Package 'flan'

May 9, 2019

Version 0.7

Date 2019-04-30

Type Package

Title FLuctuation ANalysis on Mutation Models

Author Adrien Mazoyer [aut, cre],
Remy Drouilhet [aut],
Stephane Despreaux [aut],
Bernard Ycart [aut]

Maintainer Adrien Mazoyer <mazoyer.adrien@courrier.uqam.ca>

Description Tools for fluctuations analysis of mutant cells counts.

License GPL

URL https://www.r-project.org, https://github.com/AdriMaz/flan

BugReports https://github.com/AdriMaz/flan/issues

Depends R (>= 3.0.0), lbfgsb3

Imports Rcpp (>= 0.12.5), methods

LinkingTo Rcpp, RcppArmadillo, RcppGSL

RcppModules flan_module

LazyData true

NeedsCompilation yes

SystemRequirements GNU GSL

Repository CRAN

Date/Publication 2019-05-08 22:50:09 UTC

R topics documented:

`flan-package` *Fluctuation analysis of mutant cell counts*

Description

Statistical tools for fluctuation analysis of mutant cells counts.

Details

Package:	flan
Type:	Package
Version:	0.7
Date:	2019-04-30
License:	GPL

This package contains functions dedicated to :

- Distribution (dflan, pflan, qflan, and rflan): built as the classic distribution functions, like dnorm, pnorm, qnorm, and rnorm.

- Parametric estimation (mutestim): estimates of the parameters of interest, which are the mean number of mutations (or the mutation probability) and the fitness parameter.

- Parametric testing (flan.test): built as the classic test functions, like t.test.

Author(s)

Adrien Mazoyer, Rémy Drouilhet, Stéphane Despréaux, and Bernard Ycart

Maintainer: Adrien Mazoyer <mazoyer.adrien@courrier.uqam.ca>

References

A. Mazoyer: Fluctuation analysis on mutation models with birth-date dependence Math. Biosci. (2018)

A. Mazoyer: Time inhomogeneous mutation models with birth-date dependence B. Math. Biol. (2017)

A. Mazoyer, R. Drouilhet, S. Despréaux and B. Ycart: flan: An R Package for Inference on Mutation Models. R Journal 9(1) (2017)

B. Ycart and N. Veziris: Unbiased estimates of mutation rates under fluctuating final counts. PLoS one 9(7) e101434 (2014)

B. Ycart: Fluctuation analysis with cell deaths. J. Applied Probab. Statist, 9(1):12-28 (2014)

B. Ycart: Fluctuation analysis: can estimates be trusted? One PLoS one 8(12) e80958 (2013)

A. Hamon and B. Ycart: Statistics for the Luria-Delbrück distribution. Elect. J. Statist., 6:1251-1272 (2012)

`boeal` *Number of nalidixic acid-resistant mutants from Boe et al. (1994)*

Description

Samples from Table 4 p. 2784-2785 of Boe and al (1994).

Plates with more than 512 mutants could not be counted with precision, hence the value 512 must be understood as "512 or more".

Usage

```
data(boeal)
```

Format

A list of 23 samples of mutants counts, each named "B<index of the sample>".

The i-th sample of the list includes the i-th column of the table.

Source

L. Boe, T. Tolker-Nielsen, K. M. Eegholm, H. Spliid, and A. Vrang: Fluctuation analysis of mutations to nalidixic acid resistance in *Escherichia Coli*, *J. Bacteriol.*, 176(10):2781-2787 (1994)

Examples

```
b <- unlist(boeal)              # concatenate all samples
ml <- mutestim(b)               # maximum likelihood
gf <- mutestim(b,method="GF")   # generating function
p0 <- mutestim(b,method="P0")   # P0 method
cbind(ml,gf,p0)                 # compare 3 methods

                                # test values of mutations and fitness
flan.test(b,alternative=c("greater","less"),mutations0=0.6,fitness0=1)

b1 <- unlist(boeal[1:10])       # first 10 samples
b2 <- unlist(boeal[11:20])      # next 10 samples
flan.test(list(b1,b2))          # test equality
```

david *Mutant counts and final numbers from H. L. David (1970)*

Description

Samples from Table 1 and 2 of H. L. David (1970).

Classes have been replaced by medians, 1000 for the last class.

Usage

```
data(david)
```

Format

A list of 11 lists named "D<index of the sample>". Each list includes :

- `sample` A sample of mutants counts.
- `fn` : the mean final numbers of cells, or a sample of final numbers (only for the last one).

The i-th sample of the list includes the i-th column of Table 1, except the last sample which includes Table 2.

Source

H. L. David, Probability distribution of drug-resistant mutants in unselected populations of *Mycobacterium tuberculosis*. *Appl. Microbiol.*, 20(5):810-814 (1970).

Examples

```
# samples 1-10 only give mean final numbers
D1 <- david[[1]]; D1
# estimate mutation probability
mutestim(D1$mc,mfn=D1$mfn)

# sample 11 has individual final numbers
D11 <- david[[11]]; D11
# with mean final number
mutestim(D11$mc,mfn=mean(D11$fn))
# with true final numbers
mutestim(D11$mc,D11$fn)
```

draw.clone *Graphic representation of clone growing upon a finite time.*

Description

Simulates a clone up to a time t and represents the clone as a binary tree.

Usage

```
draw.clone(t,mutprob=1.e-2,fitness=1.,death=0.,
   dist=list("lnorm",meanlog=-0.3795851,sdlog=0.3016223),
   col=c("green4","orange4"))
```

Arguments

t	time of end of experiment .
mutprob	mutation probability: numeric between 0 and 1. By default 1.e-2.
fitness	fitness parameter: numeric positive. By default 1.
death	death probability: numeric between 0 and 0.5. By default 0.
dist	lifetime distribution for mutant cells. See Details.
col	vector of size 2 of colors. The first is for the normal cells, the second for the mutant cells.

Details

This function does not provide a representation of a realistic realization of a mutation model (mutation probability too high, time of end of experiment to small).

The argument dist is a list beginning with the distribution name followed by its parameters, and must be one the 4 following distributions: "dirac", "exp", "lnorm"(meanlog, sdlog), "gamma"(shape, scale). Note that the parameters related to the "dirac" and "exp" cases are directly computed with inputs fitness and death.

See Also

rflan

Examples

```
# Luria-Delbrück model, mutation probability 1e-2, fitness 1
draw.clone(t=9,dist=list(name="exp",rate=1))

# Luria-Delbrück model, mutation probability 0.1, fitness 0.6
draw.clone(t=9,mutprob=0.1,fitness=0.6,dist=list(name="exp",rate=1))
```

```
# Haldane model, mutation probability 1e-2, fitness 1
draw.clone(t=7,dist=list(name="dirac",location=1))

# Lognormal lifetime distribution
draw.clone(t=7,fitness=0.5,death=0.1)

# Luria-Delbrück model with positive cell death probability
draw.clone(t=7,dist=list(name="exp",rate=1),death=0.2)
```

`flan.test` *Fluctuation Analysis parametric testing*

Description

Performs one-sample and two-sample Fluctuation Analysis tests on mutant counts. Returns confidence intervals and p-values, based on asymptotic normality, from the values returned by `mutestim`.

Usage

```
flan.test(mc, fn = NULL, mfn = NULL, cvfn = NULL,
             fitness = NULL, death = 0., plateff = 1.,
             model = c("LD", "H", "I"), muinf = +Inf,
             mutations0 = 1., mutprob0 = NULL, fitness0 = 1.,
             conf.level = 0.95,
             alternative = c("two.sided", "less", "greater"),
             method = c("ML", "GF", "P0"), winsor = 1024
             )
```

Arguments

`mc`	a numeric vector of mutant counts or a list of two numeric vectors of mutant counts.
`fn`	an optional numeric vector of final numbers or a list of two numeric vectors of final numbers.
`mfn`	mean final number of cells. Ignored if `fn` is non-missing.
`cvfn`	coefficient of variation of final number of cells. Ignored if `fn` is non-missing.
`fitness`	fitness parameter: ratio of growth rates of normal and mutant cells. If `fitness` is `NULL` (default) then the fitness will be estimated. Otherwise, the given value will be used to estimate the mean mutation number `mutations`
`death`	death probability. Must be smaller than 0.5. By default 0.
`plateff`	plating efficiency parameter. Must be non-larger than 1. By default 1. Available for `GF` method, and for `ML` method under `LD` model.

model	statistical lifetime model. Must be one of "LD" (default) for Luria-Delbrück model (exponential lifetimes), "H" for Haldane model (constant lifetimes), or "I" for Inhomogeneous model
muinf	parameter used only if model is I. See details.
mutations0	null hypothesis value for parameter mutations. See details.
mutprob0	null hypothesis value for parameter mutprob. See details.
fitness0	null hypothesis value for parameter fitness. See details.
conf.level	confidence level of the interval.
alternative	a character string specifying the alternative hypothesis, must be one of "two.sided" (default), "greater" or "less".
method	estimation method as a character string: one of ML (default), P0, or GF. See mutestim.
winsor	winsorization parameter: positive integer. Only used when method is ML or when method is P0 and fitness is NULL. See mutestim.

Details

flan.test tests the values of parameters mutations, mutprob, or fitness, with mutations0 mutprob0, fitness0 as null hypotheses.

If fn or mfn are given, mutprob is tested, otherwise, mutations is tested. If fitness is given, it is not tested.

muinf corresponds to the cumulative division rate on the interval [0 ; +Inf). If model is I, muinf has to be finite, else model is set to "LD".

alternative may be a two dimensional vector specifying the alternatives for the two parameters to be tested.

If mc is a list, a two-sample test is performed. In that case, the values in mutations0, mutprob0 and fitness0 apply to the difference between the two samples (with 0 as default value) For the two-sample tests, the parameters mfn, cvfn, fitness, death and plateff may be two dimensional vectors, where the first (resp. second) components are related to the first (resp. second) sample.

Value

Returns a list with class "flantest". The structure of a "flantest" object, is similar to that of a "htest" object (see also t.test). The class "flantest" contains the following components :

Tstat	the value of the computed statistic(s).
parameter	the values of fitness (if not tested), death and plateff.
p.value	the p-value(s) of the test.
conf.int	confidence interval(s) for the parameter(s) relative to the specified alternative.
estimates	the estimate(s).
null.value	the specified hypothesized value(s).
alternative	a (vector of) character string(s) describing the alternative hypothesis.
model	the statistical lifetime model.
method	method used to compute the estimate(s).
data.name	a character string giving the name of the complete data.

See Also

mutestim.

Examples

```
# one sample test, without final numbers
b <- unlist(boeal)
# is the mean mutation number greater than 0.7, and the fitness greater than 0.8?
flan.test(b, alternative = "greater", mutations0 = 0.7, fitness0 = 0.8)
# is the mean mutation number less than 0.8, given the fitness?
flan.test(b, alternative = "less", mutations0 = 0.8, fitness = 0.84)

# one sample test, with final numbers
d <- david[[11]]
flan.test(d$mc, d$fn, alternative = "less", mutprob0 = 2e-10, fitness0 = 2)

# two-sample test: test equality of parameters
b1 <- unlist(boeal[1:10])
b2 <- unlist(boeal[11:20])
flan.test(list(b1, b2))

# realistic random sample of size 100: mutation probability 1e-9,
# mean final number 1e9, coefficient of variation on final numbers 0.3,
# fitness 0.9, lognormal lifetimes, 5% mutant deaths

x <- rflan(100, mutprob = 1e-9, mfn = 1e9, cvfn = 0.3, fitness = 0.9, death = 0.05)

# test on mutations and fitness, without final numbers
flan.test(x$mc, mutations0 = 1, fitness0 = 0.9)

# test on mutprob and fitness, with final numbers
flan.test(x$mc, x$fn, mutprob0 = 1e-9, fitness0 = 0.9)

# given fitness

flan.test(x$mc, x$fn, fitness = 0.9, mutprob0 = 1e-9)

# take deaths into account
flan.test(x$mc, x$fn, mutprob0 = 1e-9, fitness0 = 0.9, death = 0.05)

# change method
flan.test(x$mc, x$fn, mutprob0 = 1e-9, fitness0 = 0.9, death = 0.05, method = "GF")

flan.test(x$mc, x$fn, mutprob0 = 1e-9, fitness0 = 0.9, death = 0.05, method = "P0")
# change model
flan.test(x$mc, x$fn, mutprob0 = 1e-9, fitness0 = 0.9, model = "H")

# Two-sample test
y <- rflan(100, mutprob = 1e-9, mfn = 1e9, cvfn = 0.3, fitness = 1.2, death = 0.05)
MC <- list(x$mc, y$mc)
FN <- list(x$fn, y$fn)
#
```

```
flan.test(mc = MC, fn = FN, fitness = c(0.9, 1.1), death = 0.05)
```

`luriadel` *Resistant bacteria counts from Luria and Delbruck (1943)*

Description

Samples of mutants counts from Table 2 of Luria and Delbrück (1943).

Usage

```
data(luriadel)
```

Format

A list of 3 samples of mutants counts, each named "LD<index of the sample>".

- The `LD1` sample includes the first four columns and the last one (experiments number 1, 10, 11, 15 and 21b) of table 2 p. 504.
- The `LD2` sample includes the fifth and the sixth columns (experiments number 16 and 17)
- The `LD3` sample includes the next to last column (experiment number 21a).

Source

D. E. Luria and M. Delbrück: Mutations of bacteria from virus sensitivity to virus resistance, *Genetics*, 28:491-511 (1943)

Examples

```
# 1st sample
mutestim(luriadel[[1]])

# compare 2nd and 3rd samples
flan.test(luriadel[2:3])
```

mutestim *Fluctuation Analysis parametric estimation*

Description

Estimates mean number of mutations, mutation probability, and fitness parameter, with different methods, under different models. Returns the estimated means and standard deviations for each parameter.

Usage

```
mutestim(mc, fn = NULL, mfn = NULL, cvfn = NULL,
                fitness = NULL, death = 0., plateff = 1.,
                model = c("LD", "H", "I"), muinf = +Inf,
                method = c("ML", "GF", "P0"),
                winsor = 1024)
```

Arguments

mc	a (non-empty) numeric vector of mutants counts.
fn	an optional (non-empty) numeric vector with same length as mc of final numbers of cells.
mfn	mean final number of cells. Ignored if fn is non-missing. By default empty.
cvfn	coefficient of variation of final number of cells. Ignored if fn is non-missing. By default empty.
fitness	fitness parameter: ratio of growth rates of normal and mutant cells. If fitness is NULL (default), then the fitness will be estimated. Otherwise, the given value will be used to estimate the mean mutation number mutations.
death	death probability. Must be smaller than 0.5. By default 0.
plateff	plating efficiency parameter. Must be non-larger than 1. By default 1. Available for GF method, and for ML method under LD model.
model	statistical lifetime model. Must be one of "LD" (default) for Luria-Delbrück model (exponential lifetimes), "H" for Haldane model (constant lifetimes), or "I" for Inhomogeneous model
muinf	parameter used only if model is I. See details.
method	estimation method as a character string: one of ML (default), P0, or GF. See details.
winsor	winsorization parameter: positive integer. Only used when method is ML or when method is P0 and fitness is NULL. See details.

Details

Method ML is the classic maximum likelihood estimation method. The maximum is computed with a BFGS (bounded) algorithm.

Method P0 uses the number of null values in the sample, therefore it can be applied only if there is at least one zero in mc. The estimate of the fitness is computed by maximum likelihood.

Method GF uses the empirical generating function of mc. Since this method is the fastest, "GF" is used to initialize the values of the estimates for methods "ML" and "P0" (if the fitness is estimated).

If fn, mfn or cvfn is non-empty, then the mutation probability is estimated instead of the mean number of mutations. If fn is non-empty and method is P0 or GF, then mfn and cvfn are computed from fn, and the estimate of of the mutation probability is deduced from the estimate of the mean number of mutations. If fn is non-empty and method is ML, the estimate of the mutation probability is directly computed.

muinf corresponds to the cumulative division rate on the interval [0 ; +Inf). If model is I, muinf has to be finite, else model is set to "LD"

The winsorization parameter winsor is used as a threshold for values in mc when maximum likelihood estimates are computed.

Value

A list containing the following components:

mutations	mean number of mutations
sd.mutations	estimated standard deviation on mean number of mutations
mutprob	mutation probability (if fn, mfn or cvfn is non-empty)
sd.mutprob	estimated standard deviation on mutation probability
fitness	estimated fitness (if argument fitness is NULL)
sd.fitness	estimated standard deviation on fitness

References

A. Mazoyer: Fluctuation analysis on mutation models with birth-date dependence. Math. Biosci 303(9): 83-100 (2018)

B. Ycart and N. Veziris: Unbiased estimates of mutation rates under fluctuating final counts. PLoS one 9(7) e101434 (2014)

B. Ycart: Fluctuation analysis with cell deaths. J. Applied Probab. Statist, 9(1):12-28 (2014)

B. Ycart: Fluctuation analysis: can estimates be trusted? One PLoS one 8(12) e80958 (2013)

A. Hamon and B. Ycart: Statistics for the Luria-Delbrück distribution. Elect. J. Statist., 6:1251-1272 (2012)

See Also

rflan, flan.test.

Examples

```
# realistic random sample of size 100: mutation probability 1e-9,
# mean final number 1e9, coefficient of variation on final numbers 0.3,
# fitness 0.9, lognormal lifetimes, 5% mutant deaths, plating efficiency 80%
x <- rflan(100, mutprob = 1e-9, mfn = 1e9, cvfn = 0.3, fitness = 0.9, death = 0.05, plateff = 0.8)

# maximum likelihood estimates with mean final number
meanfn <- mutestim(x$mc, mfn = 1e9)

# maximum likelihood estimates with final numbers
withfn <- mutestim(x$mc, x$fn)

# change model
Hmodel <- mutestim(x$mc, x$fn, model = "H")

# faster methods
GFmethod <- mutestim(x$mc, x$fn, method = "GF")
P0method <- mutestim(x$mc, x$fn, method = "P0")

# take deaths into account
withdeaths <- mutestim(x$mc, x$fn, death = 0.05, method = "GF")

# with plateff
withpef <- mutestim(x$mc, x$fn, death = 0.05, plateff = 0.8, method = "GF")

# compare results
rbind(meanfn, withfn, Hmodel, GFmethod, P0method, withdeaths, withpef)

# extreme example
x <- rflan(1000, mutations = 50, fitness = 0.5, dist = "exp")$mc
summary(x)
mutestim(x, method = "GF")
mutestim(x)
mutestim(x, winsor = 2000)

## Not run:
  # None null count in the sample: P0 can not be used.
  mutestim(x, method = "P0")

## End(Not run)
```

---

number of mutant cells distribution

The distribution of mutant cell counts

Description

Density, distribution function, quantile function and random generation for mutant cell counts.

Usage

```
dflan(m, mutations = 1., fitness = 1., death = 0., plateff = 1.,
      model = c("LD", "H", "I"), muinf = +Inf)
pflan(m, mutations = 1., fitness = 1., death = 0., plateff = 1.,
      model = c("LD", "H", "I"), muinf = +Inf, lower.tail = TRUE)
qflan(p, mutations = 1., fitness = 1., death = 0., plateff = 1.,
      model = c("LD", "H", "I"), muinf = +Inf, lower.tail = TRUE)
rflan(n, mutations = 1., mutprob = NULL, fitness = 1., death = 0., plateff = 1.,
      dist = list(name = "lnorm", meanlog = -0.3795851, sdlog = 0.3016223),
      distfn = NULL,
      mfn = 1e9, cvfn = 0,
      muih = list(mu = NULL, muinv0 = NULL),
      ...)
```

Arguments

m	vector of quantiles.
p	vector of probabilities.
n	number of observations. If length(n) > 1, the length is taken to be the number required.
mutations	mean number of mutations: positive integer. Ignored if mutprob is non-empty.
mutprob	mutation probability: numeric between 0 and 1. By default empty. See details.
fitness	fitness parameter: numeric positive. By default 1.
death	death probability: numeric between 0 and 0.5. By default 0.
plateff	plating efficiency parameter: numeric number between 0 and 1. By default 1. See Details.
dist	lifetime distribution for mutant cells. See Details.
model	statistical lifetime model. Must be one of "LD" (default) for Luria-Delbrück model (exponential lifetimes), "H" for Haldane model (constant lifetimes), or "I" for Inhomogeneous model
distfn	final number of cells distribution. Must be one of "lnorm" (default if cvfn is positive) or "gamma".
mfn	mean final number of cells: numeric positive.
cvfn	coefficient of variation of final numbers of cells: numeric, default 0. If non-zero and if mutprob is empty, compute the sample with mutprob = mutations/mfn
muih	functions for inhomogeneous model. See details.
muinf	parameter used only if model is I. See details.
lower.tail	logical: if TRUE (default), probabilities are P[X <= x] otherwise, P[X > m].
...	Arguments to be passed for muih, such that hyperparameters.

Details

The argument dist is a list beginning with the distribution name followed by its parameters, and must be one the 4 following distributions: "dirac", "exp", "lnorm"(meanlog, sdlog), "gamma"(shape, scale). Note that the parameters related to the "dirac" and "exp" cases are directly computed with inputs fitness and death.

If muih$mu is non-empty, a time dependent growth model is performed. The function muih$mu is a bivariate function which represents roughly the mean growth of clone durig an time interval. The function muih$muinv0 is the inverse function of muih$mu(0,.). If muih$muinv0 is empty, it is approximated using muih$mu.

muinf corresponds to the cumulative division rate on the interval [0 ; +Inf). It is the limit as t tends to infinity of muih$mu(0,t). If model is I, muinf has to be finite, else model is set to "LD"

If mutprob is non-empty and if cvfn is zero, the sample of mutants is computed with mutations as the product of mutprob by mfn. If cvfn is positive, the final numbers of cells are generated with the log-normal (if distfn is "lnorm" or empty) or the gamma (if distfn is "gamma") distribution with mean mfn and coefficient of variation cvfn. The sample of mutants is then generated conditionally to the sample of final numbers of cells.

A plating efficiency plateff smaller than 1 can be taken into account in the dflan, pflan and qflan functions only under LD model.

If the plating efficiency plateff is smaller than 1 in the flan function, then each mutant count mc is replaced by a binomial number with parameters size = mc and prob = plateff.

Value

dflan gives the density, pflan gives the distribution function, qflan gives the quantile function, and rflan generates a random sample.

rflan returns a list with two arguments, each with length n: a vector of integers $mc (mutant counts), and a vector of numeric $fn (final numbers of cells).

See Also

link{draw.clone}

Examples

```
#---------------------- distributions ---------------------------------

# Luria-Delbrück model, mean number of mutations 1, fitness parameter 1
dflan(0:10)
pflan(0:10)
qflan(c(0.95, 0.99))

# Luria-Delbrück model, mean number of mutations 2, fitness parameter 0.5
qflan(c(0.95, 0.99), mutations = 2, fitness = 0.5)

qflan(c(0.05, 0.01), mutations = 2, fitness = 0.5, lower.tail = FALSE)

# Haldane model, mean number of mutations 2, fitness parameter 0.5
qflan(c(0.95, 0.99), mutations = 2, fitness = 0.5, model = "H")
```

```
#---------------------- random samples ---------------------------------
# lognormal lifetime distribution, lognormal final numbers
X <- rflan(100, cvfn = 0.3)
X$mc              # mutant counts
X$fn              # final numbers

# lognormal lifetime distribution, gamma final numbers
X <- rflan(100, distfn = "gamma", cvfn = 0.3)
X$mc              # mutant counts
X$fn              # final numbers

# mean number of mutations 2, fitness parameter 0.5 (realistic model, but slow)

# lognormal lifetimes
rflan(1000, mutations = 2, fitness = 0.5)$mc
# gamma lifetimes
rflan(1000, mutations = 2, fitness = 0.5, dist = list(name = "gamma", shape = 10, scale = 0.05))$mc

# exponential lifetimes (Luria-Delbrück model, fast)
rflan(1000, mutations = 2, fitness = 0.5, dist = list("exp"))$mc

# constant lifetimes (Haldane model, fast)
rflan(1000, mutations = 2, fitness = 0.5, dist = "dirac")$mc

# specifying mutation probability and mean final number

rflan(1000, mutprob = 2e-9, mfn = 2e9, fitness = 0.5, dist = list("dirac"))$mc


# positive cell death probability
rflan(1000, mutprob = 2e-9, mfn = 2e9, death = 0.1, fitness = 0.5, dist = list("exp"))$mc

#------------- Inhomogeneous model ---------
f <- function(t,n0,nf) nf/(1+(nf/n0-1)*exp(-t))  # Logistic function: mean growth of a clone
finv <- function(s,n0,nf) -log((nf/s-1)/(nf/n0-1))  # Inverse function of f

mu <- function(s,t,n0,nf) log(f(t,n0,nf)/f(s,n0,nf))   # Definition of mu
muinv0 <- function(u,n0,nf) finv(f(0.,n0,nf)*exp(u),n0,nf)   # Inverse function of mu

rflan(1000, muih = list(mu = mu, muinv0 = muinv0), n0 = 1e2, nf = 1e6)$mc
```

`roster` *Mutant counts from Rosche and Foster (2000)*

Description

Sample from Table 3 of Rosche and Foster (2000).

Usage

```
data(roster)
```

Format

A vector of positive integers with size 52.

Source

W. A. Rosche and P. L. Foster, Determining mutation rates in bacterial populations, *Methods*, 20(1):4-17 (2000)

Examples

```
# are parameters significantly larger than 1 ?
flan.test(roster,alternative="greater",mutations0=1,fitness0=1)
```

---

werhoff *Rifampin-resistant mutant counts from Werngren and Hoffner (2003)*

Description

Samples from Table 1 p. 1522 of Werngren and Hoffner (2003).

The mean final number of cells is also given for each sample.

The coefficient of variation of the final number of cells is the same for all samples.

Usage

```
data(werhoff)
```

Format

A list of 2 elements :

- `cvfn` : the coefficient of variation of final numbers of cells, for all samples.
- `samples` : a list of 13 lists. Each list includes :
  - A sample of mutants counts named "W<index of the sample".
  - `mfn` : the mean final number of cells.

Source

J. Werngren and S. E. Hoffner: Drug susceptible Mycobacterium tuberculosis Beijing genotype does not develop mutation-conferred resistance to Rifampin at an elevated rate, *J. Clin. Microbiol.*, 41(4):1520-1524 (2003)

Examples

```
# coefficient of variation of final numbers
Wcvfn <- werhoff$cvfn
# tenth sample
W10 <- werhoff$samples[[10]]

# estimates without cvfn
without <- mutestim(W10$mc,mfn=W10$mfn)
# estimates with cvfn
with <- mutestim(W10$mc,mfn=W10$mfn,cvfn=Wcvfn)
rbind(without,with)
```

Index