Package ‘EmpiricalCalibration’

December 21, 2023

Type Package
Title Routines for Performing Empirical Calibration of Observational Study Estimates
Version 3.1.2
Date 2023-12-21
Maintainer Martijn Schuemie <schuemie@ohdsi.org>
Description Routines for performing empirical calibration of observational study estimates. By using a set of negative control hypotheses we can estimate the empirical null distribution of a particular observational study setup. This empirical null distribution can be used to compute a calibrated p-value, which reflects the probability of observing an estimated effect size when the null hypothesis is true taking both random and systematic error into account. A similar approach can be used to calibrate confidence intervals, using both negative and positive controls. For more details, see Schuemie et al. (2013) <doi:10.1002/sim.5925> and Schuemie et al. (2018) <doi:10.1073/pnas.1708282114>.

VignetteBuilder knitr
Depends R (>= 3.5.0)
Imports ggplot2 (>= 2.0.0), gridExtra, methods, rlang, Rcpp
Suggests knitr, markdown, rmarkdown, testthat, Cyclops, survival, Sequential
License Apache License 2.0
LinkingTo Rcpp
NeedsCompilation yes

BugReports https://github.com/OHDSI/EmpiricalCalibration/issues
RoxygenNote 7.2.3
Encoding UTF-8
Author  Martijn Schuemie [aut, cre] (<https://orcid.org/0000-0002-0817-5361>),
Marc Suchard [aut] (<https://orcid.org/0000-0001-9818-479X>)

Repository  CRAN

Date/Publication  2023-12-21 16:10:02 UTC

R topics documented:

calibrateConfidenceInterval ............................................... 3
calibrateLlr ................................................................. 4
calibrateP ................................................................. 4
caseControl ............................................................... 6
cohortMethod ............................................................ 7
compareEase ............................................................. 8
computeCvBinomial ...................................................... 9
computeCvPoisson .......................................................... 10
computeCvPoissonRegression ............................................ 11
computeExpectedAbsoluteSystematicError ................................ 12
computeTraditionalCi .................................................. 13
computeTraditionalIP ................................................... 14
convertNullToErrorModel ............................................... 15
evaluateCiCalibration .................................................... 16
fitMcmcNull .............................................................. 17
fitNull ................................................................. 18
fitNullNonNormalLl ..................................................... 19
fitSystematicErrorModel ................................................. 19
grahamReplication ...................................................... 21
plotCalibration .......................................................... 22
plotCalibrationEffect ................................................... 23
plotCiCalibration ....................................................... 24
plotCiCalibrationEffect .................................................. 26
plotCiCoverage .......................................................... 27
plotErrorModel .......................................................... 28
plotExpectedType1Error .................................................. 29
plotForest ............................................................... 31
plotMcmcTrace ........................................................... 32
plotTrueAndObserved .................................................... 32
sccs ................................................................. 33
simulateControls .......................................................... 34
simulateMaxSpRtData ..................................................... 35
southworthReplication .................................................. 36

Index  38
calibrateConfidenceInterval

Calibrate confidence intervals

Description

Calibrate confidence intervals

Usage

calibrateConfidenceInterval(logRr, seLogRr, model, ciWidth = 0.95)

Arguments

logRr A numeric vector of effect estimates on the log scale.

seLogRr The standard error of the log of the effect estimates. Hint: often the standard error = \(\frac{\log(<\text{lower bound 95 percent confidence interval}>)}{qnorm(0.025)}\).

model An object of type systematicErrorModel as created by the fitSystematicErrorModel function.

ciWidth The width of the confidence interval. Typically this would be .95, for the 95 percent confidence interval.

Details

Compute calibrated confidence intervals based on a model of the systematic error.

Value

A data frame with calibrated confidence intervals and point estimates.

Examples

data <- simulateControls(n = 50 * 3, mean = 0.25, sd = 0.25, trueLogRr = log(c(1, 2, 4)))
model <- fitSystematicErrorModel(data$logRr, data$seLogRr, data$trueLogRr)
newData <- simulateControls(n = 15, mean = 0.25, sd = 0.25, trueLogRr = log(c(1, 2, 4)))
result <- calibrateConfidenceInterval(newData$logRr, newData$seLogRr, model)
result
calibrateLlr

*Calibrate the log likelihood ratio*

**Description**

calibrateLlr computes calibrated log likelihood ratio using the fitted null distribution

**Usage**

calibrateLlr(null, likelihoodApproximation, twoSided = FALSE, upper = TRUE)

**Arguments**

- `null`: An object of class `null` created using the `fitNull` function or an object of class `mcmcNull` created using the `fitMcmcNull` function.
- `likelihoodApproximation`: Either a data frame containing normal, skew-normal, or custom parametric likelihood approximations, or a list of (adaptive) grid likelihood profiles.
- `twoSided`: Compute two-sided (TRUE) or one-sided (FALSE) p-value?
- `upper`: If one-sided: compute p-value for upper (TRUE) or lower (FALSE) bound?

**Value**

The calibrated log likelihood ratio.

**Examples**

data(sccs)
negatives <- sccs[sccs$groundTruth == 0, ]
null <- fitNull(negatives$logRr, negatives$seLogRr)
positive <- sccs[sccs$groundTruth == 1, ]
calibrateLlr(null, positive)

calibrateP

*Calibrate the p-value*

**Description**

calibrateP computes calibrated p-values using the fitted null distribution
Usage

calibrateP(null, logRr, seLogRr, twoSided = TRUE, upper = TRUE, ...)

## S3 method for class 'null'
calibrateP(null, logRr, seLogRr, twoSided = TRUE, upper = TRUE, ...)

## S3 method for class 'mcmcNull'
calibrateP(
  null,
  logRr,
  seLogRr,
  twoSided = TRUE,
  upper = TRUE,
  pValueOnly,
  ...
)

Arguments

null An object of class null created using the fitNull function or an object of class mcmcNull created using the fitMcmcNull function.

logRr A numeric vector of one or more effect estimates on the log scale

seLogRr The standard error of the log of the effect estimates. Hint: often the standard error = (log(<lower bound 95 percent confidence interval>) - log(<effect estimate>))/qnorm(0.025)

twoSided Compute two-sided (TRUE) or one-sided (FALSE) p-value?

upper If one-sided: compute p-value for upper (TRUE) or lower (FALSE) bound?

... Any additional parameters (currently none).

pValueOnly If true, will return only the calibrated P-value itself, not the credible interval.

Details

This function computes a calibrated two-sided p-value as described in Schuemie et al (2014).

Value

The calibrated p-value.

Methods (by class)

- calibrateP(null): Computes the calibrated P-value using asymptotic assumptions.
- calibrateP(mcmcNull): Computes the calibrated P-value and 95 percent credible interval using Markov Chain Monte Carlo (MCMC).
References

Examples

data(sccs)
negatives <- sccs[sccs$groundTruth == 0, ]
null <- fitNull(negatives$logRr, negatives$seLogRr)
positive <- sccs[sccs$groundTruth == 1, ]
calibrateP(null, positive$logRr, positive$seLogRr)

---
caseControl

Odds ratios from a case-control design

Description
Odds ratios from a case-control design

Usage
data(caseControl)

Format
A data frame with 47 rows and 4 variables:

- **drugName**: Name of the drug
- **groundTruth**: Whether the drug is a positive (1) or negative (0) control
- **logRr**: The log of the incidence rate ratio
- **seLogRr**: The standard error of the log of the incidence rate ratio

Details
A dataset containing the odds ratios (and standard errors) produced using a case-control design. The outcome is upper GI bleeding, the drug of interest (groundTruth = 1) is sertraline. Also included are 46 negative control drugs, for which we believe there to be no causal relation with upper GI bleeding. We used a database of medical records from general practices in the USA, the General Electric (GE) Centricity database, which contains data on 11.2 million subjects. We restricted on study period (start of 1990 through November 2003), age requirements (18 years or older), available time prior to event (180 days), number of controls per case (6), and risk definition window (30 days following the prescription). Controls were matched on age and sex. Cases of upper GI bleeding were identified on the basis of the occurrence of ICD-9 diagnosis codes in the problem list. These codes pertain to esophageal, gastric, duodenal, peptic, and gastrojejunal ulceration, perforation, and hemorrhage, as well as gastritis and non-specific gastrointestinal hemorrhage. For more information on this set see Schuemie et al (2014).
References

dataset(cohortMethod)

Description
Relative risks from a new-user cohort design

Usage
data(cohortMethod)

Format
A data frame with 31 rows and 4 variables:

- **drugName**: Name of the drug
- **groundTruth**: Whether the drug is a positive (1) or negative (0) control
- **logRr**: The log of the incidence rate ratio
- **seLogRr**: The standard error of the log of the incidence rate ratio

Details
A dataset containing the relative risks (and standard errors) produced using a new-user cohort design. The outcome is acute liver injury, the drug of interest (groundTruth = 1) is Isoniazid. Also included are 30 negative control drugs, for which we believe there to be no causal relation with acute liver injury. We used the Thomson MarketScan Medicare Supplemental Beneficiaries database, which contains data on 4.6 million subjects. We selected two groups (cohorts): (1) all subjects exposed to isoniazid and (2) all subjects having the ailment for which isoniazid is indicated, in this case tuberculosis, and having received at least one drug that is not known to cause acute liver injury. We removed all subjects who belonged to both groups and subjects for which less than 180 days of observation time was available prior to their first exposure to the drug in question. Acute liver injury was identified on the basis of the occurrence of ICD-9-based diagnosis codes from inpatient and outpatient medical claims and was defined broadly on the basis of codes associated with hepatic dysfunction, as have been used in prior observational database studies. The time at risk was defined as the length of exposure + 30 days, and we determined whether subjects experienced an acute liver injury during their time at risk. Using propensity score stratification, the cohorts were divided over 20 strata, and an odds ratio over all strata was computed using a Mantel-Haenszel test. The propensity score was estimated using Bayesian logistic regression using all available drug, condition, and procedure covariates occurring in the 180 days prior to first exposure, in addition to age, sex, calendar year of first exposure, Charlson index, number of drugs, number of visit days, and number of procedures. For more information on this set see Schuemie et al (2014).
compareEase

Description
Compare EASE of correlated sets of estimates

Usage
```r
compareEase(
  logRr1, 
  seLogRr1, 
  logRr2, 
  seLogRr2, 
  alpha = 0.05, 
  sampleSize = 1000 
)
```

Arguments
- `logRr1`: A numeric vector of effect estimates generated using the first method on the log scale.
- `seLogRr1`: The standard error of the log of the effect estimates generated using the first method.
- `logRr2`: A numeric vector of effect estimates generated using the second method on the log scale.
- `seLogRr2`: The standard error of the log of the effect estimates generated using the second method.
- `alpha`: The expected type I error for computing confidence intervals and p-values.
- `sampleSize`: The number of samples in the bootstraps.

Details
Compare the expected absolute systematic error (EASE) of two sets of estimates for the same set of negative controls.

Important: the two sets of estimates (logRr1 + seLogRr1 and logRr2 + seLogRr2) should be in identical order, so that for example the first item in each vector corresponds to the same negative control.
Value

A data frame with 4 columns: the point estimate, confidence interval lower bound, and upper bound for the difference between EASE in the two sets of negative controls, and a p value against the null hypothesis that the EASE is the same for the two sets.

The data frame has two attributes: ease1 and ease2, providing the EASE estimates (and confidence intervals) for the two sets, computed using bootstrapping. Note that these estimates may somewhat different from those generated using `computeExpectedAbsoluteSystematicError`, because a different approach is used to compute the confidence interval. The approach used here will more closely align with the computation of the difference in EASE.

Examples

```r
# Simulate results of first method:
ncs1 <- simulateControls(n = 50)

# Simulate second method to be more biased:
ncs2 <- ncs1
ncs2$logRr <- ncs2$logRr + rnorm(nrow(ncs2), mean = 0.1, sd = 0.1)

delta <- compareEase(
  logRr1 = ncs1$logRr,
  seLogRr1 = ncs1$seLogRr,
  logRr2 = ncs2$logRr,
  seLogRr2 = ncs2$seLogRr
)
delta
attr(delta, "ease1")
attr(delta, "ease2")
```

Description

Obtains critical values for the group and continuous sequential MaxSPRT test with Binomial data, using a Wald type upper boundary, which is flat with respect to the likelihood ratio function, and with a pre-specified upper limit on the sample size.

It is often not possible to select a critical value that corresponds to the exact alpha specified. Instead, this function will select the least conservative critical value having an alpha below the one specified, so the sequential analysis is conservative.

This function is a re-implementation of the CV.Binomial function in the Sequential package, using Monte-Carlo.
computeCvBinomial

Usage

computeCvBinomial(
  groupSizes,  
  z,  
  minimumEvents = 1,  
  alpha = 0.05,  
  sampleSize = 1e+06,  
  nullMean = 0,  
  nullSd = 0
)

Arguments

groupSizes    Vector containing the expected number of events under H0 for each test.
z            For a matched case-control analysis, z is the number of controls matched to each case under the null hypothesis. For a self-controlled analysis, z is the control time divided by the time at risk.
minimumEvents The minimum number of events needed before the null hypothesis can be rejected.
alpha         The significance level, or the type 1 error probability, which is the probability of rejecting the null hypothesis when it is true.
sampleSize    Sample size for the Monte-Carlo simulations.
nullMean      The mean of the empirical null distribution.
nullSd        The standard deviation of the empirical null distribution.

Value

The computed critical value. The 'alpha' attribute of the result indicates the selected alpha.

Examples

    groupSizes <- rep(1, 10)
    computeCvBinomial(groupSizes, z = 4)

computeCvPoisson Compute critical values for Poisson data

Description

Obtains critical values for the group and continuous sequential MaxSPRT test with Poisson data, using a Wald type upper boundary, which is flat with respect to the likelihood ratio function, and with a pre-specified upper limit on the sample size.

It is often not possible to select a critical value that corresponds to the exact alpha specified. Instead, this function will select the least conservative critical value having an alpha below the one specified, so the sequential analysis is conservative.
This function is a re-implementation of the CV.Poisson function in the Sequential package, using Monte-Carlo.

Usage

```r
computeCvPoisson(
  groupSizes,
  minimumEvents = 1,
  alpha = 0.05,
  sampleSize = 1e+06,
  nullMean = 0,
  nullSd = 0
)
```

Arguments

- `groupSizes`: Vector containing the expected number of events under H0 for each test.
- `minimumEvents`: The minimum number of events needed before the null hypothesis can be rejected.
- `alpha`: The significance level, or the type 1 error probability, which is the probability of rejecting the null hypothesis when it is true.
- `sampleSize`: Sample size for the Monte-Carlo simulations.
- `nullMean`: The mean of the empirical null distribution.
- `nullSd`: The standard deviation of the empirical null distribution.

Value

The computed critical value. The 'alpha' attribute of the result indicates the selected alpha.

Examples

```r
groupSizes <- rep(1, 10)
computeCvPoisson(groupSizes)
```

Description

Obtains critical values for the group and continuous sequential MaxSPRT test with Poisson regression data, using a Wald type upper boundary, which is flat with respect to the likelihood ratio function, and with a pre-specified upper limit on the sample size.

It is often not possible to select a critical value that corresponds to the exact alpha specified. Instead, this function will select the least conservative critical value having an alpha below the one specified, so the sequential analysis is conservative.
Usage

computeCvPoissonRegression(
    groupSizes,
    z,
    minimumEvents = 1,
    alpha = 0.05,
    sampleSize = 1e+06
)

Arguments

groupSizes Vector containing the expected number of events under H0 for each test.
z The control time divided by the time at risk.
minimumEvents The minimum number of events needed before the null hypothesis can be rejected.
alpha The significance level, or the type 1 error probability, which is the probability of rejecting the null hypothesis when it is true.
sampleSize Sample size for the Monte-Carlo simulations.

Value

The computed critical value. The 'alpha' attribute of the result indicates the selected alpha.

Examples

groupSizes <- rep(1, 10)
computeCvPoissonRegression(groupSizes, z = 4)

computeExpectedAbsoluteSystematicError

Compute the expected absolute systematic error

Description

For a random study estimate, what is the expected value of the absolute systematic error? Provides a single summary value for a null distribution. The expected systematic error of a null distribution is equal to its mean (mu), and is insensitive to the spread of the null distribution (sigma).

Taking the absolute value of the expected systematic error we can express both mean and spread of the estimated null distribution.

Usage

computeExpectedAbsoluteSystematicError(null, alpha = 0.05)
Arguments

null
An object of class null created using the fitNull function or an object of class mcmcNull created using the fitMcmcNull function.

alpha
The expected type I error for computing the credible interval.

Value

The expected absolute systematic error. If the provided null argument is of type mcmcNull, the credible interval (defined by alpha) is also returned.

See Also

compareEase for comparing the expected absolute systematic error of two sets of estimates for the same negative controls.

Examples

data(sccs)
negatives <- sccs[sccs$groundTruth == 0, ]
null <- fitNull(negatives$logRr, negatives$seLogRr)
computeExpectedAbsoluteSystematicError(null)

computeTraditionalCi

Compute the (traditional) confidence interval

Description

computeTraditionalCi computes the traditional confidence interval based on the log of the relative risk and the standard error of the log of the relative risk.

Usage

computeTraditionalCi(logRr, seLogRr, ciWidth = 0.95)

Arguments

logRr
A numeric vector of one or more effect estimates on the log scale

seLogRr
The standard error of the log of the effect estimates. Hint: often the standard error = \(\frac{\log(<\text{lower bound 95 percent confidence interval}>) - \log(<\text{effect estimate}>)}{\text{qnorm}(0.025)}\)

ciWidth
The width of the confidence interval. Typically this would be .95, for the 95 percent confidence interval.

Value

The point estimate and confidence interval
computeTraditionalP

Compute the (traditional) p-value

Description

computeTraditionalP computes the traditional two-sided p-value based on the log of the relative risk and the standard error of the log of the relative risk.

Usage

computeTraditionalP(logRr, seLogRr, twoSided = TRUE, upper = TRUE)

Arguments

logRr A numeric vector of one or more effect estimates on the log scale
seLogRr The standard error of the log of the effect estimates. Hint: often the standard error = (log(<lower bound 95 percent confidence interval>) - log(<effect estimate>))/qnorm(0.025)
twoSided Compute two-sided (TRUE) or one-sided (FALSE) p-value?
upper If one-sided: compute p-value for upper (TRUE) or lower (FALSE) bound?

Value

The (traditional) p-value.

Examples

data(sccs)
positive <- sccs[sccs$groundTruth == 1, ]
computeTraditionalP(positive$logRr, positive$seLogRr)
convertNullToErrorModel

Convert empirical null distribution to systematic error model

Description

This function converts an empirical null distribution, fitted using estimates only for negative controls, into a systematic error distribution that can be used to calibrate confidence intervals in addition to p-values.

Whereas the `fitSystematicErrorModel` uses positive controls to determine how the error distribution changes with true effect size, this function requires the user to make an assumption. The default assumption, `meanSlope = 1` and `sdSlope = 0`, specify a belief that the error distribution is the same for all true effect sizes. In many cases this assumption is likely to be correct, however, if an estimation method is biased towards the null this assumption will be violated, causing the calibrated confidence intervals to have lower than nominal coverage.

Usage

```r
convertNullToErrorModel(null, meanSlope = 1, sdSlope = 0)
```

Arguments

- `null` The empirical null distribution fitted using either the `fitNull` or the `fitMcmcNull` function.
- `meanSlope` The slope for the mean of the error distribution. A slope of 1 means the error is the same for different values of the true relative risk.
- `sdSlope` The slope for the log of the standard deviation of the error distribution. A slope of 0 means the standard deviation is the same for different values of the true relative risk.

Value

An object of type `systematicErrorModel`.

Examples

```r
data(sccs)
negatives <- sccs[sccs$groundTruth == 0, ]
null <- fitNull(negatives$logRr, negatives$seLogRr)
model <- convertNullToErrorModel(null)
positive <- sccs[sccs$groundTruth == 1, ]
calibrateConfidenceInterval(positive$logRr, positive$seLogRr, model)
```
evaluateCiCalibration \hspace{1em} Evaluate confidence interval calibration

**Description**

evaluateCiCalibration performs a leave-one-out cross-validation to evaluate the calibration confidence intervals.

**Usage**

evaluateCiCalibration(
  logRr,
  seLogRr,
  trueLogRr,
  strata = as.factor(trueLogRr),
  crossValidationGroup = 1:length(logRr),
  legacy = FALSE
)

**Arguments**

- **logRr**  
  A numeric vector of effect estimates on the log scale.

- **seLogRr**  
  The standard error of the log of the effect estimates. Hint: often the standard error = (log(<lower bound 95 percent confidence interval>) - log(<effect estimate>))/qnorm(0.025).

- **trueLogRr**  
  The true log relative risk.

- **strata**  
  Variable used to stratify the plot. Set strata = NULL for no stratification.

- **crossValidationGroup**  
  What should be the unit for the cross-validation? By default the unit is a single control, but a different grouping can be provided, for example linking a negative control to synthetic positive controls derived from that negative control.

- **legacy**  
  If true, a legacy error model will be fitted, meaning standard deviation is linear on the log scale. If false, standard deviation is assumed to be simply linear.

**Details**

The empirical calibration is performed using a leave-one-out design: The confidence interval of an effect is computed by fitting a null using all other controls.

**Value**

A data frame specifying the coverage per strata (usually true effect size) for a wide range of widths of the confidence interval. The result also includes the fraction of estimates that was below and above the confidence interval.
fitMcmcNull

Fit the null distribution using MCMC

Description
fitMcmcNull fits the null distribution to a set of negative controls using Markov Chain Monte Carlo (MCMC).

Usage
fitMcmcNull(logRr, seLogRr, iter = 1e+05)

Arguments
logRr A numeric vector of effect estimates on the log scale
seLogRr The standard error of the log of the effect estimates. Hint: often the standard error = (log(<lower bound 95 percent confidence interval>) - log(<effect estimate>))/qnorm(0.025)
iter Number of iterations of the MCMC.

Details
This is an experimental function for computing the 95 percent credible interval of a calibrated p-value using Markov-Chain Monte Carlo (MCMC).

Value
An object of type mcmcNull containing the mean and standard deviation (both on the log scale) of the null distribution, as well as the MCMC trace.

Examples
## Not run:
data(sccs)
negatives <- sccs[sccs$groundTruth == 0,]
null <- fitMcmcNull(negatives$logRr, negatives$seLogRr)
null
plotMcmcTrace(null)
positive <- sccs[sccs$groundTruth == 1,]
calibrateP(null, positive$logRr, positive$seLogRr)

## End(Not run)
fitNull

Fit the null distribution

Description

fitNull fits the null distribution to a set of negative controls.

Usage

fitNull(logRr, seLogRr)

Arguments

logRr A numeric vector of effect estimates on the log scale
seLogRr The standard error of the log of the effect estimates. Hint: often the standard error = (log(<lower bound 95 percent confidence interval>) - log(<effect estimate>))/qnorm(0.025)

Details

This function fits a Gaussian function to the negative control estimates as described in Schuemie et al (2014).

Value

An object containing the parameters of the null distribution.

References


Examples

data(sccs)
negatives <- sccs[sccs$groundTruth == 0, ]
null <- fitNull(negatives$logRr, negatives$seLogRr)
null
fitNullNonNormalLL

Fit the null distribution using non-normal log-likelihood approximations

Description

fitNullNonNormalLL fits the null distribution to a set of negative controls.

Usage

fitNullNonNormalLL(likelihoodApproximations)

Arguments

likelihoodApproximations
    Either a data frame containing normal, skew-normal, or custom parametric likelihood approximations, or a list of (adaptive) grid likelihood profiles.

Details

This function fits a Gaussian function to the negative control estimates, using non-normal approximations of the per-negative control log likelihood.

Value

An object containing the parameters of the null distribution.

Examples

```r
data(sccs)
negatives <- sccs[sccs$groundTruth == 0, ]
null <- fitNullNonNormalLL(negatives)
null
```

fitSystematicErrorModel

Fit a systematic error model

Description

Fit a systematic error model.
Usage

```r
fitSystematicErrorModel(
  logRr, 
  seLogRr, 
  trueLogRr, 
  estimateCovarianceMatrix = FALSE, 
  legacy = FALSE 
)
```

Arguments

- **logRr**: A numeric vector of effect estimates on the log scale.
- **seLogRr**: The standard error of the log of the effect estimates. Hint: often the standard error = \(\frac{\log(<\text{lower bound 95 percent confidence interval}>) - \log(<\text{effect estimate}>)}{\text{qnorm}(0.025)}\).
- **trueLogRr**: A vector of the true effect sizes.
- **estimateCovarianceMatrix**: Should a covariance matrix be computed? If so, confidence intervals for the model parameters will be available.
- **legacy**: If true, a legacy error model will be fitted, meaning standard deviation is linear on the log scale. If false, standard deviation is assumed to be simply linear.

Details

Fit a model of the systematic error as a function of true effect size. This model is an extension of the method for fitting the null distribution. The mean and log(standard deviations) of the error distributions are assumed to be linear with respect to the true effect size, and each component is therefore represented by an intercept and a slope.

Value

An object of type `systematicErrorModel`.

Examples

```r
controls <- simulateControls(n = 50 * 3, mean = 0.25, sd = 0.25, trueLogRr = log(c(1, 2, 4)))
model <- fitSystematicErrorModel(controls$logRr, controls$seLogRr, controls$trueLogRr)
model
```
Description

Relative risks from an adjusted new-user cohort design

Usage

data(grahamReplication)

Format

A data frame with 126 rows and 4 variables:

- outcomeName  Name of the outcome
- trueLogRr  The log of the true effect size. Only provided for negative and positive controls, is NA for the outcome of interest (GI bleeding).
- logRr  The log of the incidence rate ratio
- seLogRr  The standard error of the log of the incidence rate ratio

Details

A dataset containing the incidence rate ratios (and standard errors) produced using a new-user cohort design that compares new-users of dabigatran to new-users of warfarin for the outcome of GI hemorrhage. The dataset includes estimates both for the outcome of interest as well as negative and positive control outcomes. Subject are required to have 183 days of continuous observation prior to initiating treatment, be at least 65 years old at index date, and are required to have no prior exposure to warfarin or dabigatran (or any other novel anticoagulant). Furthermore, subjects are required to use the treatment for the indication of atrial fibrillation or atrial flutter, which is enforced by requiring a prior diagnosis of atrial fibrillation or flutter, and no prior diagnosis of other indications. Propensity scores are generated by fitting a model for predicting treatment assignment based on baseline patient characteristics, and are used to perform one-on-one matching. Hazard ratios are estimated through a Cox regression on the matched population. Time-at-risk is defined as starting on the day after initiating treatment and stopping when treatment is stopped, when the outcome occurs, or observation time ends, whichever comes first. The original study (Graham et al 2016) uses the Medicare database. For our replication, we use the Truven Medicare Supplementary Beneficiaries database. We analyze 15,796 dabigatran-exposed and 15,796 warfarin-exposed subjects. For more information on this set see Schuemie et al (2017).

References


---

**plotCalibration**

Create a calibration plot

**Description**

plotCalibration creates a plot showing the calibration of our calibration procedure

**Usage**

```r
plotCalibration(
  logRr,  
  seLogRr,  
  useMcmc = FALSE,  
  legendPosition = "right",  
  title,  
  fileName = NULL
)
```

**Arguments**

- `logRr` A numeric vector of effect estimates on the log scale
- `seLogRr` The standard error of the log of the effect estimates. Hint: often the standard error = \((\log(\text{lower bound 95 percent confidence interval}) - \log(\text{effect estimate}))/\text{qnorm}(0.025)\)
- `useMcmc` Use MCMC to estimate the calibrated P-value?
- `legendPosition` Where should the legend be positioned? ("none", "left", "right", "bottom", "top")
- `title` Optional: the main title for the plot
- `fileName` Name of the file where the plot should be saved, for example 'plot.png'. See the function \texttt{ggsave} in the \texttt{ggplot2} package for supported file formats.

**Details**

Creates a calibration plot showing the number of effects with \(p < \alpha\) for every level of \(\alpha\). The empirical calibration is performed using a leave-one-out design: The p-value of an effect is computed by fitting a null using all other negative controls. Ideally, the calibration line should approximate the diagonal. The plot shows both theoretical (traditional) and empirically calibrated p-values.

**Value**

A \texttt{Ggplot} object. Use the \texttt{ggsave} function to save to file.
Examples

```r
data(sccs)
negatives <- sccs[sccs$groundTruth == 0, ]
plotCalibration(negatives$logRr, negatives$seLogRr)
```

Description

`plotCalibrationEffect` creates a plot showing the effect of the calibration.

Usage

```r
plotCalibrationEffect(
  logRrNegatives,
  seLogRrNegatives,
  logRrPositives = NULL,
  seLogRrPositives = NULL,
  null = NULL,
  alpha = 0.05,
  xLabel = "Relative risk",
  title,
  showCis = FALSE,
  showExpectedAbsoluteSystematicError = FALSE,
  fileName = NULL,
  xLimits = c(0.25, 10),
  yLimits = c(0, 1.5)
)
```

Arguments

- `logRrNegatives` A numeric vector of effect estimates of the negative controls on the log scale.
- `seLogRrNegatives` The standard error of the log of the effect estimates of the negative controls.
- `logRrPositives` Optional: A numeric vector of effect estimates of the positive controls on the log scale.
- `seLogRrPositives` Optional: The standard error of the log of the effect estimates of the positive controls.
- `null` An object representing the fitted null distribution as created by the `fitNull` or `fitMcmcNull` functions. If not provided, a null will be fitted before plotting.
- `alpha` The alpha for the hypothesis test.
- `xLabel` The label on the x-axis: the name of the effect estimate.
plotCiCalibration

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>title</td>
<td>Optional: the main title for the plot</td>
</tr>
<tr>
<td>showCis</td>
<td>Show 95 percent credible intervals for the calibrated p = alpha boundary.</td>
</tr>
<tr>
<td>showExpectedAbsoluteSystematicError</td>
<td>Show the expected absolute systematic error. If null is of type mcmcNull the 95 percent credible interval will also be shown.</td>
</tr>
<tr>
<td>fileName</td>
<td>Name of the file where the plot should be saved, for example 'plot.png'. See the function ggsave in the ggplot2 package for supported file formats.</td>
</tr>
<tr>
<td>xLimits</td>
<td>Vector of length 2 for limits of the plot x axis - defaults to 0.25, 10</td>
</tr>
<tr>
<td>yLimits</td>
<td>Vector of length 2 for size limits of the y axis - defaults to 0, 1.5</td>
</tr>
</tbody>
</table>

Details

Creates a plot with the effect estimate on the x-axis and the standard error on the y-axis. Negative controls are shown as blue dots, positive controls as yellow diamonds. The area below the dashed line indicated estimates with p < 0.05. The orange area indicates estimates with calibrated p < 0.05.

Value

A Ggplot object. Use the ggsave function to save to file.

Examples

```r
data(sccs)
negatives <- sccs[sccs$groundTruth == 0, ]
positive <- sccs[sccs$groundTruth == 1, ]
plotCalibrationEffect(negatives$logRr, negatives$seLogRr, positive$logRr, positive$seLogRr)
```

plotCiCalibration

Create a confidence interval calibration plot

Description

plotCalibration creates a plot showing the calibration of our confidence interval calibration procedure

Usage

```r
plotCiCalibration(
  logRr,
  seLogRr,
  trueLogRr,
  strata = as.factor(trueLogRr),
  crossValidationGroup = 1:length(logRr),
  legacy = FALSE,
  evaluation,
  legendPosition = "top",
)```
plotCiCalibration

Arguments

logRr A numeric vector of effect estimates on the log scale.

seLogRr The standard error of the log of the effect estimates. Hint: often the standard error = (log(<lower bound 95 percent confidence interval>) - log(<effect estimate>))/qnorm(0.025).

trueLogRr The true log relative risk.

strata Variable used to stratify the plot. Set strata = NULL for no stratification.

crossValidationGroup What should be the unit for the cross-validation? By default the unit is a single control, but a different grouping can be provided, for example linking a negative control to synthetic positive controls derived from that negative control.

legacy If true, a legacy error model will be fitted, meaning standard deviation is linear on the log scale. If false, standard deviation is assumed to be simply linear.

evaluation A data frame as generated by the evaluateCiCalibration function. If provided, the logRr, seLogRr, trueLogRr, strata, and legacy arguments will be ignored.

legendPosition Where should the legend be positioned? ("none", "left", "right", "bottom", "top").

title Optional: the main title for the plot

fileName Name of the file where the plot should be saved, for example ’plot.png’. See the function ggsave in the ggplot2 package for supported file formats.

Details

Creates a calibration plot showing the fraction of effects within the confidence interval. The empirical calibration is performed using a leave-one-out design: The confidence interval of an effect is computed by fitting a null using all other controls. Ideally, the calibration line should approximate the diagonal. The plot shows the coverage for both theoretical (traditional) and empirically calibrated confidence intervals.

Value

A Ggplot object. Use the ggsave function to save to file.

Examples

```r
## Not run:
data <- simulateControls(n = 50 * 3, mean = 0.25, sd = 0.25, trueLogRr = log(c(1, 2, 4)))plotCiCalibration(data$logRr, data$seLogRr, data$trueLogRr)
## End(Not run)
```
plotCiCalibrationEffect

Plot the effect of the CI calibration

Description

Creates a plot with the effect estimate on the x-axis and the standard error on the y-axis. The plot is trellised by true effect size. Negative and positive controls are shown as blue dots. The area below the dashed line indicated estimates that are statistically significant different from the true effect size (p < 0.05). The orange area indicates estimates with calibrated p < 0.05.

Usage

plotCiCalibrationEffect(
  logRr,
  seLogRr,
  trueLogRr,
  legacy = FALSE,
  model = NULL,
  xLabel = "Relative risk",
  title,
  fileName = NULL
)

Arguments

logRr A numeric vector of effect estimates on the log scale.
seLogRr The standard error of the log of the effect estimates. Hint: often the standard error = (log(<lower bound 95 percent confidence interval>) - log(<effect estimate>))/qnorm(0.025).
trueLogRr A vector of the true effect sizes.
legacy If true, a legacy error model will be fitted, meaning standard deviation is linear on the log scale. If false, standard deviation is assumed to be simply linear.
model The fitted systematic error model. If not provided, it will be fitted on the provided data.
xLabel The label on the x-axis: the name of the effect estimate.
title Optional: the main title for the plot
fileName Name of the file where the plot should be saved, for example ’plot.png’. See the function ggsave in the ggplot2 package for supported file formats.

Value

A Ggplot object. Use the ggsave function to save to file.
Examples

data <- simulateControls(n = 50 * 3, mean = 0.25, sd = 0.25, trueLogRr = log(c(1, 2, 4)))
plotCiCalibrationEffect(data$logRr, data$seLogRr, data$trueLogRr)

plotCiCoverage

Create a confidence interval coverage plot

Description

plotCiCoverage creates a plot showing the coverage before and after confidence interval calibration at various widths of the confidence interval.

Usage

plotCiCoverage(
  logRr,
  seLogRr,
  trueLogRr,
  strata = as.factor(trueLogRr),
  crossValidationGroup = 1:length(logRr),
  legacy = FALSE,
  evaluation,
  legendPosition = "top",
  title,
  fileName = NULL
)

Arguments

logRr A numeric vector of effect estimates on the log scale.
seLogRr The standard error of the log of the effect estimates. Hint: often the standard error = (log(<lower bound 95 percent confidence interval>) - log(<effect estimate>))/qnorm(0.025).
trueLogRr The true log relative risk.
strata Variable used to stratify the plot. Set strata = NULL for no stratification.
crossValidationGroup What should be the unit for the cross-validation? By default the unit is a single control, but a different grouping can be provided, for example linking a negative control to synthetic positive controls derived from that negative control.
legacy If true, a legacy error model will be fitted, meaning standard deviation is linear on the log scale. If false, standard deviation is assumed to be simply linear.
evaluation A data frame as generated by the evaluateCiCalibration function. If provided, the logRr, seLogRr, trueLogRr, strata, and legacy arguments will be ignored.
plotErrorModel

legendPosition  Where should the legend be positioned? ("none", "left", "right", "bottom", "top").
title  Optional: the main title for the plot
fileName  Name of the file where the plot should be saved, for example 'plot.png'. See the function ggsave in the ggplot2 package for supported file formats.

Details

Creates a plot showing the fraction of effects above, within, and below the confidence interval. The empirical calibration is performed using a leave-one-out design: The confidence interval of an effect is computed by fitting a null using all other controls. The plot shows the coverage for both theoretical (traditional) and empirically calibrated confidence intervals.

Value

A Ggplot object. Use the ggsave function to save to file.

Examples

```r
## Not run:
data <- simulateControls(n = 50 * 3, mean = 0.25, sd = 0.25, trueLogRr = log(c(1, 2, 4)))
plotCiCoverage(data$logRr, data$seLogRr, data$trueLogRr)
## End(Not run)
```

plotErrorModel  *Plot the systematic error model*

Description

plotErrorModel creates a plot showing the systematic error model.

Usage

```r
plotErrorModel(
  logRr,
  seLogRr,
  trueLogRr,
  title,
  legacy = FALSE,
  fileName = NULL
)
```
plotExpectedType1Error

Arguments

logRr A numeric vector of effect estimates on the log scale.
seLogRr The standard error of the log of the effect estimates. Hint: often the standard error = (log(<lower bound 95 percent confidence interval>) - log(<effect estimate>))/qnorm(0.025).
trueLogRr The true log relative risk.
title Optional: the main title for the plot
legacy If true, a legacy error model will be fitted, meaning standard deviation is linear on the log scale. If false, standard deviation is assumed to be simply linear.
fileName Name of the file where the plot should be saved, for example ’plot.png’. See the function ggsave in the ggplot2 package for supported file formats.

Details

Creates a plot with the true effect size on the x-axis, and the mean plus and minus the standard deviation shown on the y-axis. Also shown are simple error models fitted at each true relative risk in the input.

Value

A Ggplot object. Use the ggsave function to save to file.

Examples

data <- simulateControls(n = 50 * 3, mean = 0.25, sd = 0.25, trueLogRr = log(c(1, 2, 4)))
plotErrorModel(data$logRr, data$seLogRr, data$trueLogRr)

plotExpectedType1Error

Plot the expected type 1 error as a function of standard error

Description

plotExpectedType1Error creates a plot showing the expected type 1 error as a function of standard error.

Usage

plotExpectedType1Error(
  logRrNegatives,
  seLogRrNegatives,
  seLogRrPositives,
  alpha = 0.05,
  null = NULL,
  xLabel = "Relative risk",
  title,
plotExpectedType1Error

showCis = FALSE,
showEffectSizes = FALSE,
fileName = NULL
)

Arguments

logRrNegatives  A numeric vector of effect estimates of the negative controls on the log scale.
seLogRrNegatives  The standard error of the log of the effect estimates of the negative controls.
seLogRrPositives  The standard error of the log of the effect estimates of the positive controls.
alpha  The alpha (nominal type 1 error) to be used.
null  An object representing the fitted null distribution as created by the fitNull function. If not provided, a null will be fitted before plotting.
xLabel  If showing effect sizes, what label should be used for the effect size axis?
title  Optional: the main title for the plot
showCis  Show 95 percent credible intervals for the expected type 1 error.
showEffectSizes  Show the expected effect sizes alongside the expected type 1 error?
fileName  Name of the file where the plot should be saved, for example 'plot.png'. See the function ggsave in the ggplot2 package for supported file formats.

Details

Creates a plot with the standard error on the x-axis and the expected type 1 error on the y-axis. The red line indicates the expected type 1 error given the estimated empirical null distribution if no calibration is performed. The dashed line indicated the nominal expected type 1 error rate, assuming the theoretical null distribution.

If standard errors are provided for non-negative estimates these will be plotted on the red line as yellow diamonds.

Value

A Ggplot object. Use the ggsave function to save to file.

Examples

data(sccs)
negatives <- sccs[sccs$groundTruth == 0, ]
positive <- sccs[sccs$groundTruth == 1, ]
plotExpectedType1Error(negatives$logRr, negatives$seLogRr, positive$seLogRr)
plotForest

Create a forest plot

Description

plotForest creates a forest plot of effect size estimates.

Usage

plotForest(
  logRr,
  seLogRr,
  names,
  xLabel = "Relative risk",
  title,
  fileName = NULL
)

Arguments

logRr A numeric vector of effect estimates on the log scale
seLogRr The standard error of the log of the effect estimates. Hint: often the standard error = (log(<lower bound 95 percent confidence interval>) - log(<effect estimate>))/qnorm(0.025)
names A vector containing the names of the drugs or outcomes
xLabel The label on the x-axis: the name of the effect estimate
title Optional: the main title for the plot
fileName Name of the file where the plot should be saved, for example 'plot.png'. See the function ggsave in the ggplot2 package for supported file formats.

Details

Creates a forest plot of effect size estimates (ratios). Estimates that are significantly different from 1 (alpha = 0.05) are marked in orange, others are marked in blue.

Value

A Ggplot object. Use the ggsave function to save to file.

Examples

data(sccs)
negatives <- sccs[sccs$groundTruth == 0, ]
plotForest(negatives$logRr, negatives$seLogRr, negatives$drugName)
plotMcmcTrace

Plot the MCMC trace

Description
Plot the MCMC trace

Usage
plotMcmcTrace(mcmcNull, fileName = NULL)

Arguments
mcmcNull An object of type mcmcNull as generated using the fitMcmcNull function.
fileName Name of the file where the plot should be saved, for example 'plot.png'. See the function ggsave in the ggplot2 package for supported file formats.

Details
Plot the trace of the MCMC for diagnostics purposes.

Examples
## Not run:
data(sccs)
negatives <- sccs[sccs$groundTruth == 0, ]
null <- fitMcmcNull(negatives$logRr, negatives$seLogRr)
plotMcmcTrace(null)
## End(Not run)

plotTrueAndObserved

Plot true and observed values

Description
Plot true and observed values, for example from a simulation study.

Usage
plotTrueAndObserved(
  logRr,
  seLogRr,
  trueLogRr,
  xLabel = "Relative risk",
  title,
  fileName = NULL
)
Arguments

- **logRr**: A numeric vector of effect estimates on the log scale.
- **seLogRr**: The standard error of the log of the effect estimates. Hint: often the standard error \(\frac{\ln(\text{lower bound 95 percent confidence interval}) - \ln(\text{effect estimate})}{\text{qnorm(0.025)}}\).
- **trueLogRr**: A vector of the true effect sizes.
- **xLabel**: The label on the x-axis: the name of the effect estimate.
- **title**: Optional: the main title for the plot.
- **fileName**: Name of the file where the plot should be saved, for example 'plot.png'. See the function `ggsave` in the ggplot2 package for supported file formats.

Details

Creates a forest plot of effect size estimates (ratios). Estimates that are significantly different from the true value (alpha = 0.05) are marked in orange, others are marked in blue.

Value

A Ggplot object. Use the `ggsave` function to save to file.

Examples

data <- simulateControls(n = 50 * 3, mean = 0.25, sd = 0.25, trueLogRr = log(c(1, 2, 4)))
plotTrueAndObserved(data$logRr, data$seLogRr, data$trueLogRr)

---

**scs**

*Incidence rate ratios from Self-Controlled Case Series*

data(sccs)

Description

Incidence rate ratios from Self-Controlled Case Series

Usage

data(sccs)

Format

A data frame with 46 rows and 4 variables:

- **drugName**: Name of the drug
- **groundTruth**: Whether the drug is a positive (1) or negative (0) control
- **logRr**: The log of the incidence rate ratio
- **seLogRr**: The standard error of the log of the incidence rate ratio
simulateControls

Details

A dataset containing the incidence rate ratios (and standard errors) produced using a Self-Controlled Case Series (SCCS) design. The outcome is upper GI bleeding, the drug of interest (groundTruth = 1) is sertraline. Also included are 45 negative control drugs, for which we believe there to be no causal relation with upper GI bleeding. We used a database of medical records from general practices in the USA, the General Electric (GE) Centricity database, which contains data on 11.2 million subjects. We restricted on study period (start of 1990 through November 2003), age requirements (18 years or older), available time prior to event (180 days), and risk definition window (30 days following the prescription). Time 30 days prior to the first prescription was removed to account for possible contra-indications. Cases of upper GI bleeding were identified on the basis of the occurrence of ICD-9 diagnosis codes in the problem list. These codes pertain to esophageal, gastric, duodenal, peptic, and gastrojejunal ulceration, perforation, and hemorrhage, as well as gastritis and non-specific gastrointestinal hemorrhage. For more information on this set see Schuemie et al (2014).

References


simulateControls  Simulate (negative) controls

Description

Simulate (negative) controls

Usage

simulateControls(n = 50, mean = 0, sd = 0.1, seLogRr = runif(n, min = 0.01, max = 0.2), trueLogRr = 0)

Arguments

n Number of controls to simulate.
mean The mean of the error distribution (on the log RR scale).
sd The standard deviation of the error distribution (on the log RR scale).
seLogRr The standard error of the log of the relative risk. This is recycled for the controls. The default is to sample these from a uniform distribution.
trueLogRr The true relative risk (on the log scale) used to generate these controls. This is recycled for the controls.
**simulateMaxSprtData**

Generate point estimates given known true effect sizes and standard errors

**Examples**

```r
data <- simulateControls(n = 50 * 3, mean = 0.25, sd = 0.25, trueLogRr = log(c(1, 2, 4)))
plotTrueAndObserved(data$logRr, data$seLogRr, data$trueLogRr)
```

---

**simulateMaxSprtData**  
*Simulate survival data for MaxSPRT computation*

**Description**

Simulate survival data for MaxSPRT computation

**Usage**

```r
simulateMaxSprtData(
  n = 10000,
  pExposure = 0.5,
  backgroundHazard = 0.001,
  tar = 10,
  nullMu = 0.2,
  nullSigma = 0.2,
  maxT = 100,
  looks = 10,
  numberOfNegativeControls = 50,
  numberOfPositiveControls = 1,
  positiveControlEffectSize = 4
)
```

**Arguments**

- `n`  
  Number of subjects.
- `pExposure`  
  Probability of being in target cohort.
- `backgroundHazard`  
  Background hazard (risk of the outcome per day).
- `tar`  
  Time at risk for each exposure.
- `nullMu`  
  Null distribution mean (at log HR scale).
- `nullSigma`  
  Null distribution SD (at log HR scale).
- `maxT`  
  Maximum time to simulate.
- `looks`  
  Number of (evenly spaced) looks at the data.
- `numberOfNegativeControls`  
  Number of negative controls to simulate.
numberOfPositiveControls
   Number of positive controls to simulate.

positiveControlEffectSize
   The true effect size of the positive controls.

Details

Simulate survival data for negative and positive controls. The data provides multiple looks at data accruing over time, with each look having more data than the one before. Systematic error for each outcome is drawn from the prespecified null distribution.

The outcome IDs are assigned sequentially starting at 1, with the first IDs used for the negative controls, and the latter IDs used for the positive controls.

Value

A data frame with 5 variables:

- **time** Time from index date to either the event or end of observation, whichever came first
- **outcome** Whether the outcome occurred (1) or not (0)
- **exposure** Whether the subject was exposed (TRUE) or not (FALSE)
- **lookTime** The time point when the look occurred.
- **outcomeId** A unique identifier for data corresponding to a single outcome. Lower IDs indicate negative controls, higher IDs indicate the positive control

Examples

```r
data <- simulateMaxSprtData(n = 1000)
head(data)
```

---

**southworthReplication**  *Relative risks from an unadjusted new-user cohort design*

Description

Relative risks from an unadjusted new-user cohort design

Usage

```r
data(southworthReplication)
```
Format

A data frame with 174 rows and 4 variables:

- **outcomeName**: Name of the outcome
- **trueLogRr**: The log of the true effect size. Only provided for negative and positive controls, is NA for the outcome of interest (GI bleeding).
- **logRr**: The log of the incidence rate ratio
- **seLogRr**: The standard error of the log of the incidence rate ratio

Details

A dataset containing the incidence rate ratios (and standard errors) produced using a new-user cohort design that compares new-users of dabigatran to new-users of warfarin for the outcome of GI hemorrhage. The dataset includes estimates both for the outcome of interest as well as negative and positive control outcomes. Subjects are required to have 183 days of continuous observation prior to initiating treatment, a prior diagnosis of atrial fibrillation, and are required to have no prior exposure to either dabigatran or warfarin. The study computes an incidence rate ratio without any adjustment for confounders. Time at risk is defined as the time on the drug. The original study (Southworth 2013) uses the 'Mini-Sentinel Database'. For our replication, we use the Optum databases since both databases are US insurance claims databases. We analyzed 5,982 dabigatran-exposed and 19,155 warfarin-exposed subjects. For more information on this set see Schuemie et al (2017).

References


Index

* datasets

- caseControl, 6
- cohortMethod, 7
- grahamReplication, 21
- sccs, 33
- southworthReplication, 36

- calibrateConfidenceInterval, 3
- calibrateLlr, 4
- calibrateP, 4
- caseControl, 6
- cohortMethod, 7
- compareEase, 8, 13
- computeCvBinomial, 9
- computeCvPoisson, 10
- computeCvPoissonRegression, 11
- computeExpectedAbsoluteSystematicError, 9, 12
- computeTraditionalCi, 13
- computeTraditionalP, 14
- convertNullToErrorModel, 15

- evaluateCiCalibration, 16, 25, 27

- fitMcmcNull, 15, 17
- fitNull, 15, 18
- fitNullNonNormalLl, 19
- fitSystematicErrorModel, 3, 15, 19

- grahamReplication, 21

- plotCalibration, 22
- plotCalibrationEffect, 23
- plotCiCalibration, 24
- plotCiCalibrationEffect, 26
- plotCiCoverage, 27
- plotErrorModel, 28
- plotExpectedType1Error, 29
- plotForest, 31
- plotMcmcTrace, 32
- plotTrueAndObserved, 32

- sccs, 33
- simulateControls, 34
- simulateMaxSprtData, 35
- southworthReplication, 36