vcfqc: A Package for Visualization of Variant Calls Quality

Hatoon Al Ali 181963

02 Apr 2022

Abstract

vcfqc is a an R package that allows bioinformaticians to get an overview of the quality of a VCF file, visulaize the distribution of GATK quality annotation, and set threshhold values for filteing to count the filtered-out variants. The package can be installed from https://hatoonli.github.io/vcfqc/ or from the github repository https://github.com/Hatoonli/vcfqc.

Introduction

Variant calling format (VCF) is the final file format that results from various steps of transforming sequencing data into high quality annotated genetic variants. Variants in a VCF file are prone to errors, so, different quality measurements are taken to validate the information in the file. If these measurements are not met, a further step of hard-filtering can be done to remove low quality calls (De Summa et al. 2017). Genome Analysis Toolkit (GTAK) is an established, well-documented variant calling pipeline published by the broad institute (McKenna et al. 2010). It provides various quality annotations to detect errors in the different steps of variant calling. This package visualizes these annotations and prints out GATK guidelines on hard-filtering based on the chosen parameter.

Purpose

This tool aims to guide the user on filtering GATK VCF files by visualizing the quality distribution of variants and providing GTAK filtering threshold values and tips.

Data

The package functions on GATK annotated VCF files. A GATK user needs to convert the VCF into table format using the following bash command:

```
gatk VariantsToTable -V $file\
-F CHROM -F POS -F REF -F ALT \
-F FILTER -F QUAL -F QD -F FS \
-F SOR -F MQ -F MQRankSum -F ReadPosRankSum\
-O /home/alihh/freq/stat/ksu/$file.table
```

How to use

Install the latest version of vcfqc package by copying the following commands into R console:

```
library(devtools)
devtools::install_github("Hatoonli/vcfqc", build_vignettes = TRUE)
```

Load the package

```
library(vcfqc)
```

Then choose a quality parameter from the list below and run it on your VCF file. * QD * FS * SOR * MQ * MQRankSum * ReadPosRankSum

You can read more about each parameter by inquiring about it with a question mark, for example:

```
`?`(QC)
```

You can preview the performance of the package on the example file provided in `/extdata`, to get the file pathway run the following command:

```
system.file("extdata", "HG001_GRCh38.table", package = "vcfqc")
```

and to learn more about the file, read the accompanying documentation by running the following command:

```
`?`(extdata/HG001_GRCh38.table)
```

Implementation

Choosing the package name

Before choosing the name **vcfqc**, I checked for its availability using the package **available**

```
library()
valid_package_name("vcfqc")
available_on_cran("vcfqc")
available_on_bioc("vcfqc")
available_on_github("vcfqc")
```

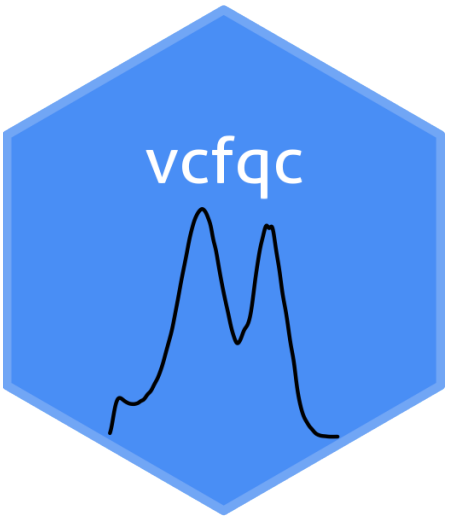
Creating a package logo

Using hexSticker, the sticker was created and saved in `/inst/figures/sticker.png`.

```
library(vcfqc)
library(ggplot2)
library(hexSticker)

p <- QD(system.file("extdata", "HG001_GRCh38.table", package = "vcfqc"))
p <- p + theme_void() + theme_transparent()
s <- sticker(p, package = "vcfqc", p_size = 20, s_x = 1, s_y = 0.75,
```

```
    s_width = 1, s_height = 1, h_fill = "#498ef5", h_color = "#71a6f5",
    filename = "inst/figures/sticker.png")
plot(s)
```



It's then turned into a logo by using the function `use_logo("inst/figures/sticker.png")` the generated text is copied and added next to the title in the `README.Rmd` file.

Building the package

This package was built using **devtools** (**dev?**). By creating the package, all required directories and files are built in the package directory.

```
usethis::create_package("vcfqc")
```

The package description is added in DESCRIPTION file. The license used is an open source licence by MIT, added using the function `use_mit_license()`.

All **vcfqc** functions are saved in the directory `\R` and generated using the devtools command `use_r()`. Information about each function was taken from GATK hard-filtering guidelines (De Summa et al. 2017) and added at the header of the function file (e.g. `QD.R`).

```
#' Plot Quality by Depth (QD)
#' @description
#' This is the ....
#' @param f VCF input file
#' @param n Threshold value (optional), default is the recommended GATK value.
#' @returns
#' A plot ...
#' @export
#' @example
#' QD(system.file("extdata", "HG001_GRCh38.table", package = "vcfqc"))
```

From this header, documentation for the function is saved in `\man` and the imported or exported objects are detailed in `NAMESPACE`.

ggplot is used for plotting the distribution of quality annotation using a simple line:

```
ggplot2::ggplot(vcf, ggplot2::aes(QD)) + ggplot2::geom_density()
```

The user gets a message for the numbers of variants not annotated (NA) and a table output that summarizes the data. Lastly, a text is printed with information about the expected plot from GTAK guidelines.

Additional files needed for the package are added under the directory `/inst`, when the package is installed all child directories of `/inst` will be installed with it. Databases in this directories are documented in `/R`. Files that are not needed for the package are added to `.Rbuildignore` so they're not copied to the user system when the package is installed. A vignette was also created to demonstarte the purpose of the package and give background information. Using `use_vignette()` created a vignette directory with the R markdown file. Finally, a website was created feo the package using **pkgdown**.

Future updates of the package can include more annotation options and a flexible annotation field where the user can enter costum values based on the variant calling pipeline they're using.

References

De Summa, Simona, Giovanni Malerba, Rosamaria Pinto, Antonio Mori, Vladan Mijatovic, and Stefania Tommasi. 2017. "GATK Hard Filtering: Tunable Parameters to Improve Variant Calling for Next Generation Sequencing Targeted Gene Panel Data." *BMC Bioinformatics* 18 (5): 57–65.

McKenna, Aaron, Matthew Hanna, Eric Banks, Andrey Sivachenko, Kristian Cibulskis, Andrew Kernytsky, Kiran Garimella, et al. 2010. "The Genome Analysis Toolkit: A MapReduce Framework for Analyzing Next-Generation DNA Sequencing Data." *Genome Research* 20 (9): 1297–1303.