Package 'gmapR'

March 26, 2013

Maintainer Cory Barr <barr.cory@gene.com>

License Artistic-2.0

Title Provides convenience methods to work with GMAP and GSNAP from within R

Type Package

Author Cory Barr, Thomas Wu, Michael Lawrence

Description GSNAP and GMAP are a pair of tools to align short-read data written by Tom Wu. This package provides convenience methods to work with GMAP and GSNAP from within R. In addition, it provides methods to tally alignment results on a per-nucleotide basis using the bam_tally tool.

Version 1.0.0

Depends R (>= 2.15.0), methods, GenomicRanges

Imports IRanges, Rsamtools (>= 1.7.4), rtracklayer (>= 1.17.15),GenomicRanges, GenomicFeatures, Biostrings, VariantAnnotation,tools, Biobase

Suggests RUnit, BSgenome.Dmelanogaster.UCSC.dm3,BSgenome.Scerevisiae.UCSC.sacCer3, VariantAnnotation,org.Hs.eg.db, TxDb.Hsapiens.UCSC.hg19.knownGene,BSgenome.Hsapiens.UCSC.hg19, LungCancerLines

Collate GmapBamReader-class.R GmapGenomeDirectory-class.R GmapGenome-class.R GmapSnpDirectory-class.R GmapSnps-class.R GsnapParam-class.R GsnapOutput-class.R atoiindex-command.R BamTallyParam-class.R bam_tally-command.R cmetindex-command.R get-genome-command.R gmap-command.R gmap_build-command.R gsnap-command.R iit-format.R iit_store-command.R info.R snpindex-command.R system.R test_gmapR_package.R makeGmapGenomePackage.R TP53Genome.R utils.R asSystemCall.R

R topics documented:

BamTallyParam-class *Class* "BamTallyParam"

Description

A BamTallyParam object stores parameters for bam_tally. The function of the same name serves as its constructor.

Usage

```
BamTallyParam(genome, which = RangesList(), cycle_breaks = NULL,
              high_base_quality = 0L,
              minimum_mapq = 0L,
              concordant_only = FALSE, unique_only = FALSE,
              primary_only = FALSE,
              min_depth = 0L, variant_strand = 0L,
              ignore_query_Ns = FALSE,
              indels = FALSE)
```

Arguments

genome	A GmapGenome object, or something coercible to one.
which	A RangesList or something coercible to one that limits the tally to that range or set of ranges. By default, the entire genome is processed.
cycle_breaks	The breaks, like those passed to cut for aggregating the per-cycle counts. If NULL, no per-cycle counts are returned.
high_base_quality	The minimum mapping quality for a read to be counted as high quality.
minimum_mapq	Minimum mapping quality for a read to be counted at all.
concordant_only	Consider only what gnsap calls “concordant” alignments.
unique_only	Consider only the uniquly mapped reads.
primary_only	Consider only primary pairs.
min_depth	The minimum number of reads overlapping a position for it to be counted.

variant_strand — The number of strands on which a variant must be seen for it to be counted. This means that a value of 0 will report reference alleles in addition to variants. A value of 1 will report only positions where a variant was seen on at least one strand, and 2 requires the variant be seen on both strands. Setting this to 1 is a good way to save resources.

ignore_query_Ns — Whether to ignore the N base pairs when counting. Can save a lot of resources when processing low quality data.

indels — Whether to return indel counts; not supported yet.

See Also

bam_tally

---

bam_tally-methods *Per-position Alignment Summaries*

Description

Given a set of alignments, for each position in the genome output counts for the reference allele and all alternate alleles. Often used as a precursor to detecting variants. Indels will be supported soon.

Usage

```
## S4 method for signature 'BamFile'
bam_tally(x, param, ...)
## S4 method for signature 'character'
bam_tally(x, param, ...)
```

Arguments

x	a BamFile object or string path to a BAM file to read
param	The BamTallyParam object with parameters for the tally operation.
...	Arguments that override settings in param.

Value

A GRanges, with a range for each position that passed the filters, and with the following elementMetadata columns:

location	A string representation of the location, of the form "chr:pos". This makes it easy, e.g., to check for the presence of a variant in another result object.
ref	The reference base at that position.
alt	The base for the alternate allele, NA for the reference allele row.
ncycles	The number of unique cycles at which the alternate allele was observed, NA for the reference allele row.
ncycles.ref	The number of unique cycles at which the reference allele was observed.
count	The number of reads with the alternate allele, NA for the reference allele row.
count.ref	The number of reads with the reference allele.

count.total	The total number of reads at that position, including reference and all alternates.
high.quality	The number of reads for the alternate allele that were above high_quality_cutoff, NA for the reference allele row.
high.quality.ref	The number of reads for the reference allele that were above high_quality_cutoff.
mean.quality	The mean mapping quality for the alternate allele, NA for the reference allele row.
mean.quality.ref	The mean mapping quality for the reference allele.
count.pos	The number of positive strand reads for the alternate allele, NA for the reference allele row.
count.pos.ref	The number of positive strand reads for the reference allele.
count.neg	The number of negative strand reads for the alternate allele, NA for the reference allele row.
count.neg.ref	The number of negative strand reads for the reference allele.

An additional column is present for each bin formed by the cycle_breaks parameter, with the read count for that bin.

Author(s)

Michael Lawrence

---

directory *Get the Path to the Location on Disk from a gmapR Class*

Description

Many objects in gmapR represent data stored on disk. The directory accessor will return this directory.

Usage

```
directory(x)
```

Arguments

x	A GmapGenome or GmapSnps object

Value

a character vector

GmapGenome-class *Class* "GmapGenome"

Description

The GmapGenome class represents a genome that has been indexed for use with the GMAP suite of tools. It is typically used as a parameter to the functions gsnap and bam_tally. This class also provides the means to index new genomes, from either a FASTA file or a BSgenome object. Genome indexes are typically stored in a centralized directory on the file system and are identified by a string key.

Constructor

GmapGenome(genome, directory = GmapGenomeDirectory(create = create), name = genomeName(genor

Creates a GmapGenome corresponding to the genome argument, which may be either a string identifier of the genome within directory, a FastaFile or DNAStringSet of the genome sequence, or a BSgenome object.

The genome index is stored in directory argument, which may be either a GmapGenomeDirectory object, or a string path.

The name argument is the actual key used for storing the genome index within directory. If genome is a string, it is taken as the key. If a FastaFile, it is the basename of the file without the extension. If a BSgenome, it is the providerVersion. Otherwise, the name must be specified. If create is TRUE, the genome index is created if one with that name does not already exist. This obviously only works if genome actually contains the genome sequence.

The first example below gives the typical and recommended usage when implementing a reproducible analysis.

Accessors

path(object): returns the path to the directory containing the genome index files.

directory(x): returns the GmapGenomeDirectory that is the parent of the directory containing the index files for this genome.

genome(x): gets the name of this genome.

seqinfo(x): gets the Seqinfo for this genome; only sequence names and lengths are available.

Author(s)

Michael Lawrence

Examples

```
## Not run:
library(BSgenome.Dmelanogaster.UCSC.dm3)
flyGG <- GmapGenome(Dmelanogaster, create = TRUE)

## access system-wide genome using a key
flyGG <- GmapGenome(genome = "dm3",
                    directory = path.expand("/usr/share/gmap"))

##create a GmapGenome from a FASTA file
fa <- system.file("extdata/hg19.MT.fasta", package="gmapR")
```

```
fastaFile <- rtracklayer::FastaFile(fa)
gmapGenome <- GmapGenome(fastaFile, create=TRUE)

## End(Not run)
```

GmapGenomeDirectory-class
Class "GmapGenomeDirectory"

Description

The GmapGenomeDirectory class stores a path to a directory containing a one or more genome-specific subdirectories, each represented by a GmapGenome. Inside those directories are the files that the GMAP suite of tools uses for alignment, tallying, and other operations. This class is typically used to create a GmapGenome object. The default directory is ~/.local/share/gmap, following the freedesktop.org XDG standard.

Constructor

GmapGenomeDirectory(path = getDefaultGmapGenomePath(), create = FALSE): Creates an object pointing to the directory at path, creating it if it does not yet exist and create is TRUE.

Methods

path(object): gets the path to the genome directory.

genome(x): gets the names of the genomes in the directory.

Author(s)

Michael Lawrence

See Also

GmapGenome-class

Examples

```
gmapGenomePath <- file.path(getwd(), "newGmapGenomeDirectory")
gmapGenomeDirectory <- GmapGenomeDirectory(gmapGenomePath, create = TRUE)
```

GmapSnpDirectory-class *Class* "GmapSnpDirectory"

Description

This class represents a directory containig one or more sets of SNPs, each corresponding to a genome. These SNP databases enable SNP-tolerant alignment with GMAP and GSNAP. If the underlying files have not been created, this class provides a means to do so.

Methods

[[<- signature(x = "GmapSnpDirectory", i = "ANY", j = "ANY"): ...

length signature(x = "GmapSnpDirectory"): ...

names signature(x = "GmapSnpDirectory"): ...

path signature(object = "GmapSnpDirectory"): ...

Author(s)

Michael Lawrence

GmapSnps-class *Class* "GmapSnps"

Description

This class represents a set of SNPs (single nucleotide polymorphisms) for use with GMAP and GSNAP (typically for SNP-tolerant alignment.)

Usage

GmapSnps(snps, directory, name = snps, create = FALSE, ...)

Arguments

snps	A path to a VCF file
directory	The directory to create the IIT files used by GMAP and GSNAP
name	If provided, the name to give the database of SNPs. If not provided, defauts to the snps argument.
create	If the directory provided in the directory argument does not exist, create it.
...	Additional arguments to be passed to the SNPs replacement method.

Objects from the Class

##TODO: doc these args Objects can be created by calls of the form GmapSnps(snps, directory, name, create).

Accessors

name(x): returns the name of the GmapSnps object

directory(x): returns the GmapGenomeDirectory that is the parent of the directory containing the index files for this GmapSnps object.

Methods

directory signature(x = "GmapSnps"): ...

Author(s)

Michael Lawrence

---

gmap_build-methods *Build Gmap/Gsnap Genome*

Description

Construct the IIT (interval index tree) needed from the GMAP suite of tools to run from a genome file. IIT files are an oligomer index and what allow GMAP and GSNAP to efficiently lookup interval information for fast genomic mapping. Fast and SNP-tolerant detection of complex variants and splicing in short reads offers an depth explication of IIT files and their use in GMAP and GSNAP.

Arguments

d	genome name
D	destination directory for installation (defaults to gmapdb directory specified at configure time
k	k-mer value for genomic index (allowed: 12..15, default 14)
S	do not order chromosomes in numeric/alphabetic order, but use order in FASTA file(s)
g	files are gzipped, so need to gunzip each file first

Methods

signature(x = "ANY", genome = "GmapGenome")

signature(x = "character", genome = "GmapGenome")

signature(x = "DNAStringSet", genome = "GmapGenome")

Examples

```
## Not run: flyGG <- GmapGenome(genome = "dm3",
                  directory = ggd)
gmap_build(x=Dmelanogaster, genome=flyGG)

## End(Not run)
```

gsnap-methods *Align a Set of Reads Using the GSNAP Aligner*

Description

Given a set of alignments, align them to a genome using the GSNAP algorithm. The GSNAP algorithm contains a number of features making it a very high quality algorithm for dealing with short reads and those from RNA-seq data in particular. Via the GsnapParam class and the gsnap function, R users are given complete control over GSNAP.

Usage

```
## S4 method for signature 'character,characterORNULL,GsnapParam'
gsnap(input_a, input_b, params,
              output = file_path_sans_ext(input_a, TRUE),
              consolidate = TRUE, ...)
```

Arguments

input_a	A path to the FASTA file containing reads to align against a GmapGenome object. If the sequencing data is single-end, this is the only FASTA file used as input.
input_b	If provided, a path to the FASTA file containing the second set of reads from paired-end sequencing data.
params	A GsnapParam object to configure the behavior of GSNAP.
output	The output path for the GSNAP alignments. The results will be saved in dirname(output). If split_output in params is TRUE, basename(output) is used as the common stem for the multiple output files. Otherwise, the results are saved to output with the "bam" extension appended.
consolidate	If GSNAP is run with multiple worker threads, each thread will output its own set of files. If consolidate is set to TRUE, these files will be merged. The default is TRUE.
...	Additional arguments to pass to GSNAP not specifically supported by the gmapR package.

Value

A GsnapOutput class.

Author(s)

Michael Lawrence

GsnapOutput-class *Class* "GsnapOutput"

Description

A GsnapOutput object stores locations of data output by the GSNAP alignment algorithm.

Objects from the Class

GsnapOutput objects are created from the gsnap function, though the function GsnapOutput can also be used as a constructor.

Coercion

In the code snippets below, x is a GsnapOutput object.

as(x, BamFile), as(x, BamFileList):
: Returns either a BamFile or BamFileList object containing paths to the output of GSNAP.

asBam(x):
: converts all gsnap SAM files to BAM files and creates the .bai index files.

Author(s)

Michael Lawrence

See Also

gsnap

GsnapParam-class *Class* "GsnapParam"

Description

A GsnapParam object stores parameters for gsnap. The function of the same name serves as its constructor.

Usage

```
GsnapParam(genome, unique_only = FALSE,
  max_mismatches = NULL, suboptimal_levels = 0L,
  mode = "standard", snps = NULL,
  npaths = if (unique_only) 1L else 100L,
  quiet_if_excessive = unique_only, nofails = unique_only,
  split_output = !unique_only,
  novelsplicing = FALSE, splicing = NULL,
  nthreads = 1L, part = NULL, batch = "2", ...)
```

Arguments

genome	A GmapGenome object to align against
unique_only	Whether only alignments with a unique match should be output. The default is FALSE.
max_mismatches	The maximum number of mismatches to allow per alignment. If NULL, then the value defaults to ((readlength + 2) / 12 - 2))
suboptimal_levels	Report suboptimal hits beyond best hit. The default is 0L.
mode	The alignment mode. It can be "standard", "cmet-stranded", "cmet-nonstranded", "atoi-stranded", or "atoi-nonstranded". The default is "standard".
snps	If not NULL, then a GmapSnps object. Provided SNPs will not count as mismatches.
npaths	The maximum number of paths to print.
quiet_if_excessive	If more than maximum number of paths are found, then no alignment from the read will be in the output.
nofails	Exclude failed alignments from output
split_output	Basename for multiple-file output, separately for nomapping, halfmapping_uniq, halfmapping_mult, unpaired_uniq, unpaired_mult, paired_uniq, paired_mult, concordant_uniq, and concordant_mult results (up to 9 files, or 10 if –fails-as-input is selected, or 3 for single-end reads)
novelsplicing	Logical indicating whether to look for novel splicing events. FALSE is the default.
splicing	If not NULL, a GmapSplices object. NULL is the default.
nthreads	The number of worker threads gsnap should use to align.
part	If not NULL, then process only the i-th out of every n sequences e.g., 0/100 or 99/100 (useful for distributing jobs to a computer farm). If NULL, then all sequences are processed. NULL is the default.
batch	This argument allows control over gsnap's memory mapping and allocation. The default is mode 2. Mode 0: {offsets=allocate, positions=mmap, genome=mmap}, Mode 1: {offsets=allocate, positions=mmap & preload,genome=mmap}, Mode 2: {offsets=allocate, positions=mmap & preload,genome=mmap & preload}, Mode 3: {offsets=allocate, positions=allocate,genome=mmap & preload}, Mode 4: {offsets=allocate, positions=allocate,genome=allocate}
...	Additional parameters for gsnap. See gsnap's full documentation for those available.

See Also

gsnap

---

internals *gmapR2 internals*

Description

Internal methods, etc, that need an alias but are not intended for public use, at least not yet.

makeGmapGenomePackage *Function to create a GmapGenome package from a GmapGenome object*

Description

A GmapGenome object is required to align reads using the GSNAP or GMAP algorithms. The makeGmapGenomePackage function allows users to save a particular GmapGenome object in an R package.

Usage

```
makeGmapGenomePackage(gmapGenome, version, maintainer, author,
destDir = ".", license = "Artistic-2.0", pkgName)
```

Arguments

gmapGenome	A GmapGenome object.
version	The version number of this package.
maintainer	The maintainer of the package. The string must contain a valid email address.
author	The author of the package
destDir	The path that the new GmapGenome package should be created at.
license	The package's license (and its version)
pkgName	The name the package should have. Though free form, names of the form GmapGenome.Organism.Source.Build are recommended. E.g., GmapGenome.Hsapiens.UCSC.hg19

Author(s)

Cory Barr

See Also

GmapGenome

Examples

```
## Not run:
library(gmapR)

if (!require(BSgenome.Dmelanogaster.UCSC.dm3)) {
  library(BiocInstaller)
  biocLite("BSgenome.Dmelanogaster.UCSC.dm3")
  library(BSgenome.Dmelanogaster.UCSC.dm3)
}

gmapGenomePath <- file.path(getwd(), "flyGenome")
if (file.exists(gmapGenomePath)) unlink(gmapGenomePath, recursive=TRUE)
ggd <- GmapGenomeDirectory(gmapGenomePath, create = TRUE)
gmapGenome <- GmapGenome(genome=Dmelanogaster,
                         directory = ggd,
```

```
                    name = "dm3",
                    create = TRUE)

makeGmapGenomePackage(gmapGenome=gmapGenome,
                      version="0.1.0",
                      maintainer="<your.name@somewhere.com>",
                      author="Your Name",
                      destDir=".",
                      license="Artistic-2.0",
                      pkgName="GmapGenome.Dmelanogaster.UCSC.dm3")


## End(Not run)
```

TP53Genome *Demo genome around TP53*

Description

Returns a GmapGenome object consisting of the UCSC hg19 sequence centered on the region of the TP53 gene, with 1 Mb flanking sequence on each side. This is intended as a test/demonstration genome and can be used, e.g., in conjunction with the LungCancerLines data package.

Usage

```
TP53Genome()
TP53Which()
```

Value

For TP53Genome, a GmapGenome object. If this is the first time the user has run this function, a side-effect will be the generation of an on-disk genome index, under the name "TP53_demo" in the default genome directory.

For TP53Which, a GRanges of the extents of the TP53 gene, translated to the space of TP53Genome.

Author(s)

Michael Lawrence, Cory Barr

Examples

```
TP53Genome()
```

Index