# Package ‘bayescount’

April 20, 2015

**Title** Power Calculations and Bayesian Analysis of Count Distributions and FECRT Data using MCMC

**Version** 0.9.99-5

**Date** 2015-04-18

**Depends** R (>= 2.14.0)

**Imports** runjags (>= 2.0.1), rjags, coda, stats, utils

**SystemRequirements** JAGS (http://mcmc-jags.sourceforge.net)

**Description** A set of functions to allow analysis of count data (such as faecal egg count data) using Bayesian MCMC methods. Returns information on the possible values for mean count, coefficient of variation and zero inflation (true prevalence) present in the data. A complete faecal egg count reduction test (FECRT) model is implemented, which returns inference on the true efficacy of the drug from the pre- and post-treatment data provided, using non-parametric bootstrapping as well as using Bayesian MCMC. Functions to perform power analyses for faecal egg counts (including FECRT) are also provided.

**License** GPL-2

**URL** `http://bayescount.sourceforge.net`

**Author** Matthew Denwood [aut, cre]

**Maintainer** Matthew Denwood <`md@sund.ku.dk`>

**NeedsCompilation** yes

**Repository** CRAN

**Date/Publication** 2015-04-20 11:46:52

## R topics documented:

## `bayescount` *Analyse Count data using MCMC*

### Description

Apply a Bayesian [zero-inflated] gamma / Weibull / lognormal / independant / simple Poisson model to count data to return possible values for mean count, coefficient of variation, and zero-inflation. A text .csv file named [name].[model].csv with the results is optionally written to the working directory (before checking if the file already exists), and an object [name].[model].results is copied to the Global environment within R. Where more than 1 model is used, a results file and object is created for each model. Convergence is assessed for each dataset by calculating the Gelman-Rubin statistic for each parameter, see `autorun.jags`. Optionally, the log likelihood for the model fit is also calculated. The time taken to complete each analysis (not including calculation of the likelihood) is also recorded. This function is a wrapper for bayescount.single(), allowing extra automation. The lower level functions in the runjags package are used for calling JAGS.

Note: The GUI interface for R in Windows may not continually refresh the output window, making it difficult to track the progress of the simulation (if silent.jags is FALSE). To avoid this, you can run the function from the terminal version of R (located in the Program Files/R/bin/ folder).

* NOTE: THIS FUNCTION IS DEPRECATED AND WILL BE REMOVED FROM BAYESCOUNT VERSION 1 - SEE `count.model` FOR AN ALTERNATIVE *

### Usage

```
bayescount(name = NA, data = NA, setnames = NA,
   model = c("ZILP"), divide.data = 1,
   scale.mean = divide.data, remove.all.zeros = TRUE,
   test = TRUE, alt.prior = FALSE, write.file = TRUE,
   adjust.zi.mean = FALSE, likelihood = FALSE,
   record.chains = FALSE, ...)
```

### Arguments

`name` a name for the analysis (character). Missing by default (function will require it to be input).

`data` either a path to a comma delimited csv file, or an existing R object containing the data. Data can be specified in one of the following ways. If a numeric vector, or as a matrix or array with 1 column only, the data are taken to be a single dataset. If a matrix, then each column of data is taken to be a dataset. If an array, then each element of the 3rd dimension represents a dataset, and each column represents repeated McMasters counts from the same sample (each row). A list containing the elements 'totals' representing the sum of the repeated McMasters counts, and 'repeats' representing the number of McMasters counts performed per sample, is also supported (these may be matrices, in which case each column is a seperate dataset). If the data is in a comma delimited file, it must be in the format specified for a single matrix. Missing data, or unused elements of non-ragged arrays, may be represented using NA, which will be removed from the data before analysis. This argument is 'NA' by default (function will require a path to the data to be input).

`setnames` either a character vector of names for each dataset, a logical value indicating if the data contains column labels in the first row, or an NA. If a character vector, the function will quit if the length does not match the length of the data. If TRUE, the dimnames[[2]] attribute of the matrix will be used if present, and failing that the first row of data is used. If NA, then the function will use the dimnames[[2]] attribute of the matrix as setnames if it is not NULL, or otherwise generate them. If FALSE, generic names are used. If data is specified as an array or list, setnames must be provided as a character vector or left as FALSE (or NA). Default NA.

`model` vector of models to use. Choices are "GP" (gamma Poisson = negative binomial), "ZIGP" (zero-inflated gamma Poisson = zero-inflated negative binomial), "LP" (lognormal Poisson), "ZILP" (zero-inflated lognormal Poisson), "WP" (Wiebull Poisson), "ZIWP" (zero-inflated Weibull Poisson), "SP" (simple Poisson), "ZISP" (zero-inflated simple Poisson) or "IP" (independant Poisson), or "all" for all of these models (case insensitive). The simple Poisson model forces each count to have the same mean, wheras the independant Poisson process allows each count to have an unrelated mean (therefore a zero-inflated version is not possible). Default "ZILP".

`divide.data` count division factor to allow egg count data in eggs per gram to be used raw (numeric). Default 1 (no transformation to data).

`scale.mean` count multiplication factor to allow results to be reported as eggs per gram. Default equal to divide.data, so that results are reported with the same mean as the input data.

`remove.all.zeros`

remove any datasets where the total number of counts is 0, since it is not appropriate to use a count model to analyse these data (logical). Some models may crash repeatedly when trying to analyse a dataset with no observations. Default TRUE.

`test` should the function briefly test the model with the first column of data before running the simulation? (logical) Affords extra 'user-proofing'. If set to FALSE and valid values are supplied for 'name', 'data' and 'setnames', the function will not require input at any point (useful for automated data analysis). Default TRUE.

`alt.prior` should the model run the [ZI] [WP|GP|LP] models using the standard or the alternative prior distribution for variability? (logical) Can also be a character value of a user-specified prior distribution. Default FALSE. Where information concerning the coefficient of variation in the data is sparse, the choice of prior distribution will have an affect on the posterior distribution for ALL parameters. It is recommended to run a simulation using both types of prior when working with small datasets, to make sure results are consistent.

`write.file` should the function write a text file to the current directory containing the results? (logical) Default TRUE. If FALSE, the text file is written during analysis and then deleted on completion of the dataset.

`adjust.zi.mean` should the mean count parameter of the zero-inflated models be adjusted to reflect the mean of the whole population? (logical) If FALSE the mean count of the zero-inflated models reflects the mean of the gamma or Poisson distribution only, if TRUE the mean includes extra zeros. Used for comparing results between zero-inflated and non zero-inflated models. Default FALSE.

`likelihood` should the (log) likelihood for the fit of each model to each dataset be calculated? (logical) The likelihood for the [ZI] LP and GP models are calculated using a likelihood function integrated over all possible values for lambda, which can take some time. The likelihood is calculated using a thinned chain of 1000 values to reduce the time taken. Likelihood calculations for the [ZI] WP model are currently not available for data with repeated observations, or for any model with either missing data or different numbers of repeated observations per sample within a dataset. Default FALSE.

`record.chains` should the final mcmc chains generated for each simulation be saved to disk for future analysis? If TRUE, an Rsave binary file is saved in the current working directory with a filename made up of the name of the analysis, model and dataset name. The function `new_unique` is used to generate the filename to ensure that no files are ever over-written. Default FALSE.

`...` further arguments to be passed directly to `autorun.jags`, including the path to JAGS, number of burnin and sampled iterations, target PSRF, time limit to extend each simulation, etc. Note that some of the default arguments to autorun.jags are changed by bayescount.single (max.time="1hr", interactive=FALSE, plots=FALSE), but these can be over-ridden.

## Value

No value is returned by this function. Instead, a text .csv file named *name*.*model*.csv with the results is optionally written to the working directory (before checking if the file already exists), and an object *name*.*model*.results is copied to the Global environment within R (this will over-write any existing object of the same name). Where more than 1 model is used, a results file and object is created for each model.

The results files consist of a table (or matrix) with the following results for each dataset:

`name` the name of the dataset. This is a dimnames attribute for matrices.

`converged` a logical flag indicating a PSRF of below the target indicating successful convergence (1), or the opposite (0). Datasets where the model initially converged but the subsequent final chains had a PSRF of more than the target are classified as 'converged', but a warning message is printed in the function and the datasets will have an error code of 4.

`error.code` a numeric code indicating the exit status of the simulation. 0 indicates no errors encountered, 1 indicates a repeatedly crashing model, 2 indicates a failure to converge within the specified time limit, 3 indicates a dataset with no observations that was removed from analysis, 4 indicates a model that initially converged but the subsequent final chains had a PSRF of more than the target, and 5 indicates an unknown error in the function not related to the model (i.e. a bug in the runjags/bayescount code...).

`samples` the number of sampled iterations calculated by `raftery.diag` to reduce the MC error to the specified level.

`samples.to.conv`
the number of sampled iterations discarded as a result of slow convergence (will be 0 if the pilot chain converged).

`parameter estimates`
the lower 95% credible interval, median estimate and upper 95% credible interval for the mean, coefficient of variation, zero-inflation, log mean, log variance, shape parameter, and scale parameter as appropriate to the model.

`multivariate.psrf`
the multivariate PSRF of the final mcmc chains. If this couldn't be calculated, NA.

`likelihood estimates`
the lower 95% credible interval, median estimate, upper 95% credible interval and maximum observed value (this is NOT equivalent to the maximum likelihood) of the likelihood for the model. For the compound distributions these likelihoods are integrated for all possible values of the Poisson means (calculated using `likelihood`), and may well be different to the deviance calculated using JAGS.

`time.taken` the total time in seconds taken to run the simulation (not including calculating the likelihood and other summary statistics).

## See Also

`autorun.jags`, `count.analysis`, and `fec.power` for power analysis methods for FEC.

## Examples

```
# run the function with all values as default, and 'name' and 'data'
# (from a local .csv file) to be input by the user when prompted:
# bayescount()

# analyse local data (2 datasets with 20 animals each with 10 repeat
# samples) using a zero-inflated lognormal Poisson model:
## Not run:
# Simulate some data:
data <- array(dim=c(20,10,2))
means1 <- rgamma(20, 10, 1)
```

```
means2 <- rgamma(20, 5, 1)
for(i in 1:20){
data[i,,1] <- rpois(10, means1[i])
data[i,,2] <- rpois(10, means2[i])
}
# Missing data is permissible but means the likelihood cannot be
# calculated - a warning will be printed:
data[sample(1:(20*10*2), 10)] <- NA
try(unlink("analysis.ZILP.csv"), silent=TRUE)
# Run the analysis:
bayescount(name="analysis", data=data, model = "ZILP",
setnames=c("Simulated group A", "Simulated group B"), likelihood=TRUE)

## End(Not run)
```

## count.analysis *Analyse Count data using MCMC*

### Description

Apply a Bayesian [zero-inflated] gamma / Weibull / lognormal / independant / simple Poisson model to count data to return possible values for mean count, coefficient of variation, and zero-inflation, as either summary statistics or mcmc objects. Convergence is assessed for each dataset by calculating the Gelman-Rubin statistic for each parameter, see `autorun.jags`. Optionally, the log likelihood for the model fit is also calculated. The time taken to complete each analysis (not including calculation of the likelihood) is also recorded. The lower level functions in the runjags package are used for calling JAGS.

Note: The GUI interface for R in Windows may not continually refresh the output window, making it difficult to track the progress of the simulation (if silent.jags is FALSE). To avoid this, you can run the function from the terminal version of R (located in the Program Files/R/bin/ folder).

### Usage

```
count.analysis(data = stop("Data must be specified"), model="ZILP",
alt.prior = FALSE, adjust.zi.mean = FALSE, raw.output = FALSE,
likelihood=FALSE, ...)
```

### Arguments

`data` an existing R object containing the data. Data can either be specified as a numeric vector of single counts for each sample, or as a matrix of repeated counts (columns) for each sample (rows). Repeated counts are modelled as part of the same Poisson process. Data can also be specified as a (ragged) list of repeated counts, with each element of the list representing a seperate sample. Finally, data can be specified as a list containing the elements 'totals' representing the sum of the repeated McMasters counts, and 'repeats' representing the number of

McMasters counts performed per sample. Missing data (or unused elements of non-ragged arrays) may be represented using NA, which will be removed from the data before analysis. The likelihood calculation is not available for ragged arrays and missing data, and will print a warning. No default.

`model` the model to use. Choices are "GP" (gamma Poisson = negative binomial), "ZIGP" (zero-inflated gamma Poisson = zero-inflated negative binomial), "LP" (lognormal Poisson), "ZILP" (zero-inflated lognormal Poisson), "WP" (Wiebull Poisson), "ZIWP" (zero-inflated Weibull Poisson), "SP" (simple Poisson), "ZISP" (zero-inflated simple Poisson) or "IP" (independant Poisson). Case insensitive. The simple Poisson model forces each count to have the same mean, wheras the independant Poisson process allows each count to have an unrelated mean (therefore a zero-inflated version is not possible). Default "ZILP".

`alt.prior` should the model run the [ZI] [WP|GP|LP] models using the standard or the alternative prior distribution for variance? (logical) Can also be a character value of a user-specified prior distribution. Default FALSE. Where information concerning overdispersion in the data is sparse, the choice of prior distribution will have an affect on the posterior distribution for ALL parameters. It is recommended to run a simulation using both types of prior when working with small datasets, to make sure results are consistent.

`adjust.zi.mean` should the mean count parameter of the zero-inflated models be adjusted to reflect the mean of the whole population? (logical) If FALSE the mean count of the zero-inflated models reflects the mean of the gamma or Poisson distribution only, if TRUE the mean includes extra zeros. Used for comparing results between zero-inflated and non zero-inflated models. Default FALSE.

`raw.output` the function can return either a summary of the results (as with bayescount), or an MCMC object representing the estimates at each iteration for both chains (including the likelihood estimates where appropriate). If TRUE, the latter is output. (logical) Default FALSE.

`likelihood` should the (log) likelihood for the fit of the model to the dataset be calculated? (logical) The likelihood for the [ZI] WP, LP and GP models are calculated using a likelihood function integrated over all possible values for lambda, which can take some time. The likelihood is calculated using a thinned chain of 1000 values to reduce the time taken. ** This option is deprecated and will be removed from bayescount version 1 **

`...` additional arguments to be passed directly to `autorun.jags`.

## Value

Either a vector similar to that obtained from `bayescount` containing an indication of the error/crash/convergence status, the number of sampled updates used, and a lower/upper 95% highest posterior density interval (see `HPDinterval`), and median estimate for each relevant parameter (optionally including the likelihood), or an MCMC object representing the estimates at each iteration for both chains (optionally including the likelihood).

## See Also

`count.model`

## Examples

```
# use a zero-inflated lognormal Poisson model to analyse some count
# data, and suppressing JAGS output:
#
## Not run:
results <- bayescount.single(data=c(0,5,3,7,0,4,3,8,0),
model="ZILP", silent.jags=TRUE)

## End(Not run)
```

---

# count.model *Analyse Count Data Using Jags*

---

## Description

Apply a Bayesian (zero-inflated) (gamma / Weibull / lognormal / independant / simple) Poisson model to count data to return possible values for mean count, variance, shape paramater, scale parameter (overdispersion or 'k') and zero-infaltion where appropriate to the model selected. This function generates the model specifications and starting values, and is used by the higher level functions bayescount and bayescount.single, but can also be called directly.

## Usage

```
count.model(data=stop("No data supplied"),
   model=stop("No model specified"), call.jags = FALSE,
   alt.prior=FALSE, monitor.lambda=FALSE,
   monitor.deviance=FALSE, ...)
```

## Arguments

`data` an existing R vector containing the data (integer vector). No default.

`model` model to use. Choices are "GP" (gamma Poisson = negative binomial), "ZIGP" (zero-inflated gamma Poisson = zero-inflated negative binomial), "LP" (lognormal Poisson), "ZILP" (zero-inflated lognormal Poisson), "WP" (Wiebull Poisson), "ZIWP" (zero-inflated Weibull Poisson), "SP" (simple Poisson), "ZISP" (zero-inflated simple Poisson) or "IP" (independant Poisson). The simple Poisson model forces each count to have the same mean, wheras the independant Poisson process allows each count to have an unrelated mean (therefore a zero-inflated version is not possible). Default "ZILP".

`call.jags` should the function run the model using run.jags? If TRUE, the model is run and the results are returned, otherwise a compiled but not updated object of class `runjags-class` is returned. NOTE: This argument is deprecated - in future only call.jags=FALSE will be permitted.

`alt.prior` should the model run the [ZI] [WP|GP|LP] models using the standard or the alternative prior distribution for variance? (logical) Can also be a character value of a user-specified prior distribution. Default FALSE. Where information concerning overdispersion in the data is sparse, the choice of prior distribution will have an affect on the posterior distribution for ALL parameters. It is recommended to run a simulation using both types of prior when working with small datasets, to make sure results are consistent.

`monitor.lambda` should the model or model specification monitor the mean of the Poisson process for each datapoint? This is required to calculate the likelihood for the Independant Poisson model only, but may be useful for other purposes. Default FALSE.

`monitor.deviance`
option to monitor the deviance of the model (using the built-in ’deviance’ monitor). Default FALSE.

`...` additional arguments to be passed directly to `autorun.jags`.

## Value

An object of class `runjags-class`

## See Also

`count.analysis`

## Examples

```

#  Return the model specification and starting values for a
# lognormal Poisson, then run the model using run.jags:

## Not run:
data <- rpois(100, rlnorm(3, 0.2))

model <- run.model(model="LP", data=data, call.jags=FALSE)
library('runjags')
results <- extend.jags(model, burnin=5000, sample=10000)


## End(Not run)
```

---

# `count.power` *Count Data Power Analysis Calculations*

---

## Description

Finds the power for a faecal egg count study with the given combination of parameters. This represents the probability that the observed empirical mean FEC will lie between the lower.limit and upper.limit specified. The power is calculated using the negative binomial distribution when considering the true mean of a single individual, or using Monte Carlo integration for more than one animal. Confidence intervals for the true power are produced for the latter.

## Usage

```
count.power(meanepg=200, g.faeces=3, sensitivity=1/25,
   replicates=1, animals=10, coeffvarrep=0.4, coeffvarind=0.3,
   coeffvargroup=0.7, true.sample=FALSE, accuracy=0.1,
   lower.limit=meanepg*(1-accuracy), upper.limit=meanepg*(1+accuracy),
   maxiterations=1000000, precision=2, confidence = 0.99,
   feedback=FALSE, forcesim=FALSE)
```

## Arguments

`meanepg` the mean egg count of the group (in EPG). If this is unknown then use a value likely to be encountered, or iterate over a distribution of values to obtain a distribution of values for power.

`g.faeces` the number of grams of faeces used per sample (must be the same for all samples).

`sensitivity` the minimum egg detection threshold or counting sensitivity of the technique used. If using the McMasters technique, this is the number of McMasters chambers counted divided by 50.

`replicates` the number of different samples (individually analysed) taken from each animal. Must be the same for all animals. This would normally be 1, but increasing this provides an effective way of improving power at the cost of additional laboratory work.

`animals` the number of animals in the group. Can be 1, in which case the ability to predict the true mean of the animal (if true.sample=TRUE) or the true mean of a group from which the animal is taken (if true.sample=FALSE) can be calculated.

`coeffvarrep` coefficient of variation between sub-samples taken from the same faecal sample, assuming 1g faeces is used per sub-sample (the effective value is automatically adjusted according to g.faeces in the function). The default value for this is taken from an intensive study of a small group of horses, but may not be accurate in other groups of animals.

`coeffvarind` coefficient of variation between samples taken from different faecal piles from the same animal over a period of days, not including that due to coeffvarrep. The default value for this is taken from an intensive study of a small group of horses, but may not be accurate in other groups of animals.

`coeffvargroup` coefficient of variation between animals. This includes the variability between samples taken from different animals within a group, not including that due to within animal variability. The default value for this is inferred from a combination of several equine datasets and an intensive study of a small group of horses, but may not be accurate in other groups of animals.

`true.sample` option to calculate the power for the population mean (if true.sample=FALSE) or the true sample mean (if true.sample=TRUE). The difference is that conceptually the true mean of a small group of animals may not reflect the mean of the population from which they are derived. If only one animal is considered, the true sample mean is the true mean of the individual whereas the population mean is the mean of a (theoretical) group of animals from which it is derived. The power will always be greater (or identical) for the true sample mean.

`accuracy` specify the lower and upper limits as the meanepg +/- this fraction of the meanepg. Ignored if non-default values are specified for lower.limit or upper.limit.

`lower.limit` lower limit for the tolerance. Specifying lower.limit and upper.limit separately allows non-symmetrical tolerance around the mean with which to calculate the power. The default is to use meanepg*(1-accuracy).

`upper.limit` upper limit for the tolerance. Specifying lower.limit and upper.limit separately allows non-symmetrical tolerance around the mean with which to calculate the power. The default is to use meanepg*(1+accuracy).

`maxiterations` the maximum number of iterations to use for the Monte Carlo integration, or the number to use if precision=NA. If precision is defined, then only the number of iterations required to estimate the power to the given precision are performed (up to a maximum of maxiterations).

`precision` the number of decimal places with which to calculate the power, unless maxiterations is reached first. A larger precision will give a more precise estimate of the true power but will take longer to calculate. Specifying this as NA performs the calculation on a fixed (=maxiterations) number of iterations.

`confidence` the degree of confidence required with which to report confidence limits for the true power when using Monte Carlo integration to report the power.

`feedback` option to display a progress indicator for calculation of the values used for Monte Carlo integration. Using feedback with some GUI versions of R may slow down the analysis considerably.

`forcesim` option to force the function to use the Monte Carlo method of approximating the power when the default would be to use a negative binomial approximation (when true.sample=FALSE and coeffvargroup is very small, or when true.sample=TRUE and animals=1). Results using the two methods should be very similar, however there may be differences due to the differenct optimisation mechanisms when calculating both lower and upper limits. In addition, Monte Carlo integration will take longer and gives a confidence interval for the true mean rather than the absolute value.

## Value

Returns a list containing the elements 'roundedci' and 'ci', which specifies the median and confidence limits (as defined by 'confidence') for the true power both rounded by 'precison' and unrounded. For analyses using the Monte Carlo integration method, 'within' 'without' and 'total' are also returned, and indicate the number of iterations for which the observed mean fell outside and inside the specified limits and the total number of iterations.

### See Also

fec.power, fecrt.analysis

## count.precision *Count Data Predicted Precision Calculations*

### Description

Finds the appropriate tolerance with which to consider the observed mean for a faecal egg count study with the given combination of power and other parameters. The precision is calculated using the negative binomial distribution when considering the true mean of a single individual, or using Monte Carlo integration for more than one animal. Confidence intervals for the true power are produced for the latter. Tolerance can be defined as either a lower limit only if upper.limit is defined, as an upper limit only if lower.limit is defined, or as both (equidistant) limits if neither are defined.

### Usage

```
count.precision(meanepg=200, g.faeces=3, sensitivity=1/25,
   replicates=1, animals=10, coeffvarrep=0.4, coeffvarind=0.3,
   coeffvargroup=0.7, true.sample=FALSE, lower.limit=NA,
   upper.limit=NA, iterations=100000, power = 0.95,
   confidence = 0.99, feedback=FALSE, forcesim=FALSE)
```

### Arguments

`meanepg` the mean egg count of the group (in EPG). If this is unknown then use a value likely to be encountered, or iterate over a distribution of values to obtain a distribution of values for tolerance.

`g.faeces` the number of grams of faeces used per sample (must be the same for all samples).

`sensitivity` the minimum egg detection threshold or counting sensitivity of the technique used. If using the McMasters technique, this is the number of McMasters chambers counted divided by 50.

`replicates` the number of different samples (individually analysed) taken from each animal. Must be the same for all animals. This would normally be 1, but increasing this provides an effective way of improving power at the cost of additional laboratory work.

`animals` the number of animals in the group. Can be 1, in which case the ability to predict the true mean of the animal (if true.sample=TRUE) or the true mean of a group from which the animal is taken (if true.sample=FALSE) can be calculated.

`coeffvarrep` coefficient of variation between sub-samples taken from the same faecal sample, assuming 1g faeces is used per sub-sample (the effective value is automatically adjusted according to g.faeces in the function). The default value for this is taken from an intensive study of a small group of horses, but may not be accurate in other groups of animals.

`coeffvarind` coefficient of variation between samples taken from different faecal piles from the same animal over a period of days, not including that due to coeffvarrep. The default value for this is taken from an intensive study of a small group of horses, but may not be accurate in other groups of animals.

`coeffvargroup` coefficient of variation between animals. This includes the variability between samples taken from different animals within a group, not including that due to within animal variability. The default value for this is inferred from a combination of several equine datasets and an intensive study of a small group of horses, but may not be accurate in other groups of animals.

`true.sample` option to calculate the power for the population mean (if true.sample=FALSE) or the true sample mean (if true.sample=TRUE). The difference is that conceptually the true mean of a small group of animals may not reflect the mean of the population from which they are derived. If only one animal is considered, the true sample mean is the true mean of the individual whereas the population mean is the mean of a (theoretical) group of animals from which it is derived. The power will always be greater (or identical) for the true sample mean.

`lower.limit` (optional) lower limit for the tolerance. If this is supplied, the required upper limit to maintain the desired power with the specified lower limit will be calculated. If neither lower limit or upper limit are specified, symmetrical limits about the mean are found that satisfy the power condition.

`upper.limit` (optional) upper limit for the tolerance. If this is supplied, the required lower limit to maintain the desired power with the specified upper limit will be calculated. If neither lower limit or upper limit are specified, symmetrical limits about the mean are found that satisfy the power condition.

`iterations` the number of iterations to use for the Monte Carlo integration. More iterations will take longer but provide a smaller confidence interval for the true power.

`power` the desired power with which to interpret the given tolerances. Default is 0.95, so that the observed mean FEC will lie within the calculated lower and upper limits 95% of the time.

`confidence` the degree of confidence required with which to report confidence limits for the true power when using Monte Carlo integration to report the power. Only used for calculating the true power after determining the tolerance limits.

`feedback` option to display a progress indicator for calculation of the values used for Monte Carlo integration. Using feedback with some GUI versions of R may slow down the analysis considerably.

`forcesim` option to force the function to use the Monte Carlo method of approximating the power when the default would be to use a negative binomial approximation (when true.sample=FALSE and coeffvargroup is very small, or when true.sample=TRUE and animals=1). Results using the two methods should be very similar, however there may be differences due to the differenct optimisation mechanisms when calculating both lower and upper limits. In addition, Monte Carlo integration will take longer and gives a confidence interval for the true mean rather than the absolute value.

### Value

Returns a list containing the elements 'limits', which is the calculated lower and upper tolerance level which provides the required power, and 'power' which specifies the median estimate and confidence intervals for the true power when using Monte Carlo integration, or the absolute value (replicated 3 times for consistency) if using the negative binomial approximation method. The true power returned may not exactly match the required power input due to the integer nature of FEC data.

### See Also

count.power, count.analysis

---

## fecrt.analysis *Analyse FECRT Data Using MCMC to Give a Probability Distribution of Values for the Mean Egg Count Reduction*

---

### Description

This function applies a Bayesian [zero-inflated] gamma Poisson (negative binomial) model to faecal egg count reduction test (FECRT) data to return possible values for the mean drug efficacy. Pre-treatment data are assumed to arise from either a gamma-Poisson or zero-inflated gamma-Poisson distribution, with post-treatment data described by a separate gamma-Poisson or zero-inflated gamma-Poisson distribution. The change in mean between these distributions is therefore the mean egg count reduction. If paired.model=TRUE, a slightly different formulation is used whereby the observed count pre-treatment is assumed to follow a compound gamma-gamma-Poisson distribution with the variability within and between animals separated. The post treatment mean for each animal is derived from the pre-treatment animal mean and FEC reduction. This formulation allows data with non-random missing post-treatment counts to be analysed correctly, and also allows data with repeat counts from an individual to be analysed - providing a method of increasing the power of the method substantially. Results are also obtained using non-parametric bootstrapping and the method advocated in the 1992 World Association for the Advancement of Veterinary Parasitology (W.A.A.V.P.) methods for the detection of anthelmintic resistance in nematodes of veterinary importance (unless the data contains repeat values or missing values). Confidence intervals for the relevant statistics are printed to file if write.file = TRUE, and returned using a custom print method. Lower level running of JAGS and assessing the simulation for convergence and required run length is done using autorun.jags.

** NOTE: THIS FUNCTION IS CURRENTLY UNDER RE-DEVELOPMENT **

Please feel free to contact the package author if you would like to collaborate on a relevant data analysis

### Usage

```
fecrt.analysis(name = NA, pre.data = NA, post.data = NA,
   data = list(pre=pre.data, post=post.data), animal.names = FALSE,
   efficacy=95, confidence=0.95, control.animals = FALSE,
   divide.data = 1, record.chains = FALSE, write.file = FALSE,
   bootstrap.iters=10000, plot.graph = TRUE, skip.mcmc = FALSE, ...)
```

## Arguments

`name` a name for the analysis (character). Missing by default (function will require it to be input).

`pre.data` the pre-treatment data. Either a numeric vector, a matrix with repeated McMasters counts from the same sample (each row) in different columns, or an array with repeat samples from the same animal in dimension 1, repeated McMasters counts from the same sample in dimension 2 (can be of length 1 if only 1 count recorded), and different animals in dimension 3. If an array, then the use.paired must be TRUE. No default. Ignored if a value is specified for data.

`post.data` the post-treatment data. Either a numeric vector, a matrix with repeated McMasters counts from the same sample (each row) in different columns, or an array with repeat samples from the same animal in dimension 1, repeated McMasters counts from the same sample in dimension 2 (can be of length 1 if only 1 count recorded), and different animals in dimension 3. If an array, then use.paired must be TRUE. No default. Ignored if a value is specified for data.

`data` either a path to a comma delimited csv file, or an existing R object containing the data. Data can be specified in one of the following ways. If a matrix, the first column is pre-treatment data, the second column is the post-treatment data, and the third column (if supplied) indicates control (1) or treatment (0) animal. Alternatively, the first column is animal names id animal.names = TRUE, with columns 2 and 3 making up the pre- and post-treatment data and optionally column 4 the treatment of control status. If the data is specified as a list, then the first element is taken as the pre-treatment data, and the second element is taken as the post-treatment data. These elements of the list can be provided as for pre.data and post.data. If the data is in a comma delimited file, it must be in the format specified for a single matrix. Missing data, or unused elements of non-ragged arrays, may be represented using NA, which will be removed from the data before analysis. This argument is taken from the specified values of pre.data and post.data by default. If a value is specified for data then arguments specified for pre.data and post.data are ignored.

`animal.names` either a character vector of names to be used for each animal specified in the data, TRUE or FALSE. If TRUE, then animal names are taken from the first column of the matrix supplied for data (a matrix must be supplied for data if animal.names is TRUE). If FALSE, then animal names are generated automatically. Default FALSE.

`efficacy` the target % efficacy of the drug used. Used to calculate the probability of resistance with the MCMC and bootstrap methods. Default 95%.

`confidence` the degree of confidence required with which to report the confidence limits for the true FEC reduction (a probability between 0 and 1).

`control.animals`
indication of which animals are to be used as controls. Should be either a vector of TRUE/FALSE (or 1/0) corresponding to whether each animal is a control (TRUE) or treatment (FALSE), or simply FALSE (the default) in which case all animals are assumed to be treated. Ignored if data is specified as a matrix (or csv file) with 3 columns, in which case the third column should reflect the treatment status. Default FALSE.

`divide.data` count division factor to allow egg count data in eggs per gram to be used raw (numeric). Default 1 (no transformation to data).

`record.chains` option to allow the MCMC chains to be recorded for future use. If TRUE, the function returns the MCMC object as part of the return value (the MCMC object is not printed using the print method). If write.file==TRUE, the results are also saved using a filename containing the name of the analysis. Default FALSE.

`write.file` option to write the results of the analysis to a text file using a filename containing the name of the analysis. The contents of the text file are identical to the return value of the function. Default FALSE.

`bootstrap.iters`
the number of bootstrap iterations to use for the bootstrap method. Default 10000.

`plot.graph` an option to plot the posterior true egg count reduction from the MCMC method graphically. If write.file==TRUE then the graph is saved as a PDF, otherwise the graph is plotted on the active device. Default FALSE.

`skip.mcmc` option to omit the MCMC analysis, and return bootstrap and WAAVP method analysis results alone. Default FALSE.

`...` other options to be passed directly to `fecrt.model` and then `autoextend.jags`.

### Value

Returns a list of the results obtained using each method, printed using a custom print method.

### References

M. J. Denwood, S. W. J. Reid, S. Love, M. K. Nielsen, L. Matthews, I. J. McKendrick, and G. T. Innocent. Comparison of three alternative methods for analysis of equine Faecal Egg Count Reduction Test data. Prev. Vet. Med. (2009), doi:10.1016/j.prevetmed.2009.11.009

### See Also

`fecrt.power` and `fecrt.power.limits` for power analysis methods for the FECRT, and `fecrt.model` and `autorun.jags` for relevent lower-level functions to which additional arguments can be passed

## `fecrt.model` *Create an MCMC model to analyse FECRT Data*

### Description

This function generates and compiles a JAGS model representation of a faecal egg count reduction test (FECRT) analysis. The return value can be updated manually using `extend.jags`

### Usage

```
fecrt.model(data, paired.model = FALSE, fix.controls = FALSE,
  fix.efficacy = TRUE, fix.variation = TRUE, zero.inflation = FALSE,
  effect.prior = "dbeta(1,1)", mean.prior = "dmouch(1)",
  precision.prior = "dmouch(1)", ...)
```

## Arguments

`data` a data frame containing the dataset to be analysed in long format, with columns of Count (the number of eggs observed), Subject (a name or number uniquely identifiyig each animal/patient) and Time (1 = Pre-treatment, 2 = Post-treatment), and optional columns of Sample (a number) and Control (1 = Treatment group, 2 = Control group). Multiple Counts are permitted from the same subject, either with the same Sample number (repeated count observations from the same processed laboratory sample) or different Sample numbers (independently analysed samples from the same individual).

`paired.model` the FECRT can be run using two compound gamma distributions to describe the variability within and between animals in place of the single gamma distribution combining the two sources of variability. When using two compound gamma distributions for pre-treatment data, the post-treatment data are paired to the pre-treatment data by animal. The advantage of using a single distribution is that the model is more identifiable, and therefore is likely to converge more quickly and return errors less frequently. The advantage of the paired model is that it allows repeat measurements tobe incorporated in the model, and non-randomly missing data (ie.protocols that involve post-treatment sampling of only animals with a high pre-treatment count) to be modelled appropriately. The simple model cannot be used when using repeat samples within animal, and will provide inaccurate results if animals are targeted for post-treatment sampling based on their pre-treatment count. Default uses the simple model (FALSE).

`fix.controls` the FECRT can be run either using control animals to estimate the overall reduction atrributable to the treatment, or assuming that the true mean of the controls does not change over time and only using this data to improve the estimate of the pre-treatment mean. The former is more in the spirit of what is intended with controls, but the latter will give smaller confidence intervals at the cost of the strong assumption that the underlying population mean has not changed between the treatment dates. Default FALSE.

`fix.efficacy` force all treatment animals to have the same reduction following treatment, so that all residual variation is absorbed by the extra-Poisson sampling variation. Setting this to FALSE requires the paired model and a substantial number of replicate samples within subjects. Default TRUE.

`fix.variation` force the post-treatment extra-Poisson variation to be the same as that pre-treatment. This parameter is difficult to estimate, so forcing it to be the same as the pre-treatment value may improve convergence, although it is a strong assumption. Note that this parameter and the efficacy variance parameter are likely to be strongly inter-dependent. Default FALSE.

`zero.inflation` option to use a zero-inflated gamma Poisson inplace of the gamma Poisson distribution. If TRUE, zero inflated distributions are used pre- and post-treatment (with the prevalence fixed between pre- and post-treatment distributions). Default FALSE.

`effect.prior` the prior distribution to use for the change in mean egg count. Any syntactically valid distribution in JAGS may be used, but it must include the default initial values of 0.01 and 1. Default is a uniform distribution between 0 and 1.

`mean.prior` the prior distribution to use for the pre-treatment mean egg count. Any syntactically valid strictly positive distribution in JAGS may be used, but it must include the default initial value of the arithmetic mean pre-treatment count. Default is DuMouchel's prior distribution.

`precision.prior`
: the prior distribution to use for the shape parameter (k) of the various gamma distributions used. Any syntactically valid strictly positive distribution in JAGS may be used, but it must include the default initial values of 0.1 and 10. Default is DuMouchel's prior distribution.

`...`
: other options to be passed directly to `autorun.jags` when setting up the model. This may include initial values, for example .RNG.seed and .RNG.name to allow the analysis to be reproducible.

## Details

Pre-treatment data are assumed to arise from either a gamma-Poisson or zero-inflated gamma-Poisson distribution, with post-treatment data described by a separate gamma-Poisson or zero-inflated gamma-Poisson distribution. The change in mean between these distributions is therefore the mean egg count reduction. A change in shape parameter of the gamma distribution is also permitted if fix.variation=FALSE. If paired.model=TRUE, a slightly different formulation is used whereby the observed count pre-treatment is assumed to follow a compound gamma-gamma-Poisson distribution with the variability within and between animals separated. The post treatment mean for each animal is derived from the pre-treatment animal mean and FEC reduction. This formulation allows data with non-random missing post-treatment counts to be analysed correctly, and also allows data with repeat counts from an individual to be analysed - providing a method of increasing the power of the method substantially. The fix.efficacy=FALSE option is only permitted for the paired.model=TRUE option, and allows the reduction estimated to vary between individuals. Problems with convergence are likely to be encountered with this option unless there is a substantial amount of replicate data avaialble within individuals.

## Value

Returns an object of class `runjags-class`

## References

M. J. Denwood, S. W. J. Reid, S. Love, M. K. Nielsen, L. Matthews, I. J. McKendrick, and G. T. Innocent. Comparison of three alternative methods for analysis of equine Faecal Egg Count Reduction Test data. Prev. Vet. Med. (2009), doi:10.1016/j.prevetmed.2009.11.009

## See Also

`fecrt.analysis` for comparisons of fitted MCMC models to bootstrapping results

## Examples

```
## Not run:
# Data in an appropriate format:
data <- data.frame(Count=rpois(80,rep(c(10,10,2,2), 20)),
Subject=rep(1:20, each=4), Time=rep(rep(1:2,each=2),40),
Sample=1:2, Control=rep(c(0,1), each=40))
# Compile the model - a paired model is required because
# there are replicate samples within an individual:
model <- fecrt.model(data, paired.model=TRUE)
# Update the model - requires runjags:
library('runjags')
results <- extend.jags(model, burnin=5000)

## End(Not run)
```

## `fecrt.power` *FECRT Power Analysis Calculations*

### Description

Finds the power for a faecal egg count reduction test with the given combination of parameters. This represents the probability that the observed empirical mean reduction will lie between the lower.limit and upper.limit specified. The power is calculated using Monte Carlo integration and confidence intervals for the true power are produced. This function can be used to determine the probability that the observed empirical mean reduction will lie within a the range (for example) 90% to 100% if the true reduction is 80% (the false negative rate if considering anthelmintic resistance), or the probability that the observed empirical mean reduction will lie within a the range (for example) 0% to 90% if the true reduction is 95% (the false positive rate if considering anthelmintic resistance)

### Usage

```
fecrt.power(meanepg=200, reduction = 80, g.faeces=3,
   sensitivity=1/25, replicates=1, animals=10, pre.coeffvarrep=0.4,
   pre.coeffvarind=0.3, pre.coeffvargroup=0.7, post.coeffvarrep=0.4,
   post.coeffvarind=0.3, post.coeffvargroup=0.7, true.sample=FALSE,
   lower.limit=0, upper.limit=90, maxiterations=1000000,
   precision=2, confidence = 0.99, feedback=FALSE)
```

### Arguments

`meanepg` the mean pre-treatment egg count of the group (in EPG). If this is unknown then use a value likely to be encountered, or iterate over a distribution of values to obtain a distribution of values for power.

`reduction` the true mean egg count reduction (in %). If this is unknown then use a value likely to be encountered, or iterate over a distribution of values to obtain a distribution of values for power.

`g.faeces` the number of grams of faeces used per sample (must be the same for all samples both pre and post treatment).

`sensitivity` the minimum egg detection threshold or counting sensitivity of the technique used. If using the McMasters technique, this is the number of McMasters chambers counted divided by 50.

`replicates` the number of different samples (individually analysed) taken from each animal. Must be the same for all animals, both pre and post treatment. This would normally be 1, but increasing this provides an effective way of improving power at the cost of additional laboratory work.

`animals` the number of animals in the group. Can be 1, in which case the ability to predict the true mean reduction for the animal (if true.sample=TRUE) or the true mean reduction for a group from which the animal is taken (if true.sample=FALSE) can be calculated. The same number of animals MUST be sample pre and post treatment.

`pre.coeffvarrep`

coefficient of variation between sub-samples taken from the same faecal sample, assuming 1g faeces is used per sub-sample (the effective value is automatically adjusted according to g.faeces in the function). The default value for this is taken from an intensive study of a small group of horses, but may not be accurate in other groups of animals. This value is applied only to the pre-treatment distributions.

`pre.coeffvarind`

coefficient of variation between samples taken from different faecal piles from the same animal over a period of days, not including that due to coeffvarrep. The default value for this is taken from an intensive study of a small group of horses, but may not be accurate in other groups of animals. This value is applied only to the pre-treatment distributions.

`pre.coeffvargroup`

coefficient of variation between animals. This includes the variability between samples taken from different animals within a group, not including that due to within animal variability. The default value for this is inferred from a combination of several equine datasets and an intensive study of a small group of horses, but may not be accurate in other groups of animals. This value is applied only to the pre-treatment distributions.

`post.coeffvarrep`

coefficient of variation between sub-samples taken from the same faecal sample, assuming 1g faeces is used per sub-sample (the effective value is automatically adjusted according to g.faeces in the function). The default value for this is taken from an intensive study of a small group of horses, but may not be accurate in other groups of animals. This value is applied only to the post-treatment distributions.

`post.coeffvarind`

coefficient of variation between samples taken from different faecal piles from the same animal over a period of days, not including that due to coeffvarrep. The default value for this is taken from an intensive study of a small group of horses, but may not be accurate in other groups of animals. This value is applied only to the post-treatment distributions.

`post.coeffvargroup`

coefficient of variation between animals. This includes the variability between samples taken from different animals within a group, not including that due to within animal variability. The default value for this is inferred from a combination of several equine datasets and an intensive study of a small group of horses, but may not be accurate in other groups of animals. This value is applied only to the post-treatment distributions.

`true.sample` option to calculate the power for the population mean reduction (if true.sample=FALSE) or the true sample mean reduction (if true.sample=TRUE). The difference is that conceptually the true mean of a small group of animals may not reflect the mean of the population from which they are derived. If only one animal is considered, the true sample mean is the true mean of the individual whereas the population mean is the mean of a (theoretical) group of animals from which it is derived. The power will always be greater (or identical) for the true sample mean.

`lower.limit` lower limit for the tolerance (in %).

`upper.limit` lower limit for the tolerance (in %).

`maxiterations` the maximum number of iterations to use for the Monte Carlo integration, or the number to use if precision=NA. If precision is defined, then only the number of iterations required to estimate the power to the given precision are performed (up to a maximum of maxiterations).

`precision` the number of decimal places with which to calculate the power, unless maxiterations is reached first. A larger precision will give a more precise estimate of the true power but will take longer to calculate. Specifying this as NA performs the calculation on a fixed (=maxiterations) number of iterations.

`confidence` the degree of confidence required with which to report confidence limits for the true power when using Monte Carlo integration to report the power.

`feedback` option to display a progress indicator for calculation of the values used for Monte Carlo integration. Using feedback with some GUI versions of R may slow down the analysis considerably.

## Value

Returns a list containing the elements ’roundedci’ and ’ci’, which specifies the median and confidence limits (as defined by ’confidence’) for the true power both rounded by ’precison’ and unrounded. The additional variables ’within’ ’without’ and ’total’ are also returned, and indicate the number of iterations for which the observed mean reduction fell outside and inside the specified limits and the total number of iterations.

## See Also

`fecrt.precision`, `fec.analysis`

## `fecrt.precision` *FECRT Predicted Precision Calculations*

## Description

Finds the appropriate tolerance with which to consider the observed mean for a faecal egg count reduction test with the given combination of power and other parameters. The precision is calculated using Monte Carlo integration, and confidence intervals for the true power are produced. Tolerance can be defined as either a lower limit only if upper.limit is defined, as an upper limit only if lower.limit is defined, or as both (equidistant) limits if neither are defined. This function can be used for example to determine the upper limit for a true reduction of 80% by setting lower.limit=0, or to determine the lower limit for a true reduction of 99% by setting upper.limit=100.

## Usage

```
fecrt.precision(meanepg=200, reduction = 95, g.faeces=3,
   sensitivity=1/25, replicates=1, animals=10, pre.coeffvarrep=0.4,
   pre.coeffvarind=0.3, pre.coeffvargroup=0.7, post.coeffvarrep=0.4,
   post.coeffvarind=0.3, post.coeffvargroup=0.7, true.sample=FALSE,
   lower.limit=NA, upper.limit=NA, iterations=100000, power = 0.95,
   confidence = 0.99, feedback=FALSE)
```

## Arguments

`meanepg` the mean pre-treatment egg count of the group (in EPG). If this is unknown then use a value likely to be encountered, or iterate over a distribution of values to obtain a distribution of values for tolerance.

`reduction` the true mean egg count reduction (in %). If this is unknown then use a value likely to be encountered, or iterate over a distribution of values to obtain a distribution of values for tolerance.

`g.faeces` the number of grams of faeces used per sample (must be the same for all samples both pre and post treatment).

`sensitivity` the minimum egg detection threshold or counting sensitivity of the technique used. If using the McMasters technique, this is the number of McMasters chambers counted divided by 50.

`replicates` the number of different samples (individually analysed) taken from each animal. Must be the same for all animals, both pre and post treatment. This would normally be 1, but increasing this provides an effective way of improving power at the cost of additional laboratory work.

`animals` the number of animals in the group. Can be 1, in which case the ability to predict the true mean reduction for the animal (if true.sample=TRUE) or the true mean reduction for a group from which the animal is taken (if true.sample=FALSE) can be calculated. The same number of animals MUST be sample pre and post treatment.

`pre.coeffvarrep`
coefficient of variation between sub-samples taken from the same faecal sample, assuming 1g faeces is used per sub-sample (the effective value is automatically adjusted according to g.faeces in the function). The default value for this is taken from an intensive study of a small group of horses, but may not be accurate in other groups of animals. This value is applied only to the pre-treatment distributions.

`pre.coeffvarind`
: coefficient of variation between samples taken from different faecal piles from the same animal over a period of days, not including that due to coeffvarrep. The default value for this is taken from an intensive study of a small group of horses, but may not be accurate in other groups of animals. This value is applied only to the pre-treatment distributions.

`pre.coeffvargroup`
: coefficient of variation between animals. This includes the variability between samples taken from different animals within a group, not including that due to within animal variability. The default value for this is inferred from a combination of several equine datasets and an intensive study of a small group of horses, but may not be accurate in other groups of animals. This value is applied only to the pre-treatment distributions.

`post.coeffvarrep`
: coefficient of variation between sub-samples taken from the same faecal sample, assuming 1g faeces is used per sub-sample (the effective value is automatically adjusted according to g.faeces in the function). The default value for this is taken from an intensive study of a small group of horses, but may not be accurate in other groups of animals. This value is applied only to the post-treatment distributions.

`post.coeffvarind`
: coefficient of variation between samples taken from different faecal piles from the same animal over a period of days, not including that due to coeffvarrep. The default value for this is taken from an intensive study of a small group of horses, but may not be accurate in other groups of animals. This value is applied only to the post-treatment distributions.

`post.coeffvargroup`
: coefficient of variation between animals. This includes the variability between samples taken from different animals within a group, not including that due to within animal variability. The default value for this is inferred from a combination of several equine datasets and an intensive study of a small group of horses, but may not be accurate in other groups of animals. This value is applied only to the post-treatment distributions.

`true.sample`
: option to calculate the power for the population mean (if true.sample=FALSE) or the true sample mean (if true.sample=TRUE). The difference is that conceptually the true mean of a small group of animals may not reflect the mean of the population from which they are derived. If only one animal is considered, the true sample mean is the true mean of the individual whereas the population mean is the mean of a (theoretical) group of animals from which it is derived. The power will always be greater (or identical) for the true sample mean.

`lower.limit`
: (optional) lower limit for the tolerance (in %). If this is supplied, the required upper limit to maintain the desired power with the specified lower limit will be calculated. If neither lower limit or upper limit are specified, symmetrical limits about the true mean reduction are found that satisfy the power condition.

`upper.limit`
: (optional) upper limit for the tolerance (in %). If this is supplied, the required lower limit to maintain the desired power with the specified upper limit will be calculated. If neither lower limit or upper limit are specified, symmetrical limits about the true mean reduction are found that satisfy the power condition.

`iterations` the number of iterations to use for the Monte Carlo integration. More iterations will take longer but provide a smaller confidence interval for the true power.

`power` the desired power with which to interpret the given tolerances. Default is 0.95, so that the observed mean FEC will lie within the calculated lower and upper limits 95% of the time.

`confidence` the degree of confidence required with which to report confidence limits for the true power when using Monte Carlo integration to report the power. Only used for calculating the true power after determining the tolerance limits.

`feedback` option to display a progress indicator for calculation of the values used for Monte Carlo integration. Using feedback with some GUI versions of R may slow down the analysis considerably.

### Value

Returns a list containing the elements 'limits', which is the calculated lower and upper tolerance level which provides the required power, and 'power' which specifies the median estimate and confidence intervals for the true power. The true power returned may not exactly match the required power input due to the integer nature of FECRT data.

### See Also

`fecrt.power`, `fecrt.analysis`

## `likelihood` *Calculate the (Log) Likelihood of Obtaining Data from a Distribution*

### Description

Function to calculate the (log) likelihood of obtaining a set of data from one of the following distributions: Poisson (P), gamma (G), lognormal (L), Weibull (W), gamma Poisson (GP), lognormal Poisson (LP), Weibull Poisson (WP), all with or without zero-inflation (ZI). For mixture models, the likelihood is calculated for the data by integrating over each possible value of lambda for each data point, which may take some time for large datasets.

* NOTE: THIS FUNCTION IS DEPRECATED AND WILL BE REMOVED FROM BAYESCOUNT VERSION 1 *

### Usage

```
likelihood(model, data, mean=NA, variance=NA,
   zi=NA, shape=NA, scale=NA,
   iterations=min(1000, (length(as.matrix(mean)[,1])*length(shape))),
   log=TRUE, silent=FALSE, raw.output=FALSE)
```

## Arguments

`model` the distribution to fit to the data. Choices are: ’IP’, ’P’, ’ZIP’, ’G’, ’ZIG’, ’L’, ’ZIL’, ’W’, ’ZIW’, ’GP’, ’ZIGP’, ’LP’, ’ZILP’, ’WP’, ’ZIWP’ (case insensitive). No default.

`data` a vector of data to fit the distribution to. Count data for the count models, continuous data for the continuous distributions.

`mean` a vector of mean values over which to calculate the likelihood. Must be a matrix with number of columns equal to the number of datapoints for the IP model only, since using the IP model each datapoint has an independant mean (this is effectively a shortcut for repeating the likelihood function for each datapoint). Mean is required for the IP, (ZI)P, (ZI)L, and (ZI)L models. Otherwise ignored (with a warning if silent=FALSE).

`variance` a vector of values for variance over which to calculate the likelihood, with length equal to length of mean. Required for the (ZI)L, and (ZI)L models. Otherwise ignored (with a warning if silent=FALSE).

`zi` a vector of values for zero-inflation, with length equal to either scale/shape or mean/variance as appropriate. Required for zero-inflated models only; if supplied for other models the appropriate zero-inflated model will be used instead (with a warning if silent=FALSE).

`shape` a vector of values for the shape parameter. Must be of length equal to that of the scale parameter. Required for the (ZI)W, (ZI)WP, (ZI)G, and (ZI)GP models. Otherwise ignored (with a warning if silent=FALSE).

`scale` a vector of values for the scale parameter. Must be of length equal to that of the shape parameter. Required for the (ZI)W, (ZI)WP, (ZI)G, and (ZI)GP models. Otherwise ignored (with a warning if silent=FALSE).

`iterations` the total number of iterative samples for which to calculate the likelihood. If this is smaller than the number of iterations provided for mean/variance/shape/scale, then the chain will be thinned to the number of iterations required (this is useful for the mixture models where integrating the likelihood for each datapoint may take some time). Default is 1000 iterations, or the length of the parameters supplied if this is shorter than 1000.

`log` choose to output the log likelihood (TRUE) or the likelihood (FALSE). Default TRUE.

`silent` should warning messages and progress indicators be supressed? Default FALSE.

`raw.output` output a vector of length equal to the number of iterations representing the likelihood at each iteration if TRUE, or a summary of the results (median and 95 percent confidence interval) if FALSE. Default FALSE.

## Value

The (log) likelihood is returned either as a vector of length equal to the number of iterations representing the likelihood at each iteration if raw.output==TRUE, or a summary of the results (median, maximum and 95% highest posterior density interval (see `HPDinterval`) if raw.output==FALSE. It is possible for the functions that use integrate to produce incalculable probabilities for some iterations, in which case the likelihood for these iterations are returned as missing data (and therefore the summary statistics for the whole simulation are also returned as missing data). If only 1 iteration of values for mean/variance/shape/scale are provided, a single value for likelihood is output.

### See Also

count.analysis

### Examples

```
# calculate the likelihood of obtaining a set of count data from
# a zero-inflated Poisson distribution with set mean and zero-inflation
# values

data <- rpois(100, 10)
data[1:15] <- 0

likelihood('ZIP', data, mean=10, zi=15)

# now calculate the likelihood for the same data using an MCMC object
# to provide the values for mean and zero-inflation

## Not run:
values <- bayescount.single(data, model='ZISP', raw.output=TRUE)$mcmc
means <- c(values[,'mean'][[1]], values[,'mean'][[2]])
zis <- (1-c(values[,'prob'][[1]], values[,'prob'][[2]]))*100
# The function outputs the prevalence of disease when raw.ouput is
# TRUE, so zero-inflation must be calculated from this

likes <- likelihood('ZIP', data, mean=means, zi=zis,
raw.output=TRUE)$likelihood
hist(likes, breaks='fd', col='red')

## End(Not run)
```

## lnormal.params *Calculate the Log-Normal Mean and Standard Deviation Using the Normal Mean and Standard Deviation*

### Description

Function to calculate the equivalent values for the mean and standard deviation of a log-normal distribution from the mean and standard deviation of the normal distribution. Outputs from this function can be used with the dlnorm() function, and with the lognormal distribution in JAGS.

### Usage

```
lnormal.params(mean, sd, coeff.variation)
```

### Arguments

| | |
|---|---|
| `mean` | either a single value or vector of values for the mean of the normal distribution. |
| `sd` | either a single value or vector of values for the standard deviation of the normal distribution. Ignored if values are supplied for coeff.variation. |
| `coeff.variation` | either a single value or vector of values for the coefficient of dispersion. |

### Value

A list with elements representing the mean of the lognormal distribution, the standard deviation of the lognormal distribution, and the coefficient of variation.

### See Also

`normal.params`

### Examples

```
## Not run:
mean <- 10
sd <- 2
lmean <- lnormal.params(mean,sd)[[1]]
lsd <- lnormal.params(mean,sd)[[2]]

curve(dnorm(x, mean, sd), from=0, to=20)
dev.new()
curve(dlnorm(x, lmean, lsd), from=0, to=20)

## End(Not run)
```

---

## `maximise.likelihood` *Calculate the Maximum Likelihood Parameters of a Continuous or Count Distribution*

---

### Description

Crude function to maximise the likelihood of one of the following distributions: Poisson (P), gamma (G), lognormal (L), Weibull (W), gamma Poisson (GP), lognormal Poisson (LP), Weibull Poisson (WP), all with or without zero-inflation (ZI). Uses the likelihood() function to calculate the likelihood at each iteration. For mixture models, the likelihood is calculated for the data by integrating over each possible value of lambda for each data point, which may take some time for large datasets. Starting values for each parameter are optional, but may improve the speed and reliability of the function if appropriate values are provided. If missing, starting values will be calculated from the data. This function is provided for interest only, and is vastly inferior as a data analysis tool to the Bayesian MCMC methods used by bayescount().

* NOTE: THIS FUNCTION IS DEPRECATED AND WILL BE REMOVED FROM BAYESCOUNT VERSION 1 *

See `count.model` for an alternative function using MCMC

## Usage

```
maximise.likelihood(data=stop("Data must be specified"),
model=stop("Please specify a distribution"), mean=NA,
variance=NA, zi=NA, shape=NA, scale=NA, silent=FALSE)
```

## Arguments

`data` a vector of data to fit the distribution to. Count data for the count models, continuous data for the continuous distributions.

`model` the distribution to fit to the data. Choices are: "P", "ZIP", "G", "ZIG", "L", "ZIL", "W", "ZIW", "GP", "ZIGP", "LP", "ZILP", "WP", "ZIWP" (case insensitive). No default.

`mean` the starting value for mean. Optional.

`variance` the starting value for variance. Optional.

`zi` the starting value for zero-inflation. Optional.

`shape` the starting value for the shape parameter. Optional.

`scale` the starting value for the scale parameter. Optional.

`silent` should warning messages and progress indicators be supressed? Default FALSE.

## Value

The values for each parameter at the maximum likelihood are output. No standard error is given (Bayesian MCMC using bayescount() gives a much better analysis)

## See Also

`count.analysis`

## Examples

```

# obtain values for mean and zero-inflation of a zero-inflated
# gamma Poisson model:

data <- rpois(100, rgamma(100, shape=1, scale=8))
data[1:15] <- 0

maximise.likelihood(data, "ZIGP")
```

## normal.params *Calculate the Normal Mean and Standard Deviation Using the Log-Normal Mean and Standard Deviation*

### Description

Function to calculate the equivalent values for the mean and standard deviation of a normal distribution from the mean and standard deviation of the log-normal distribution. Outputs from this function can be used with the dnorm() function, and with the normal distribution in JAGS.

### Usage

```
normal.params(log.mean, log.sd, coeff.variation=sqrt(exp(log.sd^2)-1))
```

### Arguments

| | |
|---|---|
| `log.mean` | either a single value or vector of values for the mean of the lognormal distribution. |
| `log.sd` | either a single value or vector of values for the standard deviation of the lognormal distribution. Ignored if values are supplied for coeff.variation. |
| `coeff.variation` | either a single value or vector of values for the coefficient of dispersion. |

### Value

A list with elements representing the mean of the normal distribution, the standard deviation of the normal distribution, and the coefficient of variation.

### See Also

`lnormal.params`

### Examples

```
## Not run:
lmean <- 2.5
lsd <- 0.2
mean <- normal.params(lmean,lsd)[[1]]
sd <- normal.params(lmean,lsd)[[2]]

curve(dlnorm(x, lmean, lsd), from=0, to=25)
dev.new()
curve(dnorm(x, mean, sd), from=0, to=25)

## End(Not run)
```

# Index