

# Package ‘bambu’

May 15, 2024

**Type** Package

**Title** Context-Aware Transcript Quantification from Long Read RNA-Seq data

**Version** 3.7.0

**Description** bambu is a R package for multi-sample transcript discovery and quantification using long read RNA-Seq data. You can use bambu after read alignment to obtain expression estimates for known and novel transcripts and genes. The output from bambu can directly be used for visualisation and downstream analysis such as differential gene expression or transcript usage.

**License** GPL-3 + file LICENSE

**Encoding** UTF-8

**ByteCompile** true

**Depends** R(>= 4.1), SummarizedExperiment(>= 1.1.6), S4Vectors(>= 0.22.1), BSgenome, IRanges

**Suggests** AnnotationDbi, Biostrings, rmarkdown, BiocFileCache, ggplot2, ComplexHeatmap, circlize, ggbio, gridExtra, knitr, testthat, BSgenome.Hsapiens.NCBI.GRCh38, TxDb.Hsapiens.UCSC.hg38.knownGene, ExperimentHub (>= 1.15.3), DESeq2, NanoporeRNASeq, purrr, apeglm, utils, DEXSeq

**Enhances** parallel

**biocViews** Alignment, Coverage, DifferentialExpression, FeatureExtraction, GeneExpression, GenomeAnnotation, GenomeAssembly, ImmunoOncology, LongRead, MultipleComparison, Normalization, RNASeq, Regression, Sequencing, Software, Transcription, Transcriptomics

**bugReports** <https://github.com/GoekeLab/bambu/issues>

**URL** <https://github.com/GoekeLab/bambu>

**RoxygenNote** 7.2.1

**LinkingTo** Rcpp, RcppArmadillo

```
Imports BiocGenerics, BiocParallel, data.table, dplyr, tidyr,  
         GenomeInfoDb, GenomicAlignments, GenomicFeatures,  
         GenomicRanges, stats, Rsamtools, methods, Rcpp, xgboost  
  
VignetteBuilder knitr  
  
LazyData true  
  
git_url https://git.bioconductor.org/packages/bambu  
git_branch devel  
git_last_commit 65fb8df  
git_last_commit_date 2024-04-30  
  
Repository Bioconductor 3.20  
Date/Publication 2024-05-15  
  
Author Ying Chen [cre, aut],  
         Andre Sim [aut],  
         Yuk Kei Wan [aut],  
         Jonathan Goeke [aut]  
  
Maintainer Ying Chen <chen_ying@gis.a-star.edu.sg>
```

Contents

bambu . . . . .	2
compareTranscripts . . . . .	6
plotBambu . . . . .	7
prepareAnnotations . . . . .	8
readFromGTF . . . . .	9
trainBambu . . . . .	10
transcriptToGeneExpression . . . . .	11
writeBambuOutput . . . . .	12
writeToGTF . . . . .	13
<b>Index</b>	<b>14</b>

---

bambu	<i>long read isoform reconstruction and quantification</i>
-------	--

---

Description

This function takes bam file of genomic alignments and performs isoform recontruction and gene and transcript expression quantification. It also allows saving of read class files of alignments, extending provided annotations, and quantification based on extended annotations. When multiple samples are provided, extended annotations will be combined across samples to allow comparison.

**Usage**

```

bambu(
  reads,
  annotations = NULL,
  genome = NULL,
  NDR = NULL,
  opt.discovery = NULL,
  opt.em = NULL,
  rcOutDir = NULL,
  discovery = TRUE,
  quant = TRUE,
  stranded = FALSE,
  ncore = 1,
  yieldSize = NULL,
  trackReads = FALSE,
  returnDistTable = FALSE,
  lowMemory = FALSE,
  fusionMode = FALSE,
  verbose = FALSE
)

```

**Arguments**

reads	A string or a vector of strings specifying the paths of bam files for genomic alignments, or a BamFile object or a BamFileList object (see Rsamtools). Alternatively string or a vector of strings specifying the read class files that are saved during previous run of <a href="#">bambu</a> .
annotations	A path to a .gtf file or a TxDb object or a GRangesList object obtained by <a href="#">prepareAnnotations</a> .
genome	A path to a fasta file or a BSGenome object.
NDR	specifying the maximum NDR rate to novel transcript output from detected read classes, defaults to an automatic recommendation
opt.discovery	A list of controlling parameters for isoform reconstruction process: <ul style="list-style-type: none"> <li><b>remove.subsetTx</b> indicating whether filter to remove read classes which are a subset of known transcripts(), defaults to TRUE</li> <li><b>min.readCount</b> specifying minimum read count to consider a read class valid in a sample, defaults to 2</li> <li><b>min.readFractionByGene</b> specifying minimum relative read count per gene, highly expressed genes will have many high read count low relative abundance transcripts that can be filtered, defaults to 0.05</li> <li><b>min.sampleNumber</b> specifying minimum sample number with minimum read count, defaults to 1</li> <li><b>min.exonDistance</b> specifying minum distance to known transcript to be considered valid as new, defaults to 35bp</li> <li><b>min.exonOverlap</b> specifying minimum number of bases shared with annotation to be assigned to the same gene id, defaults to 10bp</li> </ul>

	<p><b>min.primarySecondaryDist</b> specifying the minimum number of distance threshold, defaults to 5bp</p> <p><b>min.primarySecondaryDistStartEnd1</b> specifying the minimum number of distance threshold, used for extending annotation, defaults to 5bp</p> <p><b>min.primarySecondaryDistStartEnd2</b> specifying the minimum number of distance threshold, used for estimating distance to annotation, defaults to 5bp</p> <p><b>min.txScore.multiExon</b> specifying the minimum transcript level threshold for multi-exon transcripts during sample combining, defaults to 0</p> <p><b>min.txScore.singleExon</b> specifying the minimum transcript level threshold for single-exon transcripts during sample combining, defaults to 1</p> <p><b>fitReadClassModel</b> A boolean specifying if Bambu should attempt to train a transcript discovery model for all samples. Defaults to TRUE</p> <p><b>defaultModels</b> A model object obtained by code <a href="#">trainBambu</a> or when returnModel is TRUE</p> <p><b>returnModel</b> A boolean specifying if the trained model is output with the read-class files. Defaults to FALSE</p> <p><b>baselineFDR</b> A number between 0 - 1, specifying the false discovery rate used during NDR recommendation. Defaults to 0.1</p> <p><b>min.readFractionByEqClass</b> indicating the minimum relative read count of a subset transcript compared to all superset transcripts (ie the relative read count within the minimum equivalent class). This filter is applied on the set of annotations across all samples using the total read count, this is not a per-sample filter. Please use with caution. defaults to 0</p> <p><b>prefix</b> specifying prefix for new gene Ids (genePrefix.number), defaults to "Bambu"</p>
opt.em	<p>A list of controlling parameters for quantification algorithm estimation process:</p> <p><b>maxiter</b> specifying maximum number of run iterations, defaults to 10000</p> <p><b>degradationBias</b> correcting for degradation bias, defaults to TRUE</p> <p><b>conv</b> specifying the convergence threshold control, defaults to 0.0001</p> <p><b>minvalue</b> specifying the minvalue for convergence consideration, defaults to 0.00000001</p> <p><b>sig.digit</b> specifying the maximum significant digits of the reported estimates</p>
rcOutDir	A string variable specifying the path to where read class files will be saved.
discovery	A logical variable indicating whether annotations are to be extended. Defaults to TRUE
quant	A logical variable indicating whether quantification will be performed. If false the output type will change. Defaults to TRUE
stranded	A boolean for strandedness, defaults to FALSE.
ncore	specifying number of cores used when parallel processing is used, defaults to 1.
yieldSize	see Rsamtools.
trackReads	When TRUE read names will be tracked and output as metadata in the final output as readToTranscriptMaps detailing. the assignment of reads to transcripts. The output is a list with an entry for each sample.

returnDistTable	When TRUE the calculated distance table between read classes and annotations will be output as metadata as distTables. The output is a list with an entry for each sample.
lowMemory	Read classes will be processed by chromosomes when lowMemory is specified. This option provides an efficient way to process big samples.
fusionMode	A logical variable indicating whether run in fusion mode
verbose	A logical variable indicating whether processing messages will be printed.

## Details

Main function

## Value

bambu will output different results depending on whether *quant* mode is on. By default, *quant* is set to TRUE, so bambu will generate a *SummarizedExperiment* object that contains the transcript expression estimates. Transcript expression estimates can be accessed by *counts()*, including the following variables

**counts** expression estimates

**CPM** sequencing depth normalized estimates

**fullLengthCounts** estimates of read counts mapped as full length reads for each transcript

**uniqueCounts** counts of reads that are uniquely mapped to each transcript

Output annotations that are usually the annotations with/without novel transcripts/genes added, depending on whether *discovery* mode is on can be accessed by *rowRanges()* Transcript to gene map can be accessed by *rowData()*, with *eqClass* that defining equivalent class for each transcript

In the case when *quant* is set to FALSE, i.e., only transcript discovery is performed, bambu will report the *grangeslist* of the extended annotations.

## Examples

```
## =====
test.bam <- system.file("extdata",
  "SGNex_A549_directRNA_replicate5_run1_chr9_1_1000000.bam",
  package = "bambu")
fa.file <- system.file("extdata",
  "Homo_sapiens.GRCh38.dna_sm.primary_assembly_chr9_1_1000000.fa",
  package = "bambu")
gr <- readRDS(system.file("extdata",
  "annotationGranges_txdbGrch38_91_chr9_1_1000000.rds",
  package = "bambu"))
se <- bambu(reads = test.bam, annotations = gr,
  genome = fa.file, discovery = TRUE, quant = TRUE)
```

---

compareTranscripts	<i>compare alternatively-spliced transcripts</i>
--------------------	--

---

## Description

compare alternatively-spliced transcripts

## Usage

```
compareTranscripts(query, subject)
```

## Arguments

query	a GRangesList of transcripts
subject	a GRangesList of transcripts

## Details

This function compares two alternatively-spliced transcripts and returns a tibble object that determines the type of the alternative splicing between the query and the subject transcript. Alternative splicing includes exons skipping, intron retention, alternative 5' exon start site, alternative 3' exon end site, alternative first exon, alternative last exon, alternative transcription start site (TSS), alternative transcription end site (TES), internal first exon, internal last exon, or a mixed combination of them.

## Value

a tibble object with the following columns:

Remark: two exons, one each from query and subject transcript are defined to be equivalent if one of the exons fully covers the another exon.

- `alternativeFirstExon`: FALSE if query transcript and subject transcript have equivalent first exon. TRUE otherwise.
- `alternativeTSS`: +k if the query initiates the transcription k sites earlier than the subject transcript, -k if the query initiates the transcription k sites later than the subject transcript.
- `internalFirstExon.query`: TRUE if the first exon of the query transcript is equivalent to one of the exon of the subject transcript (except the first exon). FALSE otherwise.
- `internalFirstExon.subject`: TRUE if the first exon of the subject transcript is equivalent to one of the exon of the query transcript (except the first exon). FALSE otherwise.
- `alternativeLastExon`: FALSE if query transcript and subject transcript have equivalent last exon. TRUE otherwise.
- `alternativeTES`: +k if the query transcript ends the transcription k sites later than the subject transcript, -k if the query transcript ends the transcription k sites earlier than the subject transcript.

- `internalLastExon.query`: TRUE if the last exon of the query transcript is equivalent to one of the exon in the subject transcript (except the last exon). FALSE otherwise.
- `internalLastExon.subject`: TRUE if the last exon of the subject transcript is equivalent to one of the exon in the query transcript (except the last exon). FALSE otherwise.
- `intronRetention.subject`: k if there is/are k intron(s) in the subject transcript relative to the query transcript.
- `intronRetention.query`: k if there is/are k intron(s) in the query transcript relative to the subject transcript.
- `exonSkipping.query`: k if there is/are k exon(s) in the subject transcript (except the first and last) not in the query transcript.
- `exonSkipping.subject`: k if there is/are k exon(s) in the query transcript (except the first and last) not in the subject transcript.
- `exon5prime (splicing)`: k if there is/are k equivalent exon(s) having different 5' start site (except the 5' start site of first exon).
- `exon3prime (splicing)`: k if there is/are k equivalent exon(s) having different 3' end site (except the 3' end site of the last exon).

### See Also

Value for more details about each of the alternative splicing events.

### Examples

```
query <- readRDS(system.file("extdata",
  "annotateSpliceOverlapByDist_testQuery.rds",
  package = "bambu"))
subject <- readRDS(system.file("extdata",
  "annotateSpliceOverlapByDist_testSubject.rds",
  package = "bambu"))
compareTranscriptsTable <- compareTranscripts(query, subject)

compareTranscriptsTable
```

---

plotBambu

*plot.bambu*

---

### Description

plotSEOutput

**Usage**

```
plotBambu(
  se,
  group.variable = NULL,
  type = c("annotation", "pca", "heatmap"),
  gene_id = NULL,
  transcript_id = NULL
)
```

**Arguments**

<code>se</code>	An summarized experiment object obtained from <a href="#">bambu</a> or <a href="#">transcriptToGeneExpression</a> .
<code>group.variable</code>	Variable for grouping in plot, has to be provided if choosing to plot PCA.
<code>type</code>	plot type variable, a values of annotation for a single gene with heatmap for isoform expressions, pca, or heatmap, see details.
<code>gene_id</code>	specifying the <code>gene_id</code> for plotting gene annotation, either <code>gene_id</code> or <code>transcript_id</code> has to be provided when <code>type = "annotation"</code> .
<code>transcript_id</code>	specifying the <code>transcript_id</code> for plotting transcript annotation, either <code>gene_id</code> or <code>transcript_id</code> has to be provided when <code>type = "annotation"</code>

**Details**

[type](#) indicates the type of plots to be plotted. There are two types of plots can be chosen, PCA or heatmap.

**Value**

A heatmap plot for all samples

**Examples**

```
se <- readRDS(system.file("extdata",
  "seOutputCombined_SGNex_A549_directRNA_replicate5_run1_chr9_1_1000000.rds",
  package = "bambu"))
plotBambu(se, type = "PCA")
```

---

prepareAnnotations	<i>prepare reference annotations for long read RNA-Seq analysis with Bambu</i>
--------------------	--

---

**Description**

prepare reference annotations for long read RNA-Seq analysis with Bambu

**Usage**

```
prepareAnnotations(x)
```



**Arguments**

x                      A path to gtf file or a TxDb object.

**Details**

This function creates a reference annotation object which is used for transcript discovery and quantification in Bambu. `prepareAnnotations` can use a path to a gtf file or a TxDB object as input, and returns a annotation object that stores additional information about transcripts which is used in Bambu. For each transcript, exons are ranked from first to last exon in direction of transcription.

**Value**

A `GRangesList` object with additional details for each exon and transcript that are required by Bambu. Exons are ranked by the `exon_rank` column, corresponding to the rank in direction of transcription (from first to last exon). In addition to exon rank, gene id, transcript id, and the minimum transcript equivalent class is stored as well (a transcript equivalence class of a transcript `x` is the collection of transcripts where their exon junctions contain, in a continuous way, the exon junctions of the transcript `x`). The object is designed to be used by Bambu, and the direct access of the metadata is not recommended.

**See Also**

[`bambu()`] for transcript discovery and quantification from long read RNA-Seq.

**Examples**

```
gtf.file <- system.file("extdata",
  "Homo_sapiens.GRCh38.91_chr9_1_1000000.gtf",
  package = "bambu"
)
annotations <- prepareAnnotations(x = gtf.file)

# run bambu
test.bam <- system.file("extdata",
  "SGNex_A549_directRNA_replicate5_run1_chr9_1_1000000.bam",
  package = "bambu")
fa.file <- system.file("extdata",
  "Homo_sapiens.GRCh38.dna_sm.primary_assembly_chr9_1_1000000.fa",
  package = "bambu")
se <- bambu(reads = test.bam, annotations = annotations,
  genome = fa.file, discovery = TRUE, quant = TRUE)
```

---

readFromGTF

---

*convert a GTF file into a GRangesList*


---

**Description**

Outputs `GRangesList` object from reading a GTF file

**Usage**

```
readFromGTF(file, keep.extra.columns = NULL)
```

**Arguments**

```
file          a .gtf file
keep.extra.columns
               a vector with names of columns to keep from the the attributes in the gtf file.
               For ensembl, this could be keep.extra.columns=c('gene_name','gene_biotype',
               'transcript_biotype', 'transcript_name')
```

**Value**

grlist a GRangesList object, with two columns

**TXNAME** specifying prefix for new gene Ids (genePrefix.number), defaults to empty

**GENEID** indicating whether filter to remove read classes which are a subset of known transcripts(), defaults to TRUE

**Examples**

```
gtf.file <- system.file("extdata",
  "Homo_sapiens.GRCh38.91_chr9_1_1000000.gtf",
  package = "bambu"
)
readFromGTF(gtf.file)
```

---

trainBambu

---

*Function to train a model for use on other data*


---

**Description**

This function train a model for use on other data

**Usage**

```
trainBambu(
  rcFile = NULL,
  min.readCount = 2,
  nrounds = 50,
  NDR.threshold = 0.1,
  verbose = TRUE
)
```

**Arguments**

rcFile	summerized experiment object with read classes/ranges produced from bambu(quant = FALSE, discovery = FALSE) or rcOutdir
min.readCount	the minimum number of reads a read class is required to be have to be used for training
nrounds	xgboost hyperparameter - the number of decision trees in the final mode
NDR.threshold	the effective NDR threshold that bambu will try and match on other samples when using this model
verbose	if additional messages should be output Output - A list containing 6 objects which is passed directly into bambu(opt.discovery=list(defaultModels=trainBambu())) transcriptModelME - the model for multi-exon transcripts transcriptModelSE - the model for single-exon transcripts txScoreBaseline - the txScore used for NDR calibration for multi-exon transcripts txScoreBaselineSE - [DEPRECATED] the txScore used for NDR calibration for single-exon transcripts lmNDR = lmNDR - the linear model of the relationship between txScore and NDR used to calculate the baseline for multi-exon transcripts lmNDR.SE = lmNDR.SE - the linear model of the relationship between txScore and NDR used to calculate the baseline for single-exon transcripts NDR.threshold - the NDR threshold usd to calculate the txScoreBaseline on the lmNDR (baselineFDR)

**Details**

Function to train a model for use on other data

**Value**

It returns a model object to use in [bambu](#)

---

transcriptToGeneExpression  
*transcript to gene expression*

---

**Description**

Reduce transcript expression to gene expression

**Usage**

```
transcriptToGeneExpression(se)
```

**Arguments**

se                    a summarizedExperiment object from [bambu](#)

**Value**

A SummarizedExperiment object

## Examples

```
se <- readRDS(system.file("extdata",
  "seOutput_SGNex_A549_directRNA_replicate5_run1_chr9_1_1000000.rds",
  package = "bambu"
))
transcriptToGeneExpression(se)
```

---

**writeBambuOutput***Write Bambu results to GTF and transcript/gene-count files*

---

## Description

Write Bambu results to GTF and transcript/gene-count files

## Usage

```
writeBambuOutput(se, path, prefix = "")
```

## Arguments

se	a <a href="#">SummarizedExperiment</a> object from <a href="#">bambu</a> .
path	the destination of the output files (gtf, transcript counts, and gene counts)
prefix	the prefix of the output files

## Details

The function will write the output from Bambu to files. The annotations will be written to a .gtf file, transcript counts (total counts, CPM, full-length counts, partial-length counts, and unique counts) and gene counts will be written to .txt files.

## Examples

```
se <- readRDS(system.file("extdata",
  "seOutput_SGNex_A549_directRNA_replicate5_run1_chr9_1_1000000.rds",
  package = "bambu"
))
path <- tempdir()
writeBambuOutput(se, path)
```

---

writeToGTF	<i>write GRangeslist into GTF file</i>
------------	--

---

**Description**

Write annotation GRangesList into a GTF file

**Usage**

```
writeToGTF(annotation, file, geneIDs = NULL)
```

**Arguments**

annotation	a GRangesList object
file	the output gtf file name
geneIDs	an optional dataframe of geneIDs (column 2) with the corresponding transcriptIDs (column 1)

**Value**

gtf a GTF dataframe

**Examples**

```
outputGtfFile <- tempfile()
gr <- readRDS(system.file("extdata",
  "annotationGranges_txdbGrch38_91_chr9_1_1000000.rds",
  package = "bambu"
))
writeToGTF(gr, outputGtfFile)
```

# Index

bambu, [2](#), [3](#), [8](#), [11](#), [12](#)

compareTranscripts, [6](#)

plotBambu, [7](#)

prepareAnnotations, [3](#), [8](#)

readFromGTF, [9](#)

SummarizedExperiment, [12](#)

trainBambu, [4](#), [10](#)

transcriptToGeneExpression, [8](#), [11](#)

type, [8](#)

writeBambuOutput, [12](#)

writeToGTF, [13](#)