Package ‘ModCon’

March 12, 2024

Type Package

Title Modifying splice site usage by changing the mRNP code, while maintaining the genetic code

Version 1.10.0

Description Collection of functions to calculate a nucleotide sequence surrounding for splice donors sites to either activate or repress donor usage. The proposed alternative nucleotide sequence encodes the same amino acid and could be applied e.g. in reporter systems to silence or activate cryptic splice donor sites.

License GPL-3 + file LICENSE

Encoding UTF-8

LazyData true

VignetteBuilder knitr

Depends data.table, parallel, utils, stats, R (>= 4.1)

Suggests testthat, knitr, rmarkdown, dplyr, shinycssloaders, shiny, shinyFiles, shinydashboard, shinyjs

SystemRequirements Perl

biocViews FunctionalGenomics, AlternativeSplicing

URL https://github.com/caggtaagtat/ModCon
git_url https://git.bioconductor.org/packages/ModCon
git_branch RELEASE_3_18
git_last_commit 05c979f
git_last_commit_date 2023-10-24

Repository Bioconductor 3.18

Date/Publication 2024-03-11

Author Johannes Ptok [aut, cre] (https://orcid.org/0000-0002-0322-5649)

Maintainer Johannes Ptok <Johannes.Ptok@posteo.de>
**calculateHZEIint**  

**Calculate HZ EI integral of nucleotide sequence**

**Description**

This function calculates the HZ EI integral of a nucleotide sequence.

**Usage**

`calculateHZEIint(ntSequence)`

**Arguments**

- `ntSequence`  
  Character value of nucleotide sequence whose HZ EI integral will be calculated. It should be at least 11 nt long and only contain bases 'A', 'G', 'C', 'T'.

**Value**

Integer value stating the HZ EI integral of the given sequence `ntSequence`
Examples

```r
## Example to increase HZEI integral for a given coding sequence
x <- calculateHZEIint('ATACCAGCCAGCTATTACATTT')
```

### calculateMaxEntScanScore

`Calculate MaxEntScan score of a splice site sequence`

**Description**

This function calculates the MaxEntScan score of either splice donor (SD) or acceptor sequences (SA).

**Usage**

```r
calculateMaxEntScanScore(seqVector, ssType)
```

**Arguments**

- `seqVector` Character value of nucleotide sequence of a splice site sequence. SA sequences should be 23nt long (20 intronic, 3 exonic) and SD sequences should be 9nt long (3 exonic, 6 intronic). Only bases 'A', 'G', 'C', 'T' permitted.
- `ssType` Numeric value which indicates the type of splice site. Either '3' for an SA or '5' for an SD.

**Value**

Numeric vector stating the MaxEntScan score per splice site sequence entered with `seqVector`

**Examples**

```r
calculateMaxEntScanScore('TTCCAAACGAACCTTTTGTAGGGA',3)
calculateMaxEntScanScore('GAGGTAAGT',5)
```
**cds**

*CDs of firefly luciferase*

**Description**

Character string of the nucleotide sequence encoding the firefly luciferase.

**Usage**

`cds`

**Format**

character string

**Examples**

`cds`

---

**changeSequenceHZEI**

*Adjust HZEI integral of nucleotide sequence*

**Description**

Adjust the HZEI integral of a nucleotide sequence (min. 24nt long)

**Usage**

`changeSequenceHZEI(inSeq, increaseHZEI=TRUE, nGenerations=50, parentSize=300, startParentSize=1000, bestRate=50, semiLuckyRate=20, luckyRate=5, mutationRate=1e-04, optiRate=100, sdMaximalHBS=10, acMaximalMaxent=4, nCores=-1)`

**Arguments**

- `inSeq` Character value of nucleotide sequence (min 24nt long, only bases A, G, T or C)
- `increaseHZEI` Logical value if HZEI integral should be increased or decreased during SD degradation. If TRUE, function aims to increase HZEI integral.
- `nGenerations` Numeric value setting maximal number of generations
- `parentSize` Numeric value setting size of parent generations, generated from previous generations
- `startParentSize` Numeric value setting size of initiated parent generation of sequences
- `bestRate` Numeric value setting percentage how many of the fittest sequences are used to produce the next generation
**Codons**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>semiLuckyRate</td>
<td>Numeric value setting percentage of sequences which are selected for breeding with a probability based on the respective HZEI-score integral</td>
</tr>
<tr>
<td>luckyRate</td>
<td>Numeric value setting percentage of sequences which are randomly selected for breeding</td>
</tr>
<tr>
<td>mutationRate</td>
<td>Numeric value setting chance of each codon, to mutate randomly within a child sequence</td>
</tr>
<tr>
<td>optiRate</td>
<td>Numeric value setting level of HZEI integral optimization</td>
</tr>
<tr>
<td>sdMaximalHBS</td>
<td>Numeric value of minimal HBS of SDs which should be tried to be degraded in their intrinsic strength</td>
</tr>
<tr>
<td>acMaximalMaxent</td>
<td>Numeric value of minimal MaxEntScan score of SAs which should be tried to be degraded in their intrinsic strength</td>
</tr>
<tr>
<td>nCores</td>
<td>Numeric value setting number of cores which should be used for parallel computations. If set to <code>-1</code> all available cores are selected.</td>
</tr>
</tbody>
</table>

**Value**

Character value of a nucleotide sequence encoding the same amino acid sequence as `inSeq`, but an increased HZEI integral, due to alternative codon selection.

**Examples**

```r
## Load R packages
library('parallel')
library('utils')
library('data.table')

## Set parameters for genetic algorithm
inSeq <- 'ATGGAAGACGCCAAAAACATAAAGAAAGGCCCGGCGCCATTCTATCCGCTGGAAGATGGAACC'

## Increase HZEI integral
res <- changeSequenceHZEI(inSeq)

## Setting additional parameters
res <- changeSequenceHZEI(inSeq, increaseHZEI=TRUE, nGenerations=50, parentSize=300, startParentSize=1000, bestRate=50, semiLuckyRate=20, luckyRate=5, mutationRate=1e-04, optiRate=100, sdMaximalHBS=10, acMaximalMaxent=4, nCores=1)

## Access sequence with highest generated HZEI integral
res[[3]]
```

**Codons**

<table>
<thead>
<tr>
<th>Codons</th>
<th>Table of codons and encoded amino acids</th>
</tr>
</thead>
</table>

**Description**

Table of codons and encoded amino acids
Usage

Codons

Format

A data frame with columns:

- `ndiff`  Indicator, how many codons encode the same amino acid
- `AA` Amino acid three-letter code
- `name` Amino acid full name
- `seq` Codon sequence

Examples

Codons

```
createCodonMatrix

Create codon matrix from coding nucleotide sequence
```

Description

This function creates a codon matrix with 2 rows and as many columns as codons within the sequence.

Usage

`createCodonMatrix(cds)`

Arguments

- `cds` Character value of nucleotide sequence whose HZEI integral will be calculated. It should be at least 3 nt long and only contain bases ‘A’, ‘G’, ‘C’, ‘T’. Length must be a multiple of 3.

Value

Character matrix holding the encoded codon sequence in both rows.

Examples

```
## Example to create codon matrix
createCodonMatrix("ATGAATGATCAAAAGCTAGCC")
```
createFilialSequencePopulation

Generate new sequences by recombination

Description

This function generates new sequences from set of parental sequences through recombination.

Usage

createFilialSequencePopulation(sequenceVector, generateNrecombinedSequences)

Arguments

sequenceVector Character vector of nucleotide sequences which will be used to create new sequences through recombination.
generateNrecombinedSequences Numeric value setting number of recombined sequences which will be generated

Value

Character vector of nucleotide sequences, generated by recombination from the entered sequenceVector, holding as much filial sequences as stated in generateNrecombinedSequences. Modes of recombination are cross-over, insertion and random.

Examples

createFilialSequencePopulation(c('AAABBBCCCDDEEFFFF','GGHHHIIIJJJKKKLLL'), 3)

decreaseGTsiteStrength

Remove or degrade intrinsic strength of specific GT site while keeping the HZEI integral neutral.

Description

Degrade or remove specific GT site from a coding sequence by codon selection keeping the HZEI integral near zero.

Usage

decreaseGTsiteStrength(cds, sdSeqStartPosition)
Arguments

cds
Character value of a coding nucleotide sequence which holds the splice site of interest. Sequence length must be devidable by 3 and only contain bases 'A', 'G', 'C', 'T'.

sdSeqStartPosition
Numeric value of position of the first nucleotide of the splice donor of interest

Value

Character vector of a nucleotide sequence encoding the same amino acid as the entered cds, but the intrinsic strength of a specific GT site within the CDS is degraded as much as possible.

Examples

```r
library(data.table)
cds <- paste0('ATGGAAGACGCCAAAACATATAAGAGAAGGCCCGGCCTTCTATCCGCTGGAAGATGGAACCGCTGGAGAGCAACTGCA',
'TAAGGCTATGAAAGATACGCCCTGTCCCTGGAAACATGCTTATTTACAGATCAATCGAGGCTAGTACCTTACGCTGAGTACTTGGAAA',
'TTGCCGTCTGGTGCGAGAAAGCTTATAGCAAAATACGAGATCTGCGATGAAAGAGCTCCGCAACTTATATATCAAGTCTTGTCTACTTATT',
'GCCGGTGTTGGGGCCGTATTTATCTGGAGTTGCAATGTCGGCCGGGGAAGAGCATTATTAAATAGAACATGCTGACGGGCTTCTTATTCC',
'CAGCCCTACGGTGTTGCTTCTTGCAAAAGGGGTTGCAAAATATATGAAAGAGGCAATTTTTATTATATGGGAGACGGCCTTCTTCA',
'ATTCTAAAACGGAATACCGGGAATTTCTGCTGACGACTTCTCACTTACCCGGGTTTTATATGGAATACGTGTGGAGCCAGA',
'GTCTTTGTCGATGAGGTACGTGTCATTACCTTACTCAGGCTCTTCACTATCTACTTTTGTGCCATGAGTACGCTTCTTCTACTACGCTA',
'TTGGAAATTCTTTACTACACTCAGGAAATGCTGGTCTGCTGTGCAACATTCTTACTTCTTCTGCAAAAGGTCACTGTAATCAGGTATTAATATTATA',
'CAGGAAATTTGCTTGTGCGCCCTCTCTCTAAAGGAAGGCGCAGCTGCAGATGAGGTTCTTCAGTTGCTTCTTCTACTCAGGCAAGGATATG',
'GGTTGTGGCTTGAGCTGACACATCTGCTGTATTTTGTTCAGTTGCTTCTTCTACTCAGGCAAGGATATG',
'ATCGTGATTACGCGAGTCAAAGAACACCCGAAAAATAGGGCAGGAGAAGTTGCTTTTGTGGAGAGCTCCGCAACTTATGAAACACCCCAATCC',
'ATCTGCGAGATCACGAGGCGACTGCACAGTCAAACATGCTGGTCTGCTTCTTCTACTCAGGCAAGGATATG',
'AACTCGACGCGAAGAAATACGAGATCTTCTCAATAGGCAAGGCGGCGGAAAAAGATCCGCTGGAGAGCTCCGCAACTTATGAAACACCCCAATCC',
'ATCTGCGAGATCACGAGGCGACTGCACAGTCAAACATGCTGGTCTTCTTCTACTCAGGCAAGGATATG',
'AACTCGACGCGAAGAAATACGAGATCTTCTCAATAGGCAAGGCGGCGGAAAAAGATCCGCTGGAGAGCTCCGCAACTTATGAAACACCCCAATCC')

sdSeqStartPosition <- 1001
cdsNew <- decreaseGTsiteStrength(cds, sdSeqStartPosition)
print(cdsNew)
```

```r
degradSAs
Remove or degrade intrinsic strength of splice acceptors while adjusting HZEI integral.
```

Description

Degrade or remove splice sites of certain intrinsic strength (in MaxEntScan score) from a coding sequence by codon selection while keeping the HZEI integral up.
degradeSDs

Usage

degradeSAs(fanFunc, maxhbs=10, maxME=4, increaseHZEI=TRUE)

Arguments

fanFunc  
codon matrix with two rows (see example below)

maxhbs  
Numeric threshold which strength of internal donor sites should be degraded (in HBS)

maxME  
Numeric threshold which strength of internal acceptor sites should be degraded (in MaxEntScan score)

increaseHZEI  
Logical value if HZEI integral should be increased or decreased during SD degradation. If TRUE, function aims to increase HZEI integral.

Value

Character value of a nucleotide sequence encoding the same amino acid as the entered codon matrix fan, but the intrinsic strength of all present splice acceptor (SA) sites is degraded as much as possible, in case they exceed the given threshold maxME. Additionally, splice donor site strengths greater maxhbs are avoided, during SA degradation.

Examples

library(data.table)
sdMaximalHBS <- 10
acMaximalMaxent <- 4
increaseHZEI <- TRUE

## Initiating the Codons matrix plus corresponding amino acids
ntSequence <- 'TTTTGTCTTTTTCTGTGTGGCAGTGGGATTAGCCTCCTATCGATCTATGCGATA'

## Create Codon Matrix by splitting up the sequence by 3nt
fanFunc <- createCodonMatrix(ntSequence)
degradeSAs(fanFunc, maxhbs=sdMaximalHBS, maxME=acMaximalMaxent, increaseHZEI=increaseHZEI)

degradeSDs  
Remove or degrade intrinsic strength of splice donors while adjusting HZEI integral.

Description

Degrade or remove splice donor sites of certain intrinsic strength (in HBS) from a coding sequence by codon selection.

Usage

degradeSDs(fanFunc, maxhbs=10, increaseHZEI=TRUE)
Arguments

fanFunc       Codon matrix with two rows (see example below)
maxhbs       Numeric threshold which strength of internal donor sites should be degraded
increaseHZEI Logical value of HZEI integral should be increased or decreased during SD degradation. If TRUE, function aims to increase HZEI integral.

Value

Character value of a nucleotide sequence encoding the same amino acid as the entered codon matrix fanFunc, but the intrinsic strength of all present splice donors (SD) sites is degraded as much as possible, in case they exceed the given threshold maxhbs.

Examples

library(data.table)
## Initiaing the Codons matrix plus corresponding amino acids
ntSequence <- 'TTTTCGATGGATTAGCCTCCAGGTAAGTATCTATCGATCTATGCGATAG'
## Create Codon Matrix by splitting up the sequence by 3nt
fanFunc <- createCodonMatrix(ntSequence)
degradeSDs(fanFunc, maxhbs=10, increaseHZEI=TRUE)

generateRandomCodonsPerAA

Randomly choose Codon to encode amino acid sequence

Description

Encode amino acid sequence by random codon selection

Usage

generateRandomCodonsPerAA(aaVector)

Arguments

aaVector       Character vector of amino acids in three letter code (e.g. Met)

Value

Character value of a nucleotide sequence encoding the same amino acid as the entered by aaVector by random Codon selection.

Examples

generateRandomCodonsPerAA(c('Lys','Lys'))
getOverlappingVectorsFromVector

Create overlapping subvectors

Description
Create overlapping subvectors from large vector

Usage
getOverlappingVectorsFromVector(largeVector, subvectorLength, subvectorOverlap)

Arguments
- largeVector: Large character vector to break down into overlapping subvectors
- subvectorLength: Numeric value of length of smaller subvectors
- subvectorOverlap: Numeric value of length of subvector overlap

Value
Creates a list of overlapping subvectors from an input vector largeVector. The length of these overlapping subvectors is stated by subvectorLength and the overlap of the resulting subvectors is stated by subvectorOverlap.

Examples
getOverlappingVectorsFromVector(c(1,2,3,4), 2, 1)

hbg

Donor sequences and their HBS

Description
Donor sequences and their HBS

Usage
hbg
**Format**

A data frame with columns:

- **seq** 11nt long donor sequence
- **hbs** HBS of the donor sequence
- **special_seq** Shorter version of the donor sequence

**Examples**

hbg

---

**hex**  

*Hexamers and Z scores*

**Description**

Hexamers and Z scores

**Usage**

hex

**Format**

A data frame with columns:

- **seq** Sequence of the hexamer.
- **value** ZEI-score of the hexamer from HEXplorer.
- **first** First codon within the hexamer.
- **second** Second codon within the hexamer.
- **first_AA** First encoded amino acid within the hexamer (three letter code).
- **second_AA** Second encoded amino acid within the hexamer (three letter code).
- **AA** Both encoded amino acid within the hexamer

**Examples**

hex
increaseGTsiteStrength

Increasing intrinsic strength of specific GT site while keeping the HZEI integral neutral.

Description

Increasing intrinsic strength specific GT site from a coding sequence by codon selection keeping the HZEI integral near zero.

Usage

increaseGTsiteStrength(cds, sdSeqStartPosition)

Arguments

- **cds**: Coding nucleotide sequence which holds the splice site of interest
- **sdSeqStartPosition**: Numeric value of position of the first nucleotide of the splice donor of interest

Value

Character vector of a nucleotide sequence encoding the same amino acid as the entered cds, but the intrinsic strength of a specific GT site within the CDS is enhanced as much as possible.

Examples

```r
library(data.table)
cds <- paste0('ATGGAAGACGCAAAAAACATAAAGAAAAGGCCGCCGGCCATTCTATCCGCTGGAAGATGGAACCGCTGGAGAGCAACTGCA
TAAGGCTATGAAGAGATACGCCCTGGTTCCTGGAACAATTGCTTTTACAGATGCATATCGAGTGCGCTCGAGT',
'TGCCCTCTGGTGCAGACTATCTGACGATGATGGGTGCTGTGCTGAGTATGCGTGTGCTGGTGGGCCGAGAGCCTTATGCTCCAGAGGCCTAATCCTCGAGAAGGCTCACTGAGACTACATCAGCTATTCTGATTACACCCGAGGGGGATGATAAACCGGGCGCGGTCGGTAAAGTTGTTCCATTTTTTGAAGCGAA
GGTTGTGGATCTGGATACCGGGAAAACGCTGGGCGTTAATCAAAGAGGCGAACTGTGTGTGAGAGGTCCTATGATTATGTCCGGTTATGTAAAC
AATCCGGAAGCGACCAACGCCTTGATTGACAAGGATGGATGGCTACATTCTGGAGACATAGCTTACTGGGACGAAGACGAACACTTCTTCATCG
TTGACCGCCTGAAGTCTCTGATTAAGTACAAAGGCTATCAGGTGGCTCCCGCTGAATTGGAATCCATCTTGCTCCAACACCCCAACATCTTCGA
CGCAGGTGTCGCAGGTCTTCCCGACGATGACGCCGGTGAACTTCCCGCCGCCGTTGTTGTTTTGGAGCACGGAAAGACGATGACGGAAAAAGAG
ATCGTGGATTACGTCGCCAGTCAAGTAACAACCGCGAAAAAGTTGCGCGGAGGAGTTGTGTTTGTGGACGAAGTACCGAAAGGTCTTACCGGAA
AACTCGACGCAAGAAAAATCAGAGAGATCCTCATAAAGGCCAAGAAGGGCGGAAAGATCGCCGTG','

sdSeqStartPosition <- 1001
```
cdsNew <- increaseGTsiteStrength(cds, sdSeqStartPosition)
print(cdsNew)

---

**ModCon**

**Description**

Execute ModCon on a donor site within a coding sequence either increasing or decreasing its HZEI weight.

**Usage**

ModCon(cds, sdSeqStartPosition, upChangeCodonsIn=16, downChangeCodonsIn=16, optimizeContext=TRUE, sdMaximalHBS=10, acMaximalMaxent=4, optiRate=100, nGenerations=50, parentSize=300, startParentSize=1000, bestRate=40, semiLuckyRate=20, luckyRate=5, mutationRate=1e-04, nCores=-1)

**Arguments**

- **cds**: Character value of coding nucleotide sequence which holds the splice site of interest
- **sdSeqStartPosition**: Numeric value of the position of the first nucleotide of the splice donor of interest
- **upChangeCodonsIn**: Numeric value of number of codons to change upstream of the donor site of interest
- **downChangeCodonsIn**: Numeric value of number of codons to change downstream of the donor site of interest
- **optimizeContext**: Character value which determines, if TRUE (the default) the donor context will be adjusted to increase the splice site HEXplorer weight (SSHW), if FALSE, the SSHW will be decreased.
- **sdMaximalHBS**: Numeric value of minimal HBS of SDs which should be tried to be degraded in their intrinsic strength
- **acMaximalMaxent**: Numeric value of minimal MaxEntScan score of SAs which should be tried to be degraded in their intrinsic strength
- **optiRate**: Numeric value setting level of HZEI integral optimization
- **nGenerations**: Numeric value setting maximal number of generations
- **parentSize**: Numeric value setting size of parent generations, generated from previous generations
- **startParentSize**: Numeric value setting size of initiated parent generation of sequences
**ModCon**

**bestRate**  Numeric value setting percentage how many of the fittest sequences are used to produce the next generation

**semiLuckyRate**  Numeric value setting percentage of sequences which are selected for breeding with a probability based on the respective HZEI-score integral

**luckyRate**  Numeric value setting percentage of sequences which are randomly selected for breeding

**mutationRate**  Numeric value setting chance of each codon, to mutate randomly within a child sequence

**nCores**  Numeric value setting number of cores which should be used for parallel computations. If set to `-1` all available cores are selected.

**Value**

Creates a character value of a coding nucleotide sequence encoding the same amino acid sequence as the entered `cds`, and an alternative nucleotide surrounding around the splice donor (SD) sequence position, stated with `sdSeqStartPosition`. Depending on the entered `optimizeContext`, the SD surrounding is either adjusted aiming to enhance or decrease the splice site HEXplorer weight.

**Examples**

```r
## Load R packages
library('parallel')
library('utils')
library('data.table')

## Set parameters for simplest use of ModCon (optimizing to 100%)
cds <- paste0('ATGGGATCATCAAGAGGACGCTACGCTGATCTGTTTGCTGACAGCAATCGCATGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGATCGA','

```
## Print final cds sequence with the alternative SD nucleotide surrounding
print(finalSequence)

## More parameters can be set for use of ModCon when not optimizing to 100% (e.g. 50%)

## Execute ModCon
finalSequence <- ModCon(cds, 1001, upChangeCodonsIn=16, downChangeCodonsIn=16,
                         optimizeContext=FALSE, sdMaximalHBS=10, acMaximalMaxent=4,
                         optiRate=50, nGenerations=5, parentSize=200, startParentSize=800,
                         bestRate=50, semiLuckyRate=10, luckyRate=5, mutationRate=1e-03, nCores=1)

## Print final cds sequence with the alternative SD nucleotide surrounding
print(finalSequence)

---

### mutatePopulation

**Randomly exchange codons within a set of sequences.**

**Description**

For every codon within a set of nucleotide sequences randomly exchange the codon encoding the same amino acid to a certain chance.

**Usage**

```r
mutatePopulation(sequenceVector, codonReplacementChance)
```

**Arguments**

- `sequenceVector`: Character vector of nucleotide sequences (at least 3 nt long)
- `codonReplacementChance`: Numeric value of chance of a codons within the sequences to get exchanged to another codon encoding the same amino acid

**Value**

Creates a character vector of coding nucleotide sequences encoding the same amino acid sequence as the entered `sequenceVector`. By a mutation rate stated in `codonReplacementChance`, codons are randomly exchanged, by alternative codons encoding the same amino acid.

**Examples**

```r
mutatePopulation(c("CGCGATACGCTAGCCTACCGCATGATGGA","TGGGATATTTTAAGCGCTGACGATAGTGGA"), 0.1)
```
recombineTwoSequences Generate new sequence from recombination of two sequences

Description
This function generates a new sequence through recombination of two parental sequences using 3 modes of recombination. Either random combination of codons, crossover recombination or insertion.

Usage
recombineTwoSequences(ntSequence1, ntSequence2, preferenceVector)

Arguments
<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ntSequence1</td>
<td>Character value of a nucleotide sequence</td>
</tr>
<tr>
<td>ntSequence2</td>
<td>Character value of a nucleotide sequence</td>
</tr>
<tr>
<td>preferenceVector</td>
<td>Numeric vector of length three which indicates which modus of recombination should be preferred. The first number states the chance of random recombination, the second number indicates the chance of cross-over recombination and the third number indicates the chance of insertion recombination.</td>
</tr>
</tbody>
</table>

Value
Character value of a nucleotide sequence, generated by recombination from the entered ntSequence1 and ntSequence2. Modes of recombination are cross-over, insertion and random and mode preferences can be stated by preferenceVector.

Examples
recombineTwoSequences("AGGGCCTGGAGGAGGCTT","TAAGGGAAGCCTGGACCC",c(1,3,2))

selectBestAndMean Select best HZEI and mean

Description
From all sequences of a generation report highest HZEI integral and mean HZEI integral of all.

Usage
selectBestAndMean(sequenceVector, clusterName, increaseHZEI=TRUE)
selectMatingIndividuals

Arguments

- `sequenceVector`: Character vector of nucleotide sequences
- `clusterName`: Name of cluster generated with package parallel
- `increaseHZEI`: Logical value if HZEI integral should be increased or decreased during SD degradation. If TRUE, function aims to increase HZEI integral.

Value

Numeric vector of length 2 stating the best HZEI integral and the mean HZEI integral of a nucleotide sequence vector `sequenceVector`. Depending on the `increaseHZEI` mode, the best HZEI integral value is either the highest (for `increaseHZEI==TRUE`) or lowest (for `increaseHZEI==FALSE`).

Examples

```r
## Setup cluster
library(parallel)
nCores <- 1
clust <- makeCluster(nCores)
clusterExport(clust, list('getOverlappingVectorsFromVector', 'hex', 'calculateHZEIint'), envir = environment())
selectBestAndMean(c('CGCGATACGCTAAGCGCTACCGATAGTGGA', 'TGGGATATTTTAAGCGCTAGCTACCGATAGTGGA'), clust, increaseHZEI=TRUE)
```

selectMatingIndividuals

Selecting mating sequences from a pool of sequences

Description

Selecting sequences from a pool of nucleotide sequences based in chance and their HZEI integral.

Usage

```r
selectMatingIndividuals(inputGeneration, whoMatesBestPercent=40, whoMatesSemiRandom=20, whoMatesLuckily=5, clust, increaseHZEI=TRUE)
```

Arguments

- `inputGeneration`: Character vector of nucleotide sequences
- `whoMatesBestPercent`: Numeric value e.g. 20 (which would mean that sequences with the top 20 percent highest HZEI integral are selected for mating)
- `whoMatesSemiRandom`: Numeric value (is always lower than total number of sequences in `input_generation`)
- `whoMatesLuckily`: Numeric value (is always lower than total number of sequences in `input_generation`)

Description

Selecting sequences from a pool of nucleotide sequences based in chance and their HZEI integral.
slidingWindowHZEImanipulation

clust     Name of cluster generated with package parallel
increaseHZEI     Logical value of HZEI integral should be increased or decreased during SD degradation. If TRUE, function aims to increase HZEI integral.

Value

Character vector of nucleotide sequences which are selected from an entered vector of nucleotide sequences inputGeneration for creation of filial sequences by recombination. Sequences are selected by different criteria stated by whoMatesBestPercent, whoMatesSemiRandom, whoMatesLuckily and increaseHZEI.

Examples

```r
## Setup cluster
library(parallel)
ncores <- 1
clust <- makeCluster(ncores)
clusterExport(clust, list('getOverlappingVectorsFromVector', 'hex'), envir=environment())
selectMatingIndividuals(c('CGCGATACGCGCGATACG', 'CGCGATACGTGGGATATT', 'CTAAGCGCTCGCGATACG', 'CGCGATACGTTAAGCGCT', 'GACGATAGTCGCGATACG'), whoMatesBestPercent=40, whoMatesSemiRandom=1, whoMatesLuckily=1, clust, increaseHZEI=TRUE)
```

slidingWindowHZEImanipulation

*Quickly manipulate HZEI integral of nucleotide sequence*

Description

Quickly manipulate HZEI integral of nucleotide sequence (min. 21nt long)

Usage

```r
slidingWindowHZEImanipulation(inSeq, increaseHZEI=TRUE)
```

Arguments

- `inSeq` Character value of nucleotide sequence (min 21nt long, only bases 'A', 'G', 'T' or 'C')
- `increaseHZEI` Logical value if HZEI integral should be increased or decreased during SD degradation. If TRUE, function aims to increase HZEI integral.

Value

Character value of a nucleotide sequence encoding the same amino acid sequence as inSeq, but an increased HZEI integral, due to alternative codon selection, accomplished through sliding window optimization.
# Load R packages
library('parallel')
library('utils')
library('data.table')

# Set parameters for genetic algorithm
inSeq <- 'ATGGAAGACGCCAAAAACATAAAGAAGGCGGCTAGTGTCCATTGCGCCGATTCTATCCGTGGAAGATGGAATT'

maximizedHZEIseq <- slidingWindowHZEImanipulation(inSeq, increaseHZEI=TRUE)
minimizedHZEIseq <- slidingWindowHZEImanipulation(inSeq, increaseHZEI=FALSE)

#Access sequence with maximized HZEI integral
maximizedHZEIseq

#Access sequence with minimized HZEI integral
minimizedHZEIseq

---

**startModConApp**

Start GUI of VarCon.

**Description**

Start graphical user interface for the ModCon application.

**Usage**

```r
startModConApp()
```

**Value**

Shiny app

**Examples**

```r
startModConApp()
```
Index

* datasets
  cds, 4
  Codons, 5
  hbg, 11
  hex, 12

  calculateHZEIint, 2
  calculateMaxEntScanScore, 3
  cds, 4
  changeSequenceHZEI, 4
  Codons, 5
  createCodonMatrix, 6
  createFilialSequencePopulation, 7

  decreaseGTsiteStrength, 7
  degradeSAs, 8
  degradeSDs, 9

  generateRandomCodonsPerAA, 10
  getOverlappingVectorsFromVector, 11

  hbg, 11
  hex, 12

  increaseGTsiteStrength, 13

  ModCon, 14
  mutatePopulation, 16

  recombineTwoSequences, 17

  selectBestAndMean, 17
  selectMatingIndividuals, 18
  slidingWindowHZEImanipulation, 19
  startModConApp, 20