Rsubread

March 24, 2012

`align` *Align next-gen sequencing reads to reference genome*

Description

This is a R wrapper function for aligning reads. This function calls the underlying C function

Usage

```
align(index,readfile1,readfile2=NULL,output_file,nsubreads=10,TH1=3,TH2=1,nthrea
```

Arguments

`index`	character string giving the basename of index file. Index files should be located in the current directory.
`readfile1`	character string giving the name of file which includes sequencing reads. This will be the name of first file of paired-end data are provided. The read file should be in FASTQ or FASTA format.
`readfile2`	character string giving the name of second file when paired-end read data are provided. `NULL` by default.
`output_file`	character string giving the name of output file which includes the mapping results.
`nsubreads`	numeric value giving the number of subreads extracted from each read. 10 by default.
`TH1`	numeric value giving the consensus threshold for reporting a hit. This threshold will be applied to the first read if paired-end read data are provided. 3 by default.
`TH2`	numeric value giving the consensus threhold for the second read in a pair. 1 by default.
`nthreads`	numeric value giving the number of threads used for mapping. 1 by default.
`indels`	numeric value giving the number of insertions/deletions allowed during the mapping. 5 by default.
`phredOffset`	numeric value added to the phred score of each base in each read. `33` by default.
`tieBreakQS`	logical. If `TRUE`, use quality scores to break ties when more than one best locations were found. `FALSE` by default.

`tieBreakHamming`	logical. If `TRUE`, use Hamming distance to break ties when more than one best locations were found (to use this option, the index must be built with reference sequence included).
`unique`	logical. If `TRUE`, only uniquely mapped reads will be reported (reads mapped to multiple locations in the reference genome will not be reported). This option can be used together with option `tieBreakQS` or `tieBreakHamming`.
`min_distance`	numeric value giving the minimal distance between the pair of reads. 50 by default.
`max_distance`	numeric value giving the maximal distance between the pair of reads. 600 by default.
`PE_orientation`	character string giving the orientation of the two reads in a pair. "fr" by default, which means the first read is on the forward strand and the second read is on the reverse strand.

Details

This function implements a novel mapping strategy which uses a set of 16bp substrings (called subreads) extracted from each read to map them to the reference genome. Different from "seed-and-extend" mapping strategy, this new strategy does not have an extension step therefore it is a lot faster than the competing aligners. Due to the short length of the selected subsreads (16bp long), this strategy has a much high sensitivity than other aligners (seed length is usually around 30bp), i.e. it can align a lot more reads than competing aligners. Our evaluation results (using both simulation dataset and real dataset) showed that the accuracy of the new strategy is comparable to or slightly better than other aligners.

Two key parameters used by this new strategy are the number of subreads selected `nsubreads` and the consensus threshold for determining mapping locations `TH1` (also `TH2` for paired-end read data). We recommend using the default setting of these parameters to map reads of around 100bp long. However, users can choose to use more subreads when mapping longer reads. We recommend to set the value of consensus threshold to be 30 percent of the number of subreads used.

The C implemetation of this strategy can be found at http://subread.sourceforge.net/. This R function calls the corresponding C function to perform the alignment. Therefore, it has the mapping speed as the C program.

`buildindex` function should be called if the index has not been built for the reference genome. The index can be re-used once it has been built.

If paired-end read data is provided, file `readfile1` will assumed to contain the first read from the read pair and `readfile2` the second read.

Value

A file of SAM format which includes the mapping results.

Author(s)

Wei Shi and Yang Liao

References

Yang Liao and Wei Shi, "Subread: a superfast read aligner with high sensitivity and accuracy", In preparation.

Examples

```
# build an index using the sample reference sequence provided in the package
# and save it to the current directory
library(Rsubread)
ref <- system.file("extdata","reference.fa",package="Rsubread")
buildindex(basename="./reference_index",reference=ref)

# align the sample read data provided in this packge to the sample reference
# and save the mapping results to the current directory
reads <- system.file("extdata","reads.txt",package="Rsubread")
align(index="./reference_index",readfile1=reads,output_file="./Rsubread_alignment.SAM")
```

`atgcContent` *Calculate percentages of nucletodies A, T, G and C in a sequencing read datafile*

Description

Calculate percentages of nucletodies A, T, G and C

Usage

```
atgcContent(filename, basewise=FALSE)
```

Arguments

`filename`	character string giving the name of input FASTQ/FASTA file
`basewise`	logical. If `TRUE`, nucleotide percentages will be calculated for each base position in the read across all the reads. By default, percentages are calculated for the entire dataset.

Details

Sequencing reads could contain letter "N" besides "A", "T", "G" and "C". Percentage of "N" in the read dataset is calcuated as well.

The `basewise` calculation is useful for examining the GC bias towards the base position in the read. By default, the percentages of nucleotides in the entire dataset will be reported.

Value

A named vector containing percentages for each nucleotide type if `basewise` is `FALSE`. Otherwise, a data matrix containing nucleotide percentages for each base position of the reads.

Author(s)

Zhiyin Dai and Wei Shi

Examples

```
library(Rsubread)
reads <- system.file("extdata","reads.txt",package="Rsubread")
# Fraction of A,T,G and C in the entire dataset
x <- atgcContent(filename=reads,basewise=FALSE)
# Fraction of A,T,G and C at each base location across all the reads
xb <- atgcContent(filename=reads,basewise=TRUE)
```

`buildindex` *Build index for a reference genome for read mapping*

Description

This is an R wrapper function for building index for a reference genome. This function calls the underlying C function.

Usage

```
buildindex(basename,reference,colorspace=FALSE,memory=3700)
```

Arguments

`basename`	character string giving the basename of created index files.
`reference`	charater string giving the name of the file containing all the refernece sequences.
`colorspace`	logical. If `TRUE`, a color space index will be built. Otherwise, a base space index will be built.
`memory`	numeric value specifying the amount of memory to be requested in gigabytes

Details

A hash table will be built for the reference genome. Keys in the hash table are the 16bp sequences and hash values are their chromosomal locations. A 16bp sequence could have one or more than one chromosomal locations. They are all recorded in the hash table. Non-informative 16bp sequences, which are highly repetitive in the reference genome, are not included in the hash table.

After the index is built, reads can then be mapped to the reference genome by using `align` function.

It takes around 1 hour to build an index for human genome.

Value

Index files with basename provided in `basename`. These files are saved in the current directory.

Author(s)

Wei Shi and Yang Liao

Examples

```
# build an index using the sample reference sequence provided in the package
# and save it to the current directory
library(Rsubread)
ref <- system.file("extdata","reference.fa",package="Rsubread")
buildindex(basename="./reference_index",reference=ref)
```

`detectionCall` *Determine detection p values for each gene in an RNA-seq dataset*

Description

Use GC content adjusted background read counts to determine the detection p values for each gene

Usage

```
detectionCall(dataset, species="hg", plot=FALSE)
```

Arguments

`dataset`	a character string giving the filename of a SAM format file, which is the output of read alignment.
`species`	a character string specifing the species. Options are `hg` and `mm`.
`plot`	logical, indicating whether a density plot of detection p values will be generated.

Value

A data frame which includes detection p values and annotation information for each genes.

Author(s)

Zhiyin Dai and Wei Shi

`detectionCallAnnotation` *Generate annotation data used for calculating detection p values*

Description

This is for internal use only.

Usage

```
detectionCallAnnotation(species="hg", binsize=2000)
```

Arguments

`species`	character string specifying the species to analyase
`binsize`	binsize of integenic region

Value

Two files containing annotation information for exons regions and integenic regions, respectively.

Author(s)

Zhiyin Dai and Wei Shi

`featureCounts` *Count the number of mapped reads for each feature*

Description

Summarize read counts to features including genes and exons

Usage

```
featureCounts(SAMfiles,type="gene",species="mm",annot=NULL)
```

Arguments

`SAMfiles`	a character vector giving names of SAM format files.
`type`	a character string giving the feature type. Its value could be `gene` or `exon`.
`species`	a character string specifying the species. It can be `mm` or `hg`. Values of this argument determines which in-built annotation file will be used, if `annot` is `NULL`.
`annot`	a character string giving the name of the annotation file provided by users, which includes feature information such as chromosomal coordinates etc. This file will override the in-built annotation file chosen from using '`species`' argument.

Details

This function takes as input a set of SAM format files and assigns reads to the features. Currently, only feature types including `gene` and `exon` are supported. `gene` is the aggregation of all the exons for each gene.

There are two in-built annotation files which are used by this function to summarize reads for genes or exons for mouse and human, respectively. These annotation files include the exon annotation information downloaded from NCBI Build 37.2, including Entrez gene identifier and chromosomal coordinates for each exon. The `species` argument specifies which annotation file should be used.

Users can provide their own annotation file for read summarization as well, by using the `annot` argument. In this case, the user provided annotation file will override the in-built annotation file. The annotation file provided by users should be a tab delimited file, and its first four columns should provide gene identifiers, chromosome names, chromosomal start locations and chromosomal end locations for each exon, respectively. Below is an example:

```
entrezid chromosome chr_start chr_stop
497097 chr1 3204563 3207049
497097 chr1 3411783 3411982
497097 chr1 3660633 3661579
100503874 chr1 3637390 3640590
100503874 chr1 3648928 3648985
100038431 chr1 3670236 3671869
...
```

Although this function is designed for summarizing reads from RNA-seq experiments, it can be used to summarize reads from other next-gen sequencing experiments as well, for example ChIP-seq or other DNA sequencing experiments. Simply by setting `type` to `exon` and providing an annotation, this function will yield numbers of mapped reads for each feature.

Value

A list with the following components:

`counts`:	a data matrix containing read counts for each gene or exon.
`annotation`:	a data frame containing Entrez gene identifers, gene/exon length etc.
`targets`:	a character vector giving sample information.

Author(s)

Wei Shi

`processExons` *Obtain chromosomal coordiates of each exon using NCBI annotation*

Description

This is for internal use.

Usage

```
processExons(filename="human_seq_gene.md", species="hg")
```

Arguments

`filename`	a character string giving the name of input .md file (NCBI annotation file)
`species`	a character string specifying the species

Details

The NCBI annotation file gives the chromosomal coordinates of UTR (Untranslated region) and CDS (Coding sequence). This function uses these information to derive the chromosomal coordinates of exons. The first and last exons of genes usually contain both UTR sequence and CDS sequence.

Value

A text file containing chromosomal coordinates of each exon.

Author(s)

Zhiyin Dai and Wei Shi

`propmapped` *Obtain the proportion of mapped reads*

Description

Use mapping information stored in a SAM format file to count the number of mapped reads

Usage

```
propmapped(samfiles)
```

Arguments

`samfiles`	a character vector giving the names of SAM format files.

Details

This function counts of number of mapped reads using the mapping informtion stored in SAM format files.

Value

A data frame containing the total number of reads, number of mapped reads and proportion of mapped reads for each library.

Author(s)

Wei Shi

Examples

```
# build an index using the sample reference sequence provided in the package
# and save it to the current directory
library(Rsubread)
ref <- system.file("extdata","reference.fa",package="Rsubread")
buildindex(basename="./reference_index",reference=ref)

# align the sample read data provided in this packge to the sample reference
# and save the mapping results to the current directory
reads <- system.file("extdata","reads.txt",package="Rsubread")
align(index="./reference_index",readfile1=reads,output_file="./Rsubread_alignment.SAM")

# get the percentage of successfully mapped reads
propmapped("./Rsubread_alignment.SAM")
```

`qualityScores` *Extract quality score infromation from a sequencing read dataset*

Description

Extract quality scores and convert them to ASCII code

Usage

```
qualityScores(filename, offset=64, nreads=10000)
```

Arguments

`filename`	character string giving the name of input FASTQ file.
`offset`	numeric value giving the offset added to the original quality score, 64 by default.
`nreads`	numeric value giving the number of reads from which quality scores are extracted

Details

Quality scores are given in the form of characters in datasets which contain sequencing reads. This function extracts the quality scores and then convert them to the ASCII codes which encode these characters. These ASCII codes are then subtracted by the `offset` to obtain the original quality scores.

If the total number of reads is `n`, then every `n`/`nreads` read will be used for quality score retrieval.

Value

A data matrix containing the quality scores with rows being reads and columns being base positions in the read.

Author(s)

Zhiyin Dai and Wei Shi

Examples

```
library(Rsubread)
reads <- system.file("extdata","reads.txt",package="Rsubread")
x <- qualityScores(filename=reads,nreads=1000)
boxplot(x)
```

`sam2bed` *Convert SAM format file to BED format*

Description

SAM to BED conversion

Usage

```
sam2bed(samfile,bedfile,readlen)
```

Arguments

`samfile`	character string giving the name of input file. Input format should be in SAM format.
`bedfile`	character string giving the name of output file. Output file is in BED format.
`readlen`	numeric value giving the length of reads included in the input file.

Details

SAM format is the de facto standard format of output from read aligner. This format not only includes the mapping coordinates of the reads but also includes other using information such as mapping quality, CIGAR information and so on. This function converts a SAM format file to a BED format file, which can then be displayed in a genome browser like UCSC genome browser, IGB, IGV etc.

Value

A BED format file.

Author(s)

Wei Shi

`subjunc` *Finding exon junctions*

Description

This function uses RNA-seq reads to discover exon junction locations

Usage

```
subjunc(index,samfile,output_file,paired_end=FALSE,nthreads=1,indels=5,min_dista
```

Arguments

`index`	character string giving the basename of index file. Index files should be located in the current directory.
`samfile`	character string giving the name of SAM file generated from read alignment (eg. output from `align` function).
`output_file`	character string giving the name of output file which includes the mapping results.
`paired_end`	logical, indicating whether the data is paired ended or single ended. `FALSE` by default.
`nthreads`	numeric value giving the number of threads used for mapping. 1 by default.
`indels`	numeric value giving the number of insertions/deletions allowed during the mapping. 5 by default.
`min_distance`	numeric value giving the minimal distance between the pair of reads. 50 by default.
`max_distance`	numeric value giving the maximal distance between the pair of reads. 600 by default.
`PE_orientation`	character string giving the orientation of the two reads in a pair. "fr" by default, which means the first read is on the forward strand and the second read is on the reverse strand.

Details

This function takes as input the SAM format file generated from read alignment (eg. output from align() function), and then discover splicing points in the genome using junction reads reported in the SAM file (CIGAR strings include letter 'N'). Donor and receptor sites are used when detecting splicing points. This function can detect junction locations in the reads which are as close as 4bp from the end of the reads. It supports INDEL detection during the junction detection as well.

This function calls the underlying C program to perform the exon junction discovery. The C program can be run on its own as well, which is included in the subread software package (http://subread.sourceforge.net/)

Value

This function outputs two files: one is a BED format file which inlcudes the discovered junction locations and numbers of supporting reads, and a SAM format file which includes the mapping result for each read.

Author(s)

Wei Shi and Yang Liao

Index