

# occugene

April 19, 2009

---

binHist

*Histogram Breakpoints*

---

## Description

Returns the histogram breakpoints for fast insertion.

## Usage

```
binHist (orf, overlap=NULL, bp=6264403)
```

## Arguments

orf	2-column matrix of annotation
overlap	number position of overlap
bp	number of base pairs in genome

## Details

Returns a vector of breakpoints for the binInsertHist function.

## Value

end.pt	Position of last target
orf	orfID
overlap	Number of targets in overlap

## Author(s)

Oliver Will (owill4@yahoo.com)

## References

See the book chapter O. Will (\*\*) in \*\*.

## See Also

binInsertHist

**Examples**

```
# **
```

---

binInsert

*Insert Locations*

---

**Description**

Returns the number of ORF knockouts.

**Usage**

```
binInsert(insert, orf, returnCounts=FALSE, overlap=NULL, DEBUG=FALSE)
```

**Arguments**

insert	List of insertion locations
orf	2-column matrix of annotation
returnCounts	Return the number of insertions
overlap	Number of shared targets
DEBUG	Flag to debug the code

**Details**

Finds the number of ORFs that have an insertion given a list of locations. If the returnCounts flag is true, the function returns the number of insertions per ORF. Uses the function hist for gains in speed.

**Value**

Returns a numeric or an object

**Author(s)**

Oliver Will (owill4@yahoo.com)

**References**

See the book chapter O. Will (\*\*) in \*\*.

**Examples**

```
# **
```

---

binInsertHist      *Insert Locations Quickly*

---

### Description

Given a list of locations, returns the number of ORFs hit.

### Usage

```
binInsertHist(insert, orfHist, returnCounts=FALSE)
```

### Arguments

insert	List of insertion locations
orfHist	Histogram breakpoints
returnCounts	Return the number of insertions

### Details

Finds the number of ORFs that have an insertion given a list of locations. If the returnCounts flag is true, the function returns the number of insertions per ORF. Uses the function hist for gains in speed.

### Value

Returns a numeric or an object

### Author(s)

Oliver Will (owill4@yahoo.com)

### References

See the book chapter O. Will (\*\*) in \*\*

### See Also

binHist

### Examples

```
# **
```

---

`checkFormat`*Checks the Format of Annotation and Insertions*

---

### Description

Checks the format of the annotation and insertions.

### Usage

```
checkFormat(anno, clone)
```

### Arguments

<code>anno</code>	2-column matrix of annotation
<code>clone</code>	vector

### Details

Checks the format of the annotation and insertions list. Annotation has to be a matrix of the first and last target in the ORF. Insertions has to be a vector. Will stop if not correct format.

### Value

Returns a boolean.

### Author(s)

Oliver Will (owill4@yahoo.com)

### References

See the book chapter O. Will (\*\*) in \*\*

### Examples

```
data(sampleAnnotation)
data(sampleInsertions)
anno <- cbind(sampleAnnotation$first, sampleAnnotation$last)
clone <- sampleInsertions$position
if (checkFormat(anno, clone)) {print("Looks good.");}
```

---

delta0	<i>Number of New Knockouts</i>
--------	--------------------------------

---

**Description**

Point estimate for the number of new ORF knockouts in the next  $d$  clones.

**Usage**

```
delta0(d, anno, clone)
```

**Arguments**

<code>d</code>	Number of clones to be made
<code>anno</code>	2-column matrix of annotation
<code>clone</code>	Vector of insertions

**Details**

Use the parametric form of the cumulative occupancy distribution to estimate the number of new ORF knockouts in the next  $d$  clones.

**Value**

A numeric

**Author(s)**

Oliver Will (owill4@yahoo.com)

**References**

See the book chapter O. Will (\*\*) in \*\*

**See Also**

unbiasDelta0

**Examples**

```
data(sampleAnnotation)
data(sampleInsertions)
anno <- cbind(sampleAnnotation$first, sampleAnnotation$last)
clone <- sampleInsertions$position
delta0(10, anno, clone)
```

---

`eMult`*Expected Value of the Occupancy Distribution*

---

**Description**

Returns the expected value of the occupancy distribution based on a multinomial distribution.

**Usage**

```
eMult(n, p, iter=NULL, seed=NULL, experimental=NULL)
```

**Arguments**

<code>n</code>	number of attempts in the multinomial distribution
<code>p</code>	probabilities for landing in a specific bin
<code>iter</code>	number of iterations used in the Monte-Carlo approximation
<code>seed</code>	seed for the random number generator
<code>experimental</code>	access to other functions of multinomials

**Details**

This functions computes the expected value of the occupancy distribution for a multinomial. In other words, the expected number of bins with at least one ball. The experimental argument "oneBall" computes expected number of bins with exactly one ball and the experimental argument "nextTo" computes the expected number of bins with one ball next to a bin with zero balls. Consider any functionality through the experimental argument untested.

**Value**

Returns a numeric

**Author(s)**

Oliver Will (owill4@yahoo.com)

**References**

See the book chapter O. Will (\*\*) in \*\* for specific details about this package or Johnson, N. L. and Kotz, S. (1977) *Urn Models and Their Application: An Approach to Modern Discrete Probability Theory*. John Wiley & Sons, New York, NY.

**Examples**

```
n <- 20
p <- c(seq(10, 1, -1), 47) / 100
p <- p / sum(p)
eMult(n, p)
eMult(n, p, iter=1000, seed=4)
```

---

`etDelta`*Number of New ORF Knockouts*

---

**Description**

Estimates the number of new knockouts in next `d` clones.

**Usage**

```
etDelta(d, anno, clone)
```

**Arguments**

<code>d</code>	number of new clones
<code>anno</code>	2-column matrix of annotation
<code>clone</code>	vector

**Details**

Estimates the number of new ORF knockouts in the next `d` clones using the method outlined by Efron and Thisted.

**Value**

<code>expected</code>	Expected value
<code>variance</code>	Variance

**Author(s)**

Oliver Will (owill4@yahoo.com)

**References**

See the book chapter O. Will (\*\*) in \*\* and also Efron, B. and Thisted, R. (1976) Estimating the number of unseen species: How many words did Shakespere know? *Biometrika*. 63, 435-447.

**Examples**

```
data(sampleAnnotation)
data(sampleInsertions)
a.data <- sampleAnnotation
experiment <- sampleInsertions
orf <- cbind(a.data$first, a.data$last)
clone <- experiment$position
etDelta(10, orf, clone)
```

---

`fCumul`*Parametric Function for the Cumulative Occupancy Distribution*

---

**Description**

Returns values for parameterized cumulative occupancy distributions.

**Usage**

```
fCumul(x, b0, b1, b2)
```

**Arguments**

<code>x</code>	Point to evaluate
<code>b0</code>	Parameter b0
<code>b1</code>	Parameter b1
<code>b2</code>	Parameter b2

**Details**

Function fitted to the cumulative occupancy distribution for a multinomial distribution. Exponential model :=  $b_0 - b_1 \cdot \exp(-b_2 \cdot x)$ .

**Value**

Returns a numeric

**Author(s)**

Oliver Will (owill4@yahoo.com)

**References**

See the book chapter O. Will (\*\*) in \*\*

**Examples**

```
x <- 2
b0 <- 3
b1 <- 3
b2 <- 0.01
val <- fCumul(x, b0, b1, b2)
```



**Description**

Parameterizes the cumulative occupancy distribution.

**Usage**

```
fFit(anno, clone, TR=TRUE, b0=0, b1=0, b2=.0)
```

**Arguments**

anno	2-column matrix of annotation
clone	vector
TR	Report a trace
b0	Starting value b0
b1	Starting value b1
b2	Starting value b2

**Details**

Fits various parametric functions to the occupancy distribution for a multinomial. Using the starting values of b0=0, b1=0, and b2=0 forces the function to find starting values for you.

**Value**

Returns a object.

**Author(s)**

Oliver Will (owill4@yahoo.com)

**References**

See the book chapter O. Will (\*\*) in \*\*

**Examples**

```
data(sampleAnnotation)
data(sampleInsertions)
anno <- cbind(sampleAnnotation$first, sampleAnnotation$last)
clone <- sampleInsertions$position
TR <- TRUE
fm <- fFit(anno, clone, TR)
```

---

loadAnnotation      *Loads Annotation File*

---

**Description**

Loads and checks an annotation file.

**Usage**

```
loadAnnotation(fileName)
```

**Arguments**

fileName      Name of file

**Details**

Annotation file need four columns: idNum, first, last, and overlap.

**Value**

Returns a data frame

**Author(s)**

Oliver Will (owill4@yahoo.com)

**References**

See the book chapter O. Will (\*\*) in \*\*

**Examples**

```
# No self contained example
```

---

loadInsertions      *Load Genome Annotation File*

---

**Description**

Loads a list of insertion locations.

**Usage**

```
loadInsertions(fileName)
```

**Arguments**

fileName      Name of the file

**Details**

Loads a list of insertion locations created in a transposon mutagenesis library.

**Value**

Returns a data frame

**Author(s)**

Oliver Will (owill4@yahoo.com)

**References**

See the book chapter O. Will (\*\*) in \*\*

**Examples**

```
# No self contained example
```

---

occup2Negenes

*Convert Occupancy Format to Negenes*

---

**Description**

Convert the annotation and insertion formation of the occupancy package into the format for the negenes package.

**Usage**

```
occup2Negenes (anno, clone, INTERGENIC=FALSE)
```

**Arguments**

anno	2-column matrix of annotation
clone	vector of insertion locations
INTERGENIC	Process the intergenic region as last ORF.

**Details**

Convert the annotation and insertion formation of the occupancy package into the format for the negenes package. Of the returned data frame, column 1 is n.sites, column 2, n.sites2, column 3, counts, column 4, counts2.

**Value**

Returns a data frame

**Author(s)**

Oliver Will (owill4@yahoo.com)

## References

See the book chapter O. Will (\*\*) in \*\*

## Examples

```
data(sampleAnnotation)
data(sampleInsertions)
anno <- cbind(sampleAnnotation$first, sampleAnnotation$last)
clone <- sampleInsertions$position
occup2Negenes(anno, clone)
```

---

sampleAnnotation *Annotation for a Hypothetical Prokayote*

---

## Description

This dataset has the annotation for a hypothetical bacterium.

## Usage

```
data(sampleAnnotation)
```

## Format

A data frame containing 4 columns with 10 rows.

## Author(s)

Oliver Will (owill4@yahoo.com)

## Source

Randomly generated.

## References

See the book chapter O. Will (\*\*) in \*\*

---

sampleInsertions     *Insertions for a Hypothetical Clonal Library*

---

**Description**

Insertion locations for a simple random mutagenesis library example.

**Usage**

```
data(sampleInsertions)
```

**Format**

A data frame containing 1 column with 20 rows.

**Author(s)**

Oliver Will (owill4@yahoo.com)

**Source**

Randomly generated.

**References**

See the book chapter O. Will (\*\*) in \*\*

---

unbiasB0     *Unbiased Estimator of the Number of Non-essential ORFs*

---

**Description**

Unbiased point estimate and confidence intervals for the number of non-essential ORFs.

**Usage**

```
unbiasB0(anno, clone, iter=1000, seed=NULL, alpha=0.05, TR=TRUE)
```

**Arguments**

anno	2-column matrix of annotation
clone	Vector of insertions
iter	Number of iterations for the bootstrap
seed	Seed for the random number generator
alpha	Type I error
TR	Report a trace

**Details**

Fits a parametric function to the cumulative occupancy distribution. Uses a parametric bootstrap to correct for bias and find confidence intervals for the number of non-essential ORFs.

**Value**

b0	Unbiased point estimate
CI	Confidence interval at the alpha specified

**Author(s)**

Oliver Will (owill4@yahoo.com)

**References**

See the book chapter O. Will (\*\*) in \*\*

**See Also**

fFit

**Examples**

```
data(sampleAnnotation)
data(sampleInsertions)
anno <- cbind(sampleAnnotation$first, sampleAnnotation$last)
clone <- sampleInsertions$position
TR <- TRUE
iter <- 10
seed <- 4
unbiasB0(anno, clone, iter, seed, TR=TR)
```

---

unbiasDelta0

*Unbiased Number of New Knockouts*

---

**Description**

Unbiased point estimate and confidence intervals for the number of new ORF knockouts in the next d clones.

**Usage**

```
unbiasDelta0(d, anno, clone, iter=1000, seed=NULL, alpha=0.05, TR=TRUE)
```

**Arguments**

d	Number of new clones
anno	2-column matrix of annotation
clone	Vector of insertions
iter	Number of iterations for the bootstrap
seed	Seed for the random number generator
alpha	Type I error
TR	Report a trace

**Details**

Fits a parametric function to the cumulative occupancy distribution. Uses a parametric bootstrap to correct for bias and find confidence intervals for the number of new ORF knockouts in the next  $d$  clones.

**Value**

delta0	Unbiased point estimate
CI	Confidence interval at the alpha specified

**Author(s)**

Oliver Will (owill4@yahoo.com)

**References**

See the book chapter O. Will (\*\*) in \*\*

**See Also**

delta0

**Examples**

```
data(sampleAnnotation)
data(sampleInsertions)
anno <- cbind(sampleAnnotation$first, sampleAnnotation$last)
clone <- sampleInsertions$position
TR <- TRUE
iter <- 10
seed <- 4
unbiasDelta0(10, anno, clone, iter, seed, TR=TR)
```

---

varMult

*Variance of the Occupancy Distribution*


---

**Description**

Returns the variance of the occupancy distribution based on a multinomial distribution.

**Usage**

```
varMult(n, p, iter=NULL, seed=NULL, experimental=NULL)
```

**Arguments**

n	number of attempts in the multinomial distribution
p	probabilities for landing in a specific bin
iter	number of iterations used in the Monte-Carlo approximation
seed	seed for the random number generator
experimental	access to other functions of multinomials

**Details**

This functions computes the variance of the occupancy distribution for a multinomial. In other words, the expected number of bins with at least one ball. The experimental argument "oneBall" computes variance of bins with exactly one ball and the experimental argument "nextTo" computes the variance of bins with one ball next to a bin with zero balls. Consider any functionality through the experimental argument untested.

**Value**

Returns a numeric

**Author(s)**

Oliver Will (owill4@yahoo.com)

**References**

See the book chapter O. Will (\*\*) in \*\* for specific details about this package or Johnson, N. L. and Kotz, S. (1977) *Urn Models and Their Application: An Approach to Modern Discrete Probability Theory*. John Wiley & Sons, New York, NY.

**Examples**

```
n <- 20
p <- c(seq(10, 1, -1), 47) / 100
p <- p / sum(p)
varMult(n, p)
varMult(n, p, iter=1000, seed=4)
```



# Index

- \*Topic **attribute**
  - checkFormat, 4
- \*Topic **datasets**
  - sampleAnnotation, 12
  - sampleInsertions, 13
- \*Topic **distribution**
  - eMult, 6
  - varMult, 15
- \*Topic **iteration**
  - binHist, 1
  - binInsert, 2
  - binInsertHist, 3
- \*Topic **manip**
  - loadAnnotation, 10
  - loadInsertions, 10
  - occup2Negenes, 11
- \*Topic **models**
  - fCumul, 8
  - fFit, 9
- \*Topic **nonlinear**
  - delta0, 5
  - unbiasB0, 13
  - unbiasDelta0, 14
- \*Topic **univar**
  - etDelta, 7

binHist, 1

binInsert, 2

binInsertHist, 3

checkFormat, 4

delta0, 5

eMult, 6

etDelta, 7

fCumul, 8

fFit, 9

loadAnnotation, 10

loadInsertions, 10

occup2Negenes, 11

sampleAnnotation, 12

sampleInsertions, 13

unbiasB0, 13

unbiasDelta0, 14

varMult, 15