Package 'G4SNVHunter'

June 5, 2026

Type Package

Title Evaluating SNV-Induced Disruption of G-Quadruplex Structures

Version 1.5.0

Description G-quadruplexes (G4s) are unique nucleic acid secondary structures predominantly found in guanine-rich regions and have been shown to be involved in various biological regulatory processes.
G4SNVHunter is an R package designed to rapidly identify genomic sequences with G4-forming propensity and to accurately screen user-provided single nucleotide variants—as well as other small-scale variants such as indels and MNVs—for their potential to destabilize these structures.
This allows researchers to then screen these critical variants for deeper study, digging into how they might influence biological functions—think gene regulation, for instance—by impairing G4 formation propensity.

Encoding UTF-8

LazyData false

Depends R (>= 4.3.0)

Imports Biostrings, stats, GenomicRanges, IRanges, Rcpp, RcppRoll, data.table, Seqinfo, S4Vectors, ggplot2, cowplot, progress, ggseqlogo, viridis, ggpointdensity, tools, SummarizedExperiment, VariantAnnotation, dplyr, openxlsx, tidyr, magrittr, ggdensity

LinkingTo Rcpp

RoxygenNote 7.3.2

License MIT + file LICENSE

Suggests knitr, BiocStyle, rmarkdown, BiocManager, BSgenome.Hsapiens.UCSC.hg19, DT, rtracklayer, testthat (>= 3.0.0), RBGL

VignetteBuilder knitr

biocViews Epigenetics, SNP

News NEWS.md

URL https://github.com/rongxinzh/G4SNVHunter

BugReports https://github.com/rongxinzh/G4SNVHunter/issues

Config/testthat/edition 3

git_url https://git.bioconductor.org/packages/G4SNVHunter

git_branch devel

git_last_commit 0457e3e

git_last_commit_date 2026-04-28

Repository Bioconductor 3.24

Date/Publication 2026-06-04

Author Rongxin Zhang [cre, aut] (ORCID:
<https://orcid.org/0000-0002-1643-1185>)

Maintainer Rongxin Zhang <rongxinzhang@outlook.com>

Contents

---

checkSNV *Check the validity of SNVs*

Description

This function is deprecated and will be removed in a future version.

Usage

```
checkSNV(snv_gr = NULL, mode = "wra", ref_col = NULL, alt_col = NULL)
```

Arguments

snv_gr	A GRanges object containing SNV data.
mode	A character string specifying the checks to be performed. w for checking if all widths (w) are 1, r for checking if all ref (r) values are A, T, C, or G, a for checking if all alt (a) values are A, T, C, or G.
ref_col	Column name for the ref bases in snv_gr. Default is NULL.
alt_col	Column name for the alt bases in snv_gr. Default is NULL.

Details

This function checks whether the user-provided SNVs are single nucleotide substitutions.

Value

A logical value indicating whether the user-provided SNVs passed all checks.

Deprecated

This function is no longer supported.

Examples

```
if (!requireNamespace("BiocManager", quietly = TRUE)) {
  install.packages("BiocManager")
}

if (!requireNamespace("BSgenome.Hsapiens.UCSC.hg19", quietly = TRUE)) {
  BiocManager::install("GenomicRanges")
}

library(GenomicRanges)
gr1 <- GRanges("chr1", IRanges(start = 100, width = 1))
# check width ('w')
checkSNV(gr1, mode = "w")

gr2 <- GRanges(
  seqnames = Rle("seq1"),
  ranges = IRanges(c(100, 200, 300), width = 1),
  ref = c("A", "C", "G"),
  alt = c("T", "T", "A")
)

# check width ('w'), ref ('r'), and alt ('a')
checkSNV(gr2, mode = "wra", ref_col = "ref", alt_col = "alt")
# check width ('w') and alt ('a')
checkSNV(gr2, mode = "wa", alt_col = "alt")

gr3 <- GRanges("chr1", IRanges(start = 100, width = 10))
# widths should be all one
checkSNV(gr3, mode = "w")

gr4 <- GRanges(
  seqnames = Rle("seq1"),
  ranges = IRanges(start = 100, width = 1),
  ref = "AG",
```

```
  alt = "T"
)

# ref should be all one
checkSNV(gr4, mode = "wr", ref_col = "ref")
```

---

exportG4 *Export Predicted G4s to a File*

Description

This function exports a GRanges object containing predicted G4s generated by the G4HunterDetect function from the G4SNVHunter package, to a file in TXT, CSV, or XLSX format.

Usage

```
exportG4(
  G4 = NULL,
  filename = NULL,
  include_metadata = TRUE,
  revcomp_antisense = TRUE
)
```

Arguments

G4	A GRanges object returned by G4HunterDetect.
filename	A character string specifying the output file path. The file extension must be one of: .txt, .csv, or .xlsx.
include_metadata	A logical value. Whether to include global metadata (G4 prediction parameters) in the output. Default is TRUE.
revcomp_antisense	A logical value. Whether to reverse-complement sequences on the antisense (negative) strand. Default is TRUE.

Value

Invisibly returns a data.frame object.

Examples

```
fa_path <- system.file("extdata", "seq.fa", package = "G4SNVHunter")
seq <- loadSequence(seq_path = fa_path)
# Predict G4s
G4_detected <- G4HunterDetect(seq)

out_xlsx <- file.path(tempdir(), "results.xlsx")
out_txt  <- file.path(tempdir(), "results.txt")
out_csv  <- file.path(tempdir(), "results.csv")

exportG4(G4_detected, out_xlsx)
exportG4(G4_detected, out_txt, include_metadata = FALSE)
```

```
exportG4(G4_detected, out_csv, revcomp_antisense = FALSE)

unlink(c(out_xlsx, out_txt, out_csv))
```

---

exportMutG4 *Export G4s and Associated Variant Information to File*

Description

This function exports a GRanges object containing predicted G4 regions affected by variants to a TXT, CSV, or XLSX file. The output includes both the original and mutated G-rich sequences, along with detailed information about the associated variants.

Usage

```
exportMutG4(
  mut_G4 = NULL,
  filename = NULL,
  include_metadata = TRUE,
  revcomp_G4_seq = TRUE,
  revcomp_mutG4_seq = TRUE
)
```

Arguments

mut_G4	A GRanges object returned by filterVarImpact. Both the G4.info.sequence and the mutated.G4.seq columns must be included.
filename	A character string specifying the output file path. The file extension must be one of: .txt, .csv, or .xlsx.
include_metadata	A logical value. Whether to include global metadata (e.g., G4 prediction parameters) in the output. Default is TRUE.
revcomp_G4_seq	A logical value. Whether to reverse-complement G4 sequences on the antisense (negative) strand. Default is TRUE.
revcomp_mutG4_seq	A logical value. Whether to reverse-complement mutated G4 sequences on the antisense (negative) strand. Default is TRUE.

Value

Invisibly returns a data.frame object.

Examples

```
library(GenomicRanges)
fa_path <- system.file("extdata", "seq.fa", package = "G4SNVHunter")
seq <- loadSequence(seq_path = fa_path)
# Predict G4s
G4_detected <- G4HunterDetect(seq)
# create mut granges object
mut <- data.frame(
```

```
  chr = c("seq1", "seq5"),
  pos = c(81, 11),
  ref = c("GGGTAGGG", "A"),
  alt = c("G", "AGGGGGGGGGGGGGGG")
)

mut <- GRanges(
  seqnames = mut$chr,
  ranges = IRanges(start = mut$pos, end = mut$pos),
  strand = "*",
  ref = mut$ref,
  alt = mut$alt
)

mut_G4 <- G4VarImpact(G4_detected, mut, ref_col = "ref", alt_col = "alt")
filtered_mut_G4 <- filterVarImpact(mut_G4, score_diff_threshold = -0.2)
exportMutG4(filtered_mut_G4, "./result_mut.txt")
exportMutG4(filtered_mut_G4, "./result_mut.xlsx", include_metadata = FALSE)
exportMutG4(filtered_mut_G4, "./result_mut.csv", revcomp_mutG4_seq = FALSE)

# remove all exported files
unlink("./result_mut.txt")
unlink("./result_mut.xlsx")
unlink("./result_mut.csv")
```

`filterSNVImpact` *Filter SNV Impact GRanges Object Based on User-Defined Thresholds*

Description

This function is deprecated and will be removed in a future version.

Usage

```
filterSNVImpact(
  gr,
  raw_score_threshold = NULL,
  mut_score_threshold = NULL,
  score_diff_threshold = NULL
)
```

Arguments

`gr`	A `GRanges` object returned by the `SNVImpactG4` function.
`raw_score_threshold`	A positive numeric value no greater than 4 used as the threshold for the absolute value of `G4.info.score`. G4s with an absolute G4Hunter score exceeding this threshold will be retained. If `NULL`, this threshold is not applied.
`mut_score_threshold`	A positive numeric value no greater than 4 used as the threshold for the absolute value of `mut.score`. Mutated G4s with an absolute G4Hunter score below this threshold will be retained. If `NULL`, this threshold is not applied.

score_diff_threshold
: A negative numeric value no less than -4 used as the threshold for score.diff. G4s with a decrease in G4Hunter score greater than this threshold after variation will be retained. If NULL, this threshold is not applied.

Details

This function filters the SNV Impact GRanges object returned by the SNVImpactG4 function based on user-defined thresholds for the G4.info.score, mut.score, and score.diff parameters. This function filters SNVs that may significantly impair the formation of G4 structures using customizable filtering criteria. You are not required to specify all three threshold parameters. However, at least one threshold parameter must be provided.

Value

A filtered GRanges object, containing only the records that meet the specified threshold criteria.

Deprecated

This function is no longer supported. Use filterVarImpact instead.

See Also

SNVImpactG4 for assessing the impact of SNVs on G4 formation.

Examples

```
if (!requireNamespace("BiocManager", quietly = TRUE)) {
  install.packages("BiocManager")
}

if (!requireNamespace("GenomicRanges", quietly = TRUE)) {
  BiocManager::install("GenomicRanges")
}

if (!requireNamespace("BSgenome.Hsapiens.UCSC.hg19", quietly = TRUE)) {
  BiocManager::install("BSgenome.Hsapiens.UCSC.hg19")
}

library(GenomicRanges)
library(BSgenome.Hsapiens.UCSC.hg19)

hg19 <- BSgenome.Hsapiens.UCSC.hg19
chr21_seq <- DNAStringSet(hg19$chr21)
# Chromosome name is needed
names(chr21_seq) <- "chr21"

G4 <- G4HunterDetect(chr21_seq)

data(snp_gr)

res_snp <- SNVImpactG4(G4, snp_gr, alt_col = "alt")
filtered_snv_eff <- filterSNVImpact(res_snp,
  mut_score_threshold = 1.2,
  score_diff_threshold = -0.2
)
```

```
print(filtered_snv_eff)
```

`filterVarImpact` *Filter Variant Impact G4 Object Based on User-Defined Thresholds*

Description

This function filters the G4 (`GRanges`) object returned by the `G4VarImpact` function based on user-defined thresholds for the `G4.info.max_score`, `mutated.max_score`, and `score.diff` columns. Users are not required to specify all three thresholds; however, at least one must be provided.

Usage

```
filterVarImpact(
  mut_G4,
  raw_score_threshold = NULL,
  mut_score_threshold = NULL,
  score_diff_threshold = NULL
)
```

Arguments

`mut_G4`	A `GRanges` object returned by the `G4VarImpact` function.
`raw_score_threshold`	A positive numeric value (no greater than 4) used as the threshold for the absolute value of `G4.info.max_score`. G4s with an absolute G4Hunter max_score exceeding this threshold will be retained. If `NULL`, this threshold is not applied.
`mut_score_threshold`	A positive numeric value (no greater than 4) used as the threshold for the absolute value of `mutated.max_score`. Mutated G4s with an absolute G4Hunter max_score below this threshold will be retained. If `NULL`, this threshold is not applied.
`score_diff_threshold`	A negative numeric value (no less than -4) used as the threshold for `score.diff`. G4s with a decrease in G4Hunter score greater than this threshold after mutation will be retained. If `NULL`, this threshold is not applied.

Value

A filtered `GRanges` object, containing only the records that meet the specified threshold criteria.

See Also

`G4VarImpact` for assessing the impact of variants on G4 formation.

Examples

```
if (!requireNamespace("BiocManager", quietly = TRUE)) {
  install.packages("BiocManager")
}

if (!requireNamespace("GenomicRanges", quietly = TRUE)) {
  BiocManager::install("GenomicRanges")
}

if (!requireNamespace("BSgenome.Hsapiens.UCSC.hg19", quietly = TRUE)) {
  BiocManager::install("BSgenome.Hsapiens.UCSC.hg19")
}

library(GenomicRanges)
library(BSgenome.Hsapiens.UCSC.hg19)

hg19 <- BSgenome.Hsapiens.UCSC.hg19
chr21_seq <- DNAStringSet(hg19$chr21)
# Chromosome name is needed
names(chr21_seq) <- "chr21"

G4 <- G4HunterDetect(chr21_seq)

data(snp_gr)

res_snp <- G4VarImpact(G4, snp_gr, ref_col = "ref", alt_col = "alt")
filtered_var_eff <- filterVarImpact(res_snp,
  mut_score_threshold = 1.2,
  score_diff_threshold = -0.2
)
print(filtered_var_eff)
```

`G4HunterDetect` *Detect G4 Sequences Using the G4Hunter Algorithm*

Description

This function detects G4 sequences from a given `DNAStringSet` object using the G4Hunter algorithm.

Usage

```
G4HunterDetect(
  sequences = NULL,
  threshold = 1.5,
  window_size = 25,
  include_sequences = TRUE,
  strands = "b"
)
```

Arguments

sequences	A DNAStringSet object containing the input sequences to be analyzed.
threshold	A numeric value specifying the threshold for the G4Hunter score (absolute value). Default is 1.5. It is not recommended to set the threshold below 1.2.
window_size	An integer specifying the window size (bp) used for prediction. Default is 25. Another commonly used window size is 20. However, 25 is generally preferred.
include_sequences	A logical value (TRUE/FALSE) indicating whether to include the predicted G4 sequences in the output. Default is TRUE. Setting this parameter to FALSE can reduce memory usage, which may be beneficial for extremely large genomes. However, we strongly recommend retaining the sequence information in the output, as it is indispensable for subsequent analysis of the impact of variants on G4 formation.
strands	A character string specifying which strand(s) to consider: "b" for both strands or "p" for the positive strand only. Default is "b".

Value

A GRanges object containing the predicted G4 sequences. The GRanges object includes the following metadata columns:

score The final G4Hunter score of the predicted G4 sequence after merging and pruning.

max_score The maximum G4Hunter score observed across all sliding windows covering the G4.

sequence The sequence of the predicted G4 (if include_sequences = TRUE).

Additionally, the following parameters used during detection are stored in the metadata() of the returned GRanges object:

threshold The G4Hunter score threshold used.

window_size The window size used.

include_sequences Whether sequences were included.

strands The strand(s) considered.

If no G4 sequences are detected, an empty GRanges object is returned.

See Also

loadSequence for loading genome sequences into a DNAStringSet object.

Examples

```
if (!requireNamespace("BiocManager", quietly = TRUE)) {
  install.packages("BiocManager")
}

if (!requireNamespace("Biostrings", quietly = TRUE)) {
  BiocManager::install("Biostrings")
}

library(Biostrings)
sequences <- DNAStringSet(c(
  "AGTGAATGGGATGGGAGGAGGGACGGGGTAGTACAGCATAGCATG",
```

```
  "TAGGTAGCTACGACACCCTGCCCTACCCCTACCCCTATCTA"
))
names(sequences) <- c("seq1", "seq2")

G4s <- G4HunterDetect(sequences, threshold = 1.5, window_size = 25)
print(G4s)

seq_path <- system.file("extdata", "seq.fa", package = "G4SNVHunter")
G4s <- G4HunterDetect(loadSequence(seq_path = seq_path))
print(G4s)

seq_path <- system.file("extdata", "seq.txt", package = "G4SNVHunter")
G4s <- G4HunterDetect(loadSequence(seq_path = seq_path))
print(G4s)
```

`G4HunterScore` *Calculate the G4Hunter Score for a Given Sequence*

Description

This function calculates the G4Hunter score for a given nucleotide sequence, which reflects the ability of that sequence to form a G4 structure.

Usage

```
G4HunterScore(seq = NULL)
```

Arguments

`seq`	A single character string representing the nucleotide sequence. Must contain only the characters `A`, `T`, `C`, `G`, `U`, and `N`. The length of the sequence should not be too short (e.g., less than 10 bp).

Value

A numeric value representing the G4Hunter score for the provided sequence.

See Also

`G4HunterDetect` for detecting the G4 sequences in a given `DNAStringSet` object.

Examples

```
sequence <- "GGGTAAGGGATGGGTCGGG"
score <- G4HunterScore(sequence)
print(score)
# A negative value indicates that the G4 sequence
# is located on the reverse strand
sequence <- "GGGTAAGGGATGGGTCGGG"
score <- G4HunterScore(sequence)
print(score)
```

G4VarImpact *Evaluate the Impact of Variants (SNVs, Indels, MNVs) on G4 Sequences*

Description

This function evaluates the impact of variants (SNVs, indels, and MNVs) on G4 formation.

Usage

```
G4VarImpact(
  G4 = NULL,
  variants = NULL,
  ref_col = NULL,
  alt_col = NULL,
  mode = "s",
  sampleid_col = NULL
)
```

Arguments

G4	A GRanges object representing the G4 regions. This object must include a sequence metadata column containing the G4 sequences. It should come from the output of the G4HunterDetect function with include_sequences = TRUE.
variants	A GRanges object representing the variants. This object must include metadata columns for reference and alternative alleles.
ref_col	A character string specifying the name of the column in variants that contains the reference alleles. Default is "ref".
alt_col	A character string specifying the name of the column in variants that contains the alternative alleles. Default is "alt".
mode	A character string indicating the mode of operation. Set to "s" to evaluate the impact of individual variants on G4 regions one at a time (single-variant mode). Set to "m" to assess the combined impact of multiple variants that overlap the same G4 region within a sample (multi-variant Mode). If using "m" mode, you must specify sampleid_col.
sampleid_col	A character string specifying the name of the column in variants that contains the sample IDs. Required when mode is "m"; ignored if mode is "s".

Value

A GRanges object with variant impact results:

Mode "s" (single-variant mode) For each variant-G4 overlap:

- Original G4 metadata (G4.info.*)
- Variant information (variant.info.*)
- Mutated sequence (mutated.G4.seq)
- Annotated mutation sequence (mutated.G4.anno.seq)
- New G4Hunter max_score (mutated.max_score)
- Score difference (score.diff)

Mode "m" (multi-variant mode) For each sample-G4 combination:

- Original G4 metadata (`G4.info.*`)
- Combined variant information (`variant.info.*`)
- Mutated sequence with all variants incorporated (`mutated.G4.seq`)
- Annotated mutation sequence (`mutated.G4.anno.seq`)
- New G4Hunter max_score (`mutated.max_score`)
- Score difference (`score.diff`)

See Also

`G4HunterDetect` for detecting the G4 sequences in a given `DNAStringSet` object. `filterVarImpact` for filtering out variants with significant impact.

Examples

```
if (!requireNamespace("BiocManager", quietly = TRUE)) {
  install.packages("BiocManager")
}

if (!requireNamespace("GenomicRanges", quietly = TRUE)) {
  BiocManager::install("GenomicRanges")
}

if (!requireNamespace("BSgenome.Hsapiens.UCSC.hg19", quietly = TRUE)) {
  BiocManager::install("BSgenome.Hsapiens.UCSC.hg19")
}

library(GenomicRanges)
library(BSgenome.Hsapiens.UCSC.hg19)

# Load sequence for chromosome 21 (hg19)
hg19 <- BSgenome.Hsapiens.UCSC.hg19
chr21_seq <- DNAStringSet(hg19$chr21)
# Chromosome name is needed
names(chr21_seq) <- "chr21"

# Detect G4s in human chromosome 21
G4 <- G4HunterDetect(chr21_seq)

# Load variants
data(snv_gr)

# 's' mode; single-variant mode ('s')
# evaluating each variant individually.
res_snv_s <- G4VarImpact(G4,
                         snv_gr,
                         ref_col = "ref",
                         alt_col = "alt")
print(res_snv_s)

# 'm' mode; multi-variant mode ('m')
# evaluating the combined impact of variants on G4s.
# Grouped by the sample IDs specified in the 'sampleid_col' column.
res_snv_m <- G4VarImpact(G4,
                         snv_gr,
```

```
                          ref_col = "ref",
                          alt_col = "alt",
                          mode = "m",
                          sampleid_col = "sampleid"
)
print(res_snv_m)
```

---

loadSequence *Load Genome Sequences*

Description

This function loads genomic sequences from multiple sources, including a FASTA file, a text file with sequence identifiers and corresponding sequences, or a data frame object.

Usage

```
loadSequence(genome_seq = NULL, seq_path = NULL)
```

Arguments

genome_seq	A data frame containing sequence identifiers and corresponding sequences.
seq_path	A character string specifying the file path to a FASTA file (suffixed with .fa, .fna or .fasta) or a text file with sequence identifiers and corresponding sequences (file headers should not be provided). Ignored if genome_seq is provided.

Value

A DNAStringSet object containing the genome sequences.

Examples

```
# File path for sequences in fasta format
fa_path <- system.file("extdata", "seq.fa", package = "G4SNVHunter")
seq <- loadSequence(seq_path = fa_path)
print(seq)

# Another example
# Load sequences from data.frame
seq_df <- data.frame(
  chr = c("seq1", "seq2"),
  sequence = c(
    paste0(rep("G", 100), collapse = ""),
    paste0(rep("A", 100), collapse = "")
  )
)
seq <- loadSequence(genome_seq = seq_df)
print(seq)
```

loadVariant *Load Small Variant Data from VCF or MAF Files*

Description

This function loads variant data from either a standard VCF or MAF file and filters for small variants, such as SNVs, INDELs, and DELINs.

Usage

```
loadVariant(variant_file, file_type = c("vcf", "maf"), keep_vcf_id = TRUE)
```

Arguments

variant_file	A character string specifying the path to the variant file, which can be in either VCF or MAF format.
file_type	A character string specifying the type of the input file: either "vcf" or "maf". This parameter must be provided. If "maf" is specified, the following columns must be present in the MAF file: "Chromosome", "Start_Position", "Reference_Allele", and "Tumor_Seq_Allele2".
keep_vcf_id	A logical value indicating whether to keep the original ID field from the VCF file in the output. Default is TRUE.

Value

A GRanges object containing the variants loaded from the specified VCF or MAF file. For VCF files, only the ID, REF, and ALT metadata columns are included in the output. For MAF files, all available MAF columns are retained.

Examples

```
# load the vcf file, please do not forget to specify the file type
vcf_path <- system.file("extdata",
                        "example_variants_chr16.vcf",
                        package = "G4SNVHunter")
variants <- loadVariant(vcf_path, file_type = "vcf")
# load the maf file
maf_path <- system.file("extdata",
                        "example_variants_chr16.maf",
                        package = "G4SNVHunter")
variants <- loadVariant(maf_path, file_type = "maf")
```

plotG4Info *Plot Basic Statistics of G4s Detected by the G4HunterDetect Function*

Description

This function generates a series of plots to visualize basic statistics of G4 sequences predicted by the G4Hunter algorithm. The function produces the following plots:

- Distribution of max scores (absolute values).
- Distribution of max scores, split by strand.
- Distribution of scores (absolute values).
- Distribution of scores, split by strand.
- Distribution of sequence lengths (with lengths greater than 50 bp grouped into a single bin).
- Distribution of sequence lengths, split by strand (with lengths greater than 50 bp grouped into a single bin).

Usage

```
plotG4Info(
  G4,
  p_colors = c("#6B9ECC", "#D91117", "#0E619C", "#58AC7B", "#D91117", "#0E619C",
    "#F39C40", "#D91117", "#0E619C")
)
```

Arguments

G4	A GRanges object containing G4 sequences. The object must include at least the following metadata columns: • score: Numeric vector of scores for G4 sequences. • max_score: Numeric vector of maximum scores for G4 sequences. • sequence: Character vector of G4 sequence strings. The GRanges object must include strand information. It should come from the results returned by the G4HunterDetect function.
p_colors	A vector of colors to be used for the plots. It should contain nine color values corresponding to the different plot elements.

Value

A combined plot displays all the generated plots arranged in a grid layout.

See Also

G4HunterDetect for detecting the G4 sequences in a given DNAStringSet object.

Examples

```
seq_path <- system.file("extdata", "seq.txt", package = "G4SNVHunter")
G4 <- G4HunterDetect(loadSequence(seq_path = seq_path))
plotG4Info(G4)
```

plotImpactedG4 *Sequence Logo of the G4 Sequence and Its Mutant Counterpart*

Description

This function visualizes and compares the G4 and its mutant counterpart using sequence logos.

Usage

```
plotImpactedG4(mut_G4 = NULL, keep_gstrand = TRUE)
```

Arguments

mut_G4	A GRanges object containing the mutant G4. If multiple G4s are provided, only the first is plotted.
keep_gstrand	Logical. If TRUE (default), negative strand (C-rich) G4 will be displayed in its complementary reversed form.

Value

A ggplot2-based object containing the G4 sequence logo.

Examples

```
if (!requireNamespace("BiocManager", quietly = TRUE)) {
  install.packages("BiocManager")
}

if (!requireNamespace("GenomicRanges", quietly = TRUE)) {
  BiocManager::install("GenomicRanges")
}

if (!requireNamespace("BSgenome.Hsapiens.UCSC.hg19", quietly = TRUE)) {
  BiocManager::install("BSgenome.Hsapiens.UCSC.hg19")
}

library(GenomicRanges)
library(BSgenome.Hsapiens.UCSC.hg19)

hg19 <- BSgenome.Hsapiens.UCSC.hg19
chr21_seq <- DNAStringSet(hg19$chr21)
# Chromosome name is needed
names(chr21_seq) <- "chr21"

G4 <- G4HunterDetect(chr21_seq)

# Load SNVs
data(snp_gr)

res_snp <- G4VarImpact(G4, snp_gr, ref_col = "ref", alt_col = "alt")
plotImpactedG4(res_snp[1])
```

`plotImpactSeq` *Visualize the variants in G4 sequence*

Description

This function is deprecated and will be removed in a future version.

Usage

```
plotImpactSeq(filtered_gr, ncol = 1)
```

Arguments

`filtered_gr`	A `GRanges` object containing sequence data and G4Hunter scores. The object must have metadata columns named `G4.info.score`, `mut.score`, `G4.info.sequence`, and `mut.G4.seq`.
`ncol`	An integer specifying the number of columns in the output plot grid. Default is 1.

Details

This function plot sequence logos to visualize sequence variants caused by SNVs or SNPs, with the location of the variants highlighted by rectangles and arrows.

Value

A plot that displays the grid of sequence logos, showing the differences between the original and mutated sequences.

Deprecated

This function is no longer supported. Use `plotImpactedG4` instead.

See Also

`SNVImpactG4` for evaluating the impact of SNVs on G4 formation, and `filterSNVImpact` for filtering G4s that are significantly affected by SNVs.

Examples

```
if (!requireNamespace("GenomicRanges", quietly = TRUE)) {
  BiocManager::install("GenomicRanges")
}

library(GenomicRanges)

seq <- data.frame(chr = c("seq1", "seq2"),
                  seq = c("ATTTGGGAGGGAGGGAGGGATGATGAAAATTTTATTTATTTTATTTA",
                          "TTTATACTATTCCCTTACCCTCCCATCCCCATACGGCATCTAGATC"))
seq_gr <- loadSequence(seq)
G4 <- G4HunterDetect(seq_gr)

snv_gr <- GRanges(seqnames = c("seq1", "seq2"),
```

```
                    ranges = IRanges(start = c(18, 23), width = 1),
                    ref = c("G", "C"),
                    alt = c("C", "G"))

effect <- SNVImpactG4(G4, snv_gr, alt_col = "alt")
plotImpactSeq(effect, ncol = 2)
```

`plotSNVImpact` *Plot the Impact of SNVs on G4Hunter Scores*

Description

This function is deprecated and will be removed in a future version.

Usage

```
plotSNVImpact(gr, p_colors = c("#b22d2d", "#6ca4d6", "#2d69b0", "#1f77b4"))
```

Arguments

`gr`	A `GRanges` object containing the G4Hunter scores and associated metadata. The object should include the following columns: • `G4.info.score`: Numeric vector of the original G4Hunter scores. • `mut.score`: Numeric vector of the G4Hunter scores after mutation. • `score.diff`: Numeric vector of the differences between the original and mutant G4Hunter scores.
`p_colors`	A vector of four colors used for plotting.

Details

This function generates two plots for visualizing the impact of SNVs on G4 formation: 1. A scatter plot with density shading comparing the original G4Hunter score and the mutant G4Hunter score. 2. A density plot showing the distribution of score changes between the original and mutant G4 sequences.

Value

A combined plot (using `plot_grid`) containing two subplots: - Density scatter plot comparing original vs mutant G4Hunter scores. - Density plot of the G4Hunter score differences.

Deprecated

This function is no longer supported. Use `plotVarImpact` instead.

See Also

`SNVImpactG4` for evaluating the impact of SNVs on G4 formation, and `filterSNVImpact` for filtering G4s that are significantly affected by SNVs.

Examples

```
if (!requireNamespace("BiocManager", quietly = TRUE)) {
  install.packages("BiocManager")
}

if (!requireNamespace("GenomicRanges", quietly = TRUE)) {
  BiocManager::install("GenomicRanges")
}

if (!requireNamespace("BSgenome.Hsapiens.UCSC.hg19", quietly = TRUE)) {
  BiocManager::install("BSgenome.Hsapiens.UCSC.hg19")
}

library(GenomicRanges)
library(BSgenome.Hsapiens.UCSC.hg19)

hg19 <- BSgenome.Hsapiens.UCSC.hg19
chr21_seq <- DNAStringSet(hg19$chr21)
# Chromosome name is needed
names(chr21_seq) <- "chr21"

G4 <- G4HunterDetect(chr21_seq)

# Load SNVs
data(snp_gr)

res_snp <- SNVImpactG4(G4, snp_gr, alt_col = "alt")
plotSNVImpact(res_snp)
```

---

plotVarImpact *Plot the Impact of Variants on G4 Formation*

Description

This function generates two panels to visualize the impact of variants on G4 formation:

1. A 2D density plot of absolute maximum G4Hunter scores before and after mutation.
2. A density plot showing the distribution of score differences.

Usage

```
plotVarImpact(
  gr,
 p_colors = c(fill_2d = "#376597", diagonal = "#b22d2d", density_fill = "#6ca4d6",
    density_line = "#2d69b0", vline = "#1f77b4")
)
```

Arguments

gr	A GRanges object containing G4Hunter scores and associated metadata. Required metadata columns include: • G4.info.max_score: Numeric vector of original maximum G4Hunter scores.

- `mutated.max_score`: Numeric vector of maximum G4Hunter scores after mutation.
- `score.diff`: Numeric vector of score differences.

`p_colors` A character vector of five colors used for plotting. The vector must include:

`"fill_2d"` Fill Color for the 2D density plot.

`"diagonal"` Color for the diagonal line in the 2D density plot.

`"density_fill"` Fill color for the density plot.

`"density_line"` Line color for the density plot.

`"vline"` Color for the vertical line at zero in the density plot.

Value

A `ggplot` object combining two subplots:

1. 2D density plot of the absolute maximum G4Hunter scores before and after mutation.
2. Density plot of the maximum G4Hunter score differences.

See Also

`G4VarImpact` for computing the effects of variants on G4 formation, and `filterVarImpact` for screening G4s whose formation propensity may be impaired by variants.

Examples

```
if (!requireNamespace("BiocManager", quietly = TRUE)) {
  install.packages("BiocManager")
}

if (!requireNamespace("GenomicRanges", quietly = TRUE)) {
  BiocManager::install("GenomicRanges")
}

if (!requireNamespace("BSgenome.Hsapiens.UCSC.hg19", quietly = TRUE)) {
  BiocManager::install("BSgenome.Hsapiens.UCSC.hg19")
}

library(GenomicRanges)
library(BSgenome.Hsapiens.UCSC.hg19)

hg19 <- BSgenome.Hsapiens.UCSC.hg19
chr21_seq <- DNAStringSet(hg19$chr21)
# Chromosome name is needed
names(chr21_seq) <- "chr21"

G4 <- G4HunterDetect(chr21_seq)

# Load SNVs
data(snp_gr)

res_snp <- G4VarImpact(G4, snp_gr, ref_col = "ref", alt_col = "alt")
plotVarImpact(res_snp)
```

snp_gr *Single Nucleotide Polymorphisms GRanges Object*

Description

This dataset contains a GRanges object storing single nucleotide polymorphisms (SNPs).

Usage

```
data(snp_gr)
```

Format

A GRanges object with 30,000 SNPs.

SNVImpactG4 *Evaluate the Impact of SNVs on G4 Sequences*

Description

This function is deprecated and will be removed in a future version.

Usage

```
SNVImpactG4(
  G4 = NULL,
  snvs = NULL,
  alt_col = "alt",
  mode = "s",
  sampleid_col = NULL,
  snvid_col = NULL
)
```

Arguments

G4	A GRanges object representing the G4 regions. This object must include metadata columns for G4 sequence and G4Hunter scores. It should come from the results returned by the G4HunterDetect function.
snvs	A GRanges object representing the SNVs. This object must include metadata columns for SNV alternative alleles and, optionally, SNV IDs.
alt_col	A character string specifying the name of the column in snvs that contains the alternative alleles for the SNVs. The default is "alt".
mode	A character string indicating the mode of operation. Set to "s" to evaluate the impact of individual SNVs on G4 regions one at a time. Set to "m" to assess the combined impact of multiple SNVs that overlap the same G4 region within a sample. If using "m" mode, you must specify the sampleid_col and snvid_col parameters.

`sampleid_col` A character string specifying the name of the column in `snvs` that contains the sample IDs. This parameter is required when `mode` is `"m"`, and is ignored if `mode` is `"s"`.

`snvid_col` A character string specifying the column name in `snvs` that contains SNV IDs. Required when `mode` is `"m"`. Ignored if `mode` is `"s"`.

Details

This function evaluates the impact of SNVs on G4 formation.

Value

A `GRanges` object depending on the `mode` parameter:

Mode "s": For `mode = "s"`, the returned `GRanges` object includes the following metadata columns:

- `seqnames` Identifiers for SNVs.
- `ranges` Position of the SNVs.
- `strand` Strand of the SNVs.
- `SNV.info.*` Metadata columns related to each SNV.
- `G4.info.*` Metadata columns from the original G4 object.
- `mut.G4.seq` The mutated G4 sequence after applying the SNV change.
- `mut.G4.anno.seq` The mutated G4 sequence, with the mutated bases annotated using square brackets.
- `mut.score` The G4Hunter score of the mutated sequence.
- `score.diff` The difference between the mutated G4Hunter score and the original G4Hunter score. The value is calculated as the absolute value of the mutated G4Hunter score minus the absolute value of the original G4Hunter score.

Mode "m": For `mode = "m"`, the returned `GRanges` object includes the following metadata columns:

- `seqnames` Identifiers for G4 sequences.
- `ranges` Position of the G4 sequences (start and end).
- `strand` Strand of the G4 sequences.
- `G4.info.*` Metadata columns from the original G4 object.
- `snv.ids` Concatenated SNV IDs for all SNVs affecting the G4 region.
- `sample.ids` A semicolon-separated list of sample IDs overlapping each G4 region.
- `mut.G4.seq` The mutated G4 sequence after applying the combined SNV changes.
- `mut.G4.anno.seq` The mutated G4 sequence, with the mutated bases annotated using square brackets.
- `mut.score` The G4Hunter score of the mutated sequence.
- `score.diff` The difference between the mutated G4Hunter score and the original G4Hunter score. This value is calculated as the absolute value of the mutated G4Hunter score minus the absolute value of the original G4Hunter score.

Deprecated

This function is no longer supported. Use `G4VarImpact` instead.

See Also

`G4HunterDetect` for detecting the G4 sequences in a given `DNAStringSet` object. `G4HunterScore` for calculating the G4Hunter scores for a given sequence. `filterSNVImpact` for filtering out SNVs with significant impact.

Examples

```
if (!requireNamespace("BiocManager", quietly = TRUE)) {
  install.packages("BiocManager")
}

if (!requireNamespace("GenomicRanges", quietly = TRUE)) {
  BiocManager::install("GenomicRanges")
}

if (!requireNamespace("BSgenome.Hsapiens.UCSC.hg19", quietly = TRUE)) {
  BiocManager::install("BSgenome.Hsapiens.UCSC.hg19")
}

library(GenomicRanges)
library(BSgenome.Hsapiens.UCSC.hg19)

# Load sequence for chromosome 21 (hg19)
hg19 <- BSgenome.Hsapiens.UCSC.hg19
chr21_seq <- DNAStringSet(hg19$chr21)
# Chromosome name is needed
names(chr21_seq) <- "chr21"

# Detect G4s in human chromosome 21
G4 <- G4HunterDetect(chr21_seq)

# 's' mode
# Load SNPs
data(snp_gr)

# Obtain SNPs that overlap with G4 regions and assess their impact on G4.
# In variant-centric mode ('s'), evaluating each SNP individually.
res_snp <- SNVImpactG4(G4, snp_gr, alt_col = "alt")
print(res_snp)

# 'm' mode
# Load SNVs
data(snv_gr)

# Obtain SNVs that overlap with G4 regions and assess their impact on G4.
# In sample-centric mode ('m'), evaluate the combined impact of SNVs on G4s.
# Grouped by the sample IDs specified in 'sampleid_col'.
res_snv <- SNVImpactG4(G4, snv_gr,
  alt_col = "alt",
  mode = "m", sampleid_col = "sampleid", snvid_col = "snv_id"
)
print(res_snv)
```

---

snv_gr *Single Nucleotide Variant GRanges Object*

Description

This dataset contains a GRanges object storing single nucleotide variants (SNVs).

Usage

```
data(snv_gr)
```

Format

A GRanges object with 50,000 SNVs.

Index