appreci8R – an R/Bioconductor package for filtering SNVs and short indels with high sensitivity and high PPV

Sarah Sandmann

June 28, 2018

Contents

1 Introduction

For the use of next-generation sequencing in clinical routine valid variant calling results are crucial. However, numerous variant calling tools are available. These tools usually differ in the variant calling algorithsms, the characteristics reported along with the varaint calls, the recommended filtration strategies for the raw calls and thus, also in the output. Especially when calling variants with a low variant allele frequency (VAF), perfect results are hard to obtain. High sensitivity is usually accompanied by low positive predictive value (PPV).

appreci8R is a package for combining and filterating the output of differen variant calling tools according to the 'appreci8'-algorithm [3]. vcf as well as txt files containing variant calls can be evaluated. The number of variant calling tools to consider is unlimited (for the user interface version it is limited to 13). The final output contains a list of variant calls, classified as "probably true", "polymophism" or "artifact".

Important note: Currently, only hg19 is supported.

1.1 Loading the package

The package can be downloaded and installed with

```
> biocLite("appreci8R")
```

After installation, the package can be loaded into R by typing

```
> library(appreci8R)
```

into the R console.

appreci8R requires the R-packages *shiny*, *shinyjs*, *DT*, *VariantAnnotation*, *BSgenome*, *BSgenome.Hsapiens.UCSC.hg1*, *TxDb.Hsapiens.UCSC.hg19.knownGene*, *Homo.sapiens*, *SNPlocs.Hsapiens.dbSNP144.GRCh37*, *XtraSNPlocs.Hsapiens.d*, *rsnps*, *Biostrings*, *MafDb.1Kgenomes.phase3.hs37d5*, *MafDb.ExAC.r1.0.hs37d5*, *MafDb.ESP6500SI.V2.SSA137.hs37d5*, *MafDb.gnomADex.r2.0.1.hs37d5*, *COSMIC.67*, *rentrez*, *PolyPhen.Hsapiens.dbSNP131*, *SIFT.Hsapiens.dbSNP137*, *seqinr*, *openxlsx*, *Rsamtools*, *stringr*, *utils*, *stats*, *GenomicRanges*, *S4Vectors*, *GenomicFeatures*, *IRanges*, *VariantFiltering* and *SummarizedExperiment*. All of them are loaded automatically when using *appreci8R*.

2 appreci8R – classical

2.1 1st analysis step: Target filtration

In the 1st analysis step, all off-target calls are excluded from further analysis.

The function `filterTarget` covers two steps: reading input and target filtration. Available arguments are:

- `output_folder` The folder to write the output files into. If an empty string is provided, no files are written out.
- `caller_name` Name of the variant calling tool (only necessary if an output folder is provided).
- `caller_folder` Folder containing the variant calling results.
- `caller_file_names_add` Suffix for naming the variant calling files. If an empty string is provided, it is assumed that the files only contain the sample name, e.g. "Sample1.vcf".
- `caller_file_type` File type of the variant calling results (".vcf" or ".txt").
- `caller_snv_indel` SNVs and indels are reported in the same file (`TRUE` or `FALSE`).
- `caller_snv_names_add` Suffix for naming the variant calling files containing SNVs (only evaluated if `caller_snv_indel==TRUE`).
- `caller_indel_names_add` Suffix for naming the variant calling files containing indels (only evaluated if `caller_snv_indel==TRUE`).
- `caller_chr` Column of the variant calling input containing information on chr (default: 1).

- `caller_pos` Column of the variant calling input containing information on pos (default: 2).
- `caller_ref` Column of the variant calling input containing information on ref (default: 4).
- `caller_alt` Column of the variant calling input containing information on alt (default: 5).
- `targetRegions` Data.frame object containing the target regions to be analyzed (bed-format: 1st column chr, 2nd column 0-based start pos, 3rd column 1-based end pos).

First, all files in `caller_folder` of the file type `caller_file_type` with the suffix `caller_file_names_add` are read. Sample names are automatically derived from the file names (e.g. a sample name would be called "Sample1" if a file was called "Sample1.txt" and no suffix was defined; a sample would be called "Sample1.mutations" if a file was called "Sample1.mutations.vcf" and no suffix was defined, but "Sample1" if the suffix ".mutations" was defined).

If SNVs and indels are reported in separated files (in the same folder), `caller_snv_indel==TRUE` and `caller_snv_names_add` and `caller_indel_names_add` are defined, input from two files per sample is read and automatically combined (e.g. a sample would be called "Sample1" if files "Sample1.SNV.vcf" and "Sample1.indel.vcf" are read and `caller_snv_indel==TRUE`, `caller_snv_names_add` was defined as ".SNV" and `caller_indel_names_add` was defined as ".indel").

Subsequently, the read variant calling results are filtered according to the defined target region. All off-target calls are excluded from further analysis. A list of data.frames (one list per sample) with all on-target calls is returned. Every data.frame contains the columns: the SampleID (taken from the input file names), Chr, Pos, Ref and Alt.

Exemplary filtration of two vcf-files `Sample1.rawMutations.vcf` and `Sample2.rawMutations.vcf`:

```
> output_folder<-""
> target<-data.frame(chr = c("2","4","12","17","21","X"),
+                    start = c(25469500,106196950,12046280,7579470,36164400,15838363),
+                    end = c(25469510,106196960,12046350,7579475,36164410,15838366))
> target
```

```
  chr     start       end
1   2  25469500  25469510
2   4 106196950 106196960
3  12  12046280  12046350
4  17   7579470   7579475
5  21  36164400  36164410
6   X  15838363  15838366
```

```
> caller_folder <- system.file("extdata", package = "appreci8R")
> #Input data Sample1:
> read.table(paste(caller_folder, "/Sample1.rawMutations.vcf", sep=""),
+            header = FALSE, sep = "\t", stringsAsFactors = FALSE)
```

```
  V1       V2        V3 V4 V5      V6 V7 V8             V9                              V10
1  2 25469502 rs2276598  C  T 3432.77  . NA GT:AD:DP:GQ:PL 0/1:111,113:224:99:3461,0,3634
2 12 12044195 rs1058028  T  C 2953.77  . NA GT:AD:DP:GQ:PL   0/1:91,97:188:99:2982,0,3095
3 17  7579472 rs1042522  G  C 3074.77  . NA GT:AD:DP:GQ:PL      1/1:0,75:75:99:3103,224,0
4  X 15838366 rs2301724  C  T 3574.77  . NA GT:AD:DP:GQ:PL 0/1:161,106:267:99:3603,0,5853
```

```
> #Input data Sample2:
> read.table(paste(caller_folder,"/Sample2.rawMutations.vcf",sep=""),
+            header = FALSE, sep = "\t", stringsAsFactors = FALSE)

  V1        V2         V3 V4    V5      V6      V7 V8             V9                              V10
1  4 106196951  rs2454206  A     G 3135.77       . NA GT:AD:DP:GQ:PL   0/1:84,86:170:99:3164,0,3076
2 12  12046289 rs10649660  C CAAAG   15.85 LowQual NA GT:AD:DP:GQ:PL       0/1:52,5:57:53:53,0,2353
3 12  12046341  rs2855748  A     G 2711.77       . NA GT:AD:DP:GQ:PL   0/1:94,81:175:99:2740,0,3467
4 17   7579472  rs1042522  G     C 1264.77       . NA GT:AD:DP:GQ:PL       1/1:0,31:31:93:1293,93,0
5 20  31025689 rs41289852  G     C 4098.77       . NA GT:AD:DP:GQ:PL 0/1:120,120:240:99:4127,0,4038
6 21  36164405 rs13051066  G     T  270.77       . NA GT:AD:DP:GQ:PL       0/1:4,10:14:99:299,0,119

> filterTarget(output_folder, "GATK", caller_folder, ".rawMutations",
+              ".vcf", TRUE, "", "", targetRegions = target)

[[1]]
  SampleID Chr      Pos Ref Alt
1  Sample1   2 25469502   C   T
3  Sample1  17  7579472   G   C
4  Sample1   X 15838366   C   T

[[2]]
  SampleID Chr       Pos Ref   Alt
1  Sample2   4 106196951   A     G
2  Sample2  12  12046289   C CAAAG
3  Sample2  12  12046341   A     G
4  Sample2  17   7579472   G     C
6  Sample2  21  36164405   G     T
```

2.2 2nd analysis step: Normalization

In the 2nd analysis step, all calls are normalized with respect to reporting indels, MNVs, reporting of several alternate alleles and reporting of complex indels. Available arguments are:

- `output_folder` The folder to write the output files into. If an empty string is provided, no files are written out.
- `caller_name` Name of the variant calling tool (only necessary if an output folder is provided).
- `target_calls` List of data.frames. One list element per sample. FilterTarget()-output can directly be taken as input.
- `caller_indels_pm` Deletions are reported with a "minus" (e.g. C > -A), insertions are reported with a "plus" (e.g. C > +A) (`TRUE` or `FALSE`).
- `caller_mnvs` MNVs are reported (e.g. CA > GT instead of C > G and A > T; or CGAG > TGAT instead of C > T and G > T) (`TRUE` or `FALSE`).

The function `normalize` covers two to four normalization steps:

1. Check alternative bases: Calls containing a "comma" are split up. A call like C > A,G is converted to C > A and an additional C > G call. This enables evaluation of the output of different callers while caller1 reports C > A,G, caller2 only reports C > A and caller3 only reports C > G. This normalization step is always performed.

2. Find string differences: Calls are checked for un-mutated bases. The smallest option of reporting a variant at the left-most position is chosen. For example, CAAAC > CAAC is converted to CA > C. This normalization step is always performed.

3. Convert indels: If deletions are reported with a "minus" and insertions are reported with a "plus", these are converted. An deletion like C > -G is converted to CG > C, while an insertion like C > +G is converted to C > CG. This normalization step is only performed if `caller_indels_pm` is `TRUE`.

4. Convert MNVs: If MNVs are reported, these are converted. This enables evaluation of the output of different callers if not all callers report all mutations being part of an MNV. A call like CA > GT is split up to a C > G and an A > T variant. But also a call like CGAG > TGAT is split up to C > T and G > T (G > G and A > A are not reported as they do not pass the normalization step "Find string differences"). This normalization step is only performed if `caller_mnvs` is `TRUE`.

A data.frame with all normalized calls is returned (SampleID, Chr, Pos, Ref, Alt). If an output folder is provided, the data.frame is saved as <caller_name>.normalized.txt.

Exemplary normalization of a set of raw calls:

```
> output_folder <- ""
> caller_name <- ""
> sample1 <- data.frame(SampleID = c("Sample1","Sample1","Sample1"),
+                       Chr = c("2","17","X"),
+                       Pos = c(25469502,7579472,15838366),
+                       Ref = c("CAG","G","C"),
+                       Alt = c("TAT","C","T,A"),
+                       stringsAsFactors = FALSE)
> sample2 <- data.frame(SampleID = c("Sample2","Sample2","Sample2","sample2"),
+                       Chr = c("4","12","12","21"),
+                       Pos = c(106196951,12046289,12046341,36164405),
+                       Ref = c("A","C","A","GGG"),
+                       Alt = c("G","+AAAG","G","TGG"),
+                       stringsAsFactors = FALSE)
> #Input:
> input <- list(sample1, sample2)
> input

[[1]]
  SampleID Chr      Pos Ref Alt
1  Sample1   2 25469502 CAG TAT
2  Sample1  17  7579472   G   C
3  Sample1   X 15838366   C T,A
```

```
[[2]]
  SampleID Chr       Pos Ref   Alt
1  Sample2   4 106196951   A     G
2  Sample2  12  12046289   C +AAAG
3  Sample2  12  12046341   A     G
4  sample2  21  36164405 GGG   TGG

> normalized <- appreci8R::normalize(output_folder, caller_name, input,
+                                    TRUE, TRUE)
> normalized

   SampleID Chr       Pos Ref   Alt
2   Sample1  17   7579472   G     C
3   Sample1   X  15838366   C     T
4   Sample2   4 106196951   A     G
5   Sample2  12  12046289   C CAAAG
6   Sample2  12  12046341   A     G
31  Sample1   X  15838366   C     A
1   Sample1   2  25469502   C     T
7   sample2  21  36164405   G     T
12  Sample1   2  25469504   G     T
```

2.3 3rd analysis step: Annotation

In the 3rd analysis step, all calls are annotated (using VariantAnnotation) and filtered according to the selected locations and consequences of interest.

The function `annotate` performs annotations of all variants in the input data.frame (according to VariantAnnotation [2]) and filteres the annotated calls with respect to the selected locations and consequences of interest (based on the functions `locateVariants` and `predictCoding` of the package VariantAnnotation). At least one location of interest has to be chosen. If - among others - "coding" is selected, at least one consequence of interest has to be chosen as well. Avilable input arguments are:

- `output_folder` The folder to write the output files into. If an empty string is provided, no files are written out.
- `caller_name` Name of the variant calling tool (only necessary if an output folder is provided).
- `normalized_calls` A data.frame object. One normalized variant call per line. Necessary columns are: SampleID, Chr, Pos, Ref, Alt. Normalize()-output can directly be taken as input.
- `locations` A vector containing the locations of interest. Accepted values are: coding, intron, threeUTR, fiveUTR, intergenic, spliceSite, promoter.
- `consequences` A vector containing the consequences of interest. Accepted values are: synonymous, nonsynonymous, frameshift, nonsense, not translated.

All possible transcripts are investigated. If "coding" is chosen and a variant is annotated as "coding" in only 1 out of many transcripts, the matching transcript is, together with the annotation information, reported. All the other non-matching transcripts are not reported.

A data.frame is returned containing only the calls at the selected locations of interest and with the selected consequences of interest. Reported columns are: SampleID, Chr, Pos, Ref, Alt, Location, c. (position of variant on cDNA level), p. (position of variant on protein level), AA_ref, AA_alt, Codon_ref, Codon_alt, Consequence, Gene, GeneID, TranscriptID. TxDb.Hsapiens.UCSC.hg19.knownGene is used for annotation. Whenever there is more than one transcriptID, all of them are reported, separated by commas. The first transcriptID matches the first entry under Location, c., p. etc.

If an output folder is provided, the data.frame is saved as <caller_name>.annotated.txt.

Exemplary annotation of a set of normalized calls:

```
> output_folder <- ""
> caller_name <- ""
> input <- data.frame(SampleID = c("Sample1","Sample1","Sample2","Sample2"),
+                     Chr =c("2","X","4","21"),
+                     Pos = c (25469504,15838366,106196951,36164405),
+                     Ref = c("G","C","A","G"),
+                     Alt = c("T","A","G","T"), stringsAsFactors=FALSE)
> #Input:
> input

  SampleID Chr       Pos Ref Alt
1  Sample1   2  25469504   G   T
2  Sample1   X  15838366   C   A
3  Sample2   4 106196951   A   G
4  Sample2  21  36164405   G   T

> library(GO.db)
> library(org.Hs.eg.db)
> library(TxDb.Hsapiens.UCSC.hg19.knownGene)
> library(Biostrings)
> annotated <- annotate(output_folder, caller_name, input,
+               locations = c("coding","spliceSite"),
+               consequences = c("nonsynonymous","frameshift","nonsense"))
> annotated

  SampleID Chr       Pos Ref Alt             Location            c.          p. AA_ref AA_alt
1  Sample1   2  25469504   G   T coding,coding,coding 697,1264,1264 233,422,422  L,L,L  M,M,M
2  Sample1   X  15838366   C   A               coding           864         288      N      K
3  Sample2   4 106196951   A   G        coding,coding     5284,5347   1762,1783    I,I    V,V
    Codon_ref   Codon_alt                                Consequence                 Gene
1 CTG,CTG,CTG ATG,ATG,ATG nonsynonymous,nonsynonymous,nonsynonymous DNMT3A,DNMT3A,DNMT3A
2         AAC         AAA                             nonsynonymous               ZRSR2
3     ATA,ATA     GTA,GTA               nonsynonymous,nonsynonymous            TET2,TET2
          GeneID      TranscriptID
1 1788,1788,1788 10808,10809,10810
```

```
2           8233             75467
3    54790,54790       18308,18309
```

2.4 4th analysis step: Output combination

In the 4th analysis step, all normalized and annotated calls of the different variant calling tools are combined.

The function `combineOutput` performs a combination of the normalized and annotated variant calls from different variant calling tools. The results are sorted according to Chr, Pos, Ref, Alt and SampleID. If two callers - caller1 and caller2 - report the same variant for the same sample, it is only reported once in the output file with a "1" in the column "caller1" and another "1" in the column "caller2". Available arguments are:

- `output_folder` The folder to write the output files into. If an empty string is provided, no files are written out.
- `caller_names` Vector containing the name of the variant calling tools.
- `annotated_calls` A list of data.frames. Every list element contains the variant calls of one variant calling tool. Annotate()-output can directly be taken as input. Providing a list with only one list element, i.e. evaluating only one variant calling tool, is possible, but not assumed to be useful. The point of the appreci8-algorithm lies in the combined evaluation and filtration of the output of several variant calling tools.

A data.frame is returned containing all combined calls. Reported columns are: SampleID, Chr, Pos, Ref, Alt, Location, c. (position of variant on cDNA level), p. (position of variant on protein level), AA_ref, AA_alt, Codon_ref, Codon_alt, Consequence, Gene, GeneID, TranscriptID. In addition, one column for every variant calling tool is reported, containing a "1" for every call detected by that tool and "NA" for every call not detected by that tool. If an output folder is provided, the data.frame is saved as Results_Raw.txt.

Exemplary combination of a set of annotated calls:

```
> output_folder <- ""
> caller_name <- ""
> #Input:
> gatk <- annotated[c(1,3),]
> varscan <- annotated[c(2,3),]
> annotated<-list()
> annotated[[1]]<-gatk
> annotated[[2]]<-varscan
> annotated

[[1]]
  SampleID Chr       Pos Ref Alt             Location            c.          p. AA_ref AA_alt
1  Sample1   2  25469504   G   T coding,coding,coding 697,1264,1264 233,422,422  L,L,L  M,M,M
3  Sample2   4 106196951   A   G        coding,coding     5284,5347   1762,1783    I,I    V,V
    Codon_ref   Codon_alt                                Consequence                Gene
1 CTG,CTG,CTG ATG,ATG,ATG nonsynonymous,nonsynonymous,nonsynonymous DNMT3A,DNMT3A,DNMT3A
```

```
3     ATA,ATA     GTA,GTA               nonsynonymous,nonsynonymous            TET2,TET2
          GeneID      TranscriptID
1 1788,1788,1788 10808,10809,10810
3    54790,54790       18308,18309

[[2]]
  SampleID Chr       Pos Ref Alt      Location        c.        p. AA_ref AA_alt Codon_ref
2  Sample1   X  15838366   C   A        coding       864       288      N      K       AAC
3  Sample2   4 106196951   A   G coding,coding 5284,5347 1762,1783    I,I    V,V   ATA,ATA
  Codon_alt                Consequence      Gene      GeneID TranscriptID
2       AAA               nonsynonymous     ZRSR2        8233        75467
3   GTA,GTA nonsynonymous,nonsynonymous TET2,TET2 54790,54790  18308,18309

> combined<-combineOutput("", c("GATK","VarScan"), annotated)
> combined

   SampleID Chr       Pos Ref Alt             Location            c.          p. AA_ref AA_alt
11  Sample1   2  25469504   G   T coding,coding,coding 697,1264,1264 233,422,422  L,L,L  M,M,M
3   Sample2   4 106196951   A   G        coding,coding     5284,5347   1762,1783    I,I    V,V
2   Sample1   X  15838366   C   A               coding           864         288      N      K
     Codon_ref   Codon_alt                                  Consequence                Gene
11 CTG,CTG,CTG ATG,ATG,ATG nonsynonymous,nonsynonymous,nonsynonymous DNMT3A,DNMT3A,DNMT3A
3      ATA,ATA     GTA,GTA               nonsynonymous,nonsynonymous            TET2,TET2
2          AAC         AAA                           nonsynonymous                ZRSR2
           GeneID      TranscriptID GATK VarScan
11 1788,1788,1788 10808,10809,10810    1      NA
3     54790,54790       18308,18309    1       1
2            8233             75467   NA       1
```

2.5 5th analysis step: Coverage and BQ determination

In the 5th analysis step, all calls are evaluated with respect to coverage and basequality (using Rsamtools [1]). Calls with insufficient coverage and/or basequality are filtered from further analysis.

The function `evaluateCovAndBQ` evaluates coverage and base quality of all calls and filteres them according to user-definable thresholds. Only those calls that feature sufficient coverage and base quality are reported. Filtration is performed in a 2-step procedure. Available arguments are:

- `output_folder` The folder to write the output files into. If an empty string is provided, no files are written out.
- `combined_calls` A data.frame; necessary input object. The data.frame contains one call per line. CombineOutput()-output can directly be taken as input.
- `bam_folder` The folder containing the bam- and bai-files for every sample. The file names have to match the sample names, i.e. Sample1.bam and Sample1.bai if Sample1.vcf (without any defined suffixes) was analyzed for caller1.
- `dp` Minimum depth that is required for a call to pass the filter (default: 50).

- `nr_alt` Minimum number of reads carrying the alternate allele that is required for a call to pass the filter (default: 20).
- `vaf` Minimum VAF that is required for a call to pass the filter (default: 0.01).
- `bq` Minimum base quality that is required for a call to pass the filter (default: 15; values above 44 not recommended).
- `bq_diff` Maximum difference in basequality between reference and alternative that is allowed for a call to pass the filter (default: 7).

First, coverage is evaluated using Rsamtools `pileup` and a threshold of `min_base_quality=0`. If depth is below the defined threshold `dp` and/or the number of reads carrying the alternate allele is below the defined threshold `nr_alt` and/or VAF is below the defined threshold `VAF`, a call is excluded. However, if coverage is sufficient, base quality - which is more time-consuming - is evaluated.

To evaluate base quality, the threshold for `min_base_quality` is successively increased from 0 to 44 (thus, no values above 44 are recommended for `bq`). The average base quality for all reads carrying the reference and the alternate allele is calculated. For indels, which are always summed up as "+" by Rsamtools, no base quality can be determined. If the average base quality of the alternate allele is lower than `bq`, the call is filtered. If "average base quality reference allele" - "average base quality alternate allele" is higher than `bq_diff`, the call is filtered.

Coverage is reported with respect to the forward- and reverse reads separately. It is not yet evaluated. This is done in the 7th analysis step (`finalFiltration`).

A data.frame is returned containing all calls with sufficient coverage and base quality. Reported columns are: SampleID, Chr, Pos, Ref, Alt, Location, c. (position of variant on cDNA level), p. (position of variant on protein level), AA_ref, AA_alt, Codon_ref, Codon_alt, Consequence, Gene, GeneID, TranscriptID, Caller1 to CallerX (dependent on the number of callers that is evaluated), Nr_Ref, Nr_Alt, DP, VAF, BQ_REF, BQ_ALT, Nr_Ref_fwd, Nr_Alt_fwd, DP_fwd, VAF_rev, Nr_Ref_rev, Nr_Alt_rev, DP_rev, VAF_rev. If an output folder is provided, the data.frame is saved as Results_Frequency.txt.

Exemplary filtration of a set of combined calls:

```
> output_folder <- ""
> bam_folder <- system.file("extdata/", package = "appreci8R")
> #Input:
> combined
```

```
   SampleID Chr       Pos Ref Alt             Location            c.          p. AA_ref AA_alt
11  Sample1   2  25469504   G   T coding,coding,coding 697,1264,1264 233,422,422  L,L,L  M,M,M
3   Sample2   4 106196951   A   G        coding,coding     5284,5347   1762,1783    I,I    V,V
2   Sample1   X  15838366   C   A               coding           864         288      N      K
     Codon_ref   Codon_alt                              Consequence                 Gene
11 CTG,CTG,CTG ATG,ATG,ATG nonsynonymous,nonsynonymous,nonsynonymous DNMT3A,DNMT3A,DNMT3A
3      ATA,ATA     GTA,GTA               nonsynonymous,nonsynonymous            TET2,TET2
2          AAC         AAA                            nonsynonymous                ZRSR2
           GeneID      TranscriptID GATK VarScan
11 1788,1788,1788 10808,10809,10810    1      NA
```

```
3     54790,54790       18308,18309    1       1
2            8233             75467   NA       1

> filtered<-evaluateCovAndBQ(output_folder, combined, bam_folder, bq_diff = 20,
+                            vaf = 0.001)
> filtered

  SampleID Chr       Pos Ref Alt      Location        c.        p. AA_ref AA_alt Codon_ref
3  Sample2   4 106196951   A   G coding,coding 5284,5347 1762,1783    I,I    V,V   ATA,ATA
2  Sample1   X  15838366   C   A        coding       864       288      N      K       AAC
  Codon_alt                 Consequence      Gene      GeneID TranscriptID GATK VarScan Nr_Ref
3   GTA,GTA nonsynonymous,nonsynonymous TET2,TET2 54790,54790  18308,18309    1       1   1268
2       AAA               nonsynonymous     ZRSR2        8233        75467   NA       1   1991
  Nr_Alt   DP         VAF   BQ_REF  BQ_ALT Nr_Ref_fwd Nr_Alt_fwd DP_fwd  VAF_fwd Nr_Ref_rev
3   1283 2551 0.502940024 38.66798 38.8145        428        469    897 0.522854        840
2     31 3630 0.008539945 37.45907 18.0000        839          0   1507 0.000000       1152
  Nr_Alt_rev DP_rev    VAF_rev
3        814   1654 0.49214027
2         31   2123 0.01460198
```

2.6 6th analysis step: Characteristics determination

In the 6th analysis step, an extended set of characteristics for all calls is determined. This includes database check-ups and impact prediction on protein level.

The function `evaluateCharacteristics` determines an extended set of characteristics for the normalized, annotated, coverage and base quality filtered calls. This includes database check-ups and impact prediction on protein level. Available arguments are:

- `output_folder` The folder to write the output files into. If an empty string is provided, no files are written out.
- `frequency_calls` A data.frame object. The data.frame contains one call per line. EvaluateCovAndBQ()-output can directly be taken as input.
- `predict` String defining the prediction tool to use (accepted strings are: SIFT, Provean, PolyPhen2)
- `dbSNP` Query dbSNP (144.GRCh37) for the presence of your variants (`TRUE` or `FALSE`; default `TRUE`).
- `'1kgenomes'` Query 1000 Genomes (phase3) for the presence of your variants (`TRUE` or `FALSE`; default `TRUE`).
- `exacDB` Query ExAC (r1.0) for the presence of your variants (`TRUE` or `FALSE`; default `TRUE`).
- `espDB` Query ESP6500 (V2) for the presence of your variants (`TRUE` or `FALSE`; default `TRUE`).
- `gadDB` Query Genome Aggregation Database (r2.0.1) for the presence of your variants (`TRUE` or `FALSE`; default `TRUE`).

- `cosmicDB` Query COSMIC (67) for the presence of your variants (`TRUE` or `FALSE`; default `TRUE`).
- `clinvarDB` Query ClinVar (via rentrez) for the presence of your variants (`TRUE` or `FALSE`; default `TRUE`).

First, the database check-up is performed. At a maximum, dbSNP, 1000Genomes, ExAC, ESP6500, Genome Aggregation Database, COSMIC and ClinVar can be checked. If all the variables are set to `FALSE`, no database is checked and none will be reported. Thus, the check-up can be disabled.

Second, the impact perdiction on protein level is performed. Impact can be predicted using either SIFT, Provean or PolyPhen2. In any case, the prediction is performed on the basis of rs-IDs. For every call, the predicted effect as well as the score is reported.

A data.frame is returned containing all calls, information on their presence in the selected databases and information on the predicted effect. Reported columns are (if all databases are selected): SampleID, Chr, Pos, Ref, Alt, dbSNP (containing the rs-ID), dbSNP_MAF, G1000_AF, ExAC_AF, ESP6500_AF, GAD_AF, CosmicID, Cosmic_Counts (number of Cosmic entries), ClinVar, Prediction (damaging or neutral), Score (on the basis of the selected prediction tool), c. (variant on cDNA level containing position and variant), c.complement (complement of the variant; if c. is c.5284A>G, then c.complement is c.5284T>C), p. (variant on protein level containing position and amino acids). If an output folder is provided, the data.frame is saved as Results_Databases.txt.

Exemplary characteristics of a set of filtered calls:

```
> output_folder <- ""
> #Input:
> filtered

  SampleID Chr       Pos Ref Alt      Location        c.        p. AA_ref AA_alt Codon_ref
3  Sample2   4 106196951   A   G coding,coding 5284,5347 1762,1783    I,I    V,V   ATA,ATA
2  Sample1   X  15838366   C   A        coding       864       288      N      K       AAC
  Codon_alt                 Consequence      Gene      GeneID TranscriptID GATK VarScan Nr_Ref
3   GTA,GTA nonsynonymous,nonsynonymous TET2,TET2 54790,54790  18308,18309    1       1   1268
2       AAA               nonsynonymous     ZRSR2        8233        75467   NA       1   1991
  Nr_Alt   DP         VAF   BQ_REF  BQ_ALT Nr_Ref_fwd Nr_Alt_fwd DP_fwd  VAF_fwd Nr_Ref_rev
3   1283 2551 0.502940024 38.66798 38.8145        428        469    897 0.522854        840
2     31 3630 0.008539945 37.45907 18.0000        839          0   1507 0.000000       1152
  Nr_Alt_rev DP_rev    VAF_rev
3        814   1654 0.49214027
2         31   2123 0.01460198

> characteristics <- determineCharacteristics(output_folder, filtered,
+                                             predict = "Provean")
> characteristics

  SampleID Chr       Pos Ref Alt     dbSNP dbSNP_MAF G1000_AF ExAC_AF ESP6500_AF GAD_AF CosmicID
3  Sample2   4 106196951   A   G rs2454206    0.2304     0.23    0.27       0.29   0.30       NA
2  Sample1   X  15838366   C   A      <NA>        NA     0.44    0.47       0.49   0.47       NA
  Cosmic_Counts ClinVar Prediction  Score                 c.        c.complement                p.
```

```
3            NA      NA    Neutral -0.061 c.5284A>G,c.5347A>G c.5284T>C,c.5347T>C p.I1762V,p.I1783V
2            NA      NA       <NA>     NA            c.864C>A            c.864G>T           p.N288K
```

2.7 7th analysis step: Final filtration

In the 7th analysis step, the final filtration according to the appreci8-algorithm is performed.

The function `finalFiltration` performs the final filtration according to the appreci8-algorithm. The previously determined characteristics of the calls are evaluated and an automatic categorization of the calls is performed. Possible categories are: Probably true or Hotspot, Polymorphism and Artifact. Available arguments are:

- `output_folder` The folder to write the output files into. If an empty string is provided, no files are written out.
- `frequency_calls` A data.frame object. The data.frame contains one call per line. EvaluateCovAndBQ()-output can directly be taken as input.
- `database_calls` A data.frame object. The data.frame contains one call per line. EvaluateCharacteristics()-output can directly be taken as input.
- `combined_calls` A data.frame object. The data.frame contains one call per line. CombineOutput()-output can directly be taken as input.
- `damaging_safe` Threshold for the impact prediction score, identifying reliably damaging calls. For example, if Provean has been selected as the impact prediction tool, the internal threshold of the tool is -2.5. However, calls with a score of -2.51 are less likely to have a correct "Damaging" prediction compared to calls with a score of -12.0. A threshold of -5.0 for `damaging_safe` marks all calls with a score below -5.0 as reliably damaging.
- `tolerated_safe` Threshold for the impact prediction score, identifying reliably tolerated calls. For example, if Provean has been selected as the impact prediction tool, the internal threshold of the tool is -2.5. However, calls with a score of -2.49 are less likely to have a correct "Neutral" prediction compared to calls with a score of +7.0. A threshold of -1.0 for `tolerated_safe` marks all calls with a score above -1.0 as reliably tolerated.
- `primer` Optional: Data.frame containing primer positions in bed-format, i.e. 1st column chromosome, 2nd column 0-based start position, 3rd column 1-based end position (default: NA).
- hotspots Optional: Data.frame containing expected/hotspot mutations: 1st column gene name (e.g. ASXL1), 2nd column mutation on AA level (e.g. G12S or E628fs or I836del or G12 if any AA change at this position is considered an expected/hotspot mutation), 3rd column minimum VAF or NA (default: NA).
- `overlapTools` Vector of strings containing the names of variant calling tools that, should a call be overlappingly reported by these tools, will be rewarded.
- `nrsamples` Threshold for the number of samples. Shold a variant be detected in more than the threshold defined by `nrsamples`, it will be interpreted as possible evidence for an artifact (default: 3).

- `dp` Minimum depth that is required for a call (default: 50).
- `nr_alt` Minimum number of reads carrying the alternate allele that is required for a call (default: 20).
- `vaf` Minimum VAF that is required for a call (default: 0.01).
- `bq` Minimum base quality that is required for a call (default: 15; values above 44 not recommended).
- `bq_diff` Maximum difference in basequality between reference and alternative that is allowed for a call (default: 7).
- `detectedLow` Value added to the artifact score if more than `nrsamples` number of samples feature the same call (default: 2).
- `detectedHigh` Value added to the artifact score if more than 50% of the samples analyzed (if more than 1 sample is analyzed) feature the same call, which is not an expected/hotspot mutation (default: 2).
- `isIndel` Value added to the artifact score if the call is an indel (default: 1).
- `isIndelVAF` Value added to the artifact score if the call is an indel and the VAF is below 0.05 (default: 1).
- `detectedLowVAF` Value added to the artifact score if more than `nrsamples` number of samples feature the same call and the VAF is above 0.85 (default: 2).
- `noPrimerP` Value added to the artifact score if no primer information is provided or no primer is located at this position and the p value (Fisher's Exact Test evaluating forward and reverse reads to determine strandbias) is below 0.001 (default: 1).
- `primerPAlt` Value added to the artifact score if no primer information is provided or no primer is located at this position and the p value (Fisher's Exact Test evaluating forward and reverse reads to determine strandbias) is below 0.001 and Nr_Alt_fwd is at least `Nr_Alt/2` and Nr_Alt_rev is at least `Nr_Alt/1` (default: -1).
- `noPrimerPFwd` Value added to the artifact score if no primer information is provided or no primer is located at this position and the p value (Fisher's Exact Test evaluating forward and reverse reads to determine strandbias) is at least 0.001 and Nr_Ref_fwd is at least `(DP-Nr_Alt)/2` and Nr_Alt_fwd is below 3 (default: 1).
- `primerPFwd` Value added to the artifact score if no primer information is provided or no primer is located at this position and the p value (Fisher's Exact Test evaluating forward and reverse reads to determine strandbias) is below 0.001 and Nr_Ref_fwd is below `(DP-Nr_Alt)/2` and Nr_Alt_fwd is below 3 (default: -1).
- `noPrimerPRev` Value added to the artifact score if no primer information is provided or no primer is located at this position and the p value (Fisher's Exact Test evaluating forward and reverse reads to determine strandbias) is at least 0.001 and Nr_Ref_rev is at least `(DP-Nr_Alt)/2` and Nr_Alt_rev is below 3 (default: 1).

- `primerPRev` Value added to the artifact score if no primer information is provided or no primer is located at this position and the p value (Fisher's Exact Test evaluating forward and reverse reads to determine strandbias) is below 0.001 and Nr_Ref_rev is below `(DP-Nr_Alt)/2` and Nr_Alt_rev is below 3 (default: -1).
- `primerLocation` Value added to the artifact score if a call is located at a position where a primer is located (default: -1).
- `vafLow` Value added to the artifact score if the VAF is below 0.02 (default: 2).
- `databaseVAF` Value added to the artifact score if the VAF is below 0.10 and the variant was not found in any database (default: 1).
- `databaseHigh` Value added to the artifact score if the variant was detected in at least 50% of all samples, but not found in any database (default: 1).
- `predictionSafe` Value added to the artifact score if the variant has a reliable damaging prediction (default: -1).
- `predictionVAF` Value added to the artifact score if the variant has a reliable tolerated prediction and a VAF below 0.35 or between 0.65 and 0.85 (default: 1).
- `nrcaller4` Intermediate number of callers that overlappingly reported a variant (default: 4).
- `reward4` Value added to the artifact score if a variant is overlappingly reported by `nrcaller4` callers (default: -1).
- `nrcaller5` High number of callers that overlappingly reported a variant (default: 5).
- `reward5` Value added to the artifact score if a variant is overlappingly reported by `nrcaller5` callers (default: -1).
- `nrcaller6` Very high number of callers that overlappingly reported a variant (default: 6).
- `reward6` Value added to the artifact score if a variant is overlappingly reported by `nrcaller6` callers (default: -1).
- `oneCaller` Value added to the artifact score if a variant is only reported by one caller (default: 1).
- `BQ_AltMean` Value added to the artifact score if the base quality of a variant is below `mean(BQ_Alt)-3*(BQ_Alt)` over all variants (default: 4).
- `knownHotspot` Value added to the artifact score if a variant is an expected/hotspot mutation (default: -3).
- `overlapReward` Value added to the artifact score if a variant is overlappingly reported by the tools defined in `overlapTools` (default: -3).
- `artifactThreshold` Threshold for the artifact score, i.e. variants with a score below this threshold will be categorized as "probably true" (default: 0).

- `polyDetected` Value added to the polymorphism score if more than `nrsamples` number of samples feature the same call (default: 1).
- `polyDetectedOnce` Value added to the polymorphism score if a variant is only detected in one sample (default: -1).
- `polyDatabasesPolyLow` Intermediate number of polymorphism databases that have information on a variant (default: 2).
- `polyDatabasesPolyLowReward` Value added to the polymorphism score if a variant is present in at least `polyDatabasesPolyLow` databases (default: 1).
- `polyDatabasesPolyHigh` High number of polymorphism databases that have information on a variant (default: 4).
- `polyDatabasesPolyHighReward` Value added to the polymorphism score if a variant is present in at least `polyDatabasesPolyHigh` databases (default: 1).
- `polyDatabasesMut` Critical number of mutation databases that have information on a variant (default: 2).
- `polyDatabasesMutReward` Value added to the polymorphism score if a variant is present in at least `polyDatabasesMut` databases (default: -1).
- `polyNoDatabase` Value added to the polymorphism score if a variant is not present in any polymorphism database (default: -1).
- `polyDatabases` High number of databases that have information on a variant (default: 6).
- `polyDatabasesReward` Value added to the polymorphism score if a variant is present in at least `polyDatabases` databases (default: 1).
- `polyEffect` Value added to the polymorphism score if a variant is not a frameshift variant, not a stop gain variant and not a stop lost variant (default: 1).
- `polyVAF` Value added to the polymorphism score if the VAF of a variant is between 0.65 and 0.35 or above 0.85 (default: 1).
- `polyPrediction` Value added to the polymorphism score if the variant has a reliable tolerated prediction (default: 1).
- `polyPredictionEffect` Value added to the polymorphism score if a variant has a reliable damaging prediction or is a stop gain variant or is a stop lost variant (default: -1).
- `polyCosmic` Critical number of COSMIC entries (default: 100).
- `polyThresholdCritical` Threshold for the polymorphism score if the number of COSMIC entries is not critical, i.e. variants with at least this score will be categorized as "polymorphism" (default: 2).
- `polyThreshold` Threshold for the polymorphism score if the number of COSMIC entries is critical, i.e. variants with at least this score will be categorized as "polymorphism" (default: 3).

- `PolymorphismVAF10` Value added to the artifact score if a variant is a "polymorphism" based on the polymorphism score, but the VAF is below 0.10 (default: 5).
- `PolymorphismVAF20` Value added to the artifact score if a variant is a "polymorphism" based on the polymorphism score, but the VAF is below 0.20 (default: 2).
- `PolymorphismFrame` Value added to the artifact score if a variant is a "polymorphism" based on the polymorphism score, but it is a frameshift variant (default: 2).

Final filtration consists of several steps:

1. Frequency and base quality are re-considered. Stricter thresholds compared to `evaluateCovAndBQ` can be defined.
2. Samples with the same call are considered. Counting is based on the normalized and annotated calls, not on the coverage- and base quality filtered calls.
3. Samples with a call at the same position are considered (e.g. A>AG at pos 1 in sample 1 and A>AGG at pos 1 in sample 2 are reported as 2 calls at the same position). Counting is based on the normalized and annotated calls, not on the coverage- and base quality filtered calls.
4. Background information is considered. The number of samples with the same variant in the coverage- and base quality filtered data are considered.
5. Number of databases is considered. Dependent on the previously selected number of databases. The presence of a variant in mutation- and polymorphism databases is evaluated.
6. VAF in relation to a predicted effect is considered. Variants are marked if their VAF is typical of polymorphisms and their predicted effect is "tolerated".
7. VAF in relation to the number of samples is considered. Variants are marked if they are detected in more than `nrsamples` samples and the VAF is at least 0.85 in more than 90% of these samples.
8. Strandbias is considered. Fisher's Exact Test is performed to analyze the relation between forward-reverse and reference-alternate allele.
9. Hotspot list - if any is provided - is considered. Variants are marked if they are present on the hotspot list.
10. Final filtration is performed. The artifact- and the polymorphism score are calculated. On the basis of these two scores, a call is either classified as a probably true/hotspot call, polymorphism or artifact.

A data.frame is returned containing all calls with a predicted category. Categories can be: probably true, hotspot, polymorphism or artifact. Reported columns are: SampleID, Chr, Pos, Ref, Alt, Gene, GeneID, TranscriptID, Location, Consequence, c. (variant on cDNA level containing position and variant), c.complement (complement of the variant; if c. is c.5284A>G, then c.complement is c.5284T>C), p. (variant on protein level containing position and amino acids), Codon_ref, Codon_alt, Nr_Ref, Nr_Alt, DP, VAF, Caller1 to CallerX (dependent on the number of callers that is evaluated), dbSNP (containing the rs-ID), dbSNP_MAF, G1000_AF, ExAC_AF,

ESP6500_AF, GAD_AF, CosmicID, Cosmic_Counts (number of Cosmic entries), ClinVar, Prediction (damaging or neutral), Score (on the basis of the selected prediction tool), BQ_REF, BQ_ALT, Nr_Ref_fwd, Nr_Alt_fwd, DP_fwd, VAF_rev, Nr_Ref_rev, Nr_Alt_rev, DP_rev, VAF_rev, strandbias, nr_samples (number of samples with the same variant), nr_samples_similar (number of samples with a variant at the same position), Category.

If an output folder is provided, the data.frame is saved as Results_Final.txt. Additionally, Results_Final.xlsx is saved, containing three sheets: Mutations, Polymorphisms and Artifacts.

```
> output_folder <- ""
> #Input:
> filtered
```

```
  SampleID Chr       Pos Ref Alt      Location        c.        p. AA_ref AA_alt Codon_ref
3  Sample2   4 106196951   A   G coding,coding 5284,5347 1762,1783    I,I    V,V   ATA,ATA
2  Sample1   X  15838366   C   A        coding       864       288      N      K       AAC
  Codon_alt                 Consequence      Gene      GeneID TranscriptID GATK VarScan Nr_Ref
3   GTA,GTA nonsynonymous,nonsynonymous TET2,TET2 54790,54790  18308,18309    1       1   1268
2       AAA               nonsynonymous     ZRSR2        8233        75467   NA       1   1991
  Nr_Alt   DP         VAF   BQ_REF  BQ_ALT Nr_Ref_fwd Nr_Alt_fwd DP_fwd  VAF_fwd Nr_Ref_rev
3   1283 2551 0.502940024 38.66798 38.8145        428        469    897 0.522854        840
2     31 3630 0.008539945 37.45907 18.0000        839          0   1507 0.000000       1152
  Nr_Alt_rev DP_rev    VAF_rev
3        814   1654 0.49214027
2         31   2123 0.01460198
```

```
> characteristics
```

```
  SampleID Chr       Pos Ref Alt     dbSNP dbSNP_MAF G1000_AF ExAC_AF ESP6500_AF GAD_AF CosmicID
3  Sample2   4 106196951   A   G rs2454206    0.2304     0.23    0.27       0.29   0.30       NA
2  Sample1   X  15838366   C   A      <NA>        NA     0.44    0.47       0.49   0.47       NA
  Cosmic_Counts ClinVar Prediction  Score                  c.        c.complement                p.
3            NA      NA    Neutral -0.061 c.5284A>G,c.5347A>G c.5284T>C,c.5347T>C p.I1762V,p.I1783V
2            NA      NA       <NA>     NA            c.864C>A            c.864G>T           p.N288K
```

```
> combined
```

```
   SampleID Chr       Pos Ref Alt             Location            c.          p. AA_ref AA_alt
11  Sample1   2  25469504   G   T coding,coding,coding 697,1264,1264 233,422,422  L,L,L  M,M,M
3   Sample2   4 106196951   A   G        coding,coding     5284,5347   1762,1783    I,I    V,V
2   Sample1   X  15838366   C   A               coding           864         288      N      K
     Codon_ref   Codon_alt                                Consequence                 Gene
11 CTG,CTG,CTG ATG,ATG,ATG nonsynonymous,nonsynonymous,nonsynonymous DNMT3A,DNMT3A,DNMT3A
3      ATA,ATA     GTA,GTA               nonsynonymous,nonsynonymous            TET2,TET2
2          AAC         AAA                            nonsynonymous                ZRSR2
           GeneID      TranscriptID GATK VarScan
11 1788,1788,1788 10808,10809,10810    1      NA
3     54790,54790       18308,18309    1       1
2            8233             75467   NA       1
```

```
> final<-finalFiltration(output_folder, frequency_calls = filtered,
+                        database_calls = characteristics,
+                        combined_calls = combined, damaging_safe = -3,
+                        tolerated_safe = -1.5, overlapTools = c("VarScan"),
+                        bq_diff = 20,vaf = 0.001)
> final

  SampleID Chr       Pos Ref Alt      Gene      GeneID TranscriptID      Location
3  Sample2   4 106196951   A   G TET2,TET2 54790,54790  18308,18309 coding,coding
2  Sample1   X  15838366   C   A     ZRSR2        8233        75467        coding
                 Consequence                   c.        c.complement               p. Codon_ref
3 nonsynonymous,nonsynonymous c.5284A>G,c.5347A>G c.5284T>C,c.5347T>C p.I1762V,p.I1783V   ATA,ATA
2              nonsynonymous            c.864C>A           c.864G>T           p.N288K       AAC
  Codon_alt Nr_Ref Nr_Alt   DP         VAF GATK VarScan Called     dbSNP dbSNP_MAF G1000_AF ExAC_AF
3   GTA,GTA   1268   1283 2551 0.502940024    1       1      2 rs2454206    0.2304     0.23    0.27
2       AAA   1991     31 3630 0.008539945   NA       1      1      <NA>        NA     0.44    0.47
  ESP6500_AF GAD_AF CosmicID Cosmic_Counts ClinVar Prediction  Score   BQ_REF  BQ_ALT Nr_Ref_fwd
3       0.29   0.30       NA            NA      NA    Neutral -0.061 38.66798 38.8145        428
2       0.49   0.47       NA            NA      NA       <NA>     NA 37.45907 18.0000        839
  Nr_Alt_fwd DP_fwd  VAF_fwd Nr_Ref_rev Nr_Alt_rev DP_rev    VAF_rev   strandbias nr_samples
3        469    897 0.522854        840        814   1654 0.49214027 1.466935e-01          1
2          0   1507 0.000000       1152         31   2123 0.01460198 6.696552e-08          1
  nr_samples_similar     Category
3                  1 Polymorphism
2                  1 Artifact (4)
```

3 appreci8R - GUI

appreci8R combines and filters the output of different variant calling tools according to the 'appreci8'-algorithm. The whole analysis-pipeline consists of seven steps. Individual functions for executing these steps are available. However, an additional user-interface is available that enables easy execution of the analysis-pipeline, restarting parts of the pipeline from checkpoints, changing parameters, exporting and importing the configuration and evaluation of the results. The user-interface is opend by executing the function `appreci8Rshiny` (no arguments have to be provided).

The user-interface is structured by tabs: *Analysis*, *Overview results* and *Detailed results*.

- *Analysis*: The tab is essential for performing the whole analysis. It is further separated into 10 additional tabs:
  - `Important notes`: General information on prerequisites for performing the analysis and current limitations (only hg19).
  - `0. Preparations`: Defining the general output folder and the varaint caller input. GATK, Platypus, VarScan, FreeBayes, LoFreq, SNVer, SamTools and VarDict are available as these make up the classical appreci8-pipeline. The tools can be `added` or `removed`. For every tool the following parameters have to be defined: the folder containing the variant calling results, the output file type (vcf or txt; as vcf is a standardized format, the

columns containing chr, pos, ref and alt are assumed to be fixed; when selecting txt, the columns containing chr, pos, ref and alt have to be defined), a possible suffix for naming the output, if MNVs are reported (AG>CT instead of A>C and G>T), if SNVs and indels are reported in one file (if not, what is the suffix for the SNV files and what is the suffix for the indel files?), and how are indels reported. In addition to the eight tools, up to 5 additional tools can be added (select the number and click `Go`). In addition to the previous parameters, a name can be defined for each additionally added tool.

  - `1. Target filtration`: Upload your target region (bed file; chromosome definition without 'chr').
  - `2. Normalization`
  - `3. Annotation`: Annotates the variants and filters them according to the selected locations (coding, intron, threeUTR, fiveUTR, intergenic, spliceSite, promoter) and consequences (synonymous, nonsynonymous, frameshift, nonsense, not translated) of interest.
  - `4. Combine output`
  - `5. Evaluate Coverage and BQ`: Evaluate coverage and base quality by analyzing the bam files. Calls with insufficient coverage and or base quality are filtered. The following parameters have to be defined: the folder containing the bam- and bai files, minimum coverage, minimum number of reads carrying the alternate allele, minimum VAF, minimum base quality and the maximum difference in base quality between reference and alternative.
  - `6. Extended Set of Characteristics`: Select the databases you would like to query (dbSNP, 1000Genomes, ExAC, ESP6500, Genome Aggregation Database, COSMIC, ClinVar) and the source you would like to use for impact prediction (SIFT, Provean, PolyPhen2).
  - `7. Final Filtration`: Adjust parameters and define input for the final filtration. A bed file containing the primer positions can be uploaded (optional). A txt file containing expected/hotspot mutations (defined via gene, mutation on AA level, and an optional minimum VAF) can be uploaded (optional). Stricter thresholds for coverage and base quality can be defined (see parameters for `5. Evaluate Coverage and BQ`). The number of samples making a call striking can be defined. Dependent on the previously selected source for impact prediction (SIFT, Provean, PolyPhen2), a threshold can be defined to identify reliable damaging predictions and reliable tolerated prediction. Furthermore, the default scoring for the artifact- and the polymorphism score can be changed (just recommended for experts).
  - `Action`: The complete analysis can be executed (`Start complete analysis`). The availability of checkpoints can be checked (`Check for possible checkpoints`). If checkpoints are available, one can be selected and the analysis can be started from the selected checkpoint. The defined configuration can be exported (`Export current configuration`) as well as imported (`Import configuration`). Press `Done` if you are done with your analysis.

- *Overview results*: The tab provides an overview of the results. It is further separated into 5 additional tabs:

  - `Log`: The log of the analysis is displayed here. In case of errors, e.g. input not being available, these messages are also reported in the log.
  - `Raw Calls`: The raw number of calls per sample and per caller are reported.
  - `Calls On Target`: The number of calls on target per sample and per caller are reported.
  - `Annotaed Calls`: The number of annotated (and filtered according to annotation) calls per sample and per caller are reported.
  - `Filtered Calls`: The number of coverage and base quality filtered calls per sample is reported.

- *Detailed results*: The tab provides detailed information on the results. It is further separated into 3 additional tabs:
  - `Mutations`: Variant calls classified as likely true mutations.
  - `Polymorphisms`: Variant calls classified as likely polymorphisms.
  - `Artifacts`: Variant calls classified as likely artifacts.

No value is returned. Output files for the different analysis steps are automatically saved.

```
> appreci8Rshiny()
```

References

[1] M Morgan, H Pagès, V Obenchain, and N Hayden. Rsamtools: Binary alignment (bam), fasta, variant call (bcf) and tabix file import. http://bioconductor.org/packages/release/bioc/html/Rsamtools.html.

[2] V Obenchain, M Lawrence, V Carey, S Gogarten, P Shannon, and M Morgan. Variantannotation: a bioconductor package for exploration and annotation of genetic variants. *Bioinformatics*, 30(14), 2014.

[3] S Sandmann, M Karimi, A de Graaf, C Rohde, S Göllner, J Varghese, J Ernsting, G Walldin, B van der Reijden, C Müller-Tidow, L Malcovati, E Hellström-Lindberg, J Jansen, and M Dugas. appreci8: A pipeline for precise variant calling integrating 8 tools, 2018. https://doi.org/10.1093/bioinformatics/bty518.