

# Package ‘tRNAscanImport’

May 19, 2022

**Title** Importing a tRNAscan-SE result file as GRanges object

**Version** 1.16.0

**Date** 2020-07-18

**Description** The package imports the result of tRNAscan-SE as a GRanges object.

**License** GPL-3 + file LICENSE

**Encoding** UTF-8

**LazyData** false

**Depends** R (>= 3.5), GenomicRanges, tRNA

**Imports** methods, stringr, BiocGenerics, Biostrings, Structstrings,  
S4Vectors, IRanges, XVector, GenomeInfoDb, rtracklayer,  
BSgenome, Rsamtools

**Collate** 'tRNAscanImport.R' 'AllGenerics.R' 'tRNAscanImport-checks.R'  
'tRNAscanImport-genome.R' 'tRNAscanImport-import.R' 'utils.R'

**Suggests** BiocStyle, knitr, rmarkdown, testthat, ggplot2,  
BSgenome.Scerevisiae.UCSC.sacCer3

**RoxygenNote** 7.1.1

**VignetteBuilder** knitr

**biocViews** Software, DataImport, WorkflowStep, Preprocessing,  
Visualization

**BugReports** <https://github.com/FelixErnst/tRNAscanImport/issues>

**URL** <https://github.com/FelixErnst/tRNAscanImport>

**git\_url** <https://git.bioconductor.org/packages/tRNAscanImport>

**git\_branch** RELEASE\_3\_15

**git\_last\_commit** c9f0257

**git\_last\_commit\_date** 2022-04-26

**Date/Publication** 2022-05-19

**Author** Felix G.M. Ernst [aut, cre] (<<https://orcid.org/0000-0001-5064-0928>>)

**Maintainer** Felix G.M. Ernst <[felix.gm.ernst@outlook.com](mailto:felix.gm.ernst@outlook.com)>

## R topics documented:

get.tRNAprecursor	2
import.tRNAscanAsGRanges	3
istRNAscanGRanges	4
tRNAscanImport	5

<b>Index</b>	<b>6</b>
--------------	----------

---

get.tRNAprecursor	<i>Get tRNA precursor sequences</i>
-------------------	-------------------------------------

---

### Description

get.tRNAprecursor retrieves tRNA precursor sequences from genomic sequences. The length of 5'- and 3'-overhangs can be specified individually. The output is checked for validity against the tRNA sequences as reported by tRNAscan.

The chromosome names of tRNAscan input and genome sequences must be compatible.

### Usage

```
get.tRNAprecursor(
  input,
  genome,
  add.5prime = 50L,
  add.3prime = add.5prime,
  trim.intron = FALSE
)
```

### Arguments

input	a compatible GRanges object
genome	a <a href="#">BSgenome</a> object, a <a href="#">DNASTringSet</a> object, a <a href="#">FaFile</a> object or a character vector with a single value referring to a file, which can be coerced to a <a href="#">FaFile</a> object.
add.5prime, add.3prime	the length of overhangs as a single integer value each. (default add.5prime = 50L)
trim.intron	TRUE or FALSE: Should intron sequences be included in the precursor sequences? (default trim.intron = FALSE)

### Value

a [DNASTringSet](#) object, containing the precursor sequences.

**Examples**

```

library(BSgenome.Scerevisiae.UCSC.sacCer3)
file <- system.file("extdata",
                    file = "yeast.tRNAscan",
                    package = "tRNAscanImport")
gr <- tRNAscanImport::import.tRNAscanAsGRanges(file)
genome <- getSeq(BSgenome.Scerevisiae.UCSC.sacCer3)
names(genome) <- c(names(genome)[-17], "chrmt")
get.tRNAPrecursor(gr, genome)
# this produces an error since the seqnames do not match
## Not run:
genome <- BSgenome.Scerevisiae.UCSC.sacCer3
names(genome) <- c(names(genome)[-17], "chrmt")
get.tRNAPrecursor(gr, genome)

## End(Not run)
# ... but it can also be fixed
genome <- BSgenome.Scerevisiae.UCSC.sacCer3
seqnames(genome) <- c(seqnames(genome)[-17], "chrmt")
get.tRNAPrecursor(gr, genome)

```

---

```
import.tRNAscanAsGRanges
```

*Importing a tRNAscan output file as a GRanges object*

---

**Description**

The function `import.tRNAscanAsGRanges` will import a tRNAscan-SE output file and return the information as a GRanges object. The reported intron sequences are spliced from the result by default, but can also be returned as imported.

The function `tRNAscan2GFF` formats the output of `import.tRNAscanAsGRanges` to be GFF3 compliant.

`tRNAscanID` generates a unique tRNA ID, which is like the format used in the SGD annotation

`t*AminoAcidSingleLetter*(*Anticodon*)*ChromosomeIdentifier**optionalNumberIfOnTheSameChromosome*`

Example: `tP(UGG)L` or `tE(UUC)E1`.

**Usage**

```
import.tRNAscanAsGRanges(input, as.GFF3 = FALSE, trim.intron = TRUE)
```

```
tRNAscan2GFF(input)
```

```
tRNAscanID(input)
```

**Arguments**

- `input`                   • `import.tRNAscanAsGRanges`: a tRNAscan-SE input file  
                           • `tRNAscan2GFF`: a compatible GRanges object such as the output of `import.tRNAscanAsGRanges`
- `as.GFF3`               optional logical for `import.tRNAscanAsGRanges`: returns a gff3 compatible GRanges object directly. (default: `as.GFF3 = FALSE`)
- `trim.intron`           optional logical for `import.tRNAscanAsGRanges`: remove intron sequences. This changes the tRNA length reported. To retrieve the original length fo the tRNA gene, use the `width()` function on the GRanges object. (default: `trim.intron = TRUE`)

**Value**

a GRanges object

**References**

Chan, Patricia P., and Todd M. Lowe. 2016. "GtRNAdb 2.0: An Expanded Database of Transfer Rna Genes Identified in Complete and Draft Genomes." *Nucleic Acids Research* 44 (D1): D184–9. doi:10.1093/nar/gkv1309.

Lowe, T. M., and S. R. Eddy. 1997. "tRNAscan-Se: A Program for Improved Detection of Transfer Rna Genes in Genomic Sequence." *Nucleic Acids Research* 25 (5): 955–64.

**Examples**

```
gr <- import.tRNAscanAsGRanges(system.file("extdata",
                                           file = "yeast.tRNAscan",
                                           package = "tRNAscanImport"))
gff <- tRNAscan2GFF(gr)
identical(gff, import.tRNAscanAsGRanges(system.file("extdata",
                                                    file = "yeast.tRNAscan",
                                                    package = "tRNAscanImport"),
                                           as.GFF3 = TRUE))
```

---

`istRNAscanGRanges`           *tRNAscan compatibility check*

---

**Description**

`istRNAscanGRanges` checks whether a GRanges object contains the information expected for a tRNAscan result.

**Usage**

```
istRNAscanGRanges(gr)

## S4 method for signature 'GRanges'
istRNAscanGRanges(gr)
```

**Arguments**

gr                    the GRanges object to test

**Value**

a logical value

**Examples**

```
file <- system.file("extdata",
                    file = "yeast.tRNAscan",
                    package = "tRNAscanImport")
gr <- tRNAscanImport::import.tRNAscanAsGRanges(file)
istRNAscanGRanges(gr)
```

---

tRNAscanImport

*tRNAscanImport: Importing tRNAscan-SE output as GRanges*

---

**Description**

tRNAscan-SE can be used for prediction of tRNA genes in whole genomes based on sequence context and calculated structural features. Many tRNA annotations in genomes contain or are based on information generated by tRNAscan-SE, for example the current SGD reference genome sacCer3 for *Saccharomyces cerevisiae*. However, not all available information from tRNAscan-SE end up in the genome annotation. Among these are for example structural information, additional scores and the information, whether the conserved CCA-end is encoded in the genomic DNA. To work with this complete set of information, the tRNAscan-SE output can be parsed into a more accessible GRanges object using ‘tRNAscanImport’.

**Manual**

Please refer to the tRNAscanImport vignette for an example how to work and use the package: [tRNAscanImport](#)

**Author(s)**

Felix G M Ernst [aut]

**References**

- Chan, Patricia P., and Todd M. Lowe. 2016. “GtRNAdb 2.0: An Expanded Database of Transfer Rna Genes Identified in Complete and Draft Genomes.” *Nucleic Acids Research* 44 (D1): D184–189.. doi:10.1093/nar/gkv1309.
- Lowe, T. M., and S. R. Eddy. 1997. “tRNAscan-Se: A Program for Improved Detection of Transfer Rna Genes in Genomic Sequence.” *Nucleic Acids Research* 25 (5): 955–964.

# Index

BSgenome, [2](#)

DNASTringSet, [2](#)

FaFile, [2](#)

get.tRNAprecursor, [2](#)

import.tRNAscanAsGRanges, [3](#)

istRNAscanGRanges, [4](#)

istRNAscanGRanges, GRanges-method  
(istRNAscanGRanges), [4](#)

tRNAscan2GFF

(import.tRNAscanAsGRanges), [3](#)

tRNAscanID (import.tRNAscanAsGRanges), [3](#)

tRNAscanImport, [5](#)