MBASED Overview

May 3, 2016

1 MBASED algorithm

The package MBASED (**M**eta-analysis-**B**ased **A**llele-**S**pecific **E**xpression **D**etection) implements an algorithm for identifying instances of allele-specific gene expression in RNA count data. In one-sample analyses, MBASED identifies within-sample deviations from the null hypothesis of equal allele expression, while for two-sample analyses, significant differences in allelic ratios between samples with the same haplotypes are detected. This means that the comparison is meaningful for samples from the same individual, but not necessarily for samples from different individuals. MBASED uses principles of meta-analysis to combine information across loci within a single unit of expression (which we call `aseID`, e.g., a gene, a transcript, or an individual SNV), without relying on *a priori* knowledge of phased haplotypes. The informative loci are usually heterozygous SNVs, but could also be aggregations of SNVs or other variants that have been resolved into the same haplotype. In the extreme case of complete phasing, loci could be genes themselves, and the approach presented here reduces to a (beta-)binomial test of difference between reads mapping to different haplotypes. In the following discussion, we will use 'gene' in place of aseID and 'SNV' in place of 'locus', as these are the units we typically use in practice.

The details of the statistical approach implented by MBASED can be found in the accompanying paper. Here we briefly mention a few items pertaining to the interpretation of MBASED output.

1.1 1-Sample analysis

For each gene g with at least one expressed heterozygous locus, we test the null hypothesis $p_{hap1,g} = p_{hap2,g} = 0.5$, where $p_{hap1,g}$ and $p_{hap2,g}$ denote the underlying fractions of transcribed copies of gene g, specific to each haplotype. Note that we assume that gene g is represented by exactly two, and not more, haplotypes. The alternative hypothesis is $p_{hap1,g} \neq p_{hap2,g}$. The beta-binomial statistical model used by MBASED is flexible and allows ASE assessment in presence of pre-existing allelic bias (e.g., over-representation of reference allele-supporting reads due to enrichment protocols even in absence of any underlying ASE) and overdispersion. We show examples of this flexibility below, and the user is referred to the accompanying paper for the details of the algorithm used by MBASED. For each gene g the output of MBASED includes

1. Estimate of the underlying RNA major haplotype (allele) frequency (`majorAlleleFrequency`, MAF).
2. P-value for the test of null hypothesis of no ASE (`pValueASE`). Note that the null hypothesis can be reparametrized in terms of major haplotype frequency as $H_0 : MAF > 0.5$. The reported p-values are not adjusted for multiple hypothesis testing, and such adjustment should be carried out by the user when looking for statistical significance across multiple genes.
3. P-value for test of null hypothesis of no inter-SNV variability of ASE in the gene (only for genes with > 1 tested SNVs, `pValueHeterogeneity`). Small p-values correspond to significantly inconsistent ASE measurements at individual SNVs within a gene. If the unit of expression is a gene with multiple transcript isoforms, the observed inconsitency might indicate isoform-specific ASE. Alternatively, SNVs for such genes might exhibit background variability of ASE measures in excess of what is specified by the user (MBASED requires the user to specify the level of variability in the data, see below).

1.2 2-Sample analysis

For each gene g with at least one heterozygous locus expressed in both samples, we test the null hypothesis $p_{hap1,g,sample1} = p_{hap2,g,sample2} = p_{hap,1}$, where $p_{hap1,g,sample1}$ and $p_{hap1,g,sample1}$ denote the underlying fractions of transcribed copies of gene g, specific to haplotype 1 in each sample. No requirement is imposed that $p_{hap1,g,sample1} = 0.5$, since we are interested in differences in ASE pattern between the two samples, not in whether they both exhibit ASE. Just as in 1-sample analysis, we assume that gene g is represented by exactly two, and not more, haplotypes, and we further require that these haplotypes be the same in both samples. The alternative hypothesis is $p_{hap1,g,sample1} \neq p_{hap1,g,sample2}$. For each gene g the output of MBASED includes

1. Estimate of underlying major haplotype frequency difference, $p_{hap_{maj},g,sample1} - p_{hap_{maj},g,sample2}$, where hap_{maj} is the major of the two haplotypes in sample 1 (`majorAlleleFrequencyDifference`). This is our measure of between-sample allelic imblance (the 2-sample ASE). Note that assignment of haplotypes to 'major' and 'minor' is based exclusively on sample 1, since the 2-sample analysis is designed to look for ASE specific to sample 1. If ASE specific to sample 2 is of interest, the roles of the samples should be reversed.
2. P-value for the test of null hypothesis of no differential ASE (`pValueASE`). Analogous to 1-sample analysis, the reported p-values are not adjusted for multiple hypothesis testing, and such adjustment should be carried out by the user when looking for statistical significance across multiple genes.
3. P-value for test test of null hypothesis of no inter-SNV variability of ASE in the gene (only for genes with > 1 tested SNVs, `pValueHeterogeneity`). Small p-values correspond to significantly inconsistent 2-sample ASE measurements at individual SNVs within a gene. If the unit of expression is a gene with multiple transcript isoforms, the observed inconsitency might indicate isoform-specific allelic imbalance between the two samples.

2 Caveat Emptor

MBASED operates on read counts from heterozygous expressed SNVs. The onus is on the user to make sure that the supplied counts do in fact correspond to heterozygous SNVs. Any read counts coming from a homozygous SNV will by definition result in the detection of monoallelic expression. Further, it is up to the user to pre-filter SNVs for any known artifacts that might give rise to spurious ASE calls.

The p-values returned by MBASED are not adjusted for multiple hypothesis testing, and it is left to the user to choose the appropriate adjustment procedure and significance cutoff.

Further, to ensure the sufficient power for ASE detection, we strongly recommend that only SNVs with sufficient read coverage be retained for the analysis. What constitutes 'sufficient coverage' will depend on the multiple hypothesis testing adjustment procedure and cutoffs chosen by the user to employ on the p-values reported by MBASED. In our work, we required the depth of coverage of at least 10 reads, and higher cutoffs may be employed with more deeply sequenced samples.

Finally, while MBASED is designed to detect instances of statistically significant ASE, it is up to the user to decide what constitutes *biologically significant* ASE. Apropriate cutoffs on the reported measures of ASE (`majorAlleleFrequency` in case of 1-sample analysis, `majorAlleleFrequencyDifference` in case of 2-sample analysis) should then be employed to define the set of ASE genes for follow-up.

MBASED allows the user to provide information about pre-existing allelic bias and extra-binomial dispersion of read counts at individual loci (see examples later in the vignette), but currently does not provide explicit functionality for estimating these quantities.

See the accompanying paper for further discussion of these and other issues.

3 Examples

ASE assessment is performed by function runMBASED(). Please check the help page for a full list of arguments. Here we show some examples of its use. We start by discussing simple examples of both 1-sample and 2-sample analyses, and then show some examples of advanced usage, including adjusting for pre-existing allelic bias and extra-binomial dispersion in the data.

3.1 Input/Output format

The input and output of runMBASED() function are RangedSummarizedExperiment objects, defined in R package 'SummarizedExperiment'. Please see the RangedSummarizedExperiment help page for details. Briefly, the input object has rows corresponding to individual SNVs and columns corresponding to samples. In 1-sample analysis, there is a single column. In 2-sample analysis, there are 2 columns, one for each sample, with first column corresponding to 'sample1' and second column corresponding to 'sample2' in the language of section 1.2. The output is also a RangedSummarizedExperiment object, with rows corresponding to individual aseIDs and a single column. MBASED output at the level of individual SNV is stored as metadata on the output object. The examples below will illustrate the usage.

3.2 1-sample analysis

Suppose we are looking at 2 genes, one of which has 1 SNV and the other has 3 SNVs:

locus	location	refAllele	altAllele	refCount	altCount
gene1:SNV1	chr1:100	G	T	25	20
gene2:SNV1	chr2:1000	A	C	10	16
gene2:SNV2	chr2:1100	C	T	22	15
gene2:SNV3	chr2:1200	A	G	14	16

Table 1: 1-Sample example

If we know the true underlying haplotypes of the gene, we can run MBASED as follows (assuming all reference alleles are on the same haplotype):

```
> set.seed(988482)
> library(MBASED)
> ##a quick look at the main function
> args(runMBASED)
```

```
function (ASESummarizedExperiment, isPhased = FALSE, numSim = 0,
    BPPARAM = SerialParam())
NULL
```

```
> ## create GRanges object for SNVs of interest.
> ## note: the only required column is 'aseID'
> mySNVs <- GRanges(
+   seqnames=c('chr1', 'chr2', 'chr2', 'chr2'),
+   ranges=IRanges(start=c(100, 1000, 1100, 1200), width=1),
+   aseID=c('gene1', rep('gene2', 3)),
+   allele1=c('G', 'A', 'C', 'A'),
+   allele2=c('T', 'C', 'T', 'G')
+ )
> names(mySNVs) <- c('gene1_SNV1', 'gene2_SNV1', 'gene2_SNV2', 'gene2_SNV3')
> ## create input RangedSummarizedExperiment object
> mySample <- SummarizedExperiment(
```

```
+   assays=list(
+     lociAllele1Counts=matrix(
+       c(25, 10, 22, 14),
+       ncol=1,
+       dimnames=list(
+        names(mySNVs),
+        'mySample'
+       )
+     ),
+     lociAllele2Counts=matrix(
+       c(20,16,15,16),
+       ncol=1,
+       dimnames=list(
+        names(mySNVs),
+        'mySample'
+       )
+     )
+   ),
+   rowRanges=mySNVs
+ )
> ##example of use
> ASEresults_1s_haplotypesKnown <- runMBASED(
+   ASESummarizedExperiment=mySample,
+   isPhased=TRUE,
+   numSim=10^6,
+   BPPARAM = SerialParam()
+ )
> ## explore the return object
> class(ASEresults_1s_haplotypesKnown)

[1] "RangedSummarizedExperiment"
attr(,"package")
[1] "SummarizedExperiment"

> names(assays(ASEresults_1s_haplotypesKnown))

[1] "majorAlleleFrequency" "pValueASE"            "pValueHeterogeneity"

> assays(ASEresults_1s_haplotypesKnown)$majorAlleleFrequency

       mySample
gene1 0.5555556
gene2 0.5052529

> assays(ASEresults_1s_haplotypesKnown)$pValueASE

      mySample
gene1 0.551513
gene2 0.982877

> assays(ASEresults_1s_haplotypesKnown)$pValueHeterogeneity

      mySample
gene1       NA
gene2 0.244765

> rowRanges(ASEresults_1s_haplotypesKnown)

GRangesList object of length 2:
$gene1
```

```
GRanges object with 1 range and 3 metadata columns:
             seqnames     ranges strand |       aseID     allele1     allele2
                <Rle>  <IRanges>  <Rle> | <character> <character> <character>
  gene1_SNV1     chr1 [100, 100]      * |       gene1           G           T

$gene2
GRanges object with 3 ranges and 3 metadata columns:
             seqnames       ranges strand | aseID allele1 allele2
  gene2_SNV1     chr2 [1000, 1000]      * | gene2       A       C
  gene2_SNV2     chr2 [1100, 1100]      * | gene2       C       T
  gene2_SNV3     chr2 [1200, 1200]      * | gene2       A       G

-------
seqinfo: 2 sequences from an unspecified genome; no seqlengths
```

```
> names(metadata(ASEresults_1s_haplotypesKnown))
```

```
[1] "locusSpecificResults"
```

```
> class(metadata(ASEresults_1s_haplotypesKnown)$locusSpecificResults)
```

```
[1] "RangedSummarizedExperiment"
attr(,"package")
[1] "SummarizedExperiment"
```

```
> names(assays(metadata(ASEresults_1s_haplotypesKnown)$locusSpecificResults))
```

```
[1] "allele1IsMajor" "MAF"
```

```
> assays(metadata(ASEresults_1s_haplotypesKnown)$locusSpecificResults)$allele1IsMajor
```

```
           mySample
gene1_SNV1     TRUE
gene2_SNV1    FALSE
gene2_SNV2    FALSE
gene2_SNV3    FALSE
```

```
> assays(metadata(ASEresults_1s_haplotypesKnown)$locusSpecificResults)$MAF
```

```
            mySample
gene1_SNV1 0.5555556
gene2_SNV1 0.6153846
gene2_SNV2 0.4054054
gene2_SNV3 0.5333333
```

```
> rowRanges(metadata(ASEresults_1s_haplotypesKnown)$locusSpecificResults)
```

```
GRanges object with 4 ranges and 3 metadata columns:
             seqnames       ranges strand |       aseID     allele1     allele2
                <Rle>    <IRanges>  <Rle> | <character> <character> <character>
  gene1_SNV1     chr1 [ 100,  100]      * |       gene1           G           T
  gene2_SNV1     chr2 [1000, 1000]      * |       gene2           A           C
  gene2_SNV2     chr2 [1100, 1100]      * |       gene2           C           T
  gene2_SNV3     chr2 [1200, 1200]      * |       gene2           A           G
  -------
  seqinfo: 2 sequences from an unspecified genome; no seqlengths
```

```
> ## define function to print out the summary of ASE results
> summarizeASEResults_1s <- function(MBASEDOutput) {
+   geneOutputDF <- data.frame(
+    majorAlleleFrequency=assays(MBASEDOutput)$majorAlleleFrequency[,1],
```

```
+    pValueASE=assays(MBASEDOutput)$pValueASE[,1],
+    pValueHeterogeneity=assays(MBASEDOutput)$pValueHeterogeneity[,1]
+   )
+   lociOutputGR <- rowRanges(metadata(MBASEDOutput)$locusSpecificResults)
+   lociOutputGR$allele1IsMajor <- assays(metadata(MBASEDOutput)$locusSpecificResults)$allele1IsMajor[,1]
+   lociOutputGR$MAF <- assays(metadata(MBASEDOutput)$locusSpecificResults)$MAF[,1]
+   lociOutputList <- split(lociOutputGR, factor(lociOutputGR$aseID, levels=unique(lociOutputGR$aseID)))
+   return(
+    list(
+      geneOutput=geneOutputDF,
+      locusOutput=lociOutputList
+    )
+   )
+ }
> summarizeASEResults_1s(ASEresults_1s_haplotypesKnown)
$geneOutput
      majorAlleleFrequency pValueASE pValueHeterogeneity
gene1            0.5555556  0.551513                  NA
gene2            0.5052529  0.982877            0.244765

$locusOutput
GRangesList object of length 2:
$gene1
GRanges object with 1 range and 5 metadata columns:
             seqnames     ranges strand |       aseID     allele1     allele2
                <Rle>  <IRanges>  <Rle> | <character> <character> <character>
  gene1_SNV1     chr1 [100, 100]      * |       gene1           G           T
             allele1IsMajor       MAF
                  <logical> <numeric>
  gene1_SNV1           TRUE 0.5555556

$gene2
GRanges object with 3 ranges and 5 metadata columns:
             seqnames       ranges strand | aseID allele1 allele2 allele1IsMajor
  gene2_SNV1     chr2 [1000, 1000]      * | gene2       A       C          FALSE
  gene2_SNV2     chr2 [1100, 1100]      * | gene2       C       T          FALSE
  gene2_SNV3     chr2 [1200, 1200]      * | gene2       A       G          FALSE
                   MAF
  gene2_SNV1 0.6153846
  gene2_SNV2 0.4054054
  gene2_SNV3 0.5333333

-------
seqinfo: 2 sequences from an unspecified genome; no seqlengths
```

The output of `runMBASED()` is a RangedSummarizedExperiment object with 3 single-column assay matrices. The rows of the matrices correspond to individual aseIDs, in our case 'gene1' and 'gene2'. The matrices are majorAlleleFrequency (MBASED estimate of gene-level MAF), pValueASE (the coresponding p-value), and pValueHeterogeneity (the inter-SNV variability p-values). Note that heterogeneity p-value is `NA` for gene1, since this gene only has 1 tested SNV. The metadata slot of the return object contains a single element, 'locusSpecificResults', which is a RangedSummarizedExperiment object with information on MBASED results at the level of individual SNV. The assay matrices specify whether at a given locus allele1 belongs to the major (gene-level) haplotype ('allele1IsMajor'), and what the allele frequency is at that SNV of the allele belonging to the major (gene-level) haplotype ('MAF'). Since assignment of 'major' status to a haplotypes is based on gene-wide information, there may be cases when the locus-specific MAF (frequency of 'major' gene haplotype at the

locus) is < 0.5, as is the case for the second locus of gene2 in the previous example.

When true haplotypes are unknown, MBASED assigns alleles to 'major' and 'minor' haplotypes based on observed read counts, assigning the allele with higher than expected (under conditions of no ASE) read counts to the same 'major' haplotype. The following example illustrates this approach using the same data as before, by setting argument 'isPhased' to FALSE:

```
> summarizeASEResults_1s(
+   runMBASED(
+     ASESummarizedExperiment=mySample,
+     isPhased=FALSE,
+     numSim=10^6,
+     BPPARAM = SerialParam()
+   )
+ )
```

```
$geneOutput
      majorAlleleFrequency pValueASE pValueHeterogeneity
gene1            0.5555556  0.551245                  NA
gene2            0.5807511  0.380807             0.53758

$locusOutput
GRangesList object of length 2:
$gene1
GRanges object with 1 range and 5 metadata columns:
             seqnames     ranges strand |       aseID     allele1     allele2
                <Rle>  <IRanges>  <Rle> | <character> <character> <character>
  gene1_SNV1     chr1 [100, 100]      * |       gene1           G           T
             allele1IsMajor       MAF
                  <logical> <numeric>
  gene1_SNV1           TRUE 0.5555556

$gene2
GRanges object with 3 ranges and 5 metadata columns:
             seqnames       ranges strand | aseID allele1 allele2 allele1IsMajor
  gene2_SNV1     chr2 [1000, 1000]      * | gene2       A       C          FALSE
  gene2_SNV2     chr2 [1100, 1100]      * | gene2       C       T           TRUE
  gene2_SNV3     chr2 [1200, 1200]      * | gene2       A       G          FALSE
                   MAF
  gene2_SNV1 0.6153846
  gene2_SNV2 0.5945946
  gene2_SNV3 0.5333333

-------
seqinfo: 2 sequences from an unspecified genome; no seqlengths
```

Note that the pseudo-phasing based'major' haplotype of gene2 consists of the reference allele at SNV2 and alternative alleles at SNV1 and SNV3. Also note that by design, the gene-level estimate of MAF for the multi-locus gene2 is higher in the case of unphased data (0.58 vs. 0.505) and the SNV-level estimates are all ≥ 0.5. We rely on properly calibrated p-values to prevent these higher estimates from resulting in spurious ASE calls due to pseudo-phasing. This is accomplished by employing simulations (argument `numSim`) to obtain an approximation to null distribution of MAF estimates. We used 10^6 simulations in the previous example and we recommend at least this many simulations in practice. If parallel architecture is available (see argument `BPPARAM` and documentation for function `bplapply` in `R` package `BiocParallel`), it can be employed when many genes are tested simulatneously (parallelization is done over *genes (aseIDs)*, and not *simulations*).

To illustrate the need for simulations, we show how the results are affected by bias introduced during the pseudo-phasing

step:

```
> ##re-run analysis without simulations
> summarizeASEResults_1s(
+   runMBASED(
+     ASESummarizedExperiment=mySample,
+     isPhased=FALSE,
+     numSim=0,
+     BPPARAM = SerialParam()
+   )
+ )$geneOutput
```

```
      majorAlleleFrequency pValueASE pValueHeterogeneity
gene1            0.5555556 0.4624490                   NA
gene2            0.5807511 0.1262776            0.8144809
```

```
>
```

Notice the 3-fold decrease in pValueASE for gene2 when we skipped the simulations. See the accompanying paper for further discussion of the internal simulations used by MBASED.

The use of parallelization is illustrated in the example below (not evaluated):

```
> ##re-run analysis while parallelizing computations
> ## results are same as before
> ## with some random flactuations due to simulations
> summarizeASEResults_1s(
+   runMBASED(
+     ASESummarizedExperiment=mySample,
+     isPhased=FALSE,
+     numSim=10^6,
+     BPPARAM = MulticoreParam()
+   )
+ )$geneOutput
> ## Number of seconds it takes to run results without parallelizing:
> system.time(runMBASED(
+   ASESummarizedExperiment=mySample,
+   isPhased=FALSE,
+   numSim=10^6,
+   BPPARAM = SerialParam()
+ ))['elapsed'] ## ~ 15 sec on our machine
> ## Number of seconds it takes to run results with parallelizing:
> system.time(runMBASED(
+  ASESummarizedExperiment=mySample,
+   isPhased=FALSE,
+   numSim=10^6,
+   BPPARAM = MulticoreParam()
+ ))['elapsed'] ## ~9 sec on our machine
>
```

We also note that for a single-locus gene1, the phasing plays no role, and the simulation-based p-values are very close to each other (there's some variability due to random nature of simulations) and are approximately equal to the analytical p-value based on the binomial test:

```
> binom.test(25, 45, 0.5, 'two.sided')$p.value
```

```
[1] 0.5514843
```

Finally, let's consider a case of isoform-specific ASE. Suppose that the 3 SNVs in gene2 correspond to 3 different exons,

and that 2 transcript isoforms of gene 2 exist: isoform1 uses all 3 exons, and isoform2 uses only the last 2 exons. Further, let us assume that

- both isoforms are present in 50/50 ratio
- isoform1 shows no ASE
- isoform2 shows complete silencing of one of the alleles (monoallelic expression, an extreme form of ASE).

Here is an example of data one could observe under this toy scenario:

locus	location	refCount	altCount
gene2:SNV1	chr2:1000	23	26
gene2:SNV2	chr2:1100	65	25
gene2:SNV3	chr2:1200	30	70

Table 2: Example of isoform-specific ASE

Notice that SNV1 shows lower coverage than the other 2 SNVs, since it is only expressed by isoform1. Also note that alternative allele at SNV2 and reference allele at SNV3 are silenced in isoform2. Overall, each isoform is represented by approximately 50 reads at each SNV that is a part of that isoform.

```
> isoSpecificExampleSNVs <- mySNVs[2:4,]
> ## create input RangedSummarizedExperiment object
> isoSpecificExample <- SummarizedExperiment(
+   assays=list(
+     lociAllele1Counts=matrix(
+       c(23, 65, 30),
+       ncol=1,
+       dimnames=list(
+        names(isoSpecificExampleSNVs),
+        'mySample'
+       )
+     ),
+     lociAllele2Counts=matrix(
+       c(26,25,70),
+       ncol=1,
+       dimnames=list(
+        names(isoSpecificExampleSNVs),
+        'mySample'
+       )
+     )
+   ),
+   rowRanges=isoSpecificExampleSNVs
+ )
> summarizeASEResults_1s(
+   runMBASED(
+     ASESummarizedExperiment=isoSpecificExample,
+     isPhased=FALSE,
+     numSim=10^6,
+     BPPARAM = MulticoreParam()
+   )
+ )

$geneOutput
  majorAlleleFrequency pValueASE pValueHeterogeneity
1            0.6752393         0            0.003482
```

```
$locusOutput
GRangesList object of length 1:
$gene2
GRanges object with 3 ranges and 5 metadata columns:
             seqnames       ranges strand |       aseID     allele1     allele2
                <Rle>    <IRanges>  <Rle> | <character> <character> <character>
  gene2_SNV1     chr2 [1000, 1000]      * |       gene2           A           C
  gene2_SNV2     chr2 [1100, 1100]      * |       gene2           C           T
  gene2_SNV3     chr2 [1200, 1200]      * |       gene2           A           G
             allele1IsMajor       MAF
                  <logical> <numeric>
  gene2_SNV1          FALSE 0.5306122
  gene2_SNV2           TRUE 0.7222222
  gene2_SNV3          FALSE 0.7000000

-------
seqinfo: 2 sequences from an unspecified genome; no seqlengths
>
```

We see that MBASED reports major allele frequency of 0.675 for this gene. Since the true MAF is 0.5 for isoform1 and 1.0 for isoform2, the reproted value represents an average of sorts. The small value of heterogeneity p-value (0.003) indicates that there is strong evidence that ASE estimates are inconsistent across individual SNVs and that the reported MAF value might be misleading (it is an aggregate of isoform-specific values). Also note that the reported ASE p-value is 0. In practice, given the restricitions imposed by the number of simulations, this means that we estimate $p_{ASE} \leq \frac{1}{N_{sim}}$. In this instance, we should interpret the outcome as p-value $< 10^{-6}$.

3.3 2-sample analysis

Next, we show an example of 2-sample ASE analysis using MBASED. Suppose we want to identify instances of tumor-specifc ASE and observe the following data:

locus	location	refCountTumor	altCountTumor	refCountNormal	altCountNormal
gene1:SNV1	chr1:100	25	20	18	23
gene2:SNV1	chr2:1000	10	29	17	19
gene2:SNV2	chr2:1100	35	15	21	24
gene2:SNV3	chr2:1200	14	40	25	31
gene3:SNV1	chr3:2000	35	9	40	10

Table 3: 2-Sample example

Visual examination reveals that gene1 doesn't exhibit ASE in either sample, gene2 shows signs of ASE in tumor sample only, and gene3 shows signs of ASE in both samples.

```
> mySNVs_2s <- GRanges(
+   seqnames=c('chr1', 'chr2', 'chr2', 'chr2', 'chr3'),
+   ranges=IRanges(start=c(100, 1000, 1100, 1200, 2000), width=1),
+   aseID=c('gene1', rep('gene2', 3), 'gene3'),
+   allele1=c('G', 'A', 'C', 'A', 'T'),
+   allele2=c('T', 'C', 'T', 'G', 'G')
+ )
> names(mySNVs_2s) <- c('gene1_SNV1', 'gene2_SNV1', 'gene2_SNV2', 'gene2_SNV3', 'gene3_SNV1')
> ## create input RangedSummarizedExperiment object
> myTumorNormalExample <- SummarizedExperiment(
+   assays=list(
```

```
+     lociAllele1Counts=matrix(
+       c(
+         c(25,10,35,14,35),
+         c(18,17,21,25,40)
+       ),
+       ncol=2,
+       dimnames=list(
+        names(mySNVs_2s),
+        c('tumor', 'normal')
+       )
+     ),
+     lociAllele2Counts=matrix(
+       c(
+         c(20,29,15,40,9),
+         c(23,19,24,31,10)
+       ),
+       ncol=2,
+       dimnames=list(
+        names(mySNVs_2s),
+        c('tumor', 'normal')
+       )
+     )
+   ),
+   rowRanges=mySNVs_2s
+ )
> ##example of use
> ASEresults_2s <- runMBASED(
+   ASESummarizedExperiment=myTumorNormalExample,
+   isPhased=FALSE,
+   numSim=10^6,
+   BPPARAM = SerialParam()
+ )
> ## explore the return object
> class(ASEresults_2s)

[1] "RangedSummarizedExperiment"
attr(,"package")
[1] "SummarizedExperiment"

> names(assays(ASEresults_2s))

[1] "majorAlleleFrequencyDifference" "pValueASE"
[3] "pValueHeterogeneity"

> assays(ASEresults_2s)$majorAlleleFrequencyDifference

      tumor/normal
gene1  0.116531165
gene2  0.210128106
gene3 -0.004545455

> assays(ASEresults_2s)$pValueASE

      tumor/normal
gene1     0.266860
gene2     0.000444
gene3     0.535343
```

```
> assays(ASEresults_2s)$pValueHeterogeneity
```

```
      tumor/normal
gene1           NA
gene2     0.940698
gene3           NA
```

```
> rowRanges(ASEresults_2s)
```

```
GRangesList object of length 3:
$gene1
GRanges object with 1 range and 3 metadata columns:
             seqnames     ranges strand |       aseID     allele1     allele2
                <Rle>  <IRanges>  <Rle> | <character> <character> <character>
  gene1_SNV1     chr1 [100, 100]      * |       gene1           G           T

$gene2
GRanges object with 3 ranges and 3 metadata columns:
             seqnames       ranges strand | aseID allele1 allele2
  gene2_SNV1     chr2 [1000, 1000]      * | gene2       A       C
  gene2_SNV2     chr2 [1100, 1100]      * | gene2       C       T
  gene2_SNV3     chr2 [1200, 1200]      * | gene2       A       G

$gene3
GRanges object with 1 range and 3 metadata columns:
             seqnames       ranges strand | aseID allele1 allele2
  gene3_SNV1     chr3 [2000, 2000]      * | gene3       T       G

-------
seqinfo: 3 sequences from an unspecified genome; no seqlengths
```

```
> names(metadata(ASEresults_2s))
```

```
[1] "locusSpecificResults"
```

```
> class(metadata(ASEresults_2s)$locusSpecificResults)
```

```
[1] "RangedSummarizedExperiment"
attr(,"package")
[1] "SummarizedExperiment"
```

```
> names(assays(metadata(ASEresults_2s)$locusSpecificResults))
```

```
[1] "allele1IsMajor" "MAFDifference"
```

```
> assays(metadata(ASEresults_2s)$locusSpecificResults)$allele1IsMajor
```

```
           tumor/normal
gene1_SNV1         TRUE
gene2_SNV1        FALSE
gene2_SNV2         TRUE
gene2_SNV3        FALSE
gene3_SNV1         TRUE
```

```
> assays(metadata(ASEresults_2s)$locusSpecificResults)$MAFDifference
```

```
           tumor/normal
gene1_SNV1  0.116531165
gene2_SNV1  0.215811966
gene2_SNV2  0.233333333
```

```
gene2_SNV3  0.187169312
gene3_SNV1 -0.004545455
```

```
> rowRanges(metadata(ASEresults_2s)$locusSpecificResults)
```

```
GRanges object with 5 ranges and 3 metadata columns:
             seqnames       ranges strand |       aseID     allele1     allele2
                <Rle>    <IRanges>  <Rle> | <character> <character> <character>
  gene1_SNV1     chr1 [ 100,  100]      * |       gene1           G           T
  gene2_SNV1     chr2 [1000, 1000]      * |       gene2           A           C
  gene2_SNV2     chr2 [1100, 1100]      * |       gene2           C           T
  gene2_SNV3     chr2 [1200, 1200]      * |       gene2           A           G
  gene3_SNV1     chr3 [2000, 2000]      * |       gene3           T           G
  -------
  seqinfo: 3 sequences from an unspecified genome; no seqlengths
```

```
> ## define function to print out the summary of ASE results
> summarizeASEResults_2s <- function(MBASEDOutput) {
+   geneOutputDF <- data.frame(
+    majorAlleleFrequencyDifference=assays(MBASEDOutput)$majorAlleleFrequencyDifference[,1],
+    pValueASE=assays(MBASEDOutput)$pValueASE[,1],
+    pValueHeterogeneity=assays(MBASEDOutput)$pValueHeterogeneity[,1]
+   )
+   lociOutputGR <- rowRanges(metadata(MBASEDOutput)$locusSpecificResults)
+   lociOutputGR$allele1IsMajor <- assays(metadata(MBASEDOutput)$locusSpecificResults)$allele1IsMajor[,1]
+   lociOutputGR$MAFDifference <- assays(metadata(MBASEDOutput)$locusSpecificResults)$MAFDifference[,1]
+   lociOutputList <- split(lociOutputGR, factor(lociOutputGR$aseID, levels=unique(lociOutputGR$aseID)))
+   return(
+    list(
+      geneOutput=geneOutputDF,
+      locusOutput=lociOutputList
+    )
+   )
+ }
> summarizeASEResults_2s(ASEresults_2s)
```

```
$geneOutput
      majorAlleleFrequencyDifference pValueASE pValueHeterogeneity
gene1                    0.116531165  0.266860                  NA
gene2                    0.210128106  0.000444            0.940698
gene3                   -0.004545455  0.535343                  NA

$locusOutput
GRangesList object of length 3:
$gene1
GRanges object with 1 range and 5 metadata columns:
             seqnames     ranges strand |       aseID     allele1     allele2
                <Rle>  <IRanges>  <Rle> | <character> <character> <character>
  gene1_SNV1     chr1 [100, 100]      * |       gene1           G           T
             allele1IsMajor MAFDifference
                  <logical>     <numeric>
  gene1_SNV1           TRUE     0.1165312

$gene2
GRanges object with 3 ranges and 5 metadata columns:
             seqnames       ranges strand | aseID allele1 allele2 allele1IsMajor
```

```
  gene2_SNV1     chr2 [1000, 1000]      * | gene2       A       C          FALSE
  gene2_SNV2     chr2 [1100, 1100]      * | gene2       C       T           TRUE
  gene2_SNV3     chr2 [1200, 1200]      * | gene2       A       G          FALSE
             MAFDifference
  gene2_SNV1     0.2158120
  gene2_SNV2     0.2333333
  gene2_SNV3     0.1871693

$gene3
GRanges object with 1 range and 5 metadata columns:
             seqnames       ranges strand | aseID allele1 allele2 allele1IsMajor
  gene3_SNV1     chr3 [2000, 2000]      * | gene3       T       G           TRUE
             MAFDifference
  gene3_SNV1  -0.004545455

-------
seqinfo: 3 sequences from an unspecified genome; no seqlengths
```

>

Analogous to 1-sample analysis, the output of `runMBASED()` for 2-sample analysis is a RangedSummarizedExperiment object with 3 single-column assay matrices. The matrices are are similar to those in 1-sample case, except our measure of ASE is the difference in major allele frequency between the two samples. For example, for a single-SNV gene1, this difference is $\frac{25}{25+20} - \frac{18}{18+23} = 0.1165$. Notice that which allele is 'major' and which is 'minor' is determined entirely by read counts in the first sample (tumor, in this case).

We would like to emphasize that the 2-sample MBASED algorithm is designed to find instances of sample1-specific ASE (whatever sample1 may be) and is *not* symmetric with respect to the 2 samples. Therefore, if we were interested in cases of normal-specific ASE (e.g., loss of imprinting in cancer), we would re-run MBASED using normal sample as sample1 and tumor sample as sample2. The following examples illustrate this important point.

```
> ## single-SNV gene with strong ASE in tumor (25 vs. 5 reads)
> ## and no ASE in normal (22 vs. 23 reads):
> mySNV <- GRanges(
+   seqnames=c('chr1'),
+   ranges=IRanges(start=c(100), width=1),
+   aseID=c('gene1'),
+   allele1=c('G'),
+   allele2=c('T')
+ )
> names(mySNV) <- c('gene1_SNV1')
> summarizeASEResults_2s(
+   runMBASED(
+     ASESummarizedExperiment=SummarizedExperiment(
+       assays=list(
+         lociAllele1Counts=matrix(
+           c(
+             c(25),
+             c(22)
+           ),
+           ncol=2,
+           dimnames=list(
+            names(mySNV),
+            c('tumor', 'normal')
+           )
+         ),
```

```
+         lociAllele2Counts=matrix(
+           c(
+             c(5),
+             c(23)
+           ),
+           ncol=2,
+           dimnames=list(
+            names(mySNV),
+            c('tumor', 'normal')
+           )
+         )
+       ),
+       rowRanges=mySNV
+     ),
+     isPhased=FALSE,
+     numSim=10^6,
+     BPPARAM=SerialParam()
+   )
+ )
$geneOutput
  majorAlleleFrequencyDifference pValueASE pValueHeterogeneity
1                      0.3444444   0.00262                  NA

$locusOutput
GRangesList object of length 1:
$gene1
GRanges object with 1 range and 5 metadata columns:
             seqnames     ranges strand |       aseID     allele1     allele2
                <Rle>  <IRanges>  <Rle> | <character> <character> <character>
  gene1_SNV1     chr1 [100, 100]      * |       gene1           G           T
             allele1IsMajor MAFDifference
                  <logical>     <numeric>
  gene1_SNV1           TRUE     0.3444444

-------
seqinfo: 1 sequence from an unspecified genome; no seqlengths
> ## MBASED detects highly significant ASE.
> ##
> ## Now, suppose that the normal sample had the two allele counts
> ## exchanged (due to chance variation), :
> summarizeASEResults_2s(
+   runMBASED(
+     ASESummarizedExperiment=SummarizedExperiment(
+       assays=list(
+         lociAllele1Counts=matrix(
+           c(
+             c(25),
+             c(23) ## used to be 22
+           ),
+           ncol=2,
+           dimnames=list(
+            names(mySNV),
+            c('tumor', 'normal')
```

```
+           )
+         ),
+         lociAllele2Counts=matrix(
+           c(
+             c(5),
+             c(22) ## used to be 23
+           ),
+           ncol=2,
+           dimnames=list(
+            names(mySNV),
+            c('tumor', 'normal')
+           )
+         )
+       ),
+       rowRanges=mySNV
+     ),
+     isPhased=FALSE,
+     numSim=10^6,
+     BPPARAM=SerialParam()
+   )
+ )
$geneOutput
  majorAlleleFrequencyDifference pValueASE pValueHeterogeneity
1                      0.3222222  0.004537                  NA

$locusOutput
GRangesList object of length 1:
$gene1
GRanges object with 1 range and 5 metadata columns:
             seqnames     ranges strand |       aseID     allele1     allele2
                <Rle>  <IRanges>  <Rle> | <character> <character> <character>
  gene1_SNV1     chr1 [100, 100]      * |       gene1           G           T
             allele1IsMajor MAFDifference
                  <logical>     <numeric>
  gene1_SNV1           TRUE     0.3222222

-------
seqinfo: 1 sequence from an unspecified genome; no seqlengths
> ## We get virtually the same results
> ##
> ## Now, suppose we treated normal as sample1 and tumor as sample2
> ## Let's use original normal sample allele counts
> summarizeASEResults_2s(
+   runMBASED(
+     ASESummarizedExperiment=SummarizedExperiment(
+       assays=list(
+         lociAllele1Counts=matrix(
+           c(
+             c(22),
+             c(25)
+           ),
+           ncol=2,
+           dimnames=list(
```

```
+            names(mySNV),
+            c('normal', 'tumor')
+           )
+         ),
+         lociAllele2Counts=matrix(
+           c(
+             c(23),
+             c(5)
+           ),
+           ncol=2,
+           dimnames=list(
+            names(mySNV),
+            c('normal', 'tumor')
+           )
+         )
+       ),
+       rowRanges=mySNV
+     ),
+     isPhased=FALSE,
+     numSim=10^6,
+     BPPARAM=SerialParam()
+   )
+ )
$geneOutput
  majorAlleleFrequencyDifference pValueASE pValueHeterogeneity
1                      0.3444444  0.002414                  NA

$locusOutput
GRangesList object of length 1:
$gene1
GRanges object with 1 range and 5 metadata columns:
             seqnames     ranges strand |       aseID     allele1     allele2
                <Rle>  <IRanges>  <Rle> | <character> <character> <character>
  gene1_SNV1     chr1 [100, 100]      * |       gene1           G           T
             allele1IsMajor MAFDifference
                  <logical>     <numeric>
  gene1_SNV1          FALSE     0.3444444

-------
seqinfo: 1 sequence from an unspecified genome; no seqlengths
> ## We appear to have recovered the same result: highly significant MAF difference
> ## (but note that allele2 is now 'major', since allele classification into
> ## major/minor is based entirely on sample1)
> ## BUT: consider what happens if by chance the allelic
> ## ratio in the normal was 23/22 instead of 22/23:
> summarizeASEResults_2s(
+   runMBASED(
+     ASESummarizedExperiment=SummarizedExperiment(
+       assays=list(
+         lociAllele1Counts=matrix(
+           c(
+             c(23), ## used to be 22
+             c(25)
```

```
+           ),
+           ncol=2,
+           dimnames=list(
+            names(mySNV),
+            c('normal', 'tumor')
+           )
+         ),
+         lociAllele2Counts=matrix(
+           c(
+             c(22), ## used to be 23
+             c(5)
+           ),
+           ncol=2,
+           dimnames=list(
+            names(mySNV),
+            c('normal', 'tumor')
+           )
+         )
+       ),
+       rowRanges=mySNV
+     ),
+     isPhased=FALSE,
+     numSim=10^6,
+     BPPARAM=SerialParam()
+   )
+ )
$geneOutput
  majorAlleleFrequencyDifference pValueASE pValueHeterogeneity
1                     -0.3222222  0.999513                  NA

$locusOutput
GRangesList object of length 1:
$gene1
GRanges object with 1 range and 5 metadata columns:
             seqnames     ranges strand |       aseID     allele1     allele2
                <Rle>  <IRanges>  <Rle> | <character> <character> <character>
  gene1_SNV1     chr1 [100, 100]      * |       gene1           G           T
             allele1IsMajor MAFDifference
                  <logical>     <numeric>
  gene1_SNV1           TRUE    -0.3222222

-------
seqinfo: 1 sequence from an unspecified genome; no seqlengths
> ## MAF difference is large in size but negative and the finding is no longer
> ## significant (MBASED uses a 1-tailed significance test).
> ##
> ## We therefore strongly emphasize that the proper way to detect normal-specific ASE
> ## would be to treat normal sample as sample1 and tumor sample as sample2.
```

3.4 Adjusting for pre-exiting allelic bias

In a given sample, MBASED models the detected haplotype 1 allele counts $X_{hap1,SNV}$ at each SNV within gene g as independent random variables with

$$X_{hap1,SNV} \sim BetaBin(n = n_{SNV}, \mu = f_{hap1,SNV}(p_{hap1,g}), \rho = \rho_{SNV}) \tag{1}$$

where

$$E(X_{hap1,SNV}) = n\mu \tag{2}$$

and

$$var(X_{hap1,SNV}) = \mu(1-\mu)\rho n(n-1+frac1\rho) \tag{3}$$

If dispersion parameter $\rho_{SNV} = 0$ (this is the default model assumed by MBASED), the model reduces to the binomial model

$$X_{hap1,SNV} \sim Bin(n = n_{SNV}, p = f_{hap1,SNV}(p_{hap1,g})) \tag{4}$$

We will now assume the no-overdispersion binomial model (4) and will describe extensions to non-zero dispersions in the next section. By default, MBASED assumes that

$$f_{hap1,SNV}(p_{hap1,g}) = p_{hap1,g} \tag{5}$$

In other words, the underlying frequency of observed haplotype 1-supporting reads is equal to the frequency of haplotype 1 transcripts in the cell. In practice, this may not be the case. For example, the sample might undergo some enrichment procedure that results in global over-representation of reference allele-supporting reads. Or the aligner used to map reads to reference genome might favor one or the other allele at some SNVs due to the presence of a homologous region elsewhere in the genome. In such cases,

$$f_{hap1,SNV}(p_{hap1,g}) \neq p_{hap1,g} \tag{6}$$

We refer to the situation where one allele is favored over another in the read data, even in absence of ASE, as 'pre-existing allelic bias'. MBASED assumes that $f_{hap1,SNV}(p_{hap1,g})$ satisfies the following functional constraint:

$$f_{hap1,SNV}(p_{hap1,g}) = \frac{f_{hap1,SNV}(0.5) \times p_{hap1,g}}{f_{hap1,SNV}(0.5) \times p_{hap1,g} + f_{hap2,SNV}(0.5) \times p_{hap2,g}} \tag{7}$$

In words, $f_{hap1,SNV}(p_{hap1,g})$ is proportional to both $p_{hap1,g}$ and $f_{hap1,SNV}(0.5)$ (the underlying frequency of observed haplotype 1-supporting reads under conditions of no ASE) and is uniquely determined by these two frequencies. MBASED adjusts its estimates of ASE and corresponding statistical significance, when supplied with values of probabilities of observing allele 1-supporting reads under conditions of no ASE, $f_{hap1,SNV}(0.5)$ in our notation. As an example, consider a hypothetical case of known reference allele bias, where under conditions of no ASE ($p_{hap1,g} = p_{hap2,g} = 0.5$) the frequency of observed reference allele-supporting reads is 60% ($f_{ref,SNV}(0.5) = 0.6$ and $f_{alt,SNV}(0.5) = 0.4$) for all SNVs. Suppose we observe the following read counts:

locus	location	refCount	altCount	fractionRef
gene1:SNV1	chr1:100	20	25	0.49
gene2:SNV1	chr2:1000	17	9	0.65
gene2:SNV2	chr2:1100	22	15	0.59
gene2:SNV3	chr2:1200	20	10	0.67

Table 4: Example of reference bias in the data

Notice that gene2 is exhibiting 60/40 ratio of reference to alternative allele read counts, which is what we expect of genes with no ASE under the reference allele bias described above. In contrast, gene1 shows overexpression of alternative allele, which may actually be seen as evidence *for* ASE. MBASED allows one to specify the extent of pre-existing allelic bias by providing a corresponding matrix in the assays of the input RangedSummarizedExperiment object. By default, MBASED assumes that the background (no ASE) allelic ratio is 50/50:

```
> mySNVs
```

```
GRanges object with 4 ranges and 3 metadata columns:
             seqnames       ranges strand |       aseID     allele1     allele2
                <Rle>    <IRanges>  <Rle> | <character> <character> <character>
  gene1_SNV1     chr1 [ 100,  100]      * |       gene1           G           T
  gene2_SNV1     chr2 [1000, 1000]      * |       gene2           A           C
  gene2_SNV2     chr2 [1100, 1100]      * |       gene2           C           T
  gene2_SNV3     chr2 [1200, 1200]      * |       gene2           A           G
  -------
  seqinfo: 2 sequences from an unspecified genome; no seqlengths
```

```
> ## run MBASED with default values:
> summarizeASEResults_1s(
+   runMBASED(
+     SummarizedExperiment(
+       assays=list(
+         lociAllele1Counts=matrix(
+           c(20,17,15,20),
+           ncol=1,
+           dimnames=list(
+            names(mySNVs),
+            'mySample'
+           )
+         ),
+         lociAllele2Counts=matrix(
+           c(25,9,22,10),
+           ncol=1,
+           dimnames=list(
+            names(mySNVs),
+            'mySample'
+           )
+         )
+       ),
+       rowRanges=mySNVs
+     ),
+     isPhased=FALSE,
+     numSim=10^6,
+     BPPARAM=SerialParam()
+   )
+ )
```

```
$geneOutput
      majorAlleleFrequency pValueASE pValueHeterogeneity
gene1            0.5555556  0.551500                  NA
gene2            0.6345441  0.042719            0.544843

$locusOutput
GRangesList object of length 2:
$gene1
GRanges object with 1 range and 5 metadata columns:
             seqnames     ranges strand |       aseID     allele1     allele2
                <Rle>  <IRanges>  <Rle> | <character> <character> <character>
  gene1_SNV1     chr1 [100, 100]      * |       gene1           G           T
             allele1IsMajor       MAF
                  <logical> <numeric>
  gene1_SNV1          FALSE 0.5555556
```

```
$gene2
GRanges object with 3 ranges and 5 metadata columns:
             seqnames       ranges strand | aseID allele1 allele2 allele1IsMajor
  gene2_SNV1     chr2 [1000, 1000]      * | gene2       A       C           TRUE
  gene2_SNV2     chr2 [1100, 1100]      * | gene2       C       T          FALSE
  gene2_SNV3     chr2 [1200, 1200]      * | gene2       A       G           TRUE
                   MAF
  gene2_SNV1 0.6538462
  gene2_SNV2 0.5945946
  gene2_SNV3 0.6666667

-------
seqinfo: 2 sequences from an unspecified genome; no seqlengths
```

```
> ## we get the same results by explicitly specifying 50/50 pre-existing allelic probability
> summarizeASEResults_1s(
+   runMBASED(
+     SummarizedExperiment(
+       assays=list(
+         lociAllele1Counts=matrix(
+           c(20,17,15,20),
+           ncol=1,
+           dimnames=list(
+            names(mySNVs),
+            'mySample'
+           )
+         ),
+         lociAllele2Counts=matrix(
+           c(25,9,22,10),
+           ncol=1,
+           dimnames=list(
+            names(mySNVs),
+            'mySample'
+           )
+         ),
+         lociAllele1CountsNoASEProbs=matrix(
+           rep(0.5, 4),
+           ncol=1,
+           dimnames=list(
+            names(mySNVs),
+            'mySample'
+           )
+         )
+       ),
+       rowRanges=mySNVs
+     ),
+     isPhased=FALSE,
+     numSim=10^6,
+     BPPARAM=SerialParam()
+   )
+ )
```

```
$geneOutput
      majorAlleleFrequency pValueASE pValueHeterogeneity
```

```
gene1            0.5555556  0.551575                 NA
gene2            0.6345441  0.042472           0.545058

$locusOutput
GRangesList object of length 2:
$gene1
GRanges object with 1 range and 5 metadata columns:
             seqnames     ranges strand |       aseID     allele1     allele2
                <Rle>  <IRanges>  <Rle> | <character> <character> <character>
  gene1_SNV1     chr1 [100, 100]      * |       gene1           G           T
             allele1IsMajor       MAF
                  <logical> <numeric>
  gene1_SNV1          FALSE 0.5555556

$gene2
GRanges object with 3 ranges and 5 metadata columns:
             seqnames       ranges strand | aseID allele1 allele2 allele1IsMajor
  gene2_SNV1     chr2 [1000, 1000]      * | gene2       A       C           TRUE
  gene2_SNV2     chr2 [1100, 1100]      * | gene2       C       T          FALSE
  gene2_SNV3     chr2 [1200, 1200]      * | gene2       A       G           TRUE
                   MAF
  gene2_SNV1 0.6538462
  gene2_SNV2 0.5945946
  gene2_SNV3 0.6666667

-------
seqinfo: 2 sequences from an unspecified genome; no seqlengths
```

```
>
```

Note that we labeled 3 out of 4 reference alleles as allele1 and the corresponding alternative alleles as allele2 (the sole exception is SNV2 of gene2, where alternative allele is allele1). As can be expected, not accounting for pre-existing alleic bias results in gene2 showing strong evidence of ASE (p-value=0.04, all reference alleles are assigned to major haplotype). We can fix the problem by changing the value of `lociAllele1CountsNoASEProbs` assay matrix to the correct values of probabilites of observing allele1 under the conditions of no ASE:

```
> ##run the analysis adjusting for pre-existing allelic bias
> summarizeASEResults_1s(
+   runMBASED(
+     SummarizedExperiment(
+       assays=list(
+         lociAllele1Counts=matrix(
+           c(20,17,15,20),
+           ncol=1,
+           dimnames=list(
+            names(mySNVs),
+            'mySample'
+           )
+         ),
+         lociAllele2Counts=matrix(
+           c(25,9,22,10),
+           ncol=1,
+           dimnames=list(
+            names(mySNVs),
+            'mySample'
+           )
```

```
+         ),
+         lociAllele1CountsNoASEProbs=matrix(
+           c(0.6, 0.6, 0.4, 0.6),
+           ncol=1,
+           dimnames=list(
+            names(mySNVs),
+            'mySample'
+           )
+         )
+       ),
+       rowRanges=mySNVs
+     ),
+     isPhased=FALSE,
+     numSim=10^6,
+     BPPARAM=SerialParam()
+   )
+ )
$geneOutput
      majorAlleleFrequency pValueASE pValueHeterogeneity
gene1            0.6560217  0.046880                  NA
gene2            0.5388340  0.857726            0.685982

$locusOutput
GRangesList object of length 2:
$gene1
GRanges object with 1 range and 5 metadata columns:
             seqnames     ranges strand |       aseID     allele1     allele2
                <Rle>  <IRanges>  <Rle> | <character> <character> <character>
  gene1_SNV1     chr1 [100, 100]      * |       gene1           G           T
             allele1IsMajor       MAF
                  <logical> <numeric>
  gene1_SNV1          FALSE 0.6560217

$gene2
GRanges object with 3 ranges and 5 metadata columns:
             seqnames       ranges strand | aseID allele1 allele2 allele1IsMajor
  gene2_SNV1     chr2 [1000, 1000]      * | gene2       A       C           TRUE
  gene2_SNV2     chr2 [1100, 1100]      * | gene2       C       T           TRUE
  gene2_SNV3     chr2 [1200, 1200]      * | gene2       A       G           TRUE
                   MAF
  gene2_SNV1 0.5516396
  gene2_SNV2 0.5082244
  gene2_SNV3 0.5655568

-------
seqinfo: 2 sequences from an unspecified genome; no seqlengths
>
```

Notice that the estimate of MAF for gene2 went down from 0.63 to 0.54, and the p-value went up from 0.04 to 0.86. In contrast, MAF for gene1 went up from 0.56 to 0.66 and the p-value dropped from 0.55 to 0.05 (as we are much less likely to observe overexpression of alternative allele if the bias in the data favors the reference allele).

When 2-sample analysis is performed, the second column of `lociAllele1CountsNoASEProbs` specifies no-ASE allelic probabilities for the same alleles in sample 2, (and thus MBASED allows for possibility that pre-existing allelic biases are

different in the two samples).

In our own work, we assumed that the no-ASE probability of observing reference allele-supporting read was the same across all heterozygous loci within each sample (global reference bias) and estimated it for our data to be approximately 0.52 (see the accompanying paper for the details of the estimation procedure). A better approach would be to estimate this probability separately at each SNV based on a large number of samples known to not exhibit ASE at that locus (and keeping the protocol/sequencing/alignment pipeline constant). Currently MBASED provdies no functionality for estimating the extent of pre-existing allelic bias and the user is advised to investigate their data prior to changing the default values of the relevent function arguments.

3.5 Adjusting for overdispersion

By default, MBASED assumes that dispersion parameter ρ in model 1 is 0, leading to binomial model 4. In practice, extra-binomial dispersion may be present in the sequencing read count data and needs to be taken into account in order to assign proper statistical significance to observed levels of ASE. Below we generate some counts from beta-binomial distribution and show how model misspecification (assuming data to be binomial) affects ASE calling. We use our earlier total read counts for single-SNV gene1 and 3-SNV gene2. We also use the same value of overdispersion parameter $\rho = 0.02$ at each SNV.

```
> set.seed(5) ## we chose this seed to get a 'significant' result
> totalAlleleCounts=c(45, 26, 37, 30)
> ## simulate allele1 counts
> allele1Counts <- MBASED:::vectorizedRbetabinomMR(
+   n=4,
+   size=totalAlleleCounts,
+   mu=rep(0.5, 4),
+   rho=rep(0.02, 4)
+ )
> ## run MBASED without accounting for overdispersion
> summarizeASEResults_1s(
+   runMBASED(
+     SummarizedExperiment(
+       assays=list(
+         lociAllele1Counts=matrix(
+           allele1Counts,
+           ncol=1,
+           dimnames=list(
+            names(mySNVs),
+            'mySample'
+           )
+         ),
+         lociAllele2Counts=matrix(
+           totalAlleleCounts-allele1Counts,
+           ncol=1,
+           dimnames=list(
+            names(mySNVs),
+            'mySample'
+           )
+         )
+       ),
+       rowRanges=mySNVs
+     ),
+     isPhased=FALSE,
+     numSim=10^6,
```

```
+     BPPARAM=SerialParam()
+   )
+ )
```

```
$geneOutput
      majorAlleleFrequency pValueASE pValueHeterogeneity
gene1            0.5555556  0.551951                  NA
gene2            0.6254723  0.049582            0.025243

$locusOutput
GRangesList object of length 2:
$gene1
GRanges object with 1 range and 5 metadata columns:
             seqnames     ranges strand |       aseID     allele1     allele2
                <Rle>  <IRanges>  <Rle> | <character> <character> <character>
  gene1_SNV1     chr1 [100, 100]      * |       gene1           G           T
             allele1IsMajor       MAF
                  <logical> <numeric>
  gene1_SNV1           TRUE 0.5555556

$gene2
GRanges object with 3 ranges and 5 metadata columns:
             seqnames       ranges strand | aseID allele1 allele2 allele1IsMajor
  gene2_SNV1     chr2 [1000, 1000]      * | gene2       A       C           TRUE
  gene2_SNV2     chr2 [1100, 1100]      * | gene2       C       T          FALSE
  gene2_SNV3     chr2 [1200, 1200]      * | gene2       A       G           TRUE
                   MAF
  gene2_SNV1 0.7307692
  gene2_SNV2 0.6486486
  gene2_SNV3 0.5000000

-------
seqinfo: 2 sequences from an unspecified genome; no seqlengths
```

```
> ## this is the same as explicitly setting dispersion parameters to 0.
> summarizeASEResults_1s(
+   runMBASED(
+     SummarizedExperiment(
+       assays=list(
+         lociAllele1Counts=matrix(
+           allele1Counts,
+           ncol=1,
+           dimnames=list(
+            names(mySNVs),
+            'mySample'
+           )
+         ),
+         lociAllele2Counts=matrix(
+           totalAlleleCounts-allele1Counts,
+           ncol=1,
+           dimnames=list(
+            names(mySNVs),
+            'mySample'
+           )
+         ),
```

```
+         lociCountsDispersions=matrix(
+           rep(0, 4),
+           ncol=1,
+           dimnames=list(
+            names(mySNVs),
+            'mySample'
+           )
+         )
+       ),
+       rowRanges=mySNVs
+     ),
+     isPhased=FALSE,
+     numSim=10^6,
+     BPPARAM=SerialParam()
+   )
+ )
$geneOutput
      majorAlleleFrequency pValueASE pValueHeterogeneity
gene1            0.5555556  0.551767                  NA
gene2            0.6254723  0.049576            0.025282

$locusOutput
GRangesList object of length 2:
$gene1
GRanges object with 1 range and 5 metadata columns:
             seqnames     ranges strand |       aseID     allele1     allele2
                <Rle>  <IRanges>  <Rle> | <character> <character> <character>
  gene1_SNV1     chr1 [100, 100]      * |       gene1           G           T
             allele1IsMajor       MAF
                  <logical> <numeric>
  gene1_SNV1           TRUE 0.5555556

$gene2
GRanges object with 3 ranges and 5 metadata columns:
             seqnames       ranges strand | aseID allele1 allele2 allele1IsMajor
  gene2_SNV1     chr2 [1000, 1000]      * | gene2       A       C           TRUE
  gene2_SNV2     chr2 [1100, 1100]      * | gene2       C       T          FALSE
  gene2_SNV3     chr2 [1200, 1200]      * | gene2       A       G           TRUE
                   MAF
  gene2_SNV1 0.7307692
  gene2_SNV2 0.6486486
  gene2_SNV3 0.5000000

-------
seqinfo: 2 sequences from an unspecified genome; no seqlengths
> ## now re-run MBASED supplying the correct dispersion values:
> summarizeASEResults_1s(
+   runMBASED(
+     SummarizedExperiment(
+       assays=list(
+         lociAllele1Counts=matrix(
+           allele1Counts,
+           ncol=1,
```

```
+           dimnames=list(
+            names(mySNVs),
+            'mySample'
+           )
+         ),
+         lociAllele2Counts=matrix(
+           totalAlleleCounts-allele1Counts,
+           ncol=1,
+           dimnames=list(
+            names(mySNVs),
+            'mySample'
+           )
+         ),
+         lociCountsDispersions=matrix(
+           rep(0.02, 4),
+           ncol=1,
+           dimnames=list(
+            names(mySNVs),
+            'mySample'
+           )
+         )
+       ),
+       rowRanges=mySNVs
+     ),
+     isPhased=FALSE,
+     numSim=10^6,
+     BPPARAM=SerialParam()
+   )
+ )
$geneOutput
      majorAlleleFrequency pValueASE pValueHeterogeneity
gene1            0.5555556  0.669076                  NA
gene2            0.6263025  0.185934            0.077609

$locusOutput
GRangesList object of length 2:
$gene1
GRanges object with 1 range and 5 metadata columns:
             seqnames     ranges strand |       aseID     allele1     allele2
                <Rle>  <IRanges>  <Rle> | <character> <character> <character>
  gene1_SNV1     chr1 [100, 100]      * |       gene1           G           T
             allele1IsMajor       MAF
                  <logical> <numeric>
  gene1_SNV1           TRUE 0.5555556

$gene2
GRanges object with 3 ranges and 5 metadata columns:
             seqnames       ranges strand | aseID allele1 allele2 allele1IsMajor
  gene2_SNV1     chr2 [1000, 1000]      * | gene2       A       C           TRUE
  gene2_SNV2     chr2 [1100, 1100]      * | gene2       C       T          FALSE
  gene2_SNV3     chr2 [1200, 1200]      * | gene2       A       G           TRUE
                   MAF
  gene2_SNV1 0.7307692
  gene2_SNV2 0.6486486
```

```
  gene2_SNV3 0.5000000

-------
seqinfo: 2 sequences from an unspecified genome; no seqlengths
>
```

We observe that after taking overdispersion into account, gene1 had its ASE p-value increase from 0.55 to 0.67, and gene2 had the pvalue increase from 0.05 to 0.19. In practice, not taking overdispersion into account leads to a number of false positive ASE calls, with the number of false positives rising as the extent of overdispersion increases. Therefore it is important to have a good grasp of how much dispersion is present in the data. Also note a slight change in the estimate of major allele frequency for gene2, which results from incorporation of extra-variability into the estimate during the meta-analytic integration of ASE information from multiple SNVs within that gene.

When 2-sample analysis is performed, the second column of `lociCountsDispersions` specifies dispersion parameter values for the same SNVs in sample 2, (and thus MBASED allows for possibility that the extent of overdispersion is different in the two samples).

In our own work, we assumed that the dispersion was the same across all loci within each sample and estimated it for our data to be approximately 0.004 (see the accompanying paper for the details of the estimation procedure). A better approach would be to estimate the dispersion separately at each SNV based on a large number of samples known to not exhibit ASE at that locus (and keeping the protocol/sequencing/alignment pipeline constant). Currently MBASED provdies no functionality for estimating the extent the dispersion and the user is advised to investigate their data prior to changing the default values of the relevent function arguments.

4 Session Info

```
> sessionInfo()

R version 3.3.0 (2016-05-03)
Platform: x86_64-pc-linux-gnu (64-bit)
Running under: Ubuntu 14.04.4 LTS

locale:
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C               LC_TIME=en_US.UTF-8
 [4] LC_COLLATE=C               LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=en_US.UTF-8       LC_NAME=C                  LC_ADDRESS=C
[10] LC_TELEPHONE=C             LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C

attached base packages:
[1] stats4    parallel  stats     graphics  grDevices utils     datasets  methods
[9] base

other attached packages:
 [1] MBASED_1.6.0               SummarizedExperiment_1.2.0 Biobase_2.32.0
 [4] GenomicRanges_1.24.0       GenomeInfoDb_1.8.0         IRanges_2.6.0
 [7] S4Vectors_0.10.0           BiocParallel_1.6.0         BiocGenerics_0.18.0
[10] RUnit_0.4.31

loaded via a namespace (and not attached):
[1] zlibbioc_1.18.0 BiocStyle_2.0.0 XVector_0.12.0  tools_3.3.0
```