Package 'RADstackshelpR'

May 7, 2026

Title Optimize the De Novo Stacks Pipeline via R

Version 0.1.0

Description Offers a handful of useful wrapper functions which streamline the reading, analyzing, and visualizing of variant call format (vcf) files in R. This package was designed to facilitate an explicit pipeline for optimizing Stacks (Rochette et al., 2019) (<doi:10.1111/mec.15253>) parameters during de novo (without a reference genome) assembly and variant calling of restriction-enzyme associated DNA sequence (RADseq) data. The pipeline implemented here is based on the 2017 paper ``Lost in Parameter Space'' (Paris et al., 2017) (<doi:10.1111/2041-210X.12775>) which establishes clear recommendations for optimizing the parameters 'm', 'M', and 'n', during the process of assembling loci.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.1.1

Imports vcfR, ggplot2, ggridges, gridExtra

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Devon DeRaad [aut, cre] (ORCID: <https://orcid.org/0000-0003-3105-985X>)

Maintainer Devon DeRaad <devonderaad@gmail.com>

Repository CRAN

Date/Publication 2021-08-19 09:10:07 UTC

Contents

`optimize_bigM` *Optimize the M parameter during denovo stacks assembly*

Description

This function requires the path to stacks vcf file(s) as input. There are slots for varying the M parameter from 1-8 (as recommended by Paris et al. 2017). After running stacks with each of the M options, plug the output vcf files into this function to calculate the effect of varying M on the number of SNPs/loci built. Plug the output of this function into vis_loci() to visualize the optimal the M parameter for your dataset at the 'R80' cutoff (Paris et al. 2017).

Usage

```
optimize_bigM(
  M1 = NULL,
  M2 = NULL,
  M3 = NULL,
  M4 = NULL,
  M5 = NULL,
  M6 = NULL,
  M7 = NULL,
  M8 = NULL
)
```

Arguments

M1	Path to the input vcf file for a run when M=1
M2	Path to the input vcf file for a run when M=2
M3	Path to the input vcf file for a run when M=3
M4	Path to the input vcf file for a run when M=4
M5	Path to the input vcf file for a run when M=5
M6	Path to the input vcf file for a run when M=6
M7	Path to the input vcf file for a run when M=7
M8	Path to the input vcf file for a run when M=8

Value

A list containing four summary dataframes, 'snp' showing the number of non-missing SNPs retained in each sample at each m value, 'loci' showing the number of non-missing loci retained in each sample at each m value, 'snp.R80' showing the total number of SNPs retained at an 80% completeness cutoff, and 'loci.R80' showing the total number of polymorphic loci retained at an 80% completeness cutoff.

Examples

```
optimize_bigM(M1=system.file("extdata","bigM1.vcf.gz",package="RADstackshelpR",mustWork=TRUE))
```

optimize_m *Optimize the m parameter during denovo stacks assembly*

Description

This function requires the path to stacks vcf file(s) as input. There are slots for varying the m parameter from 3-7 (as recommended by Paris et al. 2017). After running stacks with each of the m options, plug the output vcf files into this function to calculate the effect of varying m on depth and number of SNPs/loci built. Plug the output of this function into vis_loci() to visualize the optimal the m parameter for your dataset at the 'R80' cutoff (Paris et al. 2017).

Usage

```
optimize_m(m3 = NULL, m4 = NULL, m5 = NULL, m6 = NULL, m7 = NULL)
```

Arguments

m3	Path to the input vcf file for a run when m=3
m4	Path to the input vcf file for a run when m=4
m5	Path to the input vcf file for a run when m=5
m6	Path to the input vcf file for a run when m=6
m7	Path to the input vcf file for a run when m=7

Value

A list containing five summary dataframes, 'depth' showing depth per sample for each m value, 'snp' showing the number of non-missing SNPs retained in each sample at each m value, 'loci' showing the number of non-missing loci retained in each sample at each m value, 'snp.R80' showing the total number of SNPs retained at an 80% completeness cutoff, and 'loci.R80' showing the total number of polymorphic loci retained at an 80% completeness cutoff.

Examples

```
optimize_m(m3=system.file("extdata","m3.vcf.gz",package="RADstackshelpR",mustWork=TRUE))
```

optimize_n *Optimize the n parameter during denovo stacks assembly*

Description

This function requires the path to stacks vcf file(s) as input. There are slots for varying the n parameter across M-1, M, and M-1 (as recommended by Paris et al. 2017). After running stacks with each of the n options, plug the output vcf files into this function to visualize the effect of varying m on number of SNPs and loci built to recognize which value optimizes the n parameter for your dataset at the 'R80' cutoff (Paris et al. 2017).

Usage

```
optimize_n(nequalsMminus1 = NULL, nequalsM = NULL, nequalsMplus1 = NULL)
```

Arguments

nequalsMminus1	Path to the input vcf file for a run when n=M-1
nequalsM	Path to the input vcf file for a run when n=M
nequalsMplus1	Path to the input vcf file for a run when n=M+1

Value

A dataframe showing the number of SNPs and loci retained across filtering levels for each n value

Examples

```
optimize_n(nequalsM =
system.file("extdata","nequalsm.vcf.gz",package="RADstackshelpR",mustWork=TRUE))
```

---

vis_depth *Visualize the effect of varying the m parameter on the mean depth of each sample*

Description

This function takes the list of dataframes output by optimize_m() as input. The function then uses ggplot2 to visualize the effect of m on depth.

Usage

```
vis_depth(output = NULL)
```

Arguments

output	A list containing 5 dataframes generated by optimize_m()

Value

A plot showing the depth of each sample at each given m value

Examples

```
vis_depth(output =
readRDS(system.file("extdata","optimize.m.output.RDS",package="RADstackshelpR",mustWork=TRUE)))
```

vis_loci — *Visualize the effect of varying a stacks parameter on the number of polymorphic loci retained*

Description

This function takes the list of dataframes output by optimize_m(), optimize_M(), or optimize_n() as input. The function then uses ggplot2 to visualize the effect of the given stacks on the number of polymorphic loci retained, reporting which value is optimal.

Usage

```
vis_loci(output = NULL, stacks_param = NULL)
```

Arguments

output	A list containing 5 dataframes generated by optimize_m()
stacks_param	A character string indicating the stacks parameter iterated over

Value

A plot showing the number of polymorphic loci retained at each given parameter value

Examples

```
vis_loci(output =
readRDS(system.file("extdata","optimize.m.output.RDS",package="RADstackshelpR",mustWork=TRUE)),
         stacks_param = "m")
```

vis_snps — *Visualize the effect of varying a stacks parameter on the number of SNPs retained*

Description

This function takes the list of dataframes output by optimize_m(), optimize_M(), or optimize_n() as input. The function then uses ggplot2 to visualize the effect of the given stacks on the number of SNPs retained.

Usage

```
vis_snps(output = NULL, stacks_param = NULL)
```

Arguments

output	A list containing 5 dataframes generated by optimize_m()
stacks_param	A character string indicating the stacks parameter iterated over

Value

A plot showing the number of SNPs retained at each given parameter value

Examples

```
vis_snps(output=
readRDS(system.file("extdata","optimize.m.output.RDS",package="RADstackshelpR",mustWork=TRUE)),
         stacks_param = "m")
```

Index