure 8.5 also indicates why it is better to use both the ratio and the sum dimension for genotyping, instead of just the ratio, because the latter would provide genotype densities that are too wide. The bottom right panel provides a clear demonstration. Using the sum dimension in addition allows for more accurate estimations.

Bibliography

Adorjan, P., Distler, J., Lipscher, E., Muller, F., Muller, J., Pelet, C., et al. (2002). Tumour class prediction and discovery by microarray-based dna methylation analysis. *Nucleic Acids Research*, **30**, e21.

Affymetrix (2006). BRLMM: An improved genotype calling method for the genechip human mapping 500k array set. *Technical report*, Affymetrix.

Altshuler, D., Brooks, L. D., Chakravarti, A., Collins, F. S., Daly, M. J., Donnelly, P., et al. (2005). A haplotype map of the human genome. *Nature*, **437**, 1299–1320.

Bengtsson, H., Irizarry, R., Carvalho, B. & Speed, T.P. (2008). Estimation and assessment of raw copy numbers at the single locus level. *Bioinformatics*, **24(6)**, 759–767.

Beroukhim, R., Lin, M., Park, Y., Hao, K., Zhao, X., Garraway, L.A., et al (2006). Inferring loss-of-heterozygosity from unpaired tumors using high-density oligonucleotide SNP arrays. *PLoS Comput. Biol.*, **2(5)**, e41.

Bibikova, M., Lin, Z., Zhou, L., Chudin, E., Wickham-Garcia, E., Wu, B., et al. (2006). High-throughput dna methylation profiling using universal bead arrays. *Genome Research*, **16 (3)**, 383–393.

Bowman, A.W., Crawford, E., Alexander, G., & Bowman, R.W. (2008). rpanel: Simple Interactive Controls for R Functions Using the tcltk package. *Journal of Statistical Software*, **17 (9)**, 1–18.

Bralten L.B., Kloosterhof, N.K., Gravendeel, L.A., Sacchetti, A. et al. (2010). Integrated genomic profiling identifies candidate genes implicated in glioma-genesis and a novel LEO1-SLC12A1 fusion gene. *Genes, Chromosomes and Cancer*, **49**, 509–517.

Brown, T.A. (1999). *Genomes*. Oxford, UK: BIOS Scientific Publishers Ltd.

Budinska, E., Gelnarova, E. & Schimek, M.G. (2009). MSMAD: a computionally efficient method for the analysis of noisy array CGH data. *Bioinformatics*, **25 (6)**, 703–713.

Cardoso et al. (2004). Genomic profiling by DNA amplification of laser capture microdissected tissues and array CGH. *Nucl. Ac. Res.*, **19**, 146.

Carvalho, B., Bengtsson, H., Speed, T.P. & Irizarry, R.A. (2007). Exploration, normalization, and genotype calls of high-density oligonucleotide SNP array data. *Biostatistics*, **8 (2)**, 485–499.

Carvalho, B.S., Louis, T.A. & Irizarry, R.A. (2010). Quantifying uncertainty in genotype calls. *Bioinformatics*, **26 (2)**, 242–249.

Chin, S.F., Wang. Y.Y., Thorne, N.P., Teschendorff, A.E., Pinder, S.E., Vias, M., et al. (2007). Using array-comparative genomic hybridization to define molecular portraits of primary breast cancers. *Oncogene*, **26**, 1959–1970.

Colella, S., Yau, C., Taylor, J.M., Mirza, G., Butler, H., et al. (2007). QuantiSNP: an Objective Bayes Hidden-Markov Model to detect and accurately map copy number variation using SNP genotyping data. *Nucleic Acids Res.*, **35 (6)**, 2013–2025.

Conrad, D.F., Andrews, T.D., Carter, N.P., Hurles, M.E., & Pritchard, J.K. (2006). A high-resolution survey of deletion polymorphism in the human genome. *Nat. Genet.*, **38**, 75–81.

Currie, I.D., Durban, M. & Eilers, P.H.C. (2006). Generalized linear array models with applications to multidimensional smoothing. *J. R. Statist. Soc. B* **68**, 259–280.

Diskin, S.J., Li, M., Hou, C., Yang, S., Joseph Glessner, J., Hakonarson, H., et al (2008). Adjustment of genomic waves in signal intensities from whole-genome SNP genotyping platforms. *Nucl. Ac. Res.*, **36 (19)**, e126.

Dunbar, A.J., Gondek, L.P., O'Keefe, C.L., Makishima, H., Rataul, M.S., Szpurka, H., et al.(2008). 250K Single Nucleotide Polymorphism Array Karyotyping Identifies Acquired Uniparental Disomy and Homozygous Mutations, Including Novel Missense Substitutions of c-Cbl, in Myeloid Malignancies. *Cancer Research*, **68**, 10349.

Eilers, P.H.C. (2003). A perfect smoother. *Anal Chem*, **75(14)**, 3631–3636.

Eilers, P.H.C. & DeMenezes, R. (2005). Quantile smoothing of array CGH data. *Bioinformatics*, **21 (7)**, 1146–1153.

Eilers, P.H.C. & Borgdorff, M.W. (2007). Non-parametric log-concave mixtures. *Computational Statistics & Data Analysis*. **51**, 5444–5451.

Eilers, P.H.C. & Goeman, J. (2004). Enhancing scatterplots with smoothed densities. *Bioinformatics*, **20**, 623–628.

Eilers, P.H.C. & Marx, B.D. (1996). Flexible smoothing using B-splines and penalized likelihood (with Comments and Rejoinder). *Statistical Science*, **11 (2)**, 89–121.

Eilers, P.H.C. & Marx, B.D. (2007). Multidimensional Density Smoothing with P-splines. *Proceedings of the 23rd International Workshop on Statistical Modelling*.

Fan, J. B., Gunderson, K., Bibikova, M., Yeakley, J. M., Chen, J., Wickham-Garcia, E., et al. (2006). Illumina universal bead arrays. *Methods in Enzymology*, *410*, 57–73.

Fraley, C. and Raftery, A.E. (2004). Model-based Methods of Classiffication: Using the mclust Software in Chemometrics. *Journal of Statistical Software*, **18 (6)**.

Franke, L., Kovel, C.G. de, Aulchenko, Y.S., Trynka, G., Zhernakova, A., Hunt, et al. (2008). Detection, Imputation, and Association Analysis of Small Deletions and Null Alleles on Oligonucleotide Arrays. *The American Journal of Human Genetics*, **82**, 1316-1333.

Giannoulatou, E., Yau, C., Colella, S., Ragoussis, J., & Holmes, C. (2008). GenoSNP: a variational bayes within-sample snp genotyping algorithm that does not require a reference population. *Bioinformatics*, **24 (19)**, 2209–2214.

Gravendeel, L.A., Kouwenhoven, M.C., Gevaert, O., de Rooi, J.J., Stubbs, A.P., Duijm, J.E., et al. (2009). Intrinsic gene expression profiles of gliomas are a better predictor of survival than histology. *Cancer Res.*, **69 (23)**, 9065–9072.

Griffiths, A.J.F, Gelbart, W.M., Lewontin, R.C. & Miller, J.H. (2002). *Modern Genetic Analysis: Integrating Genes and Genomes. Second Edition.* New York, USE: W.H. Freeman and Company.

The International HapMap Consortium (2003). The International HapMap project. *Nature*, **426**, 789–796.

The International HapMap Consortium (2007). A second generation human haplotype map of over 3.1 million SNPs. *Nature*, **449**, 851–861.

Hüebner, C., Petermann, I., Browning, B.L., Shelling, A.N. & Ferguson, L.R. (2007). Triallelic single nucleotide polymorphisms and genotyping error in genetic epidemiology studies: MDR1 (ABCB1) G2677/T/A as an example. *Cancer Epidemiol Biomarkers Prev.*, **16(6)**, 1185–1192.

Komura, D., Shen, F., Ishikawa, S., Fitch, K.R., Chen, W., et al. (2005). Genome-wide detection of human copy number variations using high-density DNA oligonucleotide arrays. *Genome Res.*, **16**, 1575–1584.

Korn, J.M., Kuruvilla, F.G., McCarroll, S.A., Wysoker, A., et al. (2008). Integrated genotype calling and association analysis of SNPs, common copy number polymorphisms and rare CNVs. *Nature Genetics*, **40**, 1253–1260.

Kroonenberg, P. (2008). *Applied multiway data analysis*. Hoboken, NJ: Wiley.

Lai, W.R., Johnson, M.D., Kucherlapati, R. & Park, P.J. (2005). Comparative analysis of algorithms for identifying amplifications and deletions in array CGH data. *Bioinformatics*, **21 (19)**, 3763–3770.

Leprêtre, F., Villenet, C., Quief, S., Nibourel, O., Jacquemin, C., Troussard, X. (2008). Waved aCGH: to smooth or not to smooth. *Nucl. Ac. Res.*, **38 (7)**, e94.

Lips, E., Dierssen, J., Eijk, R. van, Oosting, J., Eilers, P., Tollenaar, R., et al. (2005). Reliable high-throughput genotyping and loss-of-heterozygosity detection in formalin-fixed parafin-embedded tumors using single nucleotide polymorphism arrays. *Cancer Research*, *65*, 10188–10191.

Liu, Z., Li, A., Schulz, V., Chen, M. & Tuck, D. (2010). MixHMM: Inferring Copy Number Variation and Allelic Imbalance Using SNP Arrays and Tumor Samples Mixed with Stromal Cells. *PLoS ONE*, **5 (6)**: e10909.

Marenne, G., Rodríguez-Santiago, B., Closas, M.G., Pérez-Jurado, L., Rothman, N., et al. (2011). Assessment of copy number variation using the Illumina Infinium 1M SNP-array: a comparison of methodological approaches in the Spanish Bladder Cancer/EPICURO study. *Hum. Mutat.*, **32 (2)**, 240–248.

Marioni, J.C., Thorne, N.P., Valsesia, A., Fitzgerald, T., Redon, R., Fiegler, H., et al. (2007). Breaking the waves: improved detection of copy number variation from microarray-based comparative genomic hybridization. *Genome Biol.*, **8**, R228.

Mao, X., Young, B.D. & Lu, Y.J. (2007). The application of single nucleotide polymorphism microarrays in cancer research. *Curr. Genomics*, **8 (4)**, 219–228.

McCarroll, S.A., Hadnott, T.N., Perry, G.H., Sabeti, P.C., Zody, M.C., Barrett, J.C., et al. (2006). Common deletion polymorphisms in the human genome. *Nat. Genet.*, **38**, 86–92.

McCarroll S.A., Kuruvilla, F.G., Korn, J.M., Cawley, S., Nemesh, J., et al. (2008). Integrated detection and population-genetic analysis of SNPs and copy number variation. *Nature Genetics*, **40 (10)**, 1166–1174.

Mead, S., Poulter, M., Beck, J., Uphill, J., Jones, C., Eng Ang, C., Mein, C.A. & Collinge, J. (2008). Successful amplification of degraded DNA for use with high-throughput SNP genotyping platforms. *Human Mutation*, **29 (12)**, 1452–1458.

Morganella, S. Cerulo, L. Viglietto, G. & Ceccarelli, M. (2010). VEGA: Variational segmentation for copy number detection. *Bioinformatics*, **21 (7)**, 1146-1153.

Muggeo, V.M.R. & Adelfio, G. (2011). Efficient change point detection for genomic sequences of continuous measurements. *Bioinformatics*, **27 (2)**, 161–166.

Mukherjee, S. (2010). The emperor of all maladies: a biography of cancer. *Tantor Meida*.

Nannya, Y., Sanada, M., Nakazaki, K., Hosoya, N., Wang, L., et al. (2005). A robust algorithm for copy number detection using high-density oligonucleotide single nucleotide polymorphism genotyping arrays. *Cancer Res.*, **65**, 6071–6079.

Nicolae, D.L., Wen, X., Voight, B.F & Cox, N.J. (2006) Coverage and Characteristics of the Affymetrix GeneChip Human Mapping 100K SNP Set. *PLoS Genetics*, **2 (5)**.

Nishida, N., Koike, A., Tajima, A., Ogasawara, Y., Ishibashi, Y., et al. (2008). Evaluating the performance of Affymetrix SNP Array 6.0 platform with 400 Japanese individuals. *BMC Genomics*, **9 (1)**, 431.

R Development Core Team (2012). R: A language and environment for statistical computing. *R Foundation for Statistical Computing*, Vienna, Austria, **ISBN 3-900051-07-0**, URL http://www.R-project.org.

Rabbee, N. & Speed, T.P. (2006). A genotype calling algorithm for affymetrix SNP arrays. *Bioinformatics*, **22 (1)**, 7–12.

Reiter, L.T., Potocki, L., Chien, S., Gribskov, M. & Bier, E. (2001). A systematic analysis of human disease-associated gene sequences in Drosophila melanogaster. *Genome Research*, **11 (6)**, 1114–1125.

Rippe, R.C.A., Meulman, J.J. & Eilers, P.H.C. (2012a). Correction of Fluorescence Bias on Affymetrix Genotyping Microarrays. *Journal of Chemometrics*, **26**: 191–196. doi: 10.1002/cem.2436.

Rippe, R.C.A., Meulman, J.J. & Eilers, P.H.C. (2012b). Visualization of Genomic Changes by Segmented Smoothing Using an L_0 Penalty. *PLoS ONE*, **7(6)**: e38230. doi:10.1371/journal.pone.0038230.

Rippe, R.C.A., Eilers, P.H.C. & Meulman, J.J (2010). Efficient Semi-Parametric Genotyping. *Proceedings of the 25th International Workshop on Statistical Modelling*.

Rocke, D.M. & Durbin, D. (2003). Approximate variance-stabilizing transformations for gene-expression microarray data. *Bioinformatics*, **19**, 966–972.

Schlossmacher, E.J. (1973). An iterative technique for absolute deviations curve fitting. *Journal of the American Statistical Association*, **68(344)**, 857–859.

Sebat, J., Lakshmi, B., Troge, J., Alexander, J., Young, J., Lundin, P., et al. (2004). Large-scale copy number polymorphism in the human genome. *Science*, **305**, 525–528.

Staaf, J., Vallon-Christersson, J., Lindgren, D., Juliusson, G., Rosenquist, R., Höglund, M., Borg, A., & Ringner, M. (2008). Normalization of Illumina Infinium whole-genome SNP data improves copy number estimates and allelic intensity ratios. *BMC Bioinformatics*, **9:409**.

Smilde, A., Geladi, P., & Bro, R. (2004). *Multi-way analysis in chemistry*. Chichester, UK: Wiley.

Taylor, B.S., Barretina, J., Socci, N.D., DeCarolis, P., Ladanyi, M., et al. (2008). Functional Copy-Number Alterations in Cancer. *PLoS ONE*, **3(9)**, e3179.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *J. Royal. Statist. Soc B.*, **58 (1)**, 267–288.

Tikhomirov, A., Konkashbaev, A. & Nicolae, D.L., (2008). On Single-Array Genotype Calling Algorithms, *2008 International Conference on BioMedical Engineering and Informatics*, **vol. 1**, pp.459-462.

Tsuang, D.W., Millard, S.P., Ely, B., et al. (2010). The effect of algorithms on copy number variant detection. *PLoS One*, **5 (12)**, e14456.

Wang, K., Li, M,, Hadley, D,, Liu, R,, Glessner, J., et al. (2007). PennCNV: an integrated hidden Markov model designed for high-resolution copy number variation detection in whole-genome SNP genotyping data. *Genome Res.*, **17 (11)**, 1165–1674.

Winchester, L., Yau. C., & Ragoussis J. (2009). Comparing CNV detection methods for SNP arrays. *Brief Funct Genomic Proteomic*, **8**, 353–366.

Whittaker, E. (1923). On a new method of graduation. *Proc. Edinburgh Math. Soc.*, **41**, 63–75.

Van de Wiel, M.A., Brosens, R., Eilers, P.H.C., Kumps, C., Meijer, G.A., Menten, B., Sistermans, E., Speleman, F., Timmerman, M.E. & Ylstra, B. (2009). Smoothing waves in array CGH tumor profiles. *Bioinformatics*, **25**, 1099–1104.

Wright, M.H., Tung, C.-W., Zhao, K., Reynolds, A., McCouch, S.R. & Bustamante, C.D.] (2010). ALCHEMY: a reliable method for automated SNP genotype calling for small batch sizes and highly homozygous populations. *Bioinformatics*, **26 (23)**, 2952–2960.

Xiao, Y., Segal, M.R., Yang, Y.H. & Yeh, R.F. (2007). A multi-array multi-SNP genotyping algorithm for Affymetrix SNP microarrays. *Bioinformatics*, **23 (12)**, 1459–1467.

Zhang, D., Qian, Y., Akula, N., Alliey-Rodriguez, N., Tang, J., et al. (2011). Accuracy of CNV Detection from GWAS Data. *PLoS ONE*. **6 (1)**, e14511.

Ziegler, A. (2009). Genome-wide association studies: quality control and population-based measures. *Genetic Epidemiology*, **33**, S45–S50.

Samenvatting

Single Nucleotide Polymorphisms (SNPs) zijn variaties in het DNA, een enkele nucleotide groot. Dat is de officiële benaming wanneer een of beide allelen (één voor elk chromosoom) variëren bij meer dan 1% van de bevolking. Vaak zijn dergelijke polymorfismen onschuldig, maar in slechtere gevallen kunnen ze tumoren veroorzaken of tenminste de ontwikkeling ervan beïnvloeden. Het is om die reden dat al jaren op steeds grotere schaal onderzoek wordt gedaan naar de aanwezigheid, vorm en effecten van deze SNPs. Het is van belang te achterhalen welke allel-combinatie (het genotype) aanwezig is en of er afwijkingen zijn van het normale aantal van twee allelen. Deze afwijkingen staan bekend als copy number variaties.

Verschillende fabrikanten maken platforms die SNPs kunnen analyseren. In alle gevallen wordt hiervoor gebruik gemaakt van selectieve hybridizatie: er wordt gekeken naar specifieke posities op het genoom (en een aantal posities daaromheen vanwege stabiliteit van de reactie). Aan de moleculen worden zogeheten fluoroforen verbonden als labels om uit de resulterende fluorescenties de concentraties van de allelen A en B te bepalen. Sommige fabrikanten gebruiken één fluorofoor (bijvoorbeeld Affymetrix), andere twee (bijvoorbeeld Illumina). De mate waarin de twee fluorescentiesignalen worden geobserveerd geven de doses van de allelen A en B weer. Veel A en weinig B geeft genotype AA, omgekeerd geeft genotype BB en gelijke doses geeft AB. De totale dosis in gezond DNA bedraagt dus in principe 2.

In de praktijk blijkt dat deze signalen structurele eigenschappen bezitten. Als in één monster het signaal voor een SNP relatief sterk is ten opzichte van dat van andere SNPs in hetzelfde monster, blijkt diezelfde SNP deze eigenschap ook te hebben in andere monsters. Dit gegeven, in combinatie met verschillen tussen monsters en gegeven de verschillende signaalsterktes voor de alleldoses van elke SNP, maakt de signaalvariatie op niveau van individuele SNPs modelleerbaar. Een model hiervoor, SCALA (SNP Calibra-

tion Algorithm), wordt in dit proefschrift geïntroduceerd. Het schat ondermeer het systematische effectniveau en corrigeert daarvoor: calibratie. Het SCALA model onderscheidt twee mogelijkheden: 1) het schatten van de effecten zonder dat er genotypen bekend zijn (genotype-vrije calibratie) of 2) het per genotype schatten van een set van effectparameters voor elke SNP (genotype-gebaseerde calibratie). De laatste aanpak vereist dus wel dat het genotype van elke SNP in het betreffende monster bekend is. De geschatte effecten zijn stabiel binnen één type array. Ze gelden (dus) ook voor nieuwe arrays, en dit maakt calibratie mogelijk. Het toepassen van calibratie met de geschatte effecten uit één van bovengenoemde opties verkleint de spreiding in de signalen sterk, terwijl de alleldoses voor SNPs verhoudingsgewijs gelijk blijven en tevens de regionen met afwijkingen van twee allelen beter zichtbaar worden: de calibratie is effectief. Merk op dat de bovenstaande calibratie kan worden toegepast op onafhankelijke, nieuwe, monsters, ongeacht wel of niet bekende genotypen.

Het bepalen van genotypen uit de alleldoses is niet geheel triviaal. Hierbij wordt binnen één enkel monster gekeken naar de dose-verhouding tussen alle SNPs in dit individu. Het voordeel is dat er altijd voldoende observaties zijn voor het stabiel schatten van het gekozen model. Na transformatie van de signalen voor A en B is een puntenwolk zichtbaar met daarin drie clusters, die de genotypen AA, AB en BB vertegenwoordigen. De gebruikte methode is semi-parametrisch: er worden geen specifieke aannames gedaan over de vorm van de clusters. Het principe achter het model is dat alle punten in een vlak worden omgezet naar een 2-dimensioneel histogram. Op de telwaarden wordt een logconcaaf dichtheidsmodel geschat voor elk van de drie genotype-clusters. Een groot voordeel van deze aanpak is dat het ongevoelig is voor zowel verschillen in de vorm van de clusters als voor toename van aantal metingen: het histogram houdt dezelfde afmeting. Bovendien is het algoritme zelf zeer efficiënt, waardoor het model zowel uit inhoudelijk als praktisch oogpunt interessant is.

De beschreven aanpak is de zogenaamde 'single-array' methodiek en wordt in dit proefschrift geïntroduceerd. Het contrasteert met de tot nu toe gangbare 'multi-array' methodiek. In het geval van genotypering wil dat zeggen: voor elke individuele SNP wordt in een aantal monsters (tientallen tot zelfs

duizenden) gekeken naar de verhouding van de signalen voor elk allel. Er worden dus meerdere monsters gebruikt voor onderlinge referentie. De kwaliteit en stabiliteit van het resultaat hangt af van de hoeveelheid beschikbare gegevens, ergo van het aantal monsters. In de praktijk blijkt dat het 'single array' schatten van genotypen minstens zo goed presteert als de traditionele 'multi-array' methoden, wanneer we beide resultaten vergelijken met genotypen uit HapMap. Een echte gouden standaard bestaat niet, maar HapMap is een database die voor een aantal standaardmonsters ook de genotypen geeft, die zijn afgeleid uit een aantal algoritmen en op basis van een aantal platforms. Daarnaast kunnen door het werken met enkelvoudige arrays ook een aantal SNPs geclassificeerd worden, terwijl andere methoden dat niet konden (HapMap geeft voor deze SNPs bijvoorbeeld een 'NoCall').

De eerder genoemde signaalcalibratie is ook nuttig in het geval van genotypering voor monsters met lage kwaliteit. Calibratie van de signalen leidt tot sterk verlaagde onzekerheid in het bepalen van het genotype, voor platforms van verschillende SNP-dichtheden.

In ongezond DNA kunnen variaties in kopienummer en allel-onbalans worden geobserveerd. Voor de doses van allelen A en B betekent dit dat ze ook 0 (0), 1 (A0 of 0B), 3 (AAA, ABB, etc) of meer keren voor kunnen komen. In het schatten van deze totale alleldoses wordt gebruik gemaakt van een signaalsmoother die de observaties voor één monster gladstrijkt, zonder hierbij rekening te (hoeven) houden met referentieprofielen. Dit idee werkt, omdat in het geval van doses voor meer of minder dan twee allelen het totale signaal (de som van (log) a en (log) b) ook hoger of lager is dan in het geval van normaal weefsel (twee allelen). Het resulterende 'hoogte'-profiel van het monster is om deze redenen veel gebruikt. Echter, de effectiviteit van dit gladde profiel hangt af van de gekozen smoother. Een smoother die het aantal sprongen tussen segmenten bestraft (een penalty met de L_0 norm) vindt de afwijkende regionen, maar detecteert niet de ruis tussen regionen van verschillende niveaus. Een smoother die alleen de hoogte van signaalsprongen bestraft (een L_1 penalty) detecteert deze ruis wel, met als gevolg een onnodig onrustig profiel binnen regionen.

Voor het bepalen van de absolute alleldoses kunnen profielen van referentiemonsters (met gezond DNA, dus twee allelen) worden gebruikt. Hier is

dat echter niet nodig: een vergelijking van de penalty met L_0 norm met een ander model (VEGA) laat zien dat met de eerste zeer goede resultaten kunnen worden behaald. Bovendien zijn er indicaties dat de eerdergenoemde signaalcalibratie een verbetering van segmentering in de profielen kan opleveren. Tenslotte is de L_0 norm in de penalty ook nuttig voor het maken van gladde puntenwolken: door in één richting de bestraffing te gebruiken, wordt er *per histogram-segment* een gladde benadering gevonden.

Een uitbreiding van de bovengenoemde penalty naar een model voor het schatten van profielen voor allel-onbalans ligt hiermee voor de hand. De praktijk is echter weerbarstiger: hiervoor zijn een aantal extra stappen (en de daarmee gepaarde onzekerheid) nodig. Een eerste aanzet wordt gedaan door per segment een mengsel van verdelingen te schatten.

Vanzelfsprekend ligt er naar aanleiding van bovenstaande nog veel open voor de toekomst. Uitbreidingen van de toepassing van de voorgestelde technieken kunnen liggen in (tetraploïde) gewassen zoals aardappelen, prei, rozen, en sommige vissoorten zoals zalm.

Verder is een interessante invalshoek om tegelijkertijd genotypen, copy number-profielen en calibratieparameters te schatten. Tenslotte kan mogelijk het vervangen van "harde genotypen" door de kansen daarop de effectiviteit van het calibratiemodel verder verhogen.

Summary

Single Nucleotide Polymorphisms (SNPs) are small variations in DNA, in single nucleotides. This is the official name when one or both alleles (one for each chromosome) vary in 1% or more of the population. Many of these polymorphisms are innocent, but in worse cases they can constitute tumor development. For this (and other) reasons research on the presence, form and effects of these SNPs was performed on an increasing scale in recent years. It is important to know which combination of alleles (the genotype) is present and whether there are deviations from the normal number of two alleles. The latter deviations are known as copy number variations.

Different manufacturers deliver platforms for SNP analysis. In all cases a process called selective hybridization is applied: probes selectively react (hybridize) to specific positions on the genome. Fluorophores are attached to specific molecules to label them and use the resulting fluorescence signals to determine allele A and B concentrations. Some manufacturers use one fluorophore (Affymetrix), others use two (Illumina). The amount of observed fluorescence represent the (relative) dose of the alleles. High signal for A and low for B indicates genotype AA, and vice versa for the BB genotype. Equal signals, thus equal doses, indicate genotype AB. Hence the total allele dosage in healthy DNA equals 2.

It practice it appears that these signals have some structural properties. If in one sample the signal for a single SNP is relatively bright compared to that of other SNPs, an equivalent proportionality also extends to other samples. This, combined with differences between samples and given the signal differences relating to allele doses (genotypes), implies that the aforementioned properties can be modeled. A model that does so, SCALA (Signal Calibration Algorithm) is introduced in this thesis. It estimates the systematic effectlevel for each SNP and corrects for it: calibration. SCALA distinguishes two variants: 1) estimating effects that allow for calibration independent of

genotypes (genotype-free calibration) and 2) estimating effects conditional on genotype, hence allowing for genotype-specific calibration. The latter approach requires known genotypes in order to be applied. The estimated effects are stable within one type of chip: they hold for new arrays and hence allow for calibration. Applying calibration with the estimated effects from one of the variants reduces variance, while dose ratios remain intact and chromosomal regions with deviations from dose 2 are clearer.

Determining genotypes from allele dosage is not completely trivial. Here, we look at allele dose ratio for all SNPs within one individual. The main advantage is that we never suffer from cluster imbalance due to low minor allele frequencies. After transformation of the signals for A and B we observe data scatter with three distinct clusters, representing the genotypes AA, AB and BB. The method is of semi-parametric nature, hence not making assumptions on the cluster shapes. The idea is that the data are transformed into a 2-dimensional histogram. On the obtained counts we fit a logconcave density model for each of the three clusters. This way, the model is also insensitive to increases of the number of observations: the histogram has unchanged dimensions. The model itself is also highly efficient.

The described approach is a so-called "single array" method and is introduced in this thesis. It contrasts with mainstream methods that are "multi-array". In case of genotyping that means that each individual SNP is genotyped in a set of arrays, based on the same dosage ratio. Hence the results and their quality depend on the amount of available observations, i.e. the number of arrays. In practice the proposed single array method has at least equivalent performance when comparing both branches to HapMap genotypes. There is no gold standard, but HapMap is a database that provides derived genotypes — from a few algorithms — for a set of reference arrays. Furthermore, working with single arrays allows for genotyping of SNPs that would be otherwise undetermined.

The previously mentioned signal calibration is also useful is case of low quality arrays. After calibration we observe strongly improved cluster separation, and therefore decreased calling uncertainty, for different platforms.

In unhealthy DNA variations on the total allele dosage 2 can be found.

Dosage for alleles A and B can also be 0, 1, 3 or more, resulting e.g. in genotypes 0, A0, 0B, AAA or ABB. When estimating these dosage profiles along positions on the chromosome a signal smoother is used that irons out small differences due to signal variation. The usability of this profile depends on the smoother used. A smoother that penalizes the number of changes between segments (a penalty with L_0 norm) finds only those changes, and nothing else, while penalizing the size of the changes also picks up noise.

Here, reference profiles with healthy DNA, thus dosage 2 can be used. However, in practice it is sometimes hard to obtain these samples. A comparison to VEGA shows that using an L_0 norm in the penalty gives reliable results even without reference data. The penalty can also be used in smoothed scatterplots; by using it in one direction we obtain smoother approximations per segment.

An extension of this penalty to a model for allelic imbalance seems obvious. However, in practice it is not: it requires additional steps, thereby increasing uncertainty. A first attempt is given by fitting a mixture of distributions on histograms per segment.

Some directions are left untouched. For example, extrapolations of the techniques to tetraploid DNA (e.g. in potatoes, leek, roses and some species of fish, like salmon. Another interesting approach is to jointly model genoypes, copy numbers and calibration parameters. Replacing the hard-called genotypes by their fuzzy counterparts (the genotype probabilities) may further increase model effectiveness.

Appendices

Fluorescence bias: calibration result tables

A

Affymetrix 100k Hind

Table A.1: Results for linear models fitted on Affymetrix 100k Hind. Global model (3.1) is indicated by G. Local model (3.2) is indicated by L. Improvement of L over G is indicated by D, where D=(G-L)/G *100. Rows: chromosomes. Columns: Global (G), Local (L), Difference (D).

	AA (G)	AB (G)	BB (G)	AA (L)	AB (L)	BB (L)	D(AA)	D(AB)	D(BB)
1	0.064	0.079	0.082	0.056	0.054	0.058	12.636	31.439	29.573
2	0.064	0.080	0.080	0.057	0.055	0.056	11.466	31.410	30.893
3	0.064	0.078	0.082	0.056	0.054	0.055	12.550	31.194	32.255
4	0.064	0.080	0.080	0.056	0.054	0.056	11.266	31.844	30.190
5	0.064	0.079	0.082	0.057	0.055	0.056	11.885	30.018	31.568
6	0.065	0.079	0.086	0.056	0.055	0.057	13.250	30.765	33.516
7	0.066	0.079	0.084	0.058	0.055	0.056	12.369	29.732	33.689
8	0.063	0.077	0.080	0.056	0.055	0.056	11.718	29.350	30.075
9	0.063	0.078	0.082	0.055	0.054	0.054	12.543	30.289	34.396
10	0.064	0.077	0.081	0.056	0.053	0.055	12.484	31.148	32.682
11	0.064	0.079	0.082	0.056	0.054	0.057	12.617	30.849	30.972
12	0.064	0.078	0.081	0.056	0.055	0.055	12.069	30.381	32.807
13	0.064	0.079	0.083	0.056	0.055	0.057	12.168	30.638	32.039
14	0.065	0.080	0.081	0.057	0.055	0.056	12.259	31.339	30.680
15	0.063	0.078	0.081	0.055	0.054	0.056	11.921	30.768	30.721
16	0.064	0.076	0.080	0.056	0.054	0.056	12.611	29.531	29.702
17	0.065	0.077	0.082	0.056	0.055	0.058	13.008	28.968	29.521
18	0.064	0.078	0.079	0.058	0.055	0.057	9.578	29.753	28.458
19	0.065	0.076	0.087	0.057	0.054	0.064	12.138	28.911	26.581
20	0.064	0.079	0.082	0.056	0.055	0.058	13.100	30.296	29.698
21	0.066	0.079	0.086	0.058	0.057	0.063	12.429	28.648	26.434
22	0.070	0.084	0.084	0.060	0.056	0.056	14.437	32.807	33.557

Affymetrix 100k Xba

Table A.2: Results for linear models fitted on Affymetrix 100k Xba. Global model (3.1) is indicated by G. Local model (3.2) is indicated by L. Improvement of L over G is indicated by D, where D=(G-L)/G *100. Rows: chromosomes. Columns: Global (G), Local (L), Difference (D).

	AA (G)	AB (G)	BB (G)	AA (L)	AB (L)	BB (L)	D(AA)	D(AB)	D(BB)
1	0.064	0.079	0.092	0.053	0.050	0.061	17.229	36.306	33.778
2	0.063	0.078	0.091	0.053	0.050	0.060	16.184	36.388	34.178
3	0.062	0.077	0.093	0.050	0.049	0.060	18.109	35.428	35.488
4	0.063	0.076	0.088	0.053	0.050	0.059	14.543	35.152	33.280
5	0.064	0.078	0.093	0.052	0.050	0.060	17.832	35.968	34.820
6	0.065	0.077	0.093	0.054	0.050	0.060	17.791	34.720	35.291
7	0.063	0.079	0.093	0.053	0.050	0.059	16.759	36.039	36.285
8	0.064	0.076	0.092	0.053	0.049	0.061	16.696	34.499	33.859
9	0.063	0.077	0.091	0.052	0.050	0.061	17.069	35.527	33.610
10	0.064	0.077	0.092	0.054	0.050	0.061	16.098	35.668	34.034
11	0.063	0.078	0.090	0.052	0.050	0.060	17.135	35.276	33.261
12	0.062	0.079	0.092	0.051	0.050	0.060	17.405	36.609	34.303
13	0.062	0.075	0.089	0.052	0.049	0.059	15.186	34.321	33.141
14	0.065	0.081	0.092	0.054	0.051	0.060	16.738	37.005	34.370
15	0.063	0.076	0.094	0.053	0.049	0.064	15.142	35.686	31.605
16	0.067	0.078	0.093	0.056	0.049	0.061	17.378	37.282	34.810
17	0.063	0.080	0.093	0.052	0.049	0.061	18.379	38.781	34.504
18	0.063	0.076	0.092	0.052	0.049	0.062	17.531	35.769	32.556
19	0.064	0.080	0.089	0.053	0.054	0.061	18.100	32.901	31.616
20	0.063	0.076	0.092	0.052	0.050	0.061	17.229	33.644	32.981
21	0.063	0.078	0.091	0.054	0.052	0.058	14.833	33.170	36.242
22	0.069	0.083	0.097	0.060	0.055	0.065	12.195	34.227	33.194

Affymetrix 500k NSP

Table A.3: Results for linear models fitted on Affymetrix 500k NSP. Global model (3.1) is indicated by G. Local model (3.2) is indicated by L. Improvement of L over G is indicated by D, where D=(G-L)/G *100. Rows: chromosomes. Columns: Global (G), Local (L), Difference (D).

	AA (G)	AB (G)	BB (G)	AA (L)	AB (L)	BB (L)	D(AA)	D(AB)	D(BB)
1	0.056	0.071	0.085	0.047	0.053	0.066	16.086	26.054	22.280
2	0.056	0.071	0.085	0.047	0.053	0.066	15.852	26.208	22.844
3	0.057	0.070	0.086	0.048	0.052	0.065	15.806	25.745	23.601
4	0.056	0.072	0.085	0.048	0.054	0.067	15.330	24.969	22.022
5	0.056	0.072	0.086	0.047	0.053	0.066	16.106	25.606	23.190
6	0.056	0.071	0.086	0.047	0.053	0.066	16.179	26.166	23.653
7	0.057	0.071	0.086	0.048	0.053	0.066	16.577	25.104	23.919
8	0.056	0.071	0.085	0.047	0.053	0.065	16.588	25.174	23.247
9	0.057	0.071	0.086	0.048	0.053	0.067	16.005	25.574	22.699
10	0.056	0.072	0.085	0.047	0.053	0.066	15.288	26.221	21.970
11	0.057	0.071	0.086	0.048	0.053	0.066	15.990	25.585	22.985
12	0.056	0.071	0.085	0.047	0.053	0.065	15.792	25.442	23.168
13	0.056	0.072	0.085	0.047	0.054	0.065	16.220	25.312	23.620
14	0.057	0.073	0.085	0.048	0.054	0.065	15.966	26.126	23.845
15	0.056	0.071	0.085	0.047	0.053	0.066	16.171	25.817	22.068
16	0.056	0.070	0.084	0.047	0.052	0.064	16.116	26.055	23.642
17	0.056	0.068	0.086	0.046	0.050	0.066	16.258	26.315	23.259
18	0.056	0.071	0.085	0.047	0.053	0.065	15.924	24.556	23.179
19	0.056	0.072	0.085	0.048	0.054	0.066	15.222	24.762	22.596
20	0.056	0.068	0.085	0.048	0.050	0.066	15.138	26.028	21.611
21	0.056	0.072	0.087	0.048	0.054	0.069	15.379	24.683	20.829
22	0.056	0.072	0.086	0.048	0.055	0.068	14.331	24.456	21.003

Affymetrix 500k STY

Table A.4: Results for linear models fitted on Affymetrix 500k STY. Global model (3.1) is indicated by G. Local model (3.2) is indicated by L. Improvement of L over G is indicated by D, where D=(G-L)/G *100. Rows: chromosomes. Columns: Global (G), Local (L), Difference (D).

	AA (G)	AB (G)	BB (G)	AA (L)	AB (L)	BB (L)	D(AA)	D(AB)	D(BB)
1	0.058	0.070	0.081	0.051	0.049	0.062	12.637	29.184	23.774
2	0.059	0.069	0.081	0.051	0.049	0.061	12.732	28.980	24.618
3	0.059	0.068	0.082	0.051	0.049	0.061	13.275	28.572	25.530
4	0.058	0.069	0.081	0.051	0.049	0.061	11.988	28.515	24.337
5	0.059	0.069	0.082	0.051	0.049	0.061	13.166	29.210	25.382
6	0.059	0.069	0.083	0.051	0.049	0.063	13.419	29.090	24.622
7	0.059	0.069	0.082	0.051	0.049	0.061	13.662	28.678	25.219
8	0.059	0.069	0.081	0.051	0.049	0.061	12.746	29.379	24.440
9	0.059	0.069	0.081	0.052	0.049	0.061	12.303	29.300	24.560
10	0.059	0.069	0.080	0.052	0.049	0.061	12.290	29.601	23.334
11	0.059	0.069	0.082	0.051	0.049	0.061	13.327	28.616	24.832
12	0.059	0.069	0.082	0.051	0.049	0.062	12.856	29.549	24.441
13	0.059	0.069	0.080	0.052	0.049	0.060	11.927	29.020	24.631
14	0.058	0.070	0.081	0.051	0.049	0.062	12.346	29.496	23.564
15	0.059	0.068	0.081	0.051	0.049	0.060	12.290	28.291	25.577
16	0.059	0.068	0.081	0.052	0.048	0.061	12.784	29.296	24.526
17	0.059	0.070	0.083	0.052	0.049	0.063	12.979	29.850	23.222
18	0.058	0.070	0.081	0.051	0.050	0.061	12.455	28.635	24.401
19	0.059	0.069	0.083	0.051	0.049	0.064	13.654	29.261	22.808
20	0.059	0.068	0.081	0.051	0.048	0.061	13.047	29.260	24.870
21	0.060	0.070	0.083	0.052	0.050	0.062	13.088	28.143	25.301
22	0.060	0.070	0.080	0.053	0.050	0.061	12.155	28.656	23.503

Affymetrix SNP6.0

Table A.5: Results for linear models fitted on Affymetrix SNP6.0. Global model (3.1) is indicated by G. Local model (3.2) is indicated by L. Improvement of L over G is indicated by D, where D=(G-L)/G *100. Rows: chromosomes. Columns: Global (G), Local (L), Difference (D).

	AA (G)	AB (G)	BB (G)	AA (L)	AB (L)	BB (L)	D(AA)	D(AB)	D(BB)
1	0.063	0.087	0.089	0.052	0.064	0.060	18.148	25.624	32.323
2	0.064	0.087	0.088	0.052	0.066	0.059	18.503	24.126	33.239
3	0.064	0.086	0.089	0.052	0.065	0.059	18.177	24.676	32.900
4	0.065	0.088	0.088	0.053	0.067	0.058	17.883	23.174	33.588
5	0.064	0.086	0.089	0.052	0.065	0.059	17.800	24.526	32.892
6	0.064	0.087	0.089	0.053	0.065	0.060	17.789	24.709	32.729
7	0.064	0.086	0.089	0.052	0.065	0.060	18.187	24.282	32.781
8	0.064	0.086	0.089	0.052	0.064	0.060	18.377	24.935	32.905
9	0.064	0.086	0.089	0.052	0.065	0.060	18.461	24.477	32.663
10	0.063	0.087	0.089	0.051	0.065	0.060	18.111	25.382	32.453
11	0.064	0.088	0.088	0.052	0.067	0.059	18.515	23.982	33.351
12	0.063	0.087	0.088	0.052	0.065	0.060	18.071	24.768	32.577
13	0.064	0.088	0.088	0.053	0.067	0.058	17.412	23.015	33.601
14	0.064	0.088	0.088	0.052	0.066	0.059	18.228	24.975	33.177
15	0.063	0.085	0.089	0.051	0.063	0.060	18.018	25.994	32.369
16	0.062	0.085	0.090	0.051	0.062	0.061	18.160	27.055	32.015
17	0.062	0.086	0.091	0.051	0.064	0.062	18.480	25.964	31.926
18	0.064	0.087	0.088	0.053	0.066	0.059	17.634	24.218	33.173
19	0.063	0.085	0.092	0.050	0.064	0.063	19.627	25.199	31.754
20	0.063	0.087	0.089	0.051	0.064	0.060	18.693	26.814	32.625
21	0.065	0.087	0.088	0.053	0.067	0.059	17.904	22.514	33.437
22	0.064	0.086	0.090	0.051	0.063	0.061	19.160	26.448	31.553

Genotyping: coding scheme

B

Preparation of HapMap data for genotyping comparisons

In this section we describe the data set used in our comparisons, model settings for genotype calling, as well as the translation step to match HapMap calls to our {AA, AB, BB} format.

We compare genotype calls to those of Phase III. We only compare calls to SNPs that have matching 'RSid's. almost half of the total. We disregard the four allelotypes (A,C,G,T) and refer to homozygous genotypes as AA or BB and the heterozygous as AB.

To match our calls to those from HapMap, we need to use the same alphabet. HapMap calls are translated to A and B labels using the following R (R Development Core Team, 2011) code:

```
# create translation vector with default 5
# code contains the SCALA genotype calls
# rssel is a selection vector for matching SNP ids
# from HapMap SNP list, but in the SCALA ordering

# STEP 1:
  d = code[rssel]*0 + 5
# sort scala calls for available rs-ids in HapMap
# rsidt is the working list of HapMap rsids

# STEP 2:
  a = code[rssel][order(rsidt[rssel])]
# get aligned HapMap calls matched to rs-ids.
# hapmap is a dataframe with SNPs in rows,
```

```
# and arrays in colums
# hmsel is the SNP id list for the HapMap ordering

# STEP 3:
  b = hapmap[hmsel,samp+3][order(hapmap$rs[hmsel])]
# now a contains scala calls and
# contains hapmap calls for matching SNP id
# get all heterozygous calls

# STEP 4:
  selhetero = (b!='AA' & b!='CC' & b!='GG' & b!='TT')
# anything not homozygous is translated to 2 (AB)

# STEP 5:
  d[selhetero] = 2
# assign aligned homozygous calls


# STEP 6:
  d[a==1 & !selhetero] = 1
  d[a==3 & !selhetero] = 3
# keep NoCall seperate for later evaluation

# STEP 7:
  d[b=='NN'] = 4
```

Since genotype calls AA from either method are highly unlikely to be mistaken for BB, we can apply the above forced classification from the HapMap homozygous genotype calls into homozygous calls from SCALA.

Waves correction: result tables

C

Fit statistic

Numerical comparison in all following tables are defined as

$$d = \frac{\sum |s_i - z_i|}{n} \tag{C.1}$$

with d the normalized difference between the raw signal s and the smooth profile z (for each SNP i) on a given chromosome.

Output columns

Detailed results are provided for two tumor samples (GBM 139 and GBM 180). Results contain, for each chromosome, the difference for uncorrected data (Raw), after SCALA correction and after NoWaves correction. For both arrays, these tables are given for 4 levels of smoothing: $\lambda \in (1, 10, 100, 1000)$.

C.1 Sample GBM 139

Table C.1: Sample GBM 139; Raw vs SCALA vs NoWaves. All chromosomes for $\lambda = 1$.

Chromosome	Raw 1	SCALA 1	NoWaves 1
1	0.595	0.291	0.291
2	0.595	0.292	0.292
3	0.588	0.279	0.280
4	0.592	0.289	0.290
5	0.596	0.293	0.293
6	0.595	0.288	0.288
7	0.598	0.295	0.296
8	0.586	0.289	0.289
9	0.584	0.290	0.291
10	0.589	0.287	0.287
11	0.591	0.283	0.284
12	0.594	0.290	0.291
13	0.586	0.279	0.280
14	0.583	0.271	0.272
15	0.594	0.286	0.287
16	0.581	0.287	0.287
17	0.581	0.279	0.280
18	0.582	0.281	0.281
19	0.572	0.284	0.285
20	0.591	0.291	0.292
21	0.579	0.279	0.279
22	0.566	0.280	0.280

Table C.2: Sample GBM 139; Raw vs SCALA vs NoWaves.
All chromosomes for $\lambda = 10$.

Chromosome	Raw 10	SCALA 10	NoWaves 10
1	0.598	0.292	0.292
2	0.597	0.293	0.293
3	0.590	0.280	0.281
4	0.594	0.290	0.291
5	0.598	0.294	0.294
6	0.598	0.289	0.290
7	0.601	0.296	0.297
8	0.588	0.290	0.291
9	0.589	0.293	0.294
10	0.592	0.289	0.289
11	0.595	0.285	0.286
12	0.598	0.292	0.293
13	0.589	0.280	0.281
14	0.587	0.273	0.274
15	0.601	0.288	0.289
16	0.587	0.290	0.289
17	0.589	0.282	0.283
18	0.586	0.283	0.284
19	0.583	0.289	0.290
20	0.598	0.295	0.295
21	0.587	0.284	0.284
22	0.583	0.286	0.286

Table C.3: Sample GBM 139; Raw vs SCALA vs NoWaves. All chromosomes for λ = 100.

Chromosome	Raw 100	SCALA 100	NoWaves 100
1	0.600	0.293	0.293
2	0.600	0.293	0.294
3	0.592	0.281	0.282
4	0.596	0.291	0.291
5	0.601	0.294	0.294
6	0.600	0.290	0.291
7	0.603	0.298	0.298
8	0.592	0.292	0.292
9	0.593	0.295	0.297
10	0.594	0.290	0.290
11	0.598	0.287	0.287
12	0.600	0.293	0.294
13	0.592	0.281	0.282
14	0.590	0.274	0.275
15	0.607	0.290	0.291
16	0.591	0.292	0.292
17	0.594	0.285	0.285
18	0.591	0.285	0.286
19	0.592	0.293	0.293
20	0.603	0.297	0.297
21	0.592	0.287	0.287
22	0.594	0.290	0.289

Table C.4: Sample GBM 139; Raw vs SCALA vs NoWaves.
All chromosomes for λ = 1000.

Chromosome	Raw 1000	SCALA 1000	NoWaves 1000
1	0.602	0.293	0.293
2	0.603	0.294	0.295
3	0.594	0.281	0.282
4	0.599	0.291	0.292
5	0.603	0.295	0.295
6	0.602	0.291	0.292
7	0.605	0.298	0.299
8	0.594	0.293	0.293
9	0.598	0.299	0.300
10	0.597	0.291	0.291
11	0.601	0.288	0.289
12	0.602	0.295	0.295
13	0.593	0.282	0.283
14	0.592	0.276	0.277
15	0.610	0.292	0.292
16	0.594	0.293	0.293
17	0.598	0.286	0.287
18	0.594	0.286	0.287
19	0.599	0.295	0.296
20	0.606	0.298	0.298
21	0.596	0.289	0.289
22	0.603	0.293	0.292

C.2 Sample GBM 180

Table C.5: Sample GBM 180; Raw vs SCALA vs NoWaves. All chromosomes for $\lambda = 1$.

Chromosome	Raw 1	SCALA 1	NoWaves 1
1	0.622	0.299	0.300
2	0.620	0.300	0.301
3	0.620	0.297	0.298
4	0.620	0.299	0.300
5	0.624	0.302	0.303
6	0.618	0.296	0.297
7	0.639	0.310	0.311
8	0.611	0.297	0.297
9	0.628	0.312	0.313
10	0.619	0.297	0.298
11	0.615	0.295	0.296
12	0.639	0.315	0.315
13	0.620	0.298	0.299
14	0.620	0.292	0.293
15	0.642	0.312	0.312
16	0.626	0.309	0.310
17	0.611	0.290	0.292
18	0.614	0.294	0.295
19	0.613	0.296	0.297
20	0.620	0.300	0.300
21	0.617	0.301	0.301
22	0.617	0.304	0.304

Table C.6: Sample GBM 180; Raw vs SCALA vs NoWaves.
All chromosomes for $\lambda = 10$.

Chromosome	Raw 10	SCALA 10	NoWaves 10
1	0.624	0.300	0.301
2	0.623	0.302	0.302
3	0.623	0.299	0.300
4	0.622	0.300	0.301
5	0.627	0.304	0.304
6	0.621	0.297	0.298
7	0.642	0.311	0.312
8	0.614	0.298	0.299
9	0.633	0.314	0.315
10	0.622	0.299	0.300
11	0.620	0.297	0.298
12	0.643	0.316	0.317
13	0.624	0.300	0.301
14	0.625	0.294	0.295
15	0.650	0.315	0.315
16	0.631	0.312	0.312
17	0.619	0.294	0.295
18	0.619	0.297	0.297
19	0.626	0.302	0.303
20	0.627	0.303	0.304
21	0.627	0.306	0.306
22	0.634	0.311	0.311

Table C.7: Sample GBM 180; Raw vs SCALA vs NoWaves.
All chromosomes for λ = 100.

Chromosome	Raw 100	SCALA 100	NoWaves 100
1	0.627	0.302	0.302
2	0.625	0.302	0.303
3	0.625	0.300	0.301
4	0.625	0.301	0.302
5	0.629	0.305	0.305
6	0.623	0.298	0.299
7	0.645	0.312	0.313
8	0.617	0.299	0.300
9	0.637	0.316	0.317
10	0.625	0.300	0.301
11	0.623	0.298	0.299
12	0.646	0.318	0.318
13	0.626	0.301	0.302
14	0.628	0.296	0.297
15	0.655	0.317	0.317
16	0.636	0.314	0.314
17	0.624	0.296	0.297
18	0.623	0.299	0.300
19	0.636	0.306	0.307
20	0.632	0.305	0.305
21	0.633	0.310	0.310
22	0.648	0.316	0.317

Table C.8: Sample GBM 180; Raw vs SCALA vs NoWaves.
All chromosomes for λ = 1000.

Chromosome	Raw 1000	SCALA 1000	NoWaves 1000
1	0.629	0.302	0.303
2	0.628	0.303	0.304
3	0.626	0.301	0.301
4	0.626	0.302	0.302
5	0.631	0.305	0.306
6	0.625	0.299	0.300
7	0.647	0.313	0.314
8	0.619	0.300	0.301
9	0.640	0.319	0.319
10	0.628	0.301	0.302
11	0.625	0.299	0.300
12	0.648	0.319	0.320
13	0.628	0.302	0.303
14	0.630	0.297	0.298
15	0.659	0.318	0.319
16	0.639	0.315	0.315
17	0.628	0.298	0.299
18	0.626	0.300	0.301
19	0.644	0.309	0.310
20	0.636	0.306	0.307
21	0.638	0.313	0.314
22	0.658	0.320	0.320

Manual: SCALA Suite

D

D.1 Introduction

This software suite is a collection of programs that were created for and during a PhD project on calibration and genotyping of SNP signals. The whole framework is built on a set of two signals (one for each allele).

Signals from SNP arrays are not perfect; they contain noise. However, in practice this 'noise' has some very structural properties that can be modeled and exploited. It is not hard to imagine that in one SNP array, some SNPs of a particular genotype have a lower signal than other SNPs (of the same genotype). However, we noticed that a SNP with a lower signal behaves similarly in other arrays (of the same platform) as well. Add this to the fact that each array has its own overall signal level and that genotypes are (obviously) expressed in different signal levels for each allele, and there is a strong basis for a model.

The SCALA software models the effects described above. Signals after calibration are much more condensed, which can be beneficial in applications like genotyping (for a single array) and maps of copy numbers and loss of heterozygosity. The latter is not (yet) contained in this suite.

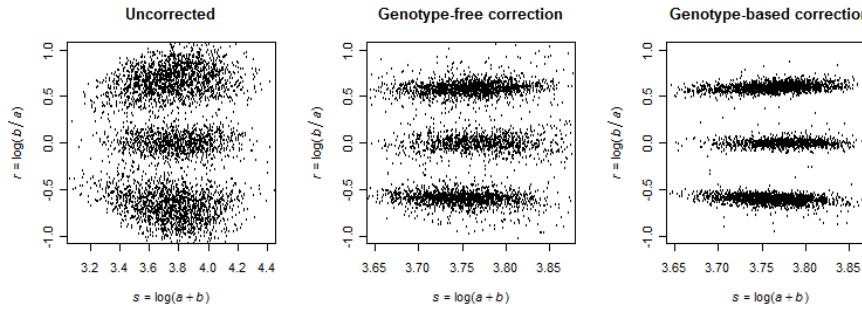
It contains a function for CEL-file conversion to the format used in SCALA (single signal per allele, no probe level information), a function to perform single array genotyping using semi-parametric mixtures on a smoothed 2-dimensional histogram, and a function to obtain signal calibration parameters.

Currently, the software handles mainly Affymetrix CEL files. To be more specific: it handles both enzymes from the 100k platform and both enzymes from the 500k platform, as well as SNP6.0 arrays.

Calibration

To illlustrate the calibration possibilities mentioned above, we provide a graphical example in Figure 1. Starting out with the uncalibrated averaged signals a and b

for allele A and B, we take $s = log(a + b)$ on the horizontal axis and $r = log(b/a)$ on the vertical axis (left panel). This orientation provides three SNP clusters: two for the homozygous genotypes AA (bottom) and BB (top), and one for the heterozygous genotype (middle). Without calibration (after plain signal conversion) this panel shows a lot of noise. However, we can reduce it in the data by using the set of α parameters from the SCALA model (middle panel) or by using Γ from the local model (right panel).



The software can perform global calibration at the conversion stage. The software also provides α sets after model fitting, so that users can perform calibration manually at any later stage.

D.2 The SCALA object class

We defined an object of class SCALA. Not because of object-specific print or plot functions (at this time), but simply to add structure to the results obtained from the different functions contained in this suite. Each of the functions add information to the object. A final object (after conversion, with calibrated signals, and genotyping) has the following structure:

```
> str(scala)

List of 10
$ meta :List of 6
  ..$ fname      : chr "ctr aff 1.CEL"
  ..$ readpath   : chr "D:/Documents/Werk/000 SCALA Suite/01 raw"
  ..$ savepath   : chr "D:/Documents/Werk/000 SCALA Suite/02 arrays"
  ..$ convertDate: chr "2010-12-30 11:21:06"
  ..$ calibrated : logi TRUE
  ..$ callDate   : chr "2010-12-30 12:32:56"
$ chr  : chr [1:262264] "20" "4" "14" "1" ...
$ pos : int [1:262264] 47874178 104894961 51975831 21039991 56554433 ...
$ rsid : chr [1:262264] "rs16994928" "rs233978" "rs2249922" "rs7553394" ...
$ X    : int [1:262264] 267 637 291 2081 772 809 328 421 277 1046 ...
$ Y    : int [1:262264] 1023 776 801 333 989 1043 1398 1359 1183 396 ...
$ Xc   : int [1:262264] 365 722 313 1174 802 804 308 335 303 1157 ...
$ Yc   : int [1:262264] 1234 799 979 315 806 835 1218 1242 1094 364 ...
$ calls: num [1:262264] 3 2 3 1 2 2 3 3 3 1 ...
$ W : num [1:262264, 1:3] 1.96e-10 2.15e-04 2.57e-08 1.00 1.72e-04 ...
 - attr(*, "class")= chr "SCALA"
```

The calibration models and (GUI-based) mapping function currently do not add to the object.

D.3 SCALA.convert: CEL file conversion

Description:

This function converts raw CEL files into aggregated signals X for allele A and Y for allele B.

Usage:

```
SCALA.convert(datatype='Affy250kNSP',calibrate=F,
              readfolder=paste(getwd(),'/01 raw',sep=''),
              savefolder=paste(getwd(),'/02 arrays',sep=''))
```

Arguments:

datatype :	'Affy50kHIND' (default), 'Affy50kXBA'
	'Affy250NSP' , 'Affy250STY'
	'AffySNP6.0'
calibrate :	TRUE (default), FALSE
readfolder :	defaults to getwd()
savefolder :	defaults to getwd()

Details:

The resulting SCALA object is automatically saved to the specified savefolder, to a file that matches [scala$meta$fname].Rdata.

If calibrate is set to T, calibration is indicated and two vectors ($Xc and $Yc) containing the calibrated signals are added after the original signals $X and $Y. The following additions and changes are made:

```
  ..$ calibrated : logi TRUE
..
$ Xc   : int [1:262264] 365 722 313 1174 802 804 308 335 303 1157 ...
$ Yc   : int [1:262264] 1234 799 979 315 806 835 1218 1242 1094 364 ...
```

See also:

SCALA.call, SCALA.global

Examples:

```
scala = SCALA.convert('Affy250kNSP',F,
              readfolder=paste(getwd(),'/01 raw',sep=''),
              savefolder=paste(getwd(),'/02 arrays',sep=''))

str(scala)

List of 7
 $ meta :List of 6
  ..$ fname      : chr "ctr aff 1.CEL"
  ..$ readpath   : chr "D:/Documents/Werk/000 SCALA Suite/01 raw"
  ..$ savepath   : chr "D:/Documents/Werk/000 SCALA Suite/02 arrays"
  ..$ convertDate: chr "2010-12-30 11:21:06"
  ..$ calibrated : logi FALSE
  ..$ callDate   : logi NA
$ chr  : chr [1:262264] "20" "4" "14" "1" ...
$ pos : int [1:262264] 47874178 104894961 51975831 21039991 56554433 ...
$ rsid : chr [1:262264] "rs16994928" "rs233978" "rs2249922" "rs7553394" ...
$ X    : int [1:262264] 267 637 291 2081 772 809 328 421 277 1046 ...
$ Y    : int [1:262264] 1023 776 801 333 989 1043 1398 1359 1183 396 ...
$ calls: logi [1:262264] NA NA NA NA NA NA NA NA ...
 - attr(*, "class")= chr "SCALA"
```

D.4 SCALA.global: calibration

Description:

This function reads all arrays in the readfolder and assumes called genotypes in the SCALA objects.

Usage:

```
params = SCALA.global(filefolder=getwd(), savefolder=getwd(),
                      filename = scala.glob.Rdata, kappa = 1e-8)
```

Arguments:

filefolder :	defaults to getwd()
savefolder :	defaults to getwd()
filename :	defaults to scala.glob.Rdata
kappa :	set value to add to avoid singularity (1e-8)

Details:

The resulting calibration parameters are returned in a separate object, instead of being added to the SCALA object. The reason for this is that the parameters are based on multiple arrays and hence should be added to each array used to obtain the calibration set.

The fields in params match to α, β and γ in the model explained in the appendix. The α values can be used to calibrate the original signal by taking

$$Xc = X/10^{\alpha}.$$

An equivalent approach can be taken for the Y signal. This is the calibration that be performed during CEL file conversion, for the currently implemented platforms.

See also:

SCALA.convert, SCALA.call

Examples:

```
params = SCALA.global()

str(params)

List of 7
 $ celfiles: chr [1:10] "ctr aff 1.CEL.Rdata" "ctr aff 2.CEL.Rdata" ...
 $ alphaX  : num [1:262217] -0.157 -0.0438 -0.0422 0.311 0.0699 ...
 $ alphaY  : num [1:262217] -0.041653 0.016274 -0.062808 0.00015 ...
 $ betaX   : num [1:10] 0.1303 -0.2242 0.1149 0.1248 0.0783 ...
 $ betaY   : num [1:10] 0.114 -0.239 0.1 0.11 0.066 ...
 $ gammaX  : num [1:3] 0.1822 0.0392 -0.2508
 $ gammaY  : num [1:3] -0.2505 0.0922 0.2386
```

D.5 SCALA.call: single array genotyping

Description:

To obtain genotype calls based on a single array, this function 'does the trick'. It uses a mixture of three semi-parametric log-concave densities and classifies each SNP into the cluster with the highest probablility.

Usage:

```
SCALA.call(scala=scala, model='s', plot=F, save=T, xbins = 100,
           ybins = 100, lambda = 10, nit=50, crit=1e-4,
           savefolder=paste(getwd(),'/02 arrays',sep=''))
```

Arguments:

scala :	expects the SCALA object as described above
model :	's': use semi-parametric model, anything other than 's' will revert to a mixture of three parametric regression models using the flexmix package ('s')
plot :	plot single array mixture (FALSE)
save :	save resulting object to file TRUE
xbins :	# of histogram bins to use on x-axis (100)
ybins :	# of histogram bins to use on y-axis (100)
lambda :	sets amount of smoothing in the histogram (10)
nit :	set maximum # of mixture iterations (50)
crit :	sets convergence threshold (1e-4)
savefolder :	defaults to getwd()

Details:

Genotype calls from any source (e.g. HapMap or CRLMM) can be added by simply replacing the $calls vector with the external calls (with AA = 1, AB = 2 and BB = 3).

The result is a change in one meta-tag ($meta$callDate) and addition of two list elements $calls and $W to the SCALA object.

```
..$ callDate   : chr "2010-12-30 12:32:56"
..
$ calls: num [1:262264] 3 2 3 1 2 2 3 3 3 1 ...
$ W : num [1:262264, 1:3] 1.96e-10 2.15e-04 2.57e-08 1.00 1.72e-04 ...
```

See also:

SCALA.convert, SCALA.global

Examples:

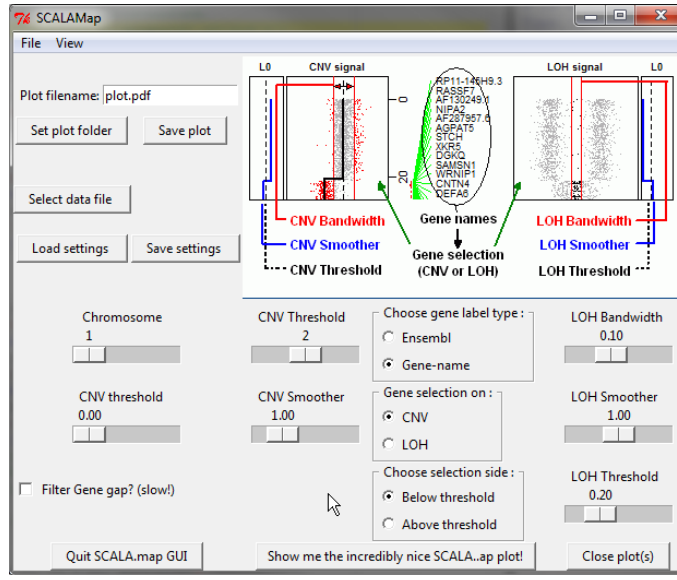
```
scala = SCALA.call(scala, xbins = 75, ybins = 75, lambda = 5)
```

D.6 SCALA.map: CNV / LOH mapping

Description:

The CNV and LOH analyses that are performed detect which genes in either the CNV or LOH signal fall below (or above) the expected threshold number of alleles that is set by the user. The selection results of this detection can be saved either per SNP or per gene. Exported results are saved in .csv format.

The mapping function is fully GUI-controlled (using the rpanel package), not commandline.



Using the GUI the user can

- select the chromosome to analyze,
- choose whether the signal subject to evaluate indicates CNV or lOH,
- where and under what number the analysis figure should be saved,
- choose between Ensembl codes or gene names in selected chromosome regions,
- adapt the signal smoother between power 0 and 2 and
- change plot (title) properties.

Usage:

```
SCALA.map(controls=NA)
```

Arguments:

This function currently only take 1 argument: a saved 'settings' file from a previous analysis.

Details:

The function call simply starts the GUI and doesn't perform any analysis until a SCALA class object is read. If calibrated signals are present, the program uses these automatically, if the $meta$calibrated is set to T.

The exported results file (.csv) contains a number of fields, summarized in the following Table.

See also:

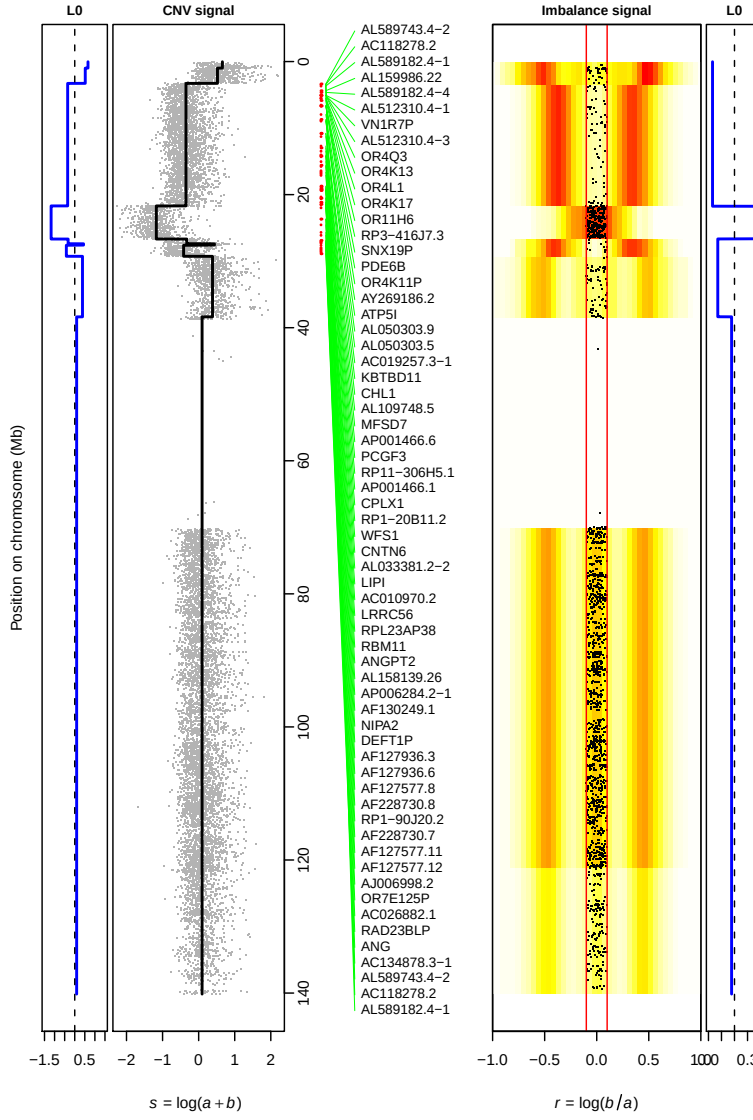
SCALA.convert, SCALA.global, SCALA.call

```
SCALA.map(controls='lastrun.Rdata')
```

The resulting SCALA.map plot:

SNP id :	Database SNP id ('rsid')
CNV sig :	CNV signal value for each SNP
LOH sig :	LOH signal value for each SNP
Position :	SNP position on the chromosome
Chrom :	Chromosome the SNP is located on
Z :	Smoothed CNV value for each SNP
SNP selected :	Indicator whether the SNP exceeds the user-defined threshold
N-level :	Copy Number level for each SNP
GeneBio :	BioMart name of the gene containing the SNP
GeneENS :	Ensembl name of the gene containing the SNP
G-Start :	Starting position of the gene
G-Stop :	Ending position of the gene
Gene selected :	Indicator whether this gene exceeds the user-defined threshold
Mean CNV :	The mean CNV signal in the gene
Mean Z :	Mean CNV smoother value in the gene
Mean LOH :	Mean LOH signal in the gene
Mean G :	Mean LOH smoother value in the gene

GBM 139.CEL chromosome 9



D.7 Appendix: The SCALA model

Theory

The SCALA model aims to find calibration values for the averaged allele intensities for each SNP.

Let $t_{ij} = \log(a_{ij})$, where the logarithms are to base 10. Let the genotypes be coded in the 3-way indicator matrix $H = [h_{ijk}]$, where $k \in \{1, 2, 3\}$ codes for the genotype. $h_{ijk} = 1$ if SNP i on array j has genotype k, otherwise $h_{ijk} = 0$. The first, global, model is written as

$$t_{ij} = \mu + \alpha_i + \beta_j + \sum_{k=1}^{3} \gamma_k h_{ijk} + e_{ij}, \tag{D.1}$$

where μ is the grand mean, α_i the effect of SNP i, and β_j the effect of array j, and γ_k the effect of genotype k. For identifiability, we introduce the constraints $\sum_i \alpha_i = 0$ and $\sum_j \beta_j = 0$. The error $e = [e_{ij}]$ is assumed to have constant variance. The model has one set of genotype parameters (γ) for all SNPs.

A refinement is to have separate genotype parameters for each SNP: $\mathbf{\Gamma} = [\gamma_{ik}]$. We call this the local model, which is specified as

$$t_{ij} = \mu + \beta_j + \sum_{k=1}^{3} \gamma_{ik} h_{ijk} + e_{ij}, \tag{D.2}$$

where we again require that $\sum_j \beta_j = 0$.

Identical models are used for the B allele, with $t_{ij} = \log(b_{ij})$.

Implementation

For the latter model, with appropriate $\boldsymbol{C}$ and $\boldsymbol{D}$, we can write

$$\boldsymbol{t} = \boldsymbol{C}\boldsymbol{\beta} + \boldsymbol{D}\boldsymbol{\gamma} + e \tag{D.3}$$

where $\boldsymbol{\beta}$ contains the n β_j parameters in (D.2) and $\boldsymbol{\gamma}$ = vec($\mathbf{\Gamma}$), i.e. the columns of $\mathbf{\Gamma} = [\gamma_{ik}]$ stacked below each other, and $\boldsymbol{t} = \text{vec}(\boldsymbol{T})$. The structure of $\boldsymbol{C}$ is simple, it can be written as $\boldsymbol{C} = \boldsymbol{I}_n \otimes \mathbf{1}_p$, where $\boldsymbol{I}_n$ is the $n \times n$ identity matrix and $\mathbf{1}_p$ is a vector of ones, of length p. The structure of $\boldsymbol{D}$ is more complex; it consists of n blocks of

diagonal matrices. Each block has three diagonal matrices $\boldsymbol{D_{jk}}$, one for each layer of $\boldsymbol{H}$, and each matrix $\boldsymbol{D_{jk}}$ contains the elements of the jth vector in the kth layer of the 3-way matrix $\boldsymbol{H}$ on its diagonal. Thus, $\boldsymbol{D}$ has dimensions $(n \times p) \times 3p$.

We do not form $\boldsymbol{C}$ and $\boldsymbol{D}$ explicitly. Instead we study the normal equations

$$\begin{bmatrix} \boldsymbol{C'C} & \boldsymbol{C'D} \\ \boldsymbol{D'C} & \boldsymbol{D'D} \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{\beta}} \\ \hat{\boldsymbol{\gamma}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{C't} \\ \boldsymbol{D't} \end{bmatrix}, \tag{D.4}$$

or

$$\begin{bmatrix} \boldsymbol{V_{11}} & \boldsymbol{V_{12}} \\ \boldsymbol{V_{21}} & \boldsymbol{V_{22}} \end{bmatrix} \begin{bmatrix} \hat{\boldsymbol{\beta}} \\ \hat{\boldsymbol{\gamma}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f_1} \\ \boldsymbol{f_2} \end{bmatrix}, \tag{D.5}$$

where $\boldsymbol{V}_{11} = \boldsymbol{C'C}$, $\boldsymbol{V}_{12} = \boldsymbol{C'D}$, $\boldsymbol{V}_{21} = \boldsymbol{D'C}$, $\boldsymbol{V}_{22} = \boldsymbol{D'D}$, $\boldsymbol{f}_1 = \boldsymbol{C't}$ and $\boldsymbol{f}_2 = \boldsymbol{C't}$. One can prove that $\boldsymbol{C'C} = p\boldsymbol{I_n}$, $\boldsymbol{D'} = \tilde{\boldsymbol{H}}$ and $\boldsymbol{D'D} = \boldsymbol{F}$, where $\tilde{\boldsymbol{H}}$ is a matrix formed by placing the three layers of $\boldsymbol{H}$ below each other. $\boldsymbol{F}$ is a $3p$ by $3p$ diagonal matrix; its first (second, third) p diagonal elements gives, for each SNP, the number of times genotype 1 (2, 3) occurs. Furthermore, $\boldsymbol{C't}$ contains the sums of the columns of $\boldsymbol{T}$, while $\boldsymbol{D't}$ is a stack of three vectors; the first (second, third) vectors contain the sum, per SNP of the elements of $\boldsymbol{t}$ corresponding to genotype 1 (2, 3).

From (D.5) it follows:

$$\hat{\boldsymbol{\gamma}} = \boldsymbol{V}_{22}^{-1}(\boldsymbol{d}_2 - \boldsymbol{V}_{21}\hat{\boldsymbol{\beta}}) \tag{D.6}$$

and hence

$$(\boldsymbol{V}_{11} - \boldsymbol{V}_{12}\boldsymbol{V}_{22}^{-1}\boldsymbol{V}_{21})\hat{\boldsymbol{\beta}} = \boldsymbol{d}_1 - \boldsymbol{V}_{12}\boldsymbol{V}_{22}^{-1}\boldsymbol{d}_2. \tag{D.7}$$

Because $\boldsymbol{V}_{22}$ is a diagonal matrix, multiplication by $\boldsymbol{V}_{22}^{-1}$ boils down to dividing the elements of a vector or the rows of a matrix by the corresponding diagonal elements of $\boldsymbol{V}_{22}$. Hence, it is not hard to compute $\boldsymbol{V}_{11} - \boldsymbol{V}_{11}\boldsymbol{V}_{12}^{-1}\boldsymbol{V}_{21}$ and to solve for $\hat{\boldsymbol{\beta}}$, a vector of moderate length. Additional efficiency can be realized by exploiting the way $\boldsymbol{V}_{21}$ is formed. Details on the latter suggestion are considered outside the scope of the current paper.

In this analysis we have ignored the fact that the system in (D.5) is singular, because the condition $\sum_j \beta_j = 0$ is not applied. An easy way out is to demand the minimum-norm solution for $\boldsymbol{\beta}$, by replacing $\boldsymbol{C'C}$ in (D.5) by $\boldsymbol{C'C} + \kappa\boldsymbol{I}$ with κ a small number.

Subject Index

Notes

Curriculum Vitae

Ralph C.A. Rippe was born on February 5, 1982 in Delft. In 2000, he graduated from the Sint Laurens-college (VWO) in Rotterdam. He studied Computer Science at Leiden University, but in 2002, he switched to Psychology. In 2006 he graduated in Methodology and Statistics (*c*um laude). His Master thesis concerned an adaptation of the Multiple Correspondence Analysis algorithm in order to work with datasets containing large design-determined chunks of missing data.

In 2006, after his graduation, he started working as a PhD candidate in the Data Theory Group in the Faculty of Social and Behavioral Science in Leiden. Originally aiming at developing methods for large (wide) datasets from Systems Biology, the project gradually changed its focus to modelling structural properties in SNP signals, after finding many interesting results in a side project. In the course of the project several internal and external cooperations were initiated; among others, with the Department of Neurology at the Erasmus Medical Center in Rotterdam.

During his thesis research, he won several awards. Among these were the Poster Award at the 2nd Channel Network Conference of the International Biometric Society in 2009, the Paper Award at the 24th International Workshop on Statistical Modeling in 2009, and the Presentation Award at the 25th International Workshop on Statistical Modeling in 2010. He was elected as PhD representative of the Interuniversity Research School for Psychometrics and Sociometrics (IOPS) for the period 2008-2010.

Currently, he is a statistician in the department of Clinical Epidemiology in the Leiden University Medical Center.

Stellingen

behorende bij het proefschrift "Advanced statistical models for SNP arrays: Signal calibration, copy number estimation and single array genotyping", van Ralph C.A. Rippe

---

1. Systematische afwijkingen in fluorescentiesignalen op SNP arrays kunnen worden gecorrigeerd met de parameters van een linear regressiemodel *(dit proefschrift)*.
2. Het genotyperen van enkelvoudige SNP monsters doet in betrouwbaarheid niet onder voor gevestigde methoden voor sets van monsters *(dit proefschrift)*.
3. De doelfunctie van de ZEN smoother is niet convex, dus is een globaal minimum niet gegarandeerd. Toch werkt het algoritme in de praktijk uitstekend *(dit proefschrift)*.
4. Golfpatronen in copy number profielen op basis van SNP fluorescentiesignalen worden onterecht als ruimtelijk effect gezien: deze "waves" zijn artefacten en kunnen worden gecorrigeerd zonder verlies van inhoudelijke informatie over het profiel *(dit proefschrift)*.
5. Recente verschuivingen van SNP chips met hoge resoluties naar platforms voor sequencing van het volledige genoom hebben tot nu toe niet geleid tot nieuwe inzichten.
6. Men vreesde voor kwalijke gevolgen voor bijvoorbeeld verzekeringsnemers door het gebruik van DNA-gegevens om levensverwachting te voorspellen. Die vrees lijkt ongegrond.
7. Toenemende modelcomplexiteit draagt niet automatisch bij aan kennis op toegepast niveau.

8. Het zichtbaar maken van nieuw werk, door implementatie in toegankelijke software, is essentieel in methoden- en statistiek–ontwikkeling.
9. In een “hot” onderzoeksgebied heeft de “in-crowd” de neiging nieuwe en/of andere inzichten van buiten de groep tegen te houden.
10. Indirect bewijs suggereert dat elke toegevoegde supervisor de projectcomplexiteit meer dan verdubbelt.
11. Het leven is wreder dan de dood; niet voor hen die sterven, maar voor hen die achterblijven.