Calling subclonal mutations with deepSNV

Moritz Gerstung and Niko Beerenwinkel

October 1, 2012

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 χ^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 α 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 and Beerenwinkel N (2012). "Reliable detection of subclonal single-nucleotide variants in tumor cell populations." *Nat Commun*, **3**, pp. 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 inte
>         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
```

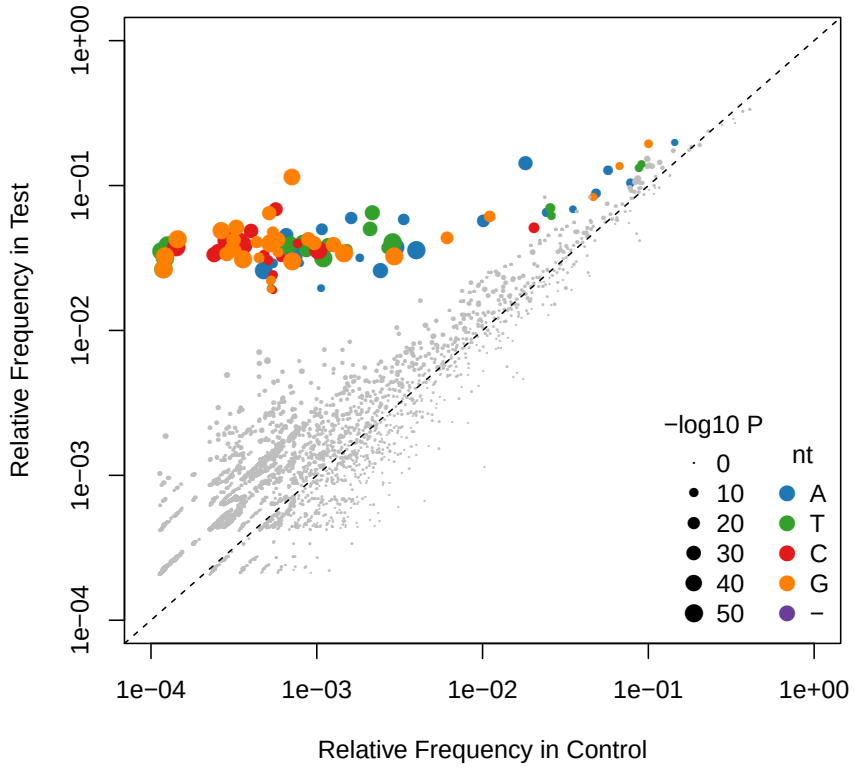
```
>         test(HIVmix)[100:110,]
```

```
        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 3317   C   T 6.535962e-72 0.03878536
2 B.FR.83.HXB2_LAI_IIIB_BRU_K034 3485   A   T 1.315137e-71 0.03502822
3 B.FR.83.HXB2_LAI_IIIB_BRU_K034 3282   A   G 6.300291e-69 0.04278264
4 B.FR.83.HXB2_LAI_IIIB_BRU_K034 3283   A   C 1.043249e-66 0.04243180
5 B.FR.83.HXB2_LAI_IIIB_BRU_K034 3263   A   G 2.557500e-66 0.03930098
6 B.FR.83.HXB2_LAI_IIIB_BRU_K034 3509   T   C 2.776494e-63 0.03485991
  sigma2.freq.var n.tst.fw cov.tst.fw n.tst.bw cov.tst.bw n.ctrl.fw cov.ctrl.fw
1    9.315960e-06       64       1370       98       2814         0        2302
2    7.506701e-06       32        729      132       3961         0        1033
3    1.185024e-05       70       1327       85       2302         0        2184
4    1.183474e-05       70       1318       84       2305         1        2170
5    1.018556e-05       71       1568       85       2369         3        2775
6    7.611893e-06       29        683      142       4105         2         943
  n.ctrl.bw cov.ctrl.bw    raw.p.val
1         0        5625 1.080682e-75
2         0        7616 4.348999e-75
3         0        4695 3.125144e-72
4         0        4687 6.899797e-70
5         0        4817 2.114335e-69
6         6        7826 3.213535e-66

> nrow(SNVs)

[1] 107

> min(SNVs$freq.var)

[1] 0.004624562
```

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$_x$.

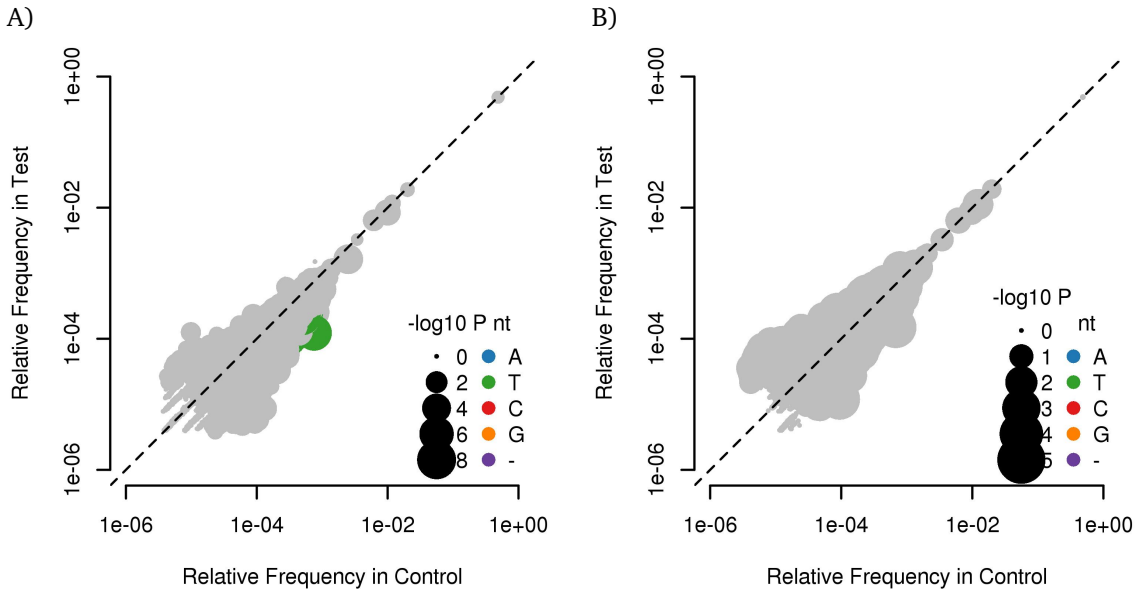


Figure 1: Scatterplot for two phiX experiments before (A) and after (B) normalization.

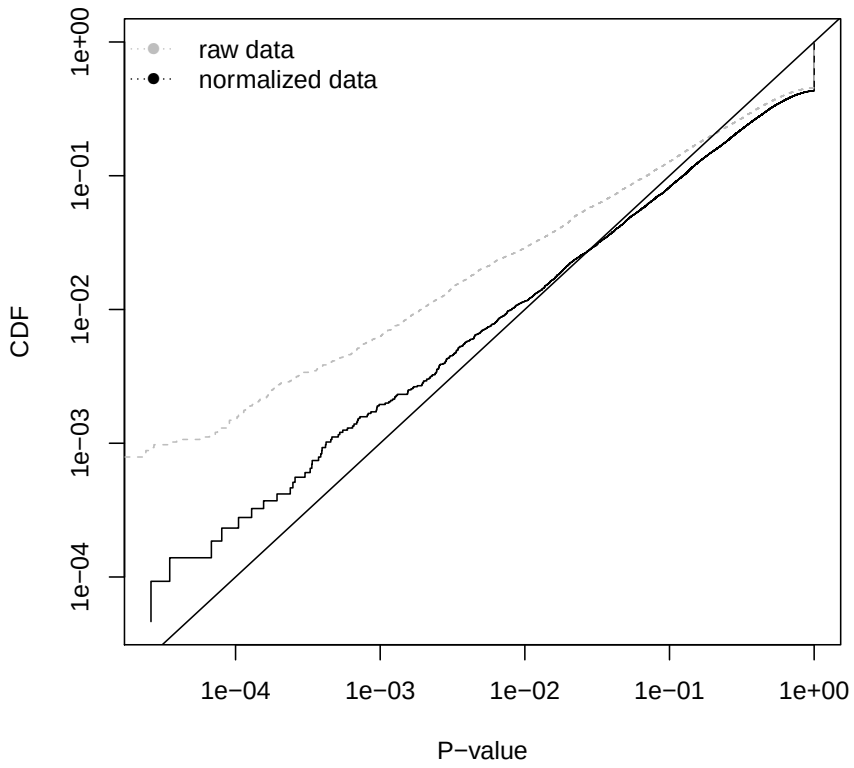
```
>         data(phiX, package="deepSNV")
>         jpeg("deepSNV-phiX.jpg", 2.5, 2.5, units="in", res=600, pointsize=7)
>         par(mar=c(4,4,1,1), bty="n")
>         plot(phiX, cex.min=.5)
>         dev.off()
```

(We use a jpg device to reduce the plot size.) From the plot in Figure 1A it appears that there is a systematic bias between the two experiments, likely because they were sequenced in different runs. We therefore normalize with:

```
>     phiN <- normalize(phiX, round=TRUE)
>         jpeg("deepSNV-phiN.jpg", 2.5, 2.5, units="in", res=600, pointsize=7)
>         par(mar=c(4,4,1,1), bty="n")
>     plot(phiN, cex.min=.5)
>         dev.off()
```

The results are shown in Figure 1B. 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",
>     legend("topleft", c("raw data", "normalized data"), pch=16, col=c("grey", "black"), bty="n"
>     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
```

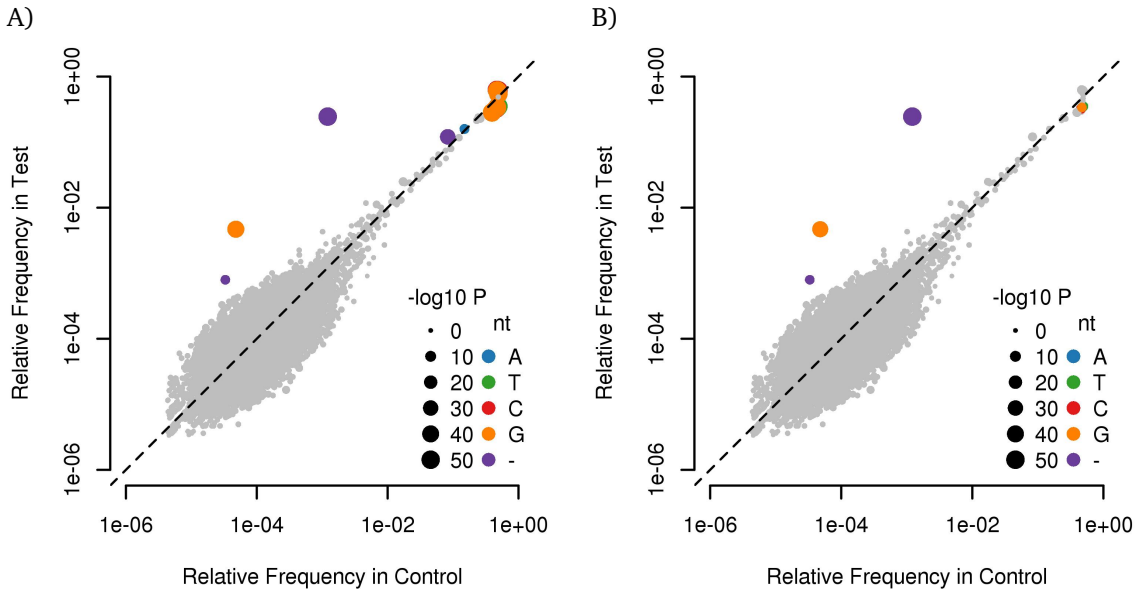


Figure 2: Scatterplot of the RCC data in a binomial (A) and beta-binomial model (B).

```
[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
```

```
>         jpeg("deepSNV-RCC-bin.jpg", 2.5, 2.5, units="in", res=600, pointsize=7)
>         par(mar=c(4,4,1,1), bty="n")
>         plot(RCC, cex.min=.5)
>         dev.off()
```

We see that a binomial model was used to generate the data. An inspection of the plot in Figure 2A, shows a long noise tail where apparently the dispersion is underestimated causing some false positives. We use a beta-binomial model instead and conservatively estimate the dispersion factor on both sides with the argument `alternative="two.sided"`:

```
> RCC.bb = estimateDispersion(RCC, alternative="two.sided")

Note: The initial object used a binomial model. Will be changed to beta-binomial.
Estimated dispersion factor 136.926849574802
```

```
>         jpeg("deepSNV-RCC-bb.jpg", 2.5, 2.5, units="in", res=600, pointsize=7)
>         par(mar=c(4,4,1,1), bty="n")
>         plot(RCC.bb, cex.min=.5)
>         dev.off()
```

The plot is shown in Figure2B. 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
1   chr3 10167762   T   C  0.000000e+00 -0.1323133971
2   chr3 10167672   A   G  0.000000e+00  0.1399872390
3   chr3 10163206   T   -  0.000000e+00  0.2440057869
4   chr3 10168683   T   G 2.134656e-304 -0.1324653452
5   chr3 10167709   C   T 6.203881e-304 -0.1440679712
6   chr3 10163012   -   C 2.512916e-211  0.1242483980
7   chr3 10166943   G   A 1.406981e-158 -0.1325781118
8   chr3 10158274   C   T 6.596999e-152 -0.1420016772
9   chr3 10163428   T   G  1.026389e-79 -0.1110488166
10  chr3 10166219   G   C  6.538408e-74  0.1591538377
11 chr17  7513782   A   G  6.137099e-57  0.0523114590
12  chr3 10167220   C   G  2.960126e-36  0.0046430287
13  chr3 10158255   A   -  7.414419e-27  0.0374533958
14  chr3 10158337   G   A  8.455105e-09 -0.1457719757
15 chr10 89710231   C   T  1.812318e-06 -0.0521225577
16 chr17  7512879   G   A  4.923709e-03  0.0095654638
17  chr3 10163208   G   -  1.908223e-02  0.0007613176
```

compared to

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

```
   chr      pos ref var         p.val      freq.var
1 chr3 10163206   T   - 6.777306e-304  0.2440057869
2 chr3 10167220   C   G  5.481366e-30  0.0046430287
3 chr3 10158274   C   T  4.185423e-03 -0.1420016772
4 chr3 10167762   T   C  8.715540e-03 -0.1323133971
5 chr3 10166943   G   A  9.499875e-03 -0.1325781118
6 chr3 10163208   G   -  2.829051e-02  0.0007613176
7 chr3 10167709   C   T  3.466312e-02 -0.1440679712
8 chr3 10168683   T   G  4.237323e-02 -0.1324653452
```

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 chr3:10163206T on 24.4% of the alleles that truncates the VHL protein. The second is a C>G conversion in the 5′-UTR of the *VHL* gene at chr3:10167220C in 0.46% of the alleles. The third variant is likely to be an alignment artifact resulting from imperfect alignments of the deletion of chr3:10163206T.

5 sessionInfo()

- R version 2.15.1 (2012-06-22), x86_64-unknown-linux-gnu
- Locale: LC_CTYPE=en_US.UTF-8, LC_NUMERIC=C, LC_TIME=en_US.UTF-8, LC_COLLATE=C, LC_MONETARY=en_US.UTF-8, LC_MESSAGES=en_US.UTF-8, LC_PAPER=C, LC_NAME=C, LC_ADDRESS=C, LC_TELEPHONE=C, LC_MEASUREMENT=en_US.UTF-8, LC_IDENTIFICATION=C
- Base packages: base, datasets, grDevices, graphics, methods, splines, stats, stats4, utils
- Other packages: BiocGenerics 0.4.0, Biostrings 2.26.0, GenomicRanges 1.10.0, IRanges 1.16.0, Rsamtools 1.10.0, VGAM 0.9-0, deepSNV 1.4.0
- Loaded via a namespace (and not attached): bitops 1.0-4.1, parallel 2.15.1, tools 2.15.1, zlibbioc 1.4.0