

# sequenza possible SNP-array usage example

Francesco Favero\*, Aron C. Eklund

November 22, 2013

## Contents

<b>1</b>	<b>Working with SNP array data</b>	<b>1</b>
1.1	Preparing the data . . . . .	1
1.1.1	Correcting logR with a normal sample, or with the mean logR value . . . .	1
1.1.2	Retrieve the homozygous position . . . . .	2
1.2	Windowing logR values. . . . .	2
1.3	Windowing B-allele frequencies values. . . . .	2
1.4	Chromosome view without mutation . . . . .	3
1.5	Segmenting with the <i>copynumber</i> package . . . . .	3
1.6	Using the Bayesian inference on segmented SNP arrays . . . . .	3
1.7	Cellularity and ploidy plot for SNP array . . . . .	4
1.8	Call for copy number variation using inferred parameters. . . . .	4
1.9	Graphical representation of copy number with SNP arrays . . . . .	6

## 1 Working with SNP array data

```
> library(sequenza)
```

### 1.1 Preparing the data

```
> data(BAF)
> data(logR)
```

```
> sample.i <- data.frame(chromosome = BAF$chrs, n.base = BAF$pos,
+                         Bf = BAF$S1, adjusted.ratio = logR$S1,
+                         depth.sample = 1, good.s.reads = 1,
+                         ref.zygosity = 'hom', stringsAsFactors = FALSE)
```

#### 1.1.1 Correcting logR with a normal sample, or with the mean logR value

Without a reference sample (normal germline sample) we can try to divide for the mean value. It would be correct to use the germline logR.

```
> #sample.i$adjusted.ratio <- 2^(sample.i$adjusted.ratio)
> #sample.i$adjusted.ratio <- sample.i$adjusted.ratio / mean(sample.i$adjusted.ratio)
> sample.i$adjusted.ratio <- 2^(sample.i$adjusted.ratio/0.55)
>
```

---

\*favero@cbs.dtu.dk

### 1.1.2 Retrieve the homozygous position

It should be available a germline sample to get the heterozygous SNP, doing in the same sample it's a risk if the sample is pure. A threshold around 0.25 or 0.35 can be picked to subset the heterozygous position on the germline. In the example we are lowering the threshold while taking the SNP from the same aberrant sample.

```
> het.lim <- 0.2
> is.het <- sample.i$Bf >= het.lim & sample.i$Bf <= 1 - het.lim
> sample.i$ref.zygosity[is.het] <- 'het'
> sample.i$Bf[sample.i$Bf >= 0.5] <- 1 - sample.i$Bf[sample.i$Bf >= 0.5]
> sample.het.i <- sample.i[is.het, ]
```

## 1.2 Windowing logR values.

```
> snp.r.win <- windowValues(x = sample.i$adjusted.ratio,
+                           positions = sample.i$n.base,
+                           chromosomes = sample.i$chromosome,
+                           window = 1e6, overlap = 1)
```

## 1.3 Windowing B-allele frequencies values.

```
> snp.b.win <- windowValues(x = sample.het.i$Bf,
+                           positions = sample.het.i$n.base,
+                           chromosomes = sample.het.i$chromosome,
+                           window = 1e6, overlap = 1)
```

## 1.4 Chromosome view without mutation

```
> chromosome.view(baf.windows = snp.b.win[[1]],  
+               ratio.windows = snp.r.win[[1]],  
+               min.N.ratio = 1)
```

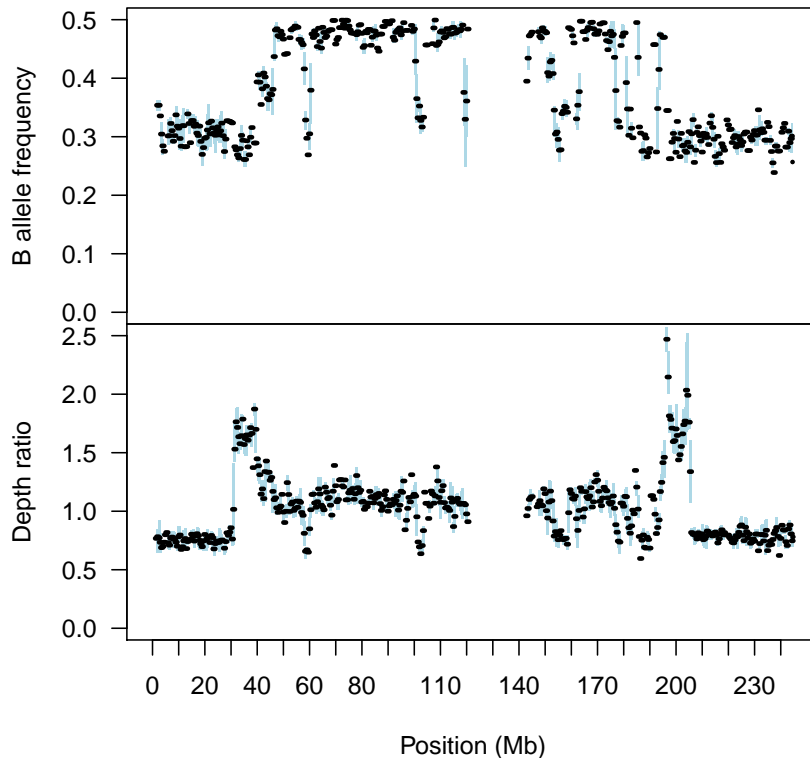


Figure 1: Plots B-allele frequencies (top) and un-logged-logR (bottom) with SNP array data.

## 1.5 Segmenting with the *copynumber* package

```
> breaks <- find.breaks(sample.het.i, gamma = 20, kmin = 15, baf.thres = c(0, 0.5))  
> seg.i <- segment.breaks(sample.i, breaks = breaks)
```

## 1.6 Using the Bayesian inference on segmented SNP arrays

```
> weights.snp <- 150 + round((seg.i$end.pos - seg.i$start.pos)/1e6, 0)  
> filter.size <- (seg.i$end.pos - seg.i$start.pos) >= 10e6  
> avg.unlogR <- mean(sample.i$adjusted.ratio, na.rm = TRUE)  
  
> CPsnp.example <- baf.model.fit(Bf = seg.i$Bf[filter.size],  
+                               depth.ratio = seg.i$depth.ratio[filter.size],  
+                               weight.ratio = weights.snp[filter.size],  
+                               weight.Bf = weights.snp[filter.size],  
+                               avg.depth.ratio = avg.unlogR,  
+                               cellularity = seq(0.1, 1, 0.01),
```

```

+                               ploidy = seq(1,7,0.1),
+                               priors.table = data.frame(CN = 2, value = 2))

> cint <- get.ci(CPsnp.example)
> cellularity <- cint$max.y
> ploidy <- cint$max.x

```

## 1.7 Cellularity and ploidy plot for SNP array

```

> cp.plot(CPsnp.example)
> cp.plot.contours(CPsnp.example, add = TRUE)

```

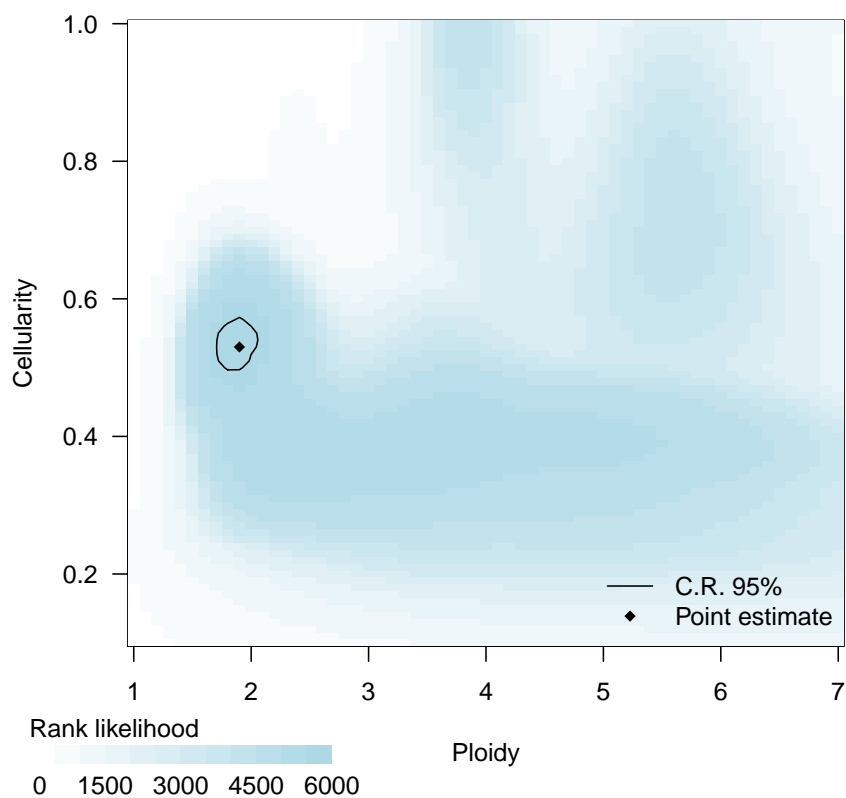


Figure 2: Result from the Bayesian inference over the defined range of cellularity and ploidy from artificial SNP array data. The color indicate the log-likelihood of the corresponding cellularity/-ploidy combinations.

## 1.8 Call for copy number variation using inferred parameters.

```

> snp.seg.cn <- baf.bayes(Bf = seg.i$Bf,
+                           depth.ratio = seg.i$depth.ratio,
+                           avg.depth.ratio = avg.unlogR,
+                           cellularity = cellularity,
+                           weight.ratio = 2 * 300,
+                           weight.Bf = 300, ratio.priority = FALSE,

```

```

+                               ploidy = ploidy, CNt.max = 10)
> segmented.snp <- cbind(seg.i, snp.seg.cn)
> head(segmented.snp[segmented.snp$chromosome == 1, ])

  chromosome start.pos  end.pos      Bf N.BAF depth.ratio N.ratio CNt A B
1          1   2189662 30490508 0.3080575   87   0.7538262   134  1 1 0
2          1   31697751 39213527 0.2817625   16   1.6662164    32  4 3 1
3          1   40285096 46296225 0.3786333   21   1.2857143    32  3 2 1
4          1   46437972 55282671 0.4791852   27   1.0502976    37  2 1 1
5          1   55913726 61908401 0.4126143   14   0.9321436    20  2 1 1
6          1   62012795 100351185 0.4781943   70   1.1108629   121  2 1 1

  L
1 -14.64082
2 -16.30625
3 -14.95819
4 -14.56076
5 -20.12817
6 -14.83072

```

## 1.9 Graphical representation of copy number with SNP arrays

```
> chromosome.view(baf.windows = snp.b.win[[1]],
+               ratio.windows = snp.r.win[[1]], min.N.ratio = 1,
+               segments = segmented.snp[segmented.snp$chromosome == "1", ],
+               cellularity = cellularity, ploidy = ploidy,
+               avg.depth.ratio = avg.unlogR, main = "1")
```

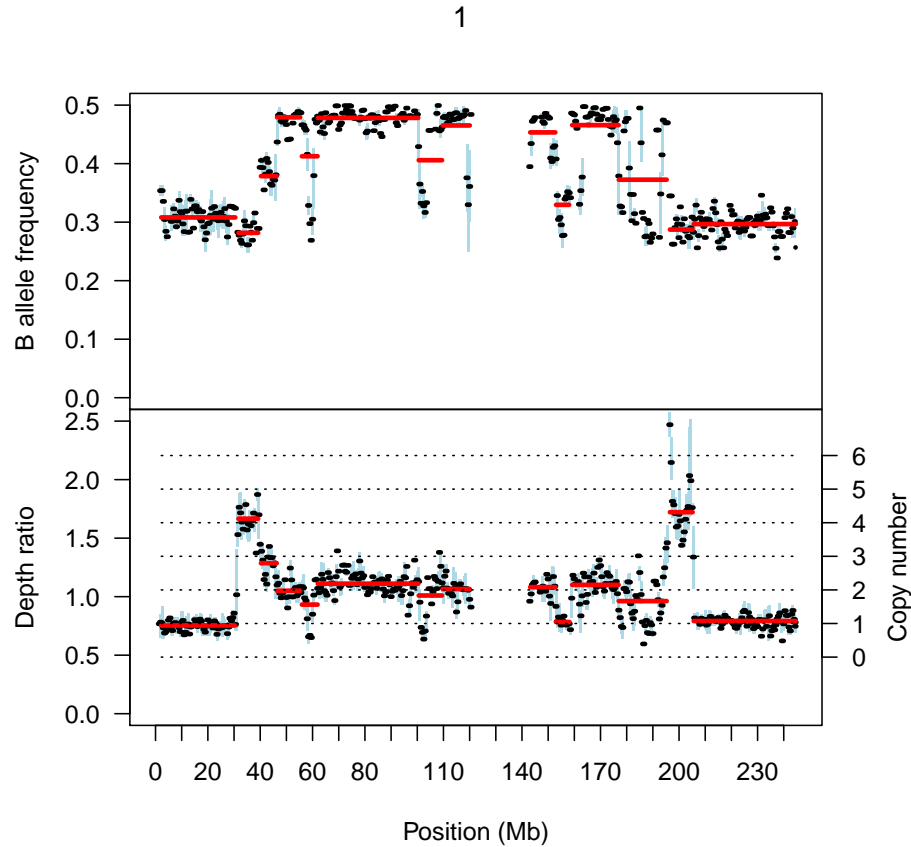


Figure 3: Plots B-allele frequencies (top) and un-logged-logR (bottom) with SNP array data. Chromosome 16. Horizontal dotted line indicate different copy number/ allele state.

```

> chromosome.view(baf.windows = snp.b.win[[16]],
+               ratio.windows = snp.r.win[[16]], min.N.ratio = 1,
+               segments = segmented.snp[segmented.snp$chromosome == "16", ],
+               cellularity = cellularity, ploidy = ploidy,
+               avg.depth.ratio = avg.unlogR, main = "16")

```

16

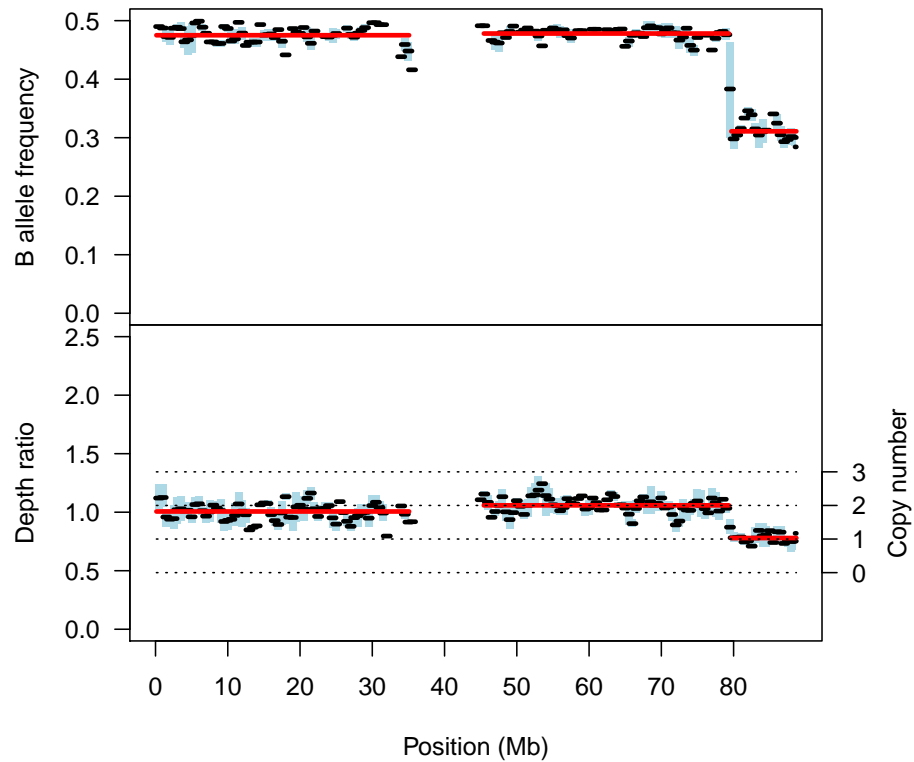


Figure 4: Plots B-allele frequencies (top) and un-logged-logR (bottom) with SNP array data. Chromosome 16. Horizontal dotted line indicate different copy number/ allele state.

```

> genome.view(seg.cn = segmented.snp, info.type = "CNt")
> legend("bottomright", bty="n", c("Tumor copy number"), col = c("red"),
+       inset = c(0, -0.4), pch=15, xpd = TRUE)

```

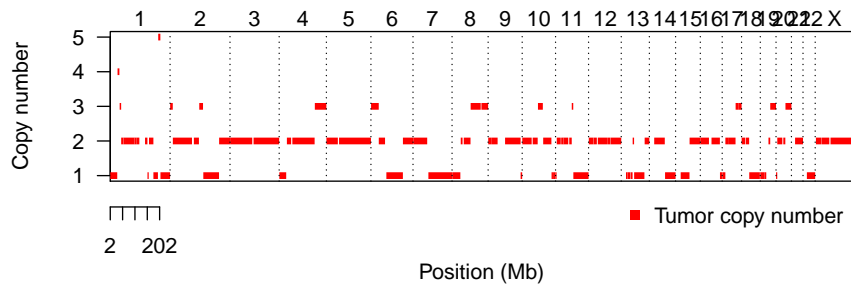


Figure 5: Genome wide copy number profile obtained from one SNP array.

```

> genome.view(seg.cn = segmented.snp, info.type = "AB")
> legend("bottomright", bty = "n", c("A-allele", "B-allele"), col= c("red", "blue"),
+       inset = c(0, -0.45), pch = 15, xpd = TRUE)

```

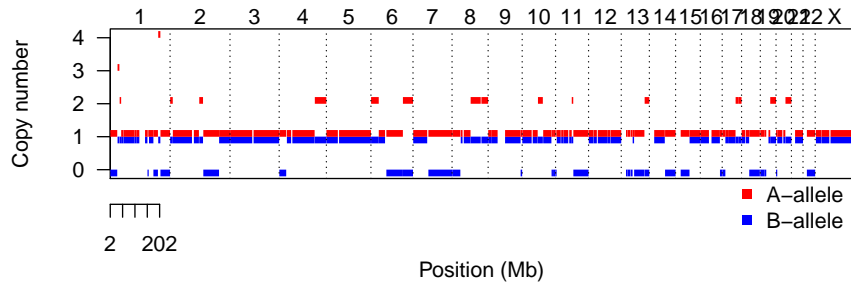


Figure 6: Genome wide A and B alleles profile, obtained from one SNP array.