**createProbeCoords**  
Create probe coordinates

**Description**
Create probe coordinates

**Usage**
```r
createProbeCoords(nrows, ncols,  
    meta_nrows = 1, meta_ncols = 1,  
    meta_padding = 5)
```

**Arguments**
- `nrows` Number of rows per sub-array
- `ncols` Number of columns per sub-array
- `meta_nrows` Number of sub-arrays per row
- `meta_ncols` Number of sub-arrays per column
- `meta_padding` Padding between sub-arrays
expression_arraywide

Generate expression for a whole array

Description

Generate expression values for a whole array

Usage

expression_arraywide(n,
noise_mean = 50, noise_sd = 5,
signal_mean = 500, signal_sd = 0.9,
highbump_percent = 5,
highbump_mean = 6000, highbump_sd = 500)

replicate_arraywide(x)

Arguments

n Number of probes
x A vector of expression values
noise_mean Mean for the noise
noise_sd Standard deviation for the noise
signal_mean Mean for the signal
signal_sd Standard deviation for the signal

Examples

# array with 10,000 probes
one_plex <- createProbeCoords(100, 100)
plot(y ~ x, data=one_plex, pch=".",
     main = "array 1x10k")

# 4x2.5k array
four_plex <- createProbeCoords(50, 50, 2, 2)
plot(y ~ x, data=four_plex, pch=".",
     main = "array 4x2.5k")
**msubseq**

highbump_percent

Percentage of probes from the ‘high bump’

highbump_mean

Mean

highbump_sd

Standad deviation

**Details**

XXX

**Value**

A vector of numerical values (and of length n, or length(x))

**Examples**

```r
y <- expression_arraywide(1000)
y2 <- replicate_arraywide(y)
library(lattice)
densityplot(~ c(y, y2), groups = rep(c(1,2), rep(length(y), 2)))
```

---

**msubseq**

*Take multiple subsequences*

**Description**

Take multiple subsequences from one sequence

**Usage**

```r
msubseq(x, ir)
```

**Arguments**

- `x` *Sequence* object
- `ir` *IRanges* object

**Details**

Take the subsequences defined by an IRanges `ir` from a Sequence `x`.

**Value**

A `DNAStringSet`.

**See Also**

`subseq`
Examples

dna_length <- 100
dna <- randomDNASequences(1, dna_length)[[1]]
ir <- randomIRanges(100, 25, 10, dna_length)
dna_chunks <- msubseq(dna, ir)

randomDNASequences create random DNA sequences

Description
Create random DNA sequences

Usage
randomDNASequences(n, w)

Arguments
n n number of DNA sequences
w width of DNA sequences (recycled as necessary)

Value
A DNAStringSet of length n

Note
Currently, all amino acids are equally probable in the sequence. A parameter to control that is planned.

Examples
# two random Affymetrix-like probes
oligos <- randomDNASequences(2, 25)
randomIRanges

Description
Create random IRanges

Usage
randomIRanges(n, width, from, to, replace = TRUE)

Arguments
- n: number of IRanges
- width: width for the IRanges
- from: starting index value for the sequence to be covered by IRanges
- to: ending index value for the sequence to be covered by IRanges
- replace: sampling with replacement if TRUE (see Details)

Details
The from and to parameters describe the underlying sequence to be covered by the ranges. To prevent having ranges outside the sequence, the end of the IRanges returned cannot be greater than end - width.
If replace is TRUE, several IRanges can have the same starting value.

Value
An IRanges object of length n.

See Also
IRanges

Examples
n <- 10
rir <- randomIRanges(n, 5, 1, 33)

# ASCII-art view
reference <- paste("|",
paste(rep("-", 33-2), collapse=""),
"|",
sep = "")
regions <- vector("character", length=n)
for (i in 1:n) {
  regions[i] <- paste(
paste(rep(" ", start(rir)[i]), collapse=""),
paste(rep("-", width(rir)[i]), collapse=""),
sep = "")}
tilingProbes

Create tiling probes or ranges

Description

Create tiling probes or ranges

Usage

tilingProbes(width, step, template_seq)
tilingIRanges(width, step, from, to)

Arguments

from
    start position for the tiling
step
    increment in the starting index between one probe and the next.
template_seq
    template sequence from which tiling probes are to be extracted
to
    end position for the tiling
width
    width for the probes

Value

tilingProbes and tilingIRanges return a DNAStringSet and a IRanges respectively.

Examples

dna <- randomDNASequences(1, 30)[[1]]
tip <- tilingProbes(10, 2, dna)

    # ASCII-art
cat(as.character(dna), "\n")
for (i in 1:length(tip)) {
    cat(paste(rep("|", (i-1)*2), collapse=""),
        as.character(tip[[i]]), "\n",
        sep="")
}
cat(as.character(dna), "\n")
Index

*T*Topic *manip*

createProbeCoords, 1
expression_arraywide, 2
msubseq, 3
randomDNASequences, 4
randomIRanges, 4
tilingProbes, 5

createProbeCoords, 1
DNAStringSet, 3, 4, 6
tilingIRanges (tilingProbes), 5
tilingProbes, 5

expression_arraywide, 2
IRanges, 3, 5, 6

msubseq, 3
randomDNASequences, 4
randomIRanges, 4
replicate_arraywide (expression_arraywide), 2

Sequence, 3
subseq, 3