# Package 'crisprScore'

April 5, 2026

**Version** 1.15.3

**Date** 2026-04-02

**Title** On-Target and Off-Target Scoring Algorithms for CRISPR gRNAs

**Depends** R (>= 4.1), crisprScoreData (>= 1.1.3)

**Imports** BiocGenerics, Biostrings, IRanges, methods, randomForest, reticulate, stringr, utils, XVector

**Suggests** BiocStyle, knitr, rmarkdown, testthat

**biocViews** CRISPR, FunctionalGenomics, FunctionalPrediction

**Description** Provides R wrappers of several on-target and off-target scoring methods for CRISPR guide RNAs (gRNAs).
The following nucleases are supported: SpCas9, AsCas12a, enAsCas12a, and Rfx-Cas13d (CasRx).
The available on-target cutting efficiency scoring methods are RuleSet1, RuleSet3, DeepHF, enPAM+GB, and CRISPRscan. Both the CFD and MIT scoring methods are available for off-target
specificity prediction. The package also provides a Lindel-derived score to predict the probability of a gRNA to produce indels inducing a frameshift for the Cas9 nuclease.
Note that DeepHF and enPAM+GB are not available on Windows machines.

**License** MIT + file LICENSE

**Encoding** UTF-8

**RoxygenNote** 7.3.3

**VignetteBuilder** knitr

**StagedInstall** no

**BugReports** https://github.com/crisprVerse/crisprScore

**URL** https://github.com/crisprVerse/crisprScore/issues

**LazyData** true

**git_url** https://git.bioconductor.org/packages/crisprScore

**git_branch** devel

**git_last_commit** cbd6f9f

**git_last_commit_date** 2026-04-02

**Repository** Bioconductor 3.23

**Date/Publication** 2026-04-05

**Author** Jean-Philippe Fortin [aut, cre, cph],
Aaron Lun [aut],
Luke Hoberecht [ctb],
Pirunthan Perampalam [ctb]

**Maintainer** Jean-Philippe Fortin <fortin946@gmail.com>

# Contents



---

`getCasRxRFScores` *Calculate on-target sgRNA activity scores for CasRx using CasRx-RF*

---

## Description

Calculate on-target sgRNA activity scores for CasRx (RfxCas13d) using the CasRx-RF algorithm.

## Usage

```
getCasRxRFScores(
  mrnaSequence,
  directRepeat = "aacccctaccaactggtcggggtttgaaac",
  binaries = NULL,
  sort = FALSE,
  verbose = TRUE
)
```

## Arguments

| | |
|---|---|
| mrnaSequence | A DNAStringSet represeting the mRNA sequence for which to extract spacer sequences and calculate scores. |
| directRepeat | String specifying the direct repeat used in the CasRx construct. |
| binaries | Named list of paths for binaries needed for CasRx-RF. Names of the list must be "RNAfold", "RNAhybrid", and "RNAplfold". Each list element is a string specifying the path of the binary. If NULL (default), binaries must be available on the PATH. |
| sort | Should spacers be sorted by score? FALSE by default. |
| verbose | Should messages be printed to console? TRUE by default. |

## Details

The function first extracts all 23mer spacer sequences targeting the mRNA sequence, and scores them for on-target activity.

## Value

A data.frame with the following columns:

- ID Character vector specifying spacer ID.
- spacer 23-mer spacer sequence.
- pfs_site coordinate of the protospacer flanking sequence (PFS).
- protospacer 23-mer protospacer sequence (reverse complement of the spacer sequence).
- PFS PFS nucleotide.
- score Raw score (not standardized).
- standardizedScore Score standardized between 0 and 1.
- quartile Quartile score (1 to 4, with 4 being the best quartile.

A scores closer to 1 indicates higher predicted on-target activity.

## Author(s)

Jean-Philippe Fortin

## References

Wessels HH, Méndez-Mancilla A, Guo X, et al. Massively parallel Cas13 screens reveal principles for guide RNA design. Nat biotechnol. 2020 Jun;38(6):722-7.

## Examples

```
if (interactive()){
    fasta <- file.path(system.file(package="crisprScore"),
        "casrxrf/test.fa")
    mrnaSequence <- Biostrings::readDNAStringSet(filepath=fasta,
        format="fasta",
        use.names=TRUE)
    results <- getCasRxRFScores(mrnaSequence)
}
```

---

`getCFDScores` *Calculate CFD off-target specificity scores*

---

## Description

Calculate cutting frequency determination (CFD) off-target specificity scores for CRISPR/Cas9 or CRISPR/CasRX.

## Usage

```
getCFDScores(spacers, protospacers, pams, nuclease = c("SpCas9", "CasRx"))
```

## Arguments

| | |
|---|---|
| `spacers` | Character vector of 20bp spacer sequences. Must be in 5' to 3' direction. For SpCas9, must be of length 20bp. For CasRx, must be at most of length 27bp. |
| `protospacers` | Character vector of 20bp protospacer sequences (target sequences). Must be in 5' to 3' direction. |
| `pams` | Character vector of PAM sequences. |
| `nuclease` | String specifying the nuclease. Either "SpCas9" (default) or "CasRx". |

## Value

**getCFDScores** returns a data.frame with `spacer`, `protospacer`, and `score` columns. The CFD score takes on a value between 0 and 1. For a given pair (on-target, off-target), a higher CFD score indicates a higher likelihood for the nuclease to cut at the off-target. Non-canonical PAM sequences are taken into account by the CFD algorithm.

## Author(s)

Jean-Philippe Fortin

## References

Doench, J., Fusi, N., Sullender, M. et al. Optimized sgRNA design to maximize activity and minimize off-target effects of CRISPR-Cas9. Nat Biotechnol 34, 184–191 (2016). https://doi.org/10.1038/nbt.3437.

## Examples

```
# Calculating MIT scores for two off-targets with respect to
# one spacer sequence:
spacer <- "AGGTGTAGTGTGTGTGATAA"
protospacer1 <- "CGGTGTAGTGTGTGTGATAA"
protospacer2 <- "CGGTGTCGTGTGTGTGATAA"
results <- getCFDScores(spacers=spacer,
    protospacers=c(protospacer1, protospacer2),
    pams=c("AGG", "CGG")
)
```

---

# getCRISPRaterScores *Calculate on-target sgRNA activity scores for Cas9 using CRISPRater*

---

## Description

Calculate on-target sgRNA activity scores for CRISPR/Cas9-induced knockout using the DeepHF scoring method. Both U6 and T7 promoters are supported. Three different versions of the SpCas9 nuclease are supported: wildtype (WT-SpCas9), high-fidelity Cas9 (SpCas9-HF1) and enhanced Cas9 (eSpCas9). Currently not supported on Windows machines.

## Usage

```
getCRISPRaterScores(sequences)
```

## Arguments

| | |
|---|---|
| sequences | Character vector of 20bp protospacer sequences. |

## Details

Input sequences for CRISPRater scoring must be 20 spacer sequences.

## Value

**getCrisprRaterScores** returns a data.frame with `sequence` and `score` columns. The CRISPRater score takes on a value between 0 and 1. A higher score indicates higher knockout efficiency.

## Author(s)

Jean-Philippe Fortin

## References

Labuhn M, Adams FF, Ng M, et al. Refined sgRNA efficacy prediction improves large-and small-scale CRISPR–Cas9 applications. Nucleic acids research. 2018 Feb 16;46(3):1375-85.

## Examples

```
spacer  <- "ATCGATGCTGATGCTAGATA" #20bp
results <- getCRISPRaterScores(spacer)
```

---

`getCRISPRscanScores` *Calculate on-target sgRNA activity scores for Cas9 using CRISPRscan*

---

## Description

Calculate on-target sgRNA activity scores for CRISPR/Cas9-induced knockout using the CRISPRscan scoring method. The method is also know as the Moreno-Mateos score. The CRISPRscan algorithm was trained using in vitro transcription of sgRNAs using a T7 promoter, and might therefore be suboptimal to predict sgRNA activity when expressed from U6 promoter.

## Usage

```
getCRISPRscanScores(sequences)
```

## Arguments

| | |
|---|---|
| `sequences` | Character vector of 35bp sequences needed for CRISPRscan scoring, see details below. |

## Details

The input sequences for Rule Set 1 scoring require 6 nucleotides upstream of the protospacer sequence, the protospacer sequence itself (23 nucleotides) and 6 nucleootides downstream of the protospacer sequence, for a total of 35 nucleotides: [6nt][20nt-spacer][NGG][6nt]. Note that a canonical PAM sequence (NGG) is required for CRISPRscan.

## Value

**getCRISPRscanScores** returns a data.frame with `sequence` and `score` columns. The CRISPRscan score takes on a value between 0 and 1. A higher score indicates higher knockout efficiency.

## Author(s)

Jean-Philippe Fortin

## References

Moreno-Mateos MA, et al. CRISPRscan: designing highly efficient sgRNAs for CRISPR-Cas9 targeting in vivo. Nature methods. 2015 Oct;12(10):982-8.

## Examples

```
flank5 <- "ACCTGG" #6bp
spacer <- "ATCGATGCTGATGCTAGATA" #20bp
pam    <- "AGG" #3bp
flank3 <- "TTGAGC" #6bp
input  <- paste0(flank5, spacer, pam, flank3)
results <- getCRISPRscanScores(input)
```

---

`getDeepHFScores` *Calculate on-target sgRNA activity scores for Cas9 using DeepHF*

---

## Description

Calculate on-target sgRNA activity scores for CRISPR/Cas9-induced knockout using the DeepHF scoring method. Both U6 and T7 promoters are supported. Three different versions of the SpCas9 nuclease are supported: wildtype (WT-SpCas9), high-fidelity Cas9 (SpCas9-HF1) and enhanced Cas9 (eSpCas9). Currently not supported on Windows machines.

## Usage

```
getDeepHFScores(
  sequences,
  enzyme = c("WT", "ESP", "HF"),
  promoter = c("U6", "T7"),
  condaEnv
)
```

## Arguments

| | |
|---|---|
| `sequences` | Character vector of 23bp protospacer sequences. |
| `enzyme` | Character string specifying the Cas9 variant. Wildtype Cas9 (WT) by default, see details below. |
| `promoter` | Character string speciyfing promoter used for expressing sgRNAs for wildtype Cas9 (must be either "U6" or "T7"). "U6" by default. |
| `condaEnv` | String specifying the path of the conda environment needed to run the DeepHF calculations. See the crisprScore vignette for instructions on how to build the environment. |

## Details

Input sequences for DeepHF scoring must be 23bpprotospacer sequences (20bp spacer sequences + 3bp PAM sequences). Only canonical PAM sequences (NGG) are allowed. Users can specify for which Cas9 they wish to score sgRNAs by using the argument `enzyme`: "WT" for Wildtype Cas9 (WT-SpCas9), "HF" for high-fidelity Cas9 (SpCas9-HF), or "ESP" for enhanced Cas9 (eSpCas9). For wildtype Cas9, users can also specify the promoter used for expressing sgRNAs using the argument `promoter` ("U6" by default).

## Value

**getDeepHFScores** returns a data.frame with `sequence` and `score` columns. The DeepHF score takes on a value between 0 and 1. A higher score indicates higher knockout efficiency.

## Author(s)

Jean-Philippe Fortin

## References

Wang, D., Zhang, C., Wang, B. et al. Optimized CRISPR guide RNA design for two high-fidelity Cas9 variants by deep learning. Nat Commun 10, 4284 (2019). https://doi.org/10.1038/s41467-019-12281-8

## Examples

```
if (interactive()){
spacer  <- "ATCGATGCTGATGCTAGATA" #20bp
pam     <- "AGG" #3bp
input   <- paste0(spacer, pam)
condaEnv <- "/Users/fortin946/miniforge3/envs/deephf-env"

# Wiltype Cas9 using U6 promoter:
results <- getDeepHFScores(input, condaEnv=condaEnv)

# Wiltype Cas9 using T7 promoter:
results <- getDeepHFScores(input, promoter="T7", condaEnv=condaEnv)

#' High-fidelity Cas9:
results <- getDeepHFScores(input, enzyme="HF",condaEnv=condaEnv)

#' Enhanced Cas9:
results <- getDeepHFScores(input, enzyme="ESP",condaEnv=condaEnv)
}
```

---

`getEnPAMGBScores` *Calculate on-target sgRNA activity scores for enCas12a using enPAM+GB*

---

## Description

Calculate on-target sgRNA activity scores for CRISPR/Cas12a-induced knockout using the enPAM+GB scoring method. Currently not supported on Windows machines.

## Usage

```
getEnPAMGBScores(sequences, condaEnv)
```

### Arguments

| | |
|---|---|
| sequences | Character vector of 34bp sequences needed for enPAM+GB scoring, see details below. |
| condaEnv | String specifying the path of the conda environment needed to run the enPAM+GB calculations. See the crisprScore vignette for instructions on how to build the environment. |

### Details

The input sequences for enPAM+GB scoring require 4 nucleotides upstream of the protospacer sequence, the protospacer sequence itself (4bp PAM sequence + 23bp spacer sequence) and 3 nucleotides downstream of the protospacer sequence, for a total of 34 nucleotides. Both canonical and non-canonical PAM sequences can be provided.

### Value

**getEnPAMGBScores** returns a data.frame with `sequence` and `score` columns.

### Author(s)

Jean-Philippe Fortin

### References

DeWeirdt, P.C., Sanson, K.R., Sangree, A.K. et al. Optimization of AsCas12a for combinatorial genetic screens in human cells. Nat Biotechnol 39, 94–104 (2021). https://doi.org/10.1038/s41587-020-0600-6.

### Examples

```
if (interactive()){
flank5 <- "CATG" #4bp
pam    <- "TTTT" #4bp
spacer <- "TTTGGGAACCAATCGATAATCAC" #23bp
flank3 <- "ATT" #3bp
input  <- paste0(flank5, pam, spacer, flank3)
condaEnv <- "/Users/fortin946/miniforge3/envs/enpamgb-env"
results <- getEnPAMGBScores(input, condaEnv)
}
```

---

## getLindelScores — *Predict frameshift ratios from CRISPR/Cas9 indel prediction using Lindel*

---

### Description

Predict frameshift ratios from CRISPR/Cas9 indel prediction using the Lindel prediction algorithm.

## Usage

```
getLindelScores(sequences, condaEnv)
```

## Arguments

| | |
|---|---|
| sequences | Character vector of 65bp sequences needed for Lindel scoring, see details below. |
| condaEnv | String specifying the path of the conda environment needed to run the DeepHF calculations. See the crisprScore vignette for instructions on how to build the environment. |

## Details

The input sequences for Lindel scoring require 13 nucleotides upstream of the protospacer sequence, the protospacer sequence itself (23 nucleotides) and 29 nucleootides downstream of the protospacer sequence, for a total of 65 nucleotides. Note that only canonical PAM sequences (NGG) are accepted by Lindel.

## Value

A data.frame with predicted frameshift ratio (between 0 and 1). A higher ratio indicates a greater chance of a frameshift indel introduced by CRISPR/Cas9-induced double-strand breaks.

## Author(s)

Jean-Philippe Fortin

## References

Wei Chen, Aaron McKenna, Jacob Schreiber, Maximilian Haeussler, Yi Yin, Vikram Agarwal, William Stafford Noble, Jay Shendure, Massively parallel profiling and predictive modeling of the outcomes of CRISPR/Cas9-mediated double-strand break repair, Nucleic Acids Research, Volume 47, Issue 15, 05 September 2019, Pages 7989–8003, https://doi.org/10.1093/nar/gkz487.

## Examples

```
if (interactive()){
flank5 <- "ACCTTTTAATCGA" #13bp
spacer <- "TGCTGATGCTAGATATTAAG" #20bp
pam    <- "TGG" #3bp
flank3 <- "CTTTTAATCGATGCTGATGCTAGATATTA" #29bp
input <- paste0(flank5, spacer, pam, flank3)
condaEnv <- "/Users/fortin946/miniforge3/envs/deephf-env"
results <- getLindelScores(input, condaEnv=condaEnv)
}
```

getMITScores *Calculate MIT off-target specificity scores for CRISPR/Cas9*

## Description

Calculate MIT off-target specificity scores for CRISPR/Cas9.

## Usage

```
getMITScores(spacers, protospacers, pams, includeDistance = TRUE)
```

## Arguments

| | |
|---|---|
| spacers | Character vector of 20bp spacer sequences. |
| protospacers | Character vector of 20bp protospacer sequences for off-targets. |
| pams | Character vector of 3nt PAM sequences. |
| includeDistance | Should distance between mismatches be considered during scoring? TRUE by default. |

## Value

**getMITScores** returns a data.frame with `spacer`, `protospacer`, and `score` columns. The MIT score takes on a value between 0 and 1. For a given pair (on-target, off-target), a higher MIT score indicates a higher likelihood for the Cas9 nuclease to cut at the off-target. Non-canonical PAM sequences are taken into account by the MIT algorithm.

## Author(s)

Jean-Philippe Fortin

## References

Hsu, P., Scott, D., Weinstein, J. et al. DNA targeting specificity of RNA-guided Cas9 nucleases. Nat Biotechnol 31, 827–832 (2013). https://doi.org/10.1038/nbt.2647.

## Examples

```
# Calculating MIT scores for two off-targets with respect to
# one spacer sequence:
spacer <- "AGGTGTAGTGTGTGTGATAA"
protospacer1 <- "CGGTGTAGTGTGTGTGATAA"
protospacer2 <- "CGGTGTCGTGTGTGTGATAA"
pams <- c("AGG", "CGG")
results <- getMITScores(spacers=spacer,
    protospacers=c(protospacer1, protospacer2),
    pams=pams
)
```

`getRuleSet1Scores` *Calculate on-target sgRNA activity scores for Cas9 using Rule Set 1*

## Description

Calculate on-target sgRNA activity scores for CRISPR/Cas9-induced knockout using the Rule Set 1 scoring method. The Rule Set 1 algorithm was an early on-target efficiency method developed by the Doench lab.

## Usage

```
getRuleSet1Scores(sequences)
```

## Arguments

| | |
|---|---|
| `sequences` | Character vector of 30bp sequences needed for Rule Set 1 scoring, see details below. |

## Details

The input sequences for Rule Set 1 scoring require 4 nucleotides upstream of the protospacer sequence, the protospacer sequence itself (23 nucleotides) and 3 nucleootides downstream of the protospacer sequence, for a total of 30 nucleotides: [4nt][20nt-spacer][NGG][3nt]. Note that a canonical PAM sequence (NGG) is required for Rule Set 1.

## Value

**getRuleSet1Scores** returns a data.frame with `sequence` and `score` columns. The Rule Set 1 score takes on a value between 0 and 1. A higher score indicates higher knockout efficiency.

## Author(s)

Jean-Philippe Fortin

## References

Doench, John G., et al. Rational design of highly active sgRNAs for CRISPR-Cas9–mediated gene inactivation. Nat Biotech 32, 1262-1267 (2014). https://doi.org/10.1038/nbt.3026.

## Examples

```
flank5 <- "ACCT" #4bp
spacer <- "ATCGATGCTGATGCTAGATA" #20bp
pam    <- "AGG" #3bp
flank3 <- "TTG" #3bp
input  <- paste0(flank5, spacer, pam, flank3)
results <- getRuleSet1Scores(input)
```

`getRuleSet3Scores` *Calculate on-target sgRNA activity scores for SpCas9 using Rule Set 3*

## Description

Calculate on-target sgRNA activity scores for CRISPR/Cas9-induced knockout using the Rule Set 3 scoring method.

## Usage

```
getRuleSet3Scores(
  sequences,
  tracrRNA = c("Hsu2013", "Chen2013"),
  mode = c("sequence", "target"),
  condaEnv
)
```

## Arguments

| | |
|---|---|
| `sequences` | Character vector of 30bp sequences needed for Rule Set 3 scoring, see details below. |
| `tracrRNA` | String specifying which tracrRNA is used. Must be either "Hsu2013" (default) or "Chen2013". |
| `mode` | String specifying which prediction mode is used. Must be either "sequence" (default) or "target". |
| `condaEnv` | String specifying the path of the conda environment needed to run the DeepHF calculations. See the crisprScore vignette for instructions on how to build the environment. |

## Details

The input sequences for Rule Set 3 scoring require 4 nucleotides upstream of the protospacer sequence, the protospacer sequence itself (20bp spacer sequence + 3bp PAM sequence ) and 3 nucleootides downstream of the protospacer sequence, for a total of 30 nucleotides.

## Value

**getRuleSet3Scores** returns a data.frame with `sequence` and `score` columns. The getRuleSet3Scores score is similar to a Z-score.A higher score indicates higher knockout efficiency.

## Author(s)

Jean-Philippe Fortin

## References

doi: https://doi.org/10.1101/2022.06.27.497780

## Examples

```
if (interactive()){
flank5 <- "ACCG" #4bp
spacer <- "AATCGATGCTGATGCTAGAT" #20bp
pam    <- "AGG" #3bp
flank3 <- "AAT" #3bp
input  <- paste0(flank5, spacer, pam, flank3)
condaEnv <- "/Users/fortin946/miniforge3/envs/rs3-env"
results <- getRuleSet3Scores(input, condaEnv=condaEnv, tracrRNA="Chen2013")
}
```

---

# scoringMethodsInfo *data.frame detailing available scoring methods*

---

## Description

data.frame detailing available scoring methods with information needed to extract nucleotide sequences needed by each scoring algorithm.

## Usage

```
data(scoringMethodsInfo)
```

## Format

A data frame with 6 columns:

**method** name of the scoring method

**nuclease** nuclease compatible with the scoring method

**left** upstream offset (relative to PAM site) to extract nucleotide sequence needed for scoring

**right** downstream offset (relative to PAM site) to extract nucleotide sequence needed for scoring

**type** type of the scoring algorithm (on-target or off-target)

**label** proper case-sensitive method name for labeling

**len** length of the nucleotide sequence needed for scoring

# Index