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

Rippe, R.C.A.

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.

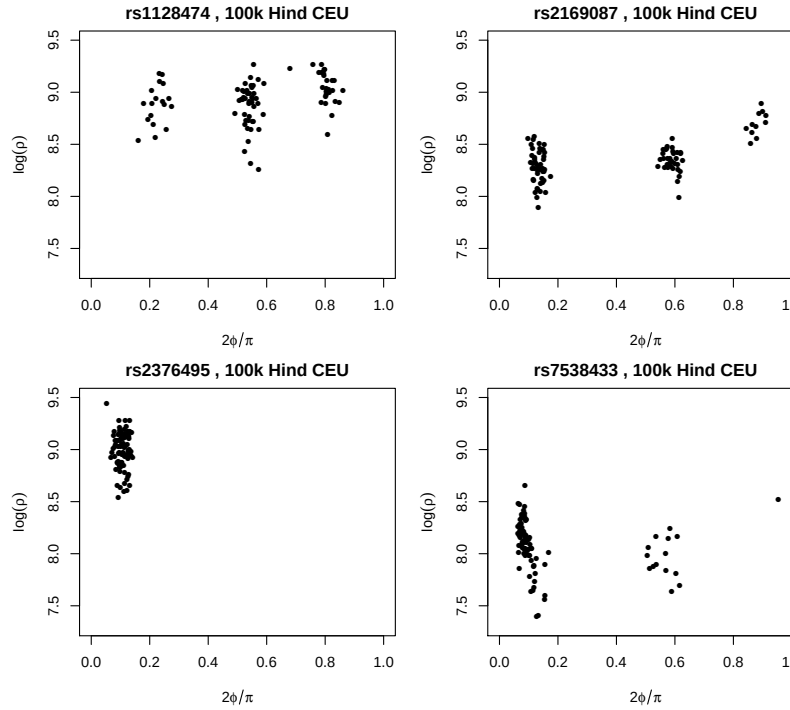

Figure 3.1: Multi-array genotyping for four separate SNPs in a sample set from the CEU HapMap population. Top row: a clear three genotype division without minor allele frequency problem. Bottom row: genotype clusters with minor allele frequency problems.

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.

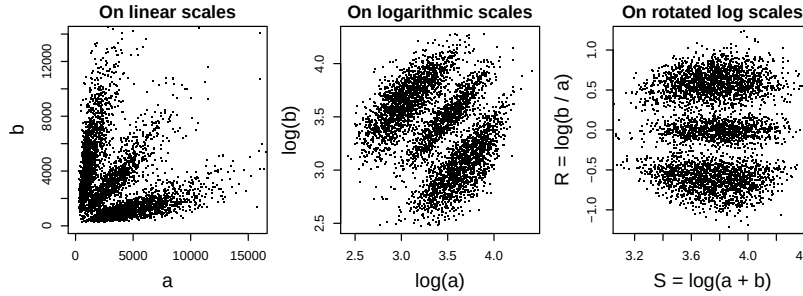

Figure 3.2: Illustration of signal transformation. Signal a (b) represents allele A (B). The left panel shows the signals on linear scales. The middle panel shows the same signals on logarithmic scale. The right panel shows transformed signals to $s = \log(a + b)$ on the x-axis and $r = \log(b/a)$ on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

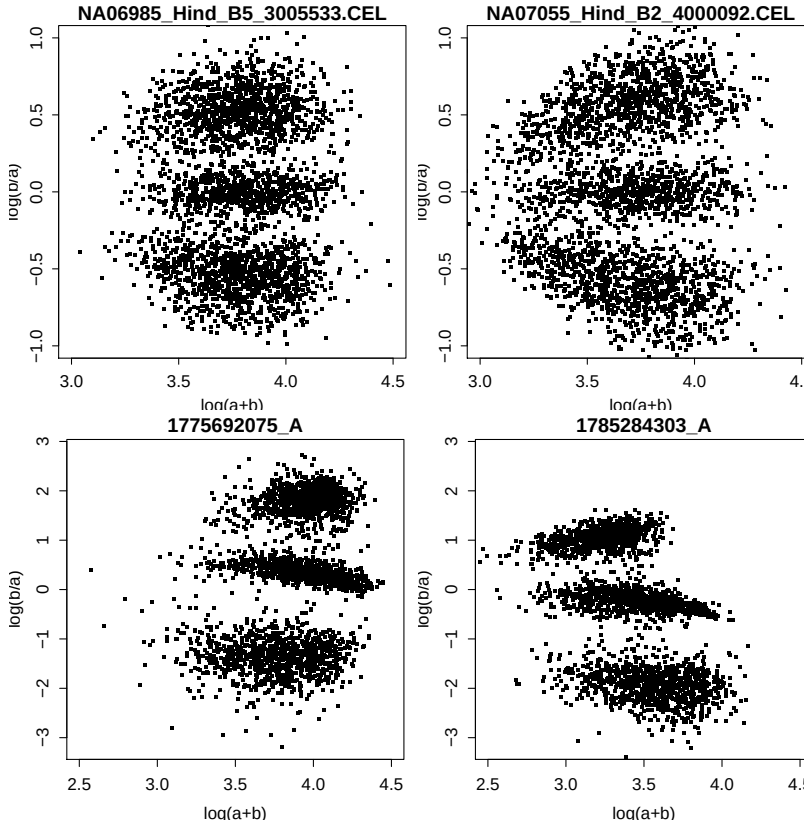

Figure 3.3: Single HapMap Affymetrix 100k Hind samples (NA06985, NA07055 from left to right) in top panels, typical Illumina arrays in bottom panels. SNPs are shown for chromosome 1.

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 $i = 1, \ldots, n$.

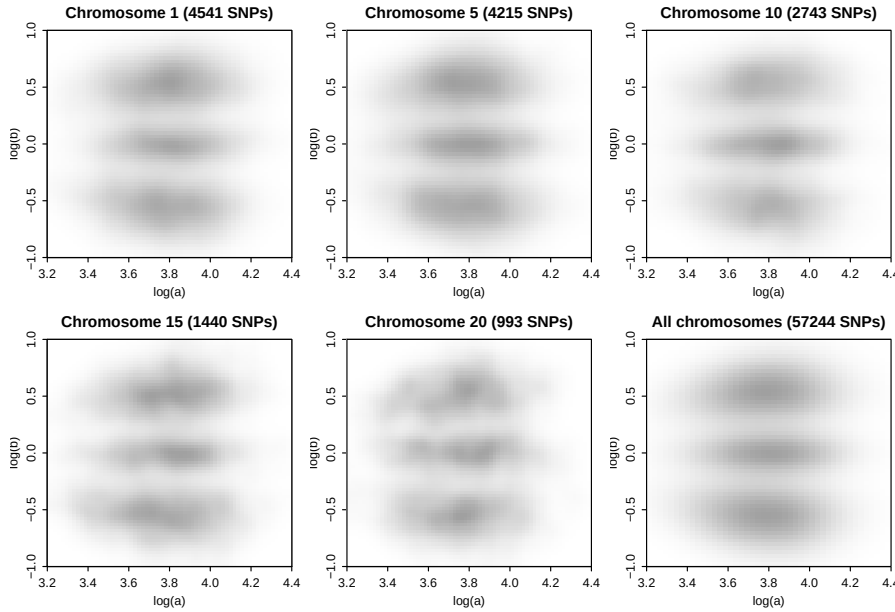

Figure 3.4: Genotype clusters in HapMap sample NA06985 for five individual chromosomes and genotype clusters over all chromosomes (bottom right panel). There is only a difference in SNP density, but not in scale or cluster separation.

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

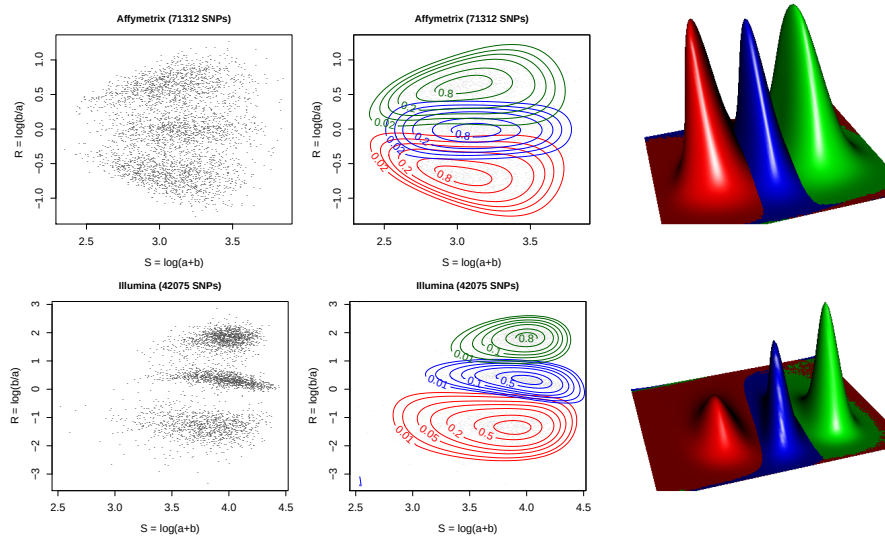

Figure 3.5: Top row: a typical Affymetrix SNP6.0 array. Bottom row: a typical Illumina HumanHap550 array. Left panels : a random selection of 3500 SNPs om chromosome 1 plotted as dots. Middle panels: observations and contour lines of semi-parametric mixture components. Normalized contours (mode set to 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

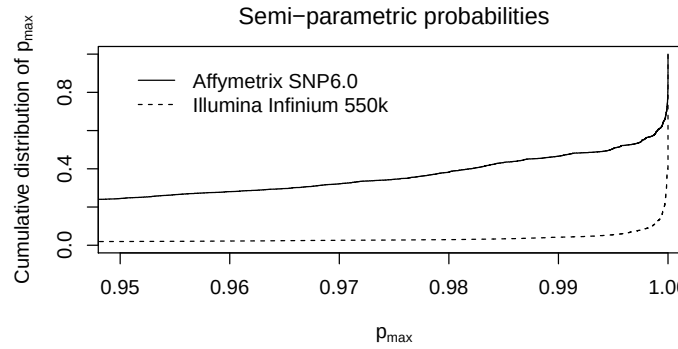

Figure 3.6: Comparison of semi-parametric probability distributions: symmetric Affymetrix (left) vs. asymmetric Illumina (right).

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-

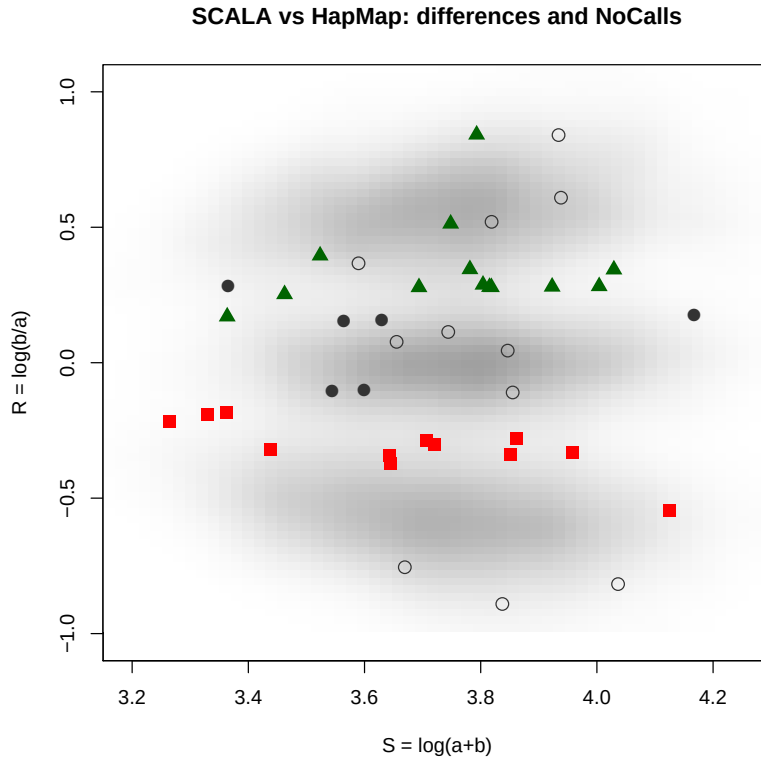

Figure 3.7: Example of SCALA call disagreements with HapMap for chromosome 1 on Affymetrix 100k Hind array NA06991. Some Hapmap AB genotypes called as AA (red squares) or BB (green triangles) by SCALA. HapMap NN calls (circles) can be genotyped with high (open) or low (filled) probability.

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-

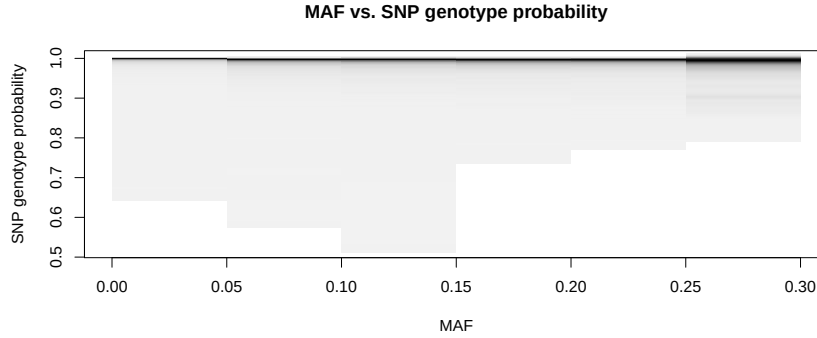

Figure 3.8: Dark color: high density, light color: low density. In data for all chromosomes, p_{max} is still high in low to moderate MAF situations, albeit with somewhat higher variance for lowest MAF.

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.