# Calling subclonal mutations with `deepSNV`

Moritz Gerstung and Niko Beerenwinkel

May 19, 2021

## Contents



## 1 Introduction

This package provides algorithms for calling single nucleotide variants in deep sequencing experiments of polyclonal samples. The package uses a clonal control experiment for estimating the local error rate and tests whether the observed nucleotide frequencies differ between test and control. The basic model is a binomial model for the counts $X_{i,j}$ and $Y_{i,j}$ of nucleotide $j$ at position $i$, in the test and the control experiment, respectively:

$$\begin{aligned} X_{i,j} &\sim \mathrm{Bin}(n_i, p_{i,j}) \\ Y_{i,j} &\sim \mathrm{Bin}(m_i, q_{i,j}). \end{aligned} \tag{1}$$

Here $n_i$ and $m_i$ denote the coverage in the two experiments, and $p_{i,j}$ and $q_{i,j}$ are learned from the data. The presence of an SNV in the test experiment amounts to testing the hypothesis $H_1 : p_{i,j} > q_{i,j}$ against the null-hypothesis $H_0 : p_{i,j} = q_{i,j}$. The deepSNV algorithm uses likelihood ratio test with a $\chi^2_1$-distribtution.

As an alternative to the binomial distribution, a beta-binomial model can be used that has a global parameter of overdispersion:

$$\begin{aligned} X_{i,j} &\sim \mathrm{BB}(n_i, \alpha, p_{i,j}) \\ Y_{i,j} &\sim \mathrm{BB}(m_i, \alpha, q_{i,j}). \end{aligned} \tag{2}$$

The parameter $\alpha$ defines a parameter that quantifies the overdispersion of the model, shared across sites and nucleotides. This parametrization is equivalent to setting $\beta_{i,j} = \alpha(1 - p_{i,j})/p_{i,j}$. For small $p_{i,j}$, one obtains a variance of $\mathrm{E}[X_{i,j}] \approx n_i p_{i,j} + (n_i p_{i,j})^2/\alpha$.

All parameters are determined by a maximum likelihood criterion, where for $p_{i,j}$ (and similarly for $q_{i,j}$) a methods-of-moments approximation is used, $\alpha/(\alpha + \hat{\beta}_{i,j}) = X_{i,j}/n_i$. The binomial model arises from the beta-binomial model in the limit $\alpha \to \infty$.

To achieve a higher specifitiy the test is performed on both strands separately, and the resulting p-values are combined into a single one using either the product, average, or maximum as a statistic and their corresponding distributions under a uniform for computing a joint p-value. For more information and for citing the `deepSNV` package please use:

- Gerstung M, Beisel C, Rechsteiner M, Wild P, Schraml P, Moch H, Beerenwinkel N (2012). "Reliable detection of subclonal single-nucleotide variants in tumor cell populations." *Nat Commun*, **3**, 811.

## 2 Working example

In this example, we load some real world data. The data of length 1,512 nt were sequenced with a Roche 454 Junior sequencer at about 500x coverage. They consist of a mixture of two HIV clones at 10% and 90%(test) and a clonal control. The data were aligned to the HXB2 reference genome with novoalign, and can be downloaded from the authors' website, or attached . We first load the package and define the genomic `region` of interest:

```
    library(deepSNV)
        regions <- data.frame(chr="B.FR.83.HXB2_LAI_IIIB_BRU_K034", start = 2074, stop=3585)
```

Now the data can be loaded from the remote .bam files with the `deepSNV` command (not run)

```
# HIVmix <- deepSNV(test = "http://www.bsse.ethz.ch/cbg/software/deepSNV/data/test.bam",
#                   control = "http://www.bsse.ethz.ch/cbg/software/deepSNV/data/control.bam",
#                   regions=regions, q=10)
```

The `data.frame regions` contains the genomic region to be parsed from the two files by the method `deepSNV`. The additional parameter `q=10` specifies that only nucleotides with PHRED higher than 10 are counted. As this might fail in the absence of a running internet connection, we load the resulting object that comes along with the deepSNV package:

```
        data(HIVmix) # Attach the data instead, as it could fail in routine checks without internet
        show(HIVmix)

## Data:  1512 positions x  10 characters
## Model:  bin
## Alternative:  greater
## Combine Method:  fisher
## P-Values:
##              A         T         C         G         -
## [1,]        NA 0.5965736 0.5965736 0.5965736 0.5965736
## [2,] 0.5965736 0.5965736 0.5965736        NA 0.5965736
## [3,]        NA 0.5965736 0.5965736 0.5965736 0.5965736
## [4,] 0.5965736 0.5965736        NA 0.5965736 0.5965736
## [5,]        NA 0.5965736 0.5965736 0.5965736 0.5965736
## [6,] 0.5965736 0.5965736 0.5965736        NA 0.5965736
## ...
##                 A         T         C         G         -
## [1507,]        NA 0.5965736 0.5965736 0.5965736 0.5965736
## [1508,] 0.8465736 0.5965736 0.8465736        NA 0.5965736
## [1509,] 0.5965736 0.5965736        NA 0.5965736 0.5965736
## [1510,] 0.8465736 0.5965736        NA 0.5965736 0.5965736
## [1511,]        NA 0.5965736 0.5965736 0.4737885 0.5965736
## [1512,] 1.0000000        NA 0.8465736 0.8465736 0.8465736
```

The counts are stored in the slots `test` and `control`:

```
        control(HIVmix)[100:110,]

##          A    T    C    G - a   t  c   g _
##  [1,] 1170    0    0    0 0 163   0  0   1 0
##  [2,]    0    0    0 1170 0   1   0  0 147 0
##  [3,]    0 1170    0    0 0   0 118  0   3 6
##  [4,]    0 1170    0    0 0   0 125  0   0 0
##  [5,]    0 1170    0    0 0   1  99 10   0 0
##  [6,]    0    0 1170    0 0   1   0 91   5 0
##  [7,]    0    0    0 1168 0   0   0  1  85 0
```

```
##  [8,]    0    0    0 1169 0   0   1  0  94 0
##  [9,]    0    0    0 1170 0   0   3  0  99 0
## [10,]    0 1170    0    1 0   0 109  0   0 0
## [11,]    0 1173    0    0 0   0 115  0   0 0
```

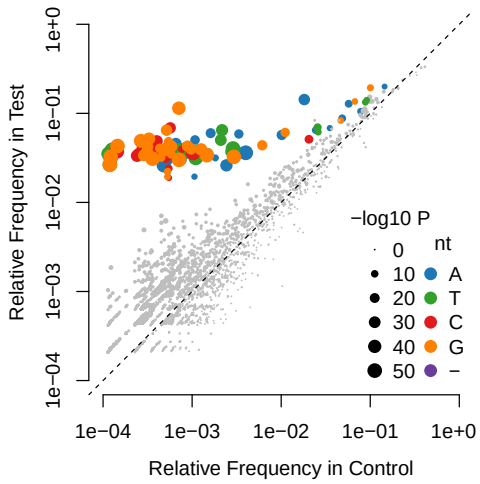
```
        test(HIVmix)[100:110,]
```

```
##         A   T   C   G - a  t  c  g _
##  [1,] 442   0   0   0 0 70  0  0  0 0
##  [2,]   0   0   0 441 0  0  0  0 66 0
##  [3,]   0 427  13   0 0  0 50  6  0 3
##  [4,]   0 440   0   0 0  0 56  0  0 0
##  [5,]   0 440   0   0 0  0 42  7  0 0
##  [6,]   1   1 437   0 0  0  1 38  4 0
##  [7,]  10   0   2 424 0  3  0  1 33 0
##  [8,]   0   0   0 429 0  0  0  0 36 0
##  [9,]   0   0   0 425 0  0  0  0 38 0
## [10,]   0 425   0   0 0  0 42  0  0 0
## [11,]   0 425   0   0 0  0 47  0  0 0
```

Uppercase nucleotides are from the reference strand, lowercase nucleotides from the reverse. Also note the strand bias.

A visual representation of the data can be obtained with the `plot` method:

```
        plot(HIVmix)
```




One realizes that there are many variants nicely separated by the test at the topleft corner, althought the noise level also extends along the diagonal to similar frequencies. Grey dots have a $P$-values smaller than 0.05.

Significant SNVs are tabularized with the `summary` command:

```
        SNVs <- summary(HIVmix, sig.level=0.05, adjust.method="BH")
        head(SNVs)
```

```
##                              chr  pos ref var        p.val   freq.var
## 1  B.FR.83.HXB2_LAI_IIIB_BRU_K034 2127   G   A 4.191236e-07 0.02903526
## 53 B.FR.83.HXB2_LAI_IIIB_BRU_K034 2130   T   C 1.802070e-09 0.03076923
## 73 B.FR.83.HXB2_LAI_IIIB_BRU_K034 2139   A   G 7.753362e-11 0.03362573
## 74 B.FR.83.HXB2_LAI_IIIB_BRU_K034 2141   A   G 5.771900e-11 0.03333333
## 54 B.FR.83.HXB2_LAI_IIIB_BRU_K034 2150   A   C 9.639517e-06 0.01763908
## 2  B.FR.83.HXB2_LAI_IIIB_BRU_K034 2151   G   A 5.804568e-10 0.02815013
##    sigma2.freq.var n.tst.fw cov.tst.fw n.tst.bw cov.tst.bw n.ctrl.fw
## 1     4.892000e-05       17        581        2         47         2
```

```
## 53    4.733728e-05       16        597        4         53         0
## 73    4.916043e-05       16        599        7         85         0
## 74    4.830918e-05       16        599        7         91         0
## 54    2.393362e-05       10        609        3        128         0
## 2     3.773476e-05       16        614        5        132         0
##    cov.ctrl.fw n.ctrl.bw cov.ctrl.bw    raw.p.val
## 1         1537         0         103 6.306257e-09
## 53        1534         0         104 2.353895e-11
## 73        1535         0         158 9.614784e-13
## 74        1535         0         181 7.062179e-13
## 54        1538         0         283 1.577897e-07
## 2         1539         0         287 7.486050e-12
```

```
        nrow(SNVs)
```

```
## [1] 107
```

```
        min(SNVs$freq.var)
```

```
## [1] 0.004378763
```

We chose a significance level of `sig.level=0.05` and Benjamini-Hochberg correction for multiple testing (`adjust.method="BH"`). The test selected 107 variants. This compares to

```
        sum(RF(test(HIVmix), total=T) > 0.01 & RF(test(HIVmix), total=T) < 0.95)
```

```
## [1] 417
```

candidate variants with frequencies above 0.01! In this experiment we also know the truth from direct Sanger sequencing of the clones before pooling. Load the data and study the confusion matrix with:

```
        data(trueSNVs, package="deepSNV")
        table(p.adjust(p.val(HIVmix), method="BH") < 0.05, trueSNVs)
```

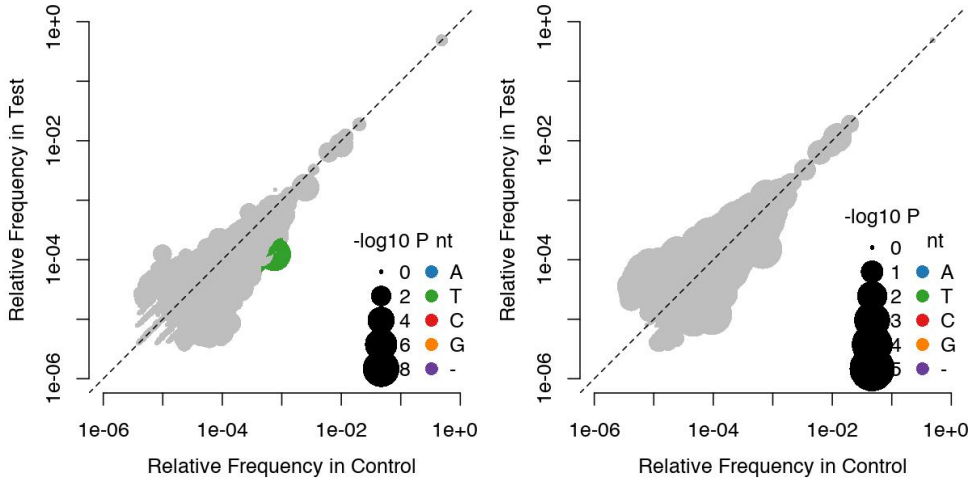
```
##         trueSNVs
##          FALSE TRUE
##   FALSE  5933    8
##   TRUE     14   93
```

So 93 of 101 SNVs could be recovered by the experiments.

## 3 Normalization

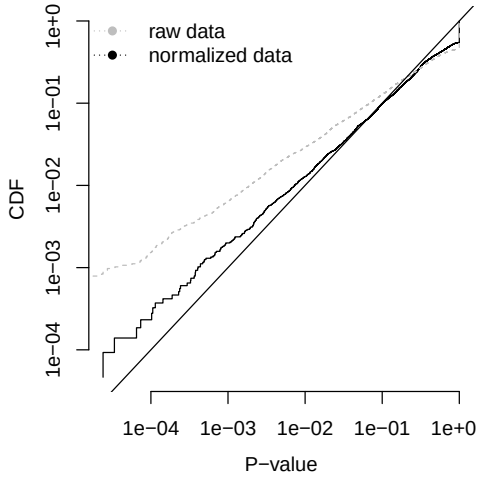
We want to further assess the null model with experimental data from two homogeneous replicates. In particular we want to analyze whether the empirical distribution of the p-values is uniform. The data we study comes from two phiX sequences sequenced on separate runs on a GAII$_x$.

```
        ## Load data (unnormalized)
        data(phiX, package="deepSNV")
        plot(phiX, cex.min=.5)
        ## Normalize data
        phiN <- normalize(phiX, round=TRUE)
        plot(phiN, cex.min=.5)
```




From the left plot it appears that there is a systematic bias between the two experiments, likely because they were sequenced in different runs. The normalized data is shown in the second plot. Now the points now symmetrically scatter around the diagonal and all p-values are within the expected range:

```
    p.norm <- p.val(phiN)
n <- sum(!is.na(p.norm))
qqplot(p.norm, seq(1/n,1, length.out=n), log="xy", type="S", xlab="P-value", ylab="CDF")
    p.val <- p.val(phiX)
points(sort(p.val[!is.na(p.val)]), seq(1/n,1, length.out=n), pch=16, col="grey", type="S", lty=2)
legend("topleft", c("raw data", "normalized data"), pch=16, col=c("grey", "black"), bty="n", lty=
abline(0,1)
```




After normalization the cumulative distribution of the p-values is close to the diagonal, even for the smallest values. Hence the p-values accurately measure the probability of type-1 errors.

## 4 Overdispersion

In some situations, the variance of the binomial model is too small, for example for templates with long repeats or heavy PCR amplification for target selection. An alternative model is the beta-binomial distribution that allows for a larger variance.

We load a data-set from two deep sequencing experiments of four genes extracted from a metastatic renal cell carcinoma with sequenced on separate lanes of a GAII$_x$:

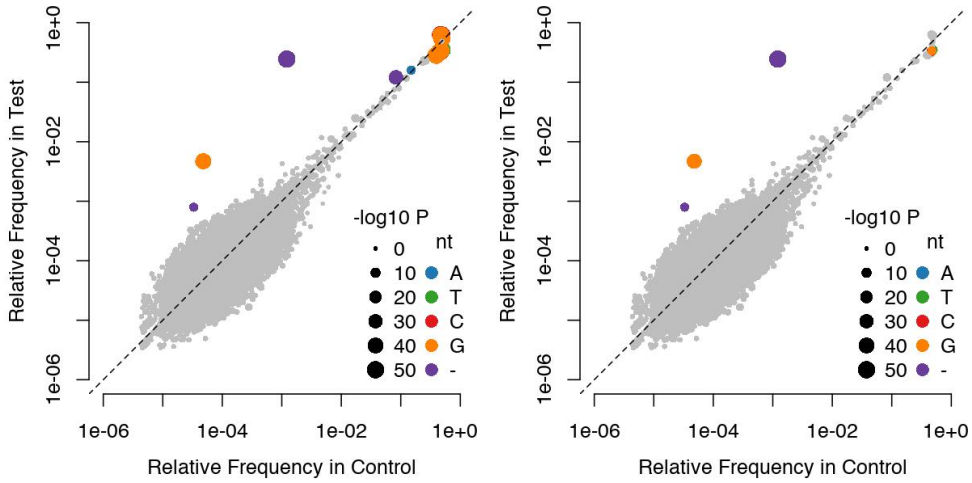
```
data("RCC", package="deepSNV")
show(RCC)

## Data:  14813 positions x  10 characters
```

```
## Model:  bin
## Alternative:  two.sided
## Combine Method:  average
## P-Values:
##              A         T         C         G -
## [1,]        NA 1.0000000 1.0000000 1.0000000 1
## [2,]        NA 1.0000000 0.8395837 1.0000000 1
## [3,]        NA 0.8901412 1.0000000 0.9144178 1
## [4,] 0.5000127 0.5969902        NA 0.5717729 1
## [5,]        NA 0.9977519 0.5858255 0.6729237 1
## [6,] 1.0000000 0.9810956 0.7530014        NA 1
## ...
##                   A         T         C  G -
## [14808,] 0.9733845 1.0000000 0.8896366 NA 1
## [14809,] 1.0000000 1.0000000        NA  1 1
## [14810,]        NA 0.9201851 0.5000416  1 1
## [14811,] 0.9414135 1.0000000 0.9944664 NA 1
## [14812,]        NA 1.0000000 0.8270151  1 1
## [14813,]        NA 1.0000000 1.0000000  1 1
```

```
        plot(RCC, cex.min=.5)
        RCC.bb = estimateDispersion(RCC, alternative="two.sided")
```

```
## Note: The initial object used a binomial model. Will be changed to beta-binomial.
## Estimated dispersion factor 136.926969848769
```

```
        plot(RCC.bb, cex.min=.5)
```




We see that a binomial model was used to generate the data. An inspection of the first plot shows a long noise tail where apparently the dispersion is underestimated causing some false positives. In the second plot we used a beta-binomial model instead and conservatively estimate the dispersion factor on both sides with the argument `alternative="two.sided"`.

The log-likelihood of the two models are:

```
        RCC.bb@log.lik
```

```
## [1] -270110.7
```

```
        RCC@log.lik
```

```
## [1] -283851.7
```

```
    RCC.bb@log.lik - RCC@log.lik
```

```
## [1] 13740.93
```

```
        log(4*nrow(test(RCC)))

## [1] 10.98955
```

Note that the difference is larger than $\log(n)$, the difference in BIC of the two models.
If we compare the number of called SNVs, we find

```
        summary(RCC, adjust.method="bonferroni")[,1:6]

##       chr      pos ref var         p.val      freq.var
## 6   chr10 89710231   C   T  1.812318e-06 -0.0522069105
## 1   chr17  7512879   G   A  4.923709e-03  0.0095680379
## 10  chr17  7513782   A   G  6.137099e-57  0.0523098637
## 15   chr3 10158255   A   -  7.414419e-27  0.0374605077
## 4    chr3 10158274   C   T 6.596999e-152 -0.1420210172
## 2    chr3 10158337   G   A  8.455105e-09 -0.1458150689
## 7    chr3 10163012   -   C 2.512916e-211  0.1242571351
## 16   chr3 10163206   T   -  0.000000e+00  0.2440387988
## 17   chr3 10163208   G   -  1.908223e-02  0.0007724394
## 11   chr3 10163428   T   G  1.026389e-79 -0.1111055718
## 8    chr3 10166219   G   C  6.538408e-74  0.1591560973
## 3    chr3 10166943   G   A 1.406981e-158 -0.1326056510
## 12   chr3 10167220   C   G  2.960126e-36  0.0046531880
## 13   chr3 10167672   A   G  0.000000e+00  0.1399833662
## 5    chr3 10167709   C   T 6.203881e-304 -0.1440968910
## 9    chr3 10167762   T   C  0.000000e+00 -0.1323353964
## 14   chr3 10168683   T   G 2.134656e-304 -0.1324914089
```

compared to

```
        tab <- summary(RCC.bb, adjust.method="bonferroni")[,1:6]
        tab

##    chr      pos ref var         p.val      freq.var
## 2 chr3 10158274   C   T  4.185434e-03 -0.1420210172
## 7 chr3 10163206   T   - 6.774267e-304  0.2440387988
## 8 chr3 10163208   G   -  2.693886e-02  0.0007724394
## 1 chr3 10166943   G   A  9.499952e-03 -0.1326056510
## 5 chr3 10167220   C   G  5.461018e-30  0.0046531880
## 3 chr3 10167709   C   T  3.466443e-02 -0.1440968910
## 4 chr3 10167762   T   C  8.715616e-03 -0.1323353964
## 6 chr3 10168683   T   G  4.237354e-02 -0.1324914089
```

A closer inspection will show that the variants with a negative change in frequency are all known SNPs on chromosome 3, which drop in frequency due to loss of heterozygousity in the tumor. The remaining variants have positive frequencies. The first is a deletion of chr3:10158274C on -14.2% of the alleles that truncates the VHL protein. The second is a C>G conversion in the 5′-UTR of the *VHL* gene at chr3:10163206T in 24.4% of the alleles. The third variant is likely to be an alignment artifact resulting from imperfect alignments of the deletion of chr3:10158274C.

## 5 sessionInfo()

- R version 4.1.0 RC (2021-05-10 r80283), `x86_64-apple-darwin17.0`
- Locale: `C/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8`
- Running under: `macOS Mojave 10.14.6`
- Matrix products: default

- BLAS: `/Library/Frameworks/R.framework/Versions/4.1/Resources/lib/libRblas.dylib`
- LAPACK: `/Library/Frameworks/R.framework/Versions/4.1/Resources/lib/libRlapack.dylib`
- Base packages: base, datasets, grDevices, graphics, methods, parallel, splines, stats, stats4, utils
- Other packages: Biobase 2.52.0, BiocGenerics 0.38.0, Biostrings 2.60.0, GenomeInfoDb 1.28.0, GenomicRanges 1.44.0, IRanges 2.26.0, MatrixGenerics 1.4.0, Rsamtools 2.8.0, S4Vectors 0.30.0, SummarizedExperiment 1.22.0, VGAM 1.1-5, VariantAnnotation 1.38.0, XVector 0.32.0, deepSNV 1.38.0, knitr 1.33, matrixStats 0.58.0
- Loaded via a namespace (and not attached): AnnotationDbi 1.54.0, BSgenome 1.60.0, BiocFileCache 2.0.0, BiocIO 1.2.0, BiocParallel 1.26.0, DBI 1.1.1, DelayedArray 0.18.0, GenomeInfoDbData 1.2.6, GenomicAlignments 1.28.0, GenomicFeatures 1.44.0, KEGGREST 1.32.0, Matrix 1.3-3, R6 2.5.0, RCurl 1.98-1.3, RSQLite 2.2.7, Rcpp 1.0.6, Rhtslib 1.24.0, XML 3.99-0.6, assertthat 0.2.1, biomaRt 2.48.0, bit 4.0.4, bit64 4.0.5, bitops 1.0-7, blob 1.2.1, bslib 0.2.5.1, cachem 1.0.5, compiler 4.1.0, crayon 1.4.1, curl 4.3.1, dbplyr 2.1.1, digest 0.6.27, dplyr 1.0.6, ellipsis 0.3.2, evaluate 0.14, fansi 0.4.2, fastmap 1.1.0, filelock 1.0.2, generics 0.1.0, glue 1.4.2, grid 4.1.0, highr 0.9, hms 1.1.0, htmltools 0.5.1.1, httr 1.4.2, jquerylib 0.1.4, jsonlite 1.7.2, lattice 0.20-44, lifecycle 1.0.0, magrittr 2.0.1, memoise 2.0.0, pillar 1.6.1, pkgconfig 2.0.3, png 0.1-7, prettyunits 1.1.1, progress 1.2.2, purrr 0.3.4, rappdirs 0.3.3, restfulr 0.0.13, rjson 0.2.20, rlang 0.4.11, rmarkdown 2.8, rtracklayer 1.52.0, sass 0.4.0, stringi 1.6.2, stringr 1.4.0, tibble 3.1.2, tidyselect 1.1.1, tools 4.1.0, utf8 1.2.1, vctrs 0.3.8, xfun 0.23, yaml 2.2.1, zlibbioc 1.38.0