qcCHIP User's Guide

Xiang Liu xiang.liu@moffitt.org
Mingxiang Teng mingxiang.teng@moffitt.org
Department of Biostatistics and Bioinformatics
Moffit Cancer Center, Tampa, FL, USA

2025-09-02

Contents

Introduction

Clonal hematopoiesis (CH) is a molecular biomarker associated with various adverse outcomes in healthy and disease individuals. Detecting CHs usually involves genomic sequencing of individual blood samples followed by rigorous bioinformatics data filtering. We report an R package, qcCHIP, a bioinformatics pipeline to identify CH from sequencing data by implementing a series of quality control filters and permutation-based parameter optimization.

qcCHIP takes inputs in VCF format and filter CH mutations using technical, functional, individual and populations metrics associated to the mutations. It allows customized parameter settings of those metrics as well as parameter optimization using permutation analysis with cohort-specific characteristics. For more detail, please refer to our manuscript and the help pages of R functions.

Install package

Install *qcCHIP* package via *devtools*.

```
library(devtools)
devtools::install_github("https://github.com/tenglab/qcCHIP.git",force=T)
#> -- R CMD build ------------------------------------------------------------------
#>      checking for file ‘/private/var/folders/3s/9r3z6h_n0z3889fx5pn38svw0021h2/T/RtmppHw0th/remotes1
#>   -  preparing ‘qcCHIP’:
#>      checking DESCRIPTION meta-information ...  v  checking DESCRIPTION meta-information
#>   -  checking for LF line-endings in source and make files and shell scripts
#>   -  checking for empty or unneeded directories
#>   -  building ‘qcCHIP_0.0.0.9000.tar.gz’
#>      Warning: invalid uid value replaced by that for user 'nobody'
#>
#>
```

Getting Started

Load the package in R.

```
library(qcCHIP)
library(GenomicRanges)
```

Preparing Input Files

qcCHIP works on mutation calls to optimize parameters and filter CHs. For each blood sample, mutations should be first called with blood sequencing data using tools like MuTect2. Usually, a VCF file will be generated by these tools to document mutations for a given sample. As qcCHIP is built to work on a clinical cohort instead of a single sample, the main functions of qcCHIP take input a merged mutation file from all samples for a given cohort. qcCHIP package provides an internal function *vcf2input* to merge multiple VCF files into an annotated text file, which will serve as the input of main functions. The merged mutation text file contains the following columns. The users can also manually built their merged mutation file.

1. Chr: chromosome of variant. Exp: chr1, chr2,chrX.
2. Start: start posation of variant.
3. End: end posation of variant.
4. Ref: reference allele.
5. Alt: alternative allele.
6. TLOD: TLOD or Qual Info from vcf file.
7. SOR: SOR Info from vcf file.
8. AD_alt: Allelic depths for the alt alleles from vcf file.
9. AF: AF or VAF from blood sample vcf file.
10. DP: DP from vcf file.
11. SAF: SAF info from vcf file.
12. SAR: SAR info from vcf file.
13. SampleID: sample ID or variant.
14. Func.refGene: function annotation from refGene.
15. ExonicFunc.refGene: exonic function annotation from refGene. (nonsynonymous SNV and synoymous SNV values need to be named as "nonsynonymous SNV" and"synonymous SNV")
16. cosmic70: if the variant is exist in cosmic database. (empty value needs to be ".")
17. tumor_AF: optional, AF or VAF from tumor sample vcf file.
18. non_cancer_AF_popmax: optional, non cancer AF value from gnomad database.

19. Alt_dpGAP_PopFreq: optional, ALT population frequency from dpGAP databse.

To note: all empty value should be noted as "." in the merged text file.

```
# demo example of input merged file
input_path<- system.file("extdata","demo_input.txt",package="qcCHIP")
in_f <- read.table(input_path,sep="\t",header=T)

# name of each variable
colnames(in_f)
#>  [1] "Chr"                  "Start"                "End"
#>  [4] "Ref"                  "Alt"                  "TLOD"
#>  [7] "SOR"                  "AD_alt"               "AF"
#> [10] "DP"                   "SAF"                  "SAR"
#> [13] "tumor_AF"             "SampleID"             "Func.refGene"
#> [16] "Gene.refGene"         "GeneDetail.refGene"   "ExonicFunc.refGene"
#> [19] "AAChange.refGene"     "cosmic70"             "non_cancer_AF_popmax"
#> [22] "ALT_dpGAp"            "Ref_dpGAP_PopFreq"    "Alt_dpGAP_PopFreq"

# examples of each variable
head(in_f)
#>     Chr     Start       End  Ref Alt   TLOD   SOR AD_alt    AF  DP SAF SAR
#> 1 chr16   3728301   3728304 CGCT   C  17.34 0.768      9 0.021 463   5   4
#> 2 chr17  31169974  31169974    C   A    3.3 4.975     11 0.051 142   0  11
#> 3  chrX  15820243  15820243    C   T 228.43 1.483     72 0.984  72  47  25
#> 4  chr7 140734494 140734494    T  TA    7.8 0.919     14 0.051 227   6   8
#> 5 chr21  43093000  43093000    G  GT   6.93 0.356      8 0.085  95  16   0
#> 6  chr7 102257421 102257422   GA   G   5.28 0.899      7 0.059 121   9   7
#>   tumor_AF   SampleID Func.refGene  Gene.refGene
#> 1    0.018 sample_155       exonic        CREBBP
#> 2    0.025 sample_155       exonic           NF1
#> 3    0.996 sample_155       exonic         ZRSR2
#> 4    0.000 sample_155         UTR3          BRAF
#> 5    0.141 sample_155         UTR3 U2AF1;U2AF1L5
#> 6    0.000 sample_155         UTR3          CUX1
#>
#> 1
#> 2
#> 3
#> 4
#> 5 NM_006758:c.*102C>AC;NM_001025203:c.*102C>AC;NM_001025204:c.*102C>AC;NM_001320650:c.*102C>AC;NM_00
#> 6
#>           ExonicFunc.refGene
#> 1 nonframeshift substitution
#> 2          nonsynonymous SNV
#> 3             synonymous SNV
#> 4                          .
#> 5                          .
#> 6                          .
#>                                                                                                      A
#> 1    CREBBP:NM_001079846:exon30:c.6629_6632delinsG:p.Q2210del,CREBBP:NM_004380:exon31:c.6743_6746del
#> 2 NF1:NM_000267:exon5:c.C563A:p.A188E,NF1:NM_001042492:exon5:c.C563A:p.A188E,NF1:NM_001128147:exon5:
#> 3                                                                            ZRSR2:NM_005089:exon10:
#> 4
#> 5
#> 6
#>   cosmic70 non_cancer_AF_popmax ALT_dpGAp Ref_dpGAP_PopFreq Alt_dpGAP_PopFreq
#> 1        .             2.03E-05         .                 .                 .
#> 2        .                    .         .                 .                 .
#> 3        1               0.6222         .                 .                 .
#> 4        .                    .         .                 .                 .
#> 5        .                    .         .                 .                 .
#> 6        .                    .         .                 .                 .
```

The function *vcf2input* can convert raw or annotated VCF files into an input merged file for the main functions of qcCHIP. For raw VCF files, as they don't contain the required information of *Func.refGene*, *ExonicFunc.refGene*, and *cosmic70*, converting with raw VCF files require users to manually add the corresponding columns to the merged file.

Converting with raw VCF files

Below are the examples of how to convert raw VCF files into a merged text file. For demo purpose, VCF files generated from 10 cancer patients are documented in the package and used here.

```
# Path to raw vcf files
raw_vcf_path<- system.file("extdata/raw_vcf",package="qcCHIP")

# Create data.frame of samples
sample_df <- data.frame(blood=list.files(raw_vcf_path)[! list.files(raw_vcf_path) %in%
                                                         grep("*_tumor\\.vcf\\.gz",list.files(raw_vcf_pa
                        tumor=grep("*_tumor\\.vcf\\.gz",list.files(raw_vcf_path),value=T))
head(sample_df)
#>                blood                  tumor
#> 1  sample_1.vcf.gz  sample_1_tumor.vcf.gz
#> 2 sample_10.vcf.gz sample_10_tumor.vcf.gz
#> 3  sample_2.vcf.gz  sample_2_tumor.vcf.gz
#> 4  sample_3.vcf.gz  sample_3_tumor.vcf.gz
#> 5  sample_4.vcf.gz  sample_4_tumor.vcf.gz
#> 6  sample_5.vcf.gz  sample_5_tumor.vcf.gz

# 1. Converting without paired tumor sample
input_1 <- vcf2input(vcf_path = raw_vcf_path,
                   sample_list=sample_df$blood)
head(input_1)
#>    Chr    Start      End Ref Alt      SampleID   TLOD   SOR AD_alt    AF  DP
#> 1 chr1  1786850  1786850   G   T sample_1_st_g   4.08 0.375      3 0.103  30
#> 2 chr1  1786996  1786996   C   G sample_1_st_g  43.36 0.333     17 0.469  36
#> 3 chr1 36471458 36471458   A   G sample_1_st_g 187.69  0.43     72 0.500 148
#> 4 chr1 64833405 64833405   A   C sample_1_st_g  18.22 0.593      8 0.532  14
#> 5 chr1 64837976 64837976   C   T sample_1_st_g  14.64 1.244      7 0.269  30
#> 6 chr1 64841324 64841324   G   T sample_1_st_g   3.08 1.142      3 0.062  76
#>   SAF SAR
#> 1   2   1
#> 2   9   8
#> 3  36  36
#> 4   3   5
#> 5   5   2
#> 6   1   2

# 2. Converting with paired tumor sample
input_2 <- vcf2input(vcf_path = raw_vcf_path,
                  sample_list=sample_df$blood,
                  tumor=T,
                  tumor_path=raw_vcf_path,
                  tumor_list=sample_df$tumor)
head(input_2)
#>      Start Ref Alt      SampleID      End  Chr   TLOD   SOR AD_alt    AF  DP
#> 1  1786850   G   T sample_1_st_g  1786850 chr1   4.08 0.375      3 0.103  30
#> 2  1786996   C   G sample_1_st_g  1786996 chr1  43.36 0.333     17 0.469  36
#> 3 36471458   A   G sample_1_st_g 36471458 chr1 187.69  0.43     72 0.500 148
#> 4 64833405   A   C sample_1_st_g 64833405 chr1  18.22 0.593      8 0.532  14
#> 5 64837976   C   T sample_1_st_g 64837976 chr1  14.64 1.244      7 0.269  30
#> 6 64841324   G   T sample_1_st_g 64841324 chr1   3.08 1.142      3 0.062  76
#>   SAF SAR    SampleID_t tumor_AF
#> 1   2   1             .        .
#> 2   9   8 sample_1_st_t    0.216
#> 3  36  36 sample_1_st_t     0.16
#> 4   3   5 sample_1_st_t    0.534
#> 5   5   2 sample_1_st_t    0.483
#> 6   1   2             .        .
```

Converting with annotated VCF files

Currently, qcCHIP can take the following annotation information. These annotation can be generated by ANNOVAR tool.

1. Function name, gene name, exonic function, AAchange, and GeneDetail information from refGene database;
2. COSMIC info from COSMIC database (e.g. COSMIC70);
3. Non cancer AF population max from gnomAD database.

Currently, function name, exonic function, and COSMIC are required annotation information by qcCHIP.

```
# Path to annotated vcf files
annot_vcf_path<- system.file("extdata/annot_vcf",package="qcCHIP")

# Create data.frame of samples
sample_df <- data.frame(blood=list.files(annot_vcf_path)[! list.files(annot_vcf_path) %in%
                                                      grep("*_tumor\\.hg38_multianno\\.vcf",list.files(a
                        tumor=grep("*_tumor\\.hg38_multianno\\.vcf",list.files(annot_vcf_path),value=T))
head(sample_df)
#>                             blood                              tumor
#> 1  sample_1.hg38_multianno.vcf  sample_1_tumor.hg38_multianno.vcf
#> 2 sample_10.hg38_multianno.vcf sample_10_tumor.hg38_multianno.vcf
#> 3  sample_2.hg38_multianno.vcf  sample_2_tumor.hg38_multianno.vcf
#> 4  sample_3.hg38_multianno.vcf  sample_3_tumor.hg38_multianno.vcf
#> 5  sample_4.hg38_multianno.vcf  sample_4_tumor.hg38_multianno.vcf
#> 6  sample_5.hg38_multianno.vcf  sample_5_tumor.hg38_multianno.vcf
```

```r
# example of annotated VCF files
library(vcfR)
exp_vcf <- read.vcfR(system.file("extdata/annot_vcf","sample_1.hg38_multianno.vcf",package="qcCHIP"))
#> Scanning file to determine attributes.
#> File attributes:
#>   meta lines: 3468
#>   header_line: 3469
#>   variant count: 213
#>   column count: 10
#> Meta line 1000 read in.Meta line 2000 read in.Meta line 3000 read in.Meta line 3468 read in.
#> All meta lines processed.
#> gt matrix initialized.
#> Character matrix gt created.
#>   Character matrix gt rows: 213
#>   Character matrix gt cols: 10
#>   skip: 0
#>   nrows: 213
#>   row_num: 0
#> Processed variant: 213
#> All variants processed

# check name of annotation information
strsplit(head(exp_vcf@fix,n=1)[,8],";")
#> $INFO
#>  [1] "AS_SB_TABLE=19,8|2,1"
#>  [2] "DP=32"
#>  [3] "ECNT=1"
#>  [4] "FS=0"
#>  [5] "MBQ=30,20"
#>  [6] "MFRL=182,144"
#>  [7] "MMQ=60,60"
#>  [8] "MPOS=10"
#>  [9] "POPAF=7.3"
#> [10] "SOR=0.375"
#> [11] "TLOD=4.08"
#> [12] "ANNOVAR_DATE=2020-06-08"
#> [13] "Func.refGeneWithVer=UTR3"
#> [14] "Gene.refGeneWithVer=GNB1"
#> [15] "GeneDetail.refGeneWithVer=NM_001282538.1:c.*213C>A\\x3bNM_001282539.1:c.*213C>A\\x3bNM_002074.
#> [16] "ExonicFunc.refGeneWithVer=."
#> [17] "AAChange.refGeneWithVer=."
#> [18] "cosmic70=."
#> [19] "AF=."
#> [20] "AF_popmax=."
#> [21] "AF_male=."
#> [22] "AF_female=."
#> [23] "AF_raw=."
#> [24] "AF_afr=."
#> [25] "AF_sas=."
#> [26] "AF_amr=."
#> [27] "AF_eas=."
#> [28] "AF_nfe=."
#> [29] "AF_fin=."
#> [30] "AF_asj=."
#> [31] "AF_oth=."
#> [32] "non_topmed_AF_popmax=."
#> [33] "non_neuro_AF_popmax=."
#> [34] "non_cancer_AF_popmax=."
#> [35] "controls_AF_popmax=."
#> [36] "ALLELE_END"

# Converting annotated VCFs with paired tumor samples. Annotated with refGene, cosmic, and gnomAD.
input_3 <- vcf2input(vcf_path = annot_vcf_path,
                  sample_list=sample_df$blood,
                  tumor=T,
                  tumor_path=annot_vcf_path,
                  tumor_list=sample_df$tumor,
                  refGene=T,
                  refGene_func_name="Func.refGeneWithVer",
                  refGene_gene_name="Gene.refGeneWithVer",
                  refGene_Exonicfunc_name="ExonicFunc.refGeneWithVer",
                  refGene_AAchange_name=NA,
                  refGene_GeneDetail_name=NA,
                  cosmic=T,
                  cosmic_name="cosmic70",
                  gnomad=T,
                  gnomad_name="non_cancer_AF_popmax")

head(input_3)
#>      Start Ref Alt      SampleID      End  Chr   TLOD   SOR AD_alt    AF  DP
#> 1  1786850   G   T sample_1_st_g  1786850 chr1   4.08 0.375      3 0.103  30
#> 2  1786996   C   G sample_1_st_g  1786996 chr1  43.36 0.333     17 0.469  36
#> 3 36471458   A   G sample_1_st_g 36471458 chr1 187.69  0.43     72 0.500 148
#> 4 64833405   A   C sample_1_st_g 64833405 chr1  18.22 0.593      8 0.532  14
#> 5 64837976   C   T sample_1_st_g 64837976 chr1  14.64 1.244      7 0.269  30
#> 6 64841324   G   T sample_1_st_g 64841324 chr1   3.08 1.142      3 0.062  76
#>   SAF SAR    SampleID_t tumor_AF Func.refGene Gene.refGene ExonicFunc.refGene
#> 1   2   1             .        .         UTR3         GNB1                  .
#> 2   9   8 sample_1_st_t    0.216         UTR3         GNB1                  .
#> 3  36  36 sample_1_st_t     0.16       exonic        CSF3R     synonymous_SNV
#> 4   3   5 sample_1_st_t    0.534         UTR3         JAK1                  .
#> 5   5   2 sample_1_st_t    0.483       exonic         JAK1     synonymous_SNV
#> 6   1   2             .        .       exonic         JAK1           stopgain
#>   cosmic70 non_cancer_AF_popmax Alt_dpGAP_PopFreq
#> 1        .                    .                 .
#> 2        .                    .                 .
#> 3        .               0.7273                 .
#> 4        .                    .                 .
#> 5        .               0.1083                 .
#> 6        .                    .                 .
```

The output file can be directly adopted by the main filtering and permutation functions.

```
# find CH with default metrics
ch_candidate <- CHIPfilter(input_3,tumor_sample=T)
#> [1] "Perform population metrics"
#> [1] "Perform technique metrics"
#> [1] "Perform individual metrics with paired tumor sample"
#> [1] "Perform functional metrics"
#> [1] "No not nonsunonymous metrics find, skip"
#> [1] "Perform gnomad metrics only"
#> [1] "No blacklist region bed file find, skip"
head(ch_candidate)
#>          Start Ref Alt       SampleID       End  Chr   TLOD   SOR AD_alt    AF
#> 347  150068243 AGC   - sample_10_st_g 150068245 chr5  18.15 1.508      8 0.036
#> 391   48792153   G   C sample_10_st_g  48792153 chrX 205.62 0.716     85 0.296
#> 2170 102255385   -  AA  sample_9_st_g 102255385 chr7  31.03 0.284     49 0.187
#> 2176 102257898   -   T  sample_9_st_g 102257898 chr7  14.91 0.565     19 0.233
#>       DP SAF SAR     SampleID_t tumor_AF Func.refGene Gene.refGene
#> 347  291   6   2 sample_10_st_t    0.013       exonic       SETBP1
#> 391  299  40  45 sample_10_st_t    0.026       exonic       BCORL1
#> 2170 240 125  85  sample_9_st_t    0.113       exonic        SRSF2
#> 2176  79   4  33  sample_9_st_t    0.142       exonic         JAK3
#>      ExonicFunc.refGene                                        cosmic70
#> 347   nonsynonymous_SNV                                               .
#> 391      synonymous_SNV                                               .
#> 2170     synonymous_SNV ID\\x3dCOSM4130674\\x3bOCCURENCE\\x3d1(thyroid)
#> 2176     synonymous_SNV                                               .
#>      non_cancer_AF_popmax Alt_dpGAP_PopFreq               v_name           loci
#> 347                0.2566                 . chr5:150068243:AGC:- chr5:150068243
#> 391                     .                 .    chrX:48792153:G:C  chrX:48792153
#> 2170                    1                 .  chr7:102255385:-:AA chr7:102255385
#> 2176                    .                 .   chr7:102257898:-:T chr7:102257898
#>                                mut_sample
#> 347  chr5:150068243:AGC:-|sample_10_st_g
#> 391     chrX:48792153:G:C|sample_10_st_g
#> 2170   chr7:102255385:-:AA|sample_9_st_g
#> 2176    chr7:102257898:-:T|sample_9_st_g
```

Basic Usage of The *qcCHIP* Function

In this section, we use *qcCHIP* function to perform permutation and evaluate the effects of parameter cutoff values. Currently, *qcCHIP* function supports optimization for five metrics: VAF, DP, SOR, SAF/SAR and cohort mutation prevalence. As demonstrated in our manuscript, permutation consistency usually increases with higher parameter values with the exception that larger SOR decreases consistency. This is because that larger SOR means higher strand bias and lower reliability of mutation calls. In our manuscript, we demonstrated that the infection points of permutation consistency curves indicate the best parameter cutoffs. In practice, users should determine the infection points of permutation consistency curves: 1) with the increase of consistency stratified by VAF, DP, SAF/SAR and cohort mutation prevalence; 2) with the decrease of consistency stratified by SOR. In addition, users are suggested to take full consideration across consistency, precision and recall curves to determine the best parameter cutoffs.

Below we demo the permutation analysis and generate figures and summary for three parameters: VAF, DP, and cohort mutation prevalence. The figures and summary are used for determination of optimal parameter cutoffs used in the next section.

Run *qcCHIP* with different VAF cutoffs

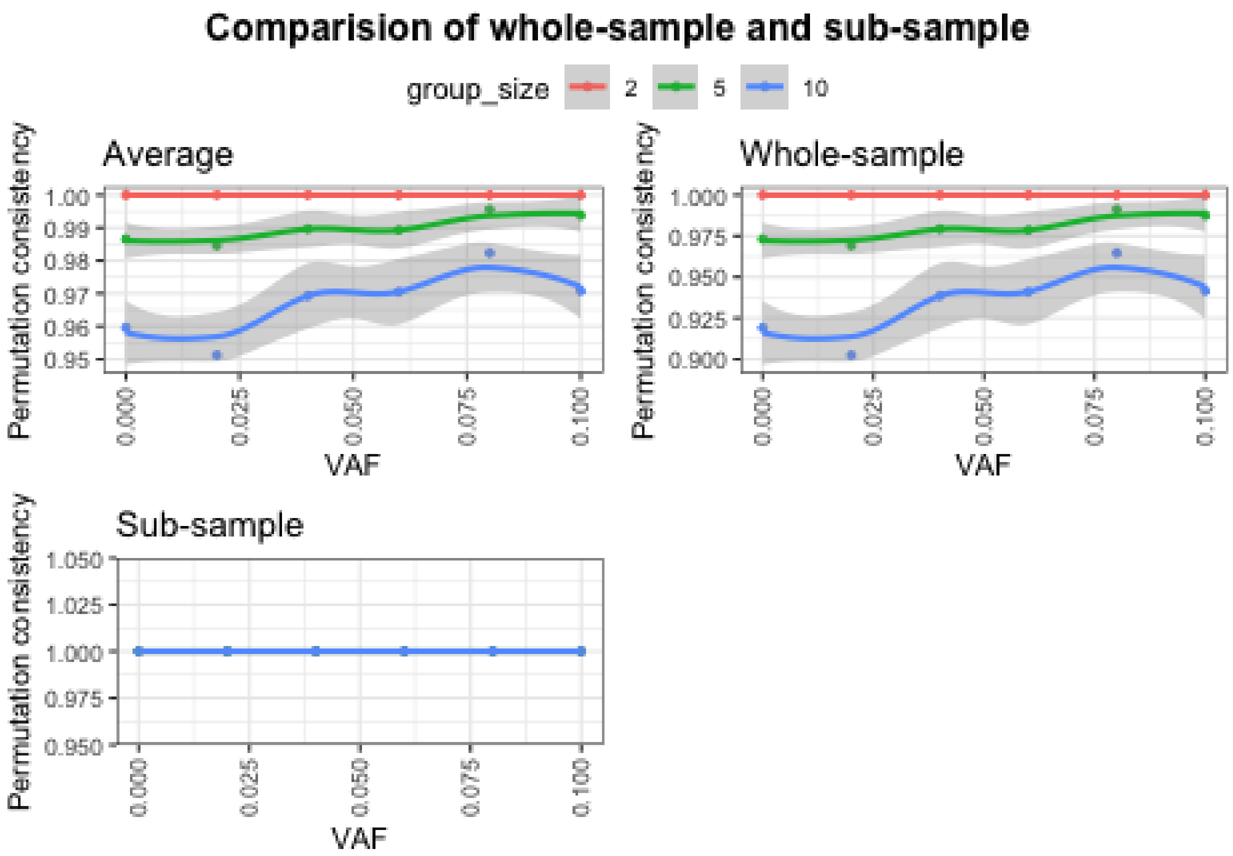
The hypothesis is that a CH should hold a reliable VAF to be considered as true somatic mutation. This section demonstrates how different settings of minimum VAF affect the permutation consistency. Please refer to our manuscript for more detail. For computing efficiency, we only demo with 10 samples here, which is less-powered compared to the results presented in our manuscript.

```r
# input file
input_path<- system.file("extdata","demo_input.txt",package="qcCHIP")
in_f <- read.table(input_path,sep="\t",header=T)

# create test directory
out_dir <- paste0(getwd(),"/vaf_test")
vaf_permut <- qcCHIP(in_f,out_path = out_dir
                     ,metric_min = 0,
                     metric_step = 0.02,
                     metric_max = 0.1,
                     core=1,
                     show_info = F)

# example of comparision summary output
head(vaf_permut$summary_df)
#>   metric_name metric_setting group_size permut_index var_n_whole var_n_sub
#> 1         VAF              0          2            1         255       255
#> 2         VAF              0          2            2         255       255
#> 3         VAF              0          2            3         255       255
#> 4         VAF              0          2            4         255       255
#> 5         VAF              0          2            5         255       255
#> 6         VAF              0          2            6         255       255
#>   union_n common_n whole_only sub_only common_whole common_sub
#> 1     255      255          0        0            1          1
#> 2     255      255          0        0            1          1
#> 3     255      255          0        0            1          1
#> 4     255      255          0        0            1          1
#> 5     255      255          0        0            1          1
#> 6     255      255          0        0            1          1

# permutation consistency plot
vaf_permut$figs
```



Run *qcCHIP* with different DP cutoffs

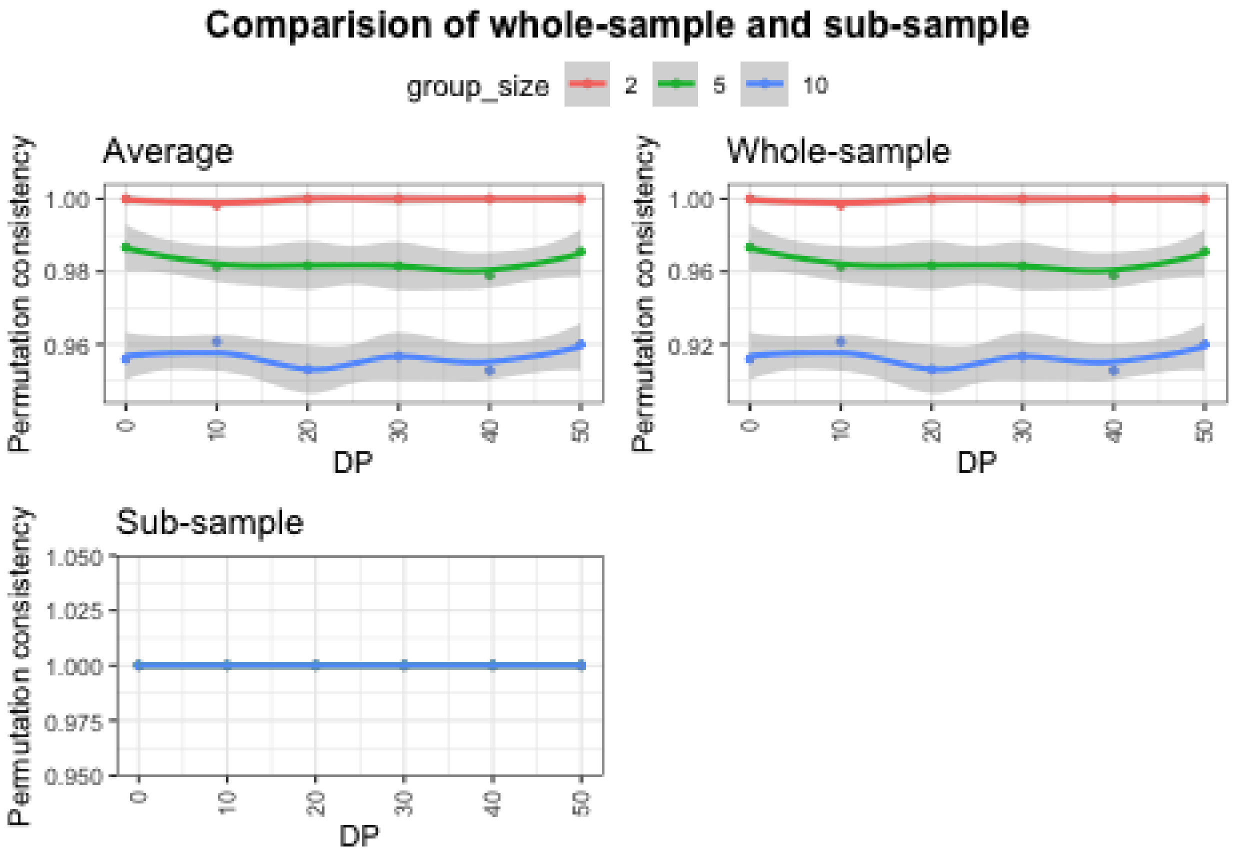
The hypothesis is that a CH should hold a reliable DP to be considered as true somatic mutation. This section demonstrates how different settings of minimum DP affect the permutation consistency. Please refer to our manuscript for more detail.

```
# input file
input_path<- system.file("extdata","demo_input.txt",package="qcCHIP")
in_f <- read.table(input_path,sep="\t",header=T)

# create test directory
out_dir <- paste0(getwd(),"/DP_test")
DP_permut <- qcCHIP(in_f,out_path = out_dir,permut_metrics = "DP",
                    metric_min = 0,
                    metric_step = 10,
                    metric_max = 50,
                    core=1,
                    show_info = F)

# example of comparision summary output
head(DP_permut$summary_df)
#>   metric_name metric_setting group_size permut_index var_n_whole var_n_sub
#> 1          DP              0          2            1         184       184
#> 2          DP              0          2            2         184       184
#> 3          DP              0          2            3         184       184
#> 4          DP              0          2            4         184       184
#> 5          DP              0          2            5         184       184
#> 6          DP              0          2            6         184       184
#>   union_n common_n whole_only sub_only common_whole common_sub
#> 1      184      184          0        0            1          1
#> 2      184      184          0        0            1          1
#> 3      184      184          0        0            1          1
#> 4      184      184          0        0            1          1
#> 5      184      184          0        0            1          1
#> 6      184      184          0        0            1          1

# permutation consistency plot
DP_permut$figs
```



Run *qcCHIP* with different mutation prevalence cutoffs

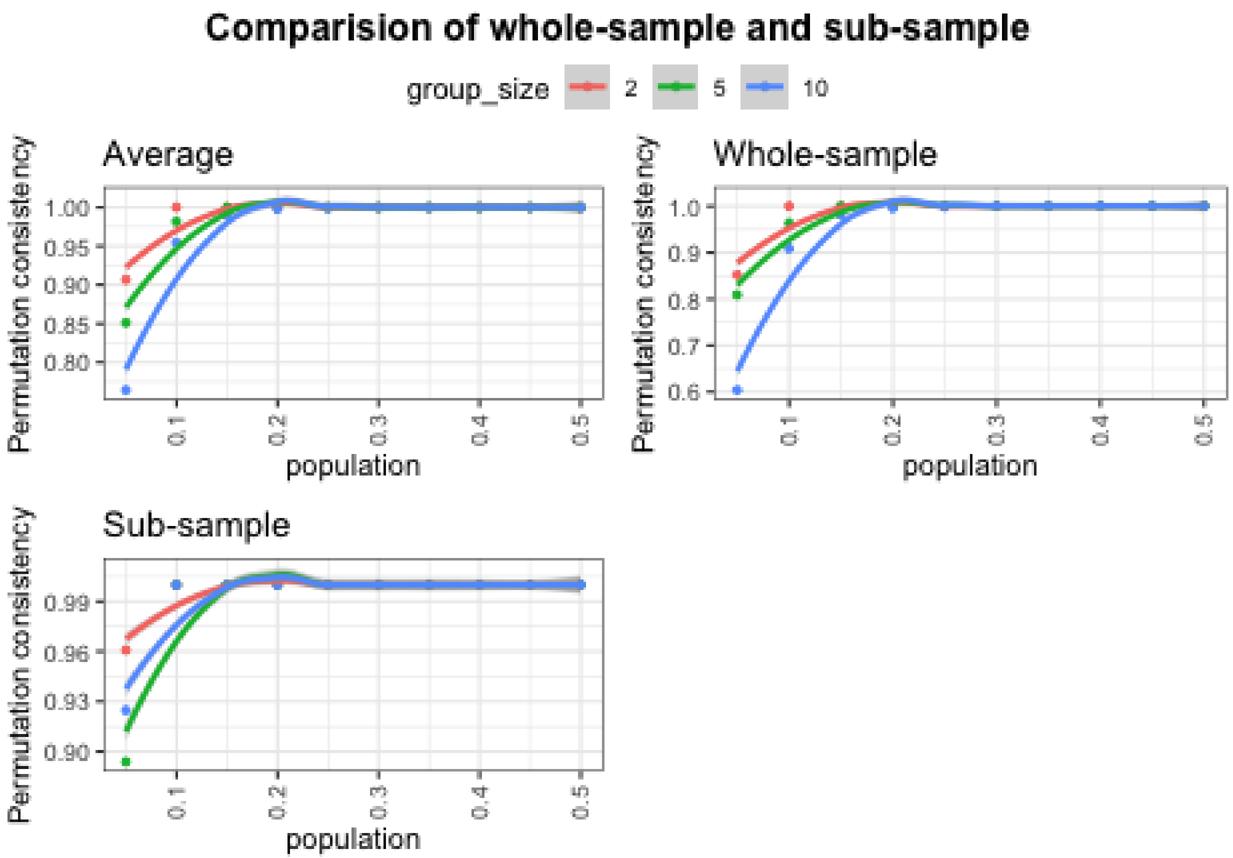
The hypothesis is that a true CH shouldn't be over prevalent in a given cohort as CHs are relatively rare events. This section demonstrates how different settings of maximum mutation prevalence affect the permutation consistency. Please refer to our manuscript for more detail.

```
# input file
input_path<- system.file("extdata","demo_input.txt",package="qcCHIP")
in_f <- read.table(input_path,sep="\t",header=T)
```

```
# create test directory
out_dir <- paste0(getwd(),"/population_test")
pop_permut <- qcCHIP(in_f,out_path = out_dir,permut_metrics = "population",
                    metric_min = 0.05,
                    metric_step = 0.05,
                    metric_max = 0.5,
                    core=1,
                    show_info = F)

# example of comparison summary output
head(pop_permut$summary_df)
#>   metric_name metric_setting group_size permut_index var_n_whole var_n_sub
#> 1  population           0.05          2            1         137       121
#> 2  population           0.05          2            2         137       126
#> 3  population           0.05          2            3         137       125
#> 4  population           0.05          2            4         137       124
#> 5  population           0.05          2            5         137       123
#> 6  population           0.05          2            6         137       103
#>   union_n common_n whole_only sub_only common_whole common_sub
#> 1     141      117         20        4        0.854      0.967
#> 2     147      116         21       10        0.847      0.921
#> 3     146      116         21        9        0.847      0.928
#> 4     142      119         18        5        0.869      0.960
#> 5     139      121         16        2        0.883      0.984
#> 6     137      103         34        0        0.752      1.000

# permutation consistency plot
pop_permut$figs
```



Basic Usage of The *CHIPfilter* Function

Finally, we use *CHIPfilter* to filter CH candidates based on a variety of quality metrics (detailed in the man page of *CHIPfilter*). The output will be a subset of input file which pass the QC filtering. It is recommended that users first run *qcCHIP* function to determine optimal metric values. Nevertheless, users can pick their customized metric values and directly apply this function for CH filtering. Some features of *CHIPfilter* are described below.

```
# input file
input_path<- system.file("extdata","demo_input.txt",package="qcCHIP")
in_f <- read.table(input_path,sep="\t",header=T)

# exclude blacklist region
bf_path<- system.file("extdata","demo_blacklist.bed",package="qcCHIP")

bl_f <- read.table(bf_path,sep = "\t",header=F)

# run default setting
out_1 <- CHIPfilter(in_f)
#> [1] "Perform population metrics"
#> [1] "Perform technique metrics"
#> [1] "No paired tumor sample, skip"
#> [1] "Perform functional metrics"
#> [1] "No not nonsunonymous metrics find, skip"
#> [1] "Perform gnomad metrics only"
#> [1] "No blacklist region bed file find, skip"

# use different metric values
out_2 <- CHIPfilter(in_f,max_percent=0.02,DP_min = 40,VAF_min=0.002,info=F)

# fitler with paired tumor sample
out_3 <- CHIPfilter(in_f,tumor_sample = T,tumor_VAF_min = 0.02,info=F)

# filter with gnomAD or dpGAP reference file
out_4 <- CHIPfilter(in_f,gnomad = F,dpGAP = F,info=F)

# filter with blacklist region
out_5 <- CHIPfilter(in_f,blacklist_f = bl_f,info=F)

# check the number of CHIP
length(unique(out_1$mut_sample))
#> [1] 182
length(unique(out_2$mut_sample))
#> [1] 101
length(unique(out_3$mut_sample))
#> [1] 136
length(unique(out_4$mut_sample))
#> [1] 201
length(unique(out_5$mut_sample))
#> [1] 178
```