Package ‘PoolTestR’

February 15, 2021

Title Prevalence and Regression for Pool-Tested (Group-Tested) Data
Version 0.1.1
URL https://github.com/AngusMcLure/PoolTestR
BugReports https://github.com/AngusMcLure/PoolTestR

Description
An easy-to-use tool for working with presence/absence tests on 'pooled' or 'grouped' samples. The primary application is for estimating prevalence of a marker in a population based on the results of tests on pooled specimens. This sampling method is often employed in surveillance of rare conditions in humans or animals (e.g. molecular xenomonitoring). The package was initially conceived as an R-based alternative to the molecular xenomonitoring software, 'PoolScreen' <https://sites.uab.edu/statgenetics/software/>. However, it goes further, allowing for estimates of prevalence to be adjusted for hierarchical sampling frames, and perform flexible mixed-effect regression analyses (McLure et al. pre-print <arXiv:2012.05405>). The package is currently in early stages, however more features are planned or in the works: e.g. adjustments for imperfect test specificity/sensitivity, functions for helping with optimal experimental design, and functions for spatial modelling.

License GPL (>= 3)
Encoding UTF-8
LazyData true
Biarch true

Depends R (>= 3.4.0)
Imports dplyr, methods, Rcpp (>= 0.12.0), rstan (>= 2.18.1), rstantools (>= 2.0.0), brms, stats, lme4, stringr, progress

LinkingTo BH (>= 1.66.0), Rcpp (>= 0.12.0), RcppEigen (>= 0.3.3.3.0), rstan (>= 2.18.1), StanHeaders (>= 2.18.0)

SystemRequirements GNU make
RoxygenNote 7.1.1
Language en-AU
NeedsCompilation yes
PoolTestR-package

Author  Angus McLure [aut, cre] (<https://orcid.org/0000-0003-2551-3059>)
Maintainer  Angus McLure <angus.mclure@anu.edu.au>
Repository  CRAN
Date/Publication  2021-02-15 05:50:07 UTC

R topics documented:

<table>
<thead>
<tr>
<th>R topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>PoolTestR-package</td>
<td>2</td>
</tr>
<tr>
<td>ExampleData</td>
<td>3</td>
</tr>
<tr>
<td>getPrevalence</td>
<td>4</td>
</tr>
<tr>
<td>HierPoolPrev</td>
<td>6</td>
</tr>
<tr>
<td>PoolLink</td>
<td>8</td>
</tr>
<tr>
<td>PoolPrev</td>
<td>10</td>
</tr>
<tr>
<td>PoolReg</td>
<td>12</td>
</tr>
<tr>
<td>PoolRegBayes</td>
<td>14</td>
</tr>
<tr>
<td>SimpleExampleData</td>
<td>17</td>
</tr>
<tr>
<td>TruePrev</td>
<td>17</td>
</tr>
</tbody>
</table>

Index  19

---

PoolTestR-package  The 'PoolTestR' package.

Description

This is a package for working with presence/absence tests on pooled or grouped samples. The primary application is for estimating prevalence of a marker in a population based on the results of tests on pooled specimens. This sampling method is often employed in surveillance of rare conditions in humans or animals (e.g. livestock or xenomonitoring). The package was initially conceived as an R-based alternative to the xenomonitoring software, PoolScreen. However, it goes further, allowing for estimates of prevalence to adjusted for hierarchy in sampling frames, and perform flexible (mixed effect) regression analyses. The package is currently in early stages, however more features are planned or in the works: e.g. adjustments for imperfect test specificity/sensitivity, functions for helping with optimal experimental design, and functions for spatial modelling.

References

A synthetic dataset mimicking a realistic hierarchical sampling frame. Simulated samples are taken from across three regions (A, B, and C) in which the vectors have a low (0.5%), medium (2%), and high (4%) prevalence of the marker of interest. Ten villages are chosen within each region, and traps are placed at ten sites within each village. Every site is sampled once a year over three years (0, 1, and 2). Prevalence is not uniform within each region or over time. At baseline (year 0), prevalence varies between villages within each region around the mean for the region, and prevalence varies between sites within each village around the mean for the village. Consequently though the prevalence is different for each site, two sites within the same village are likely to have a more similar prevalence than two sites in different villages, or two sites in different regions. On average the prevalence is declining over time (odds ratio of 0.8 per year), however, the growth rate varies between villages. Consequently two sites in different villages with similar prevalence at baseline may have different prevalence by the third year, and prevalence may go up in some villages. Each year the traps at each site catch a negative binomial number (mean 200, dispersion 5) of vectors. The catch size at each site and year is independent. Each year, the catches at each site are pooled into groups of 25 with an additional pool for any remainder (e.g. a catch of 107 vectors will be pooled into 4 pools of 25 and one pool of 7). Test results on each pool are simulated assuming the test has perfect sensitivity and specificity.

Usage

ExampleData

Format

A data frame with 6 variables:

- **NumInPool** Number of specimens in pool. Range = 1:25
- **Region** ID of the region the pool was taken from. "A", "B", or "C"
- **Village** ID of village pool was taken from. Includes name of region e.g. "B-3" is village 3 from region B
- **Site** ID of village pool was taken from. Includes name of region and village e.g. "B-3-7" is site 7 from village 3 from region B
- **Result** Result of test on pool; 0 = negative, 1 = positive
- **Year** Year of sampling. Years are 0, 1, or 2

Details

The 'true' model can be summarised in formula notation as:

\[
\text{Result} \sim \text{Region} + \text{Year} + (1 + \text{Year} | \text{Village}) + (1 | \text{Site})
\]

where the coefficient for Year is log(0.8), the standard deviation for intercept random effects for village and site are both 0.5, the standard deviation for the year random effect for village is 0.2 and the random effects are all uncorrelated/independent.
getPrevalence

Predicting Prevalence from a Mixed or Fixed Effect Logistic Regression with Presence/Absence Tests on Pooled Samples

Description

This function works somewhat like a predict or fitted generic function returning the model predicted prevalence for a given set of data; however, as the quantity of interest (prevalence) is neither on the response or link scale we do not use either of these generic functions. Further, when the model accounts for the hierarchical structure of the sampling frame (e.g. Region/Village/Site), it is common to want to know the predicted values at each level of sampling (e.g. Prevalence at each region, village or site) so these are calculated automatically.

Usage

getPrevalence(model, newdata = NULL, re.form = NULL)

Arguments

- **model**: An object returned by [PoolReg()]{r} or [PoolRegBayes()]{r}
- **newdata**: The data for which prevalence needs to be estimated/predicted. If not provided, defaults to using the data used to train the model (i.e. returns the fitted values of the prevalence)
- **re.form**: A description of which random effects to include in the prediction. If omitted, getPrevalence automatically tests to see if there are any random effect terms. If not, it just returns the estimates based on population effects. If there are random effects, it tests to see if the random effect variables form a nested hierarchical structure. If so, in addition to the estimates based on population effects only, it will estimate at different levels of the nested hierarchical structure in order of increasing granularity. For manual control you can set to NA for population effects only, or a one-sided formula specifying the form of the random effects to include in estimates, or a list of such objects.

Value

A list with at least one field PopulationEffects and an additional field for every random/group effect variable. The field PopulationEffects contains a data.frame with the prevalence estimated based only the fixed/population effects. When the intercept is the only fixed/population effect, this is just the population mean (possibly adjusted for random/group effects). When there are group effects terms, getPrevalence attempts to order these with respect to 'granularity' and extract the prevalence estimates for these random effects; e.g. if the random/group effects included are there to account for a hierarchical sampling frame with levels 'Village' and 'Site' with a formula like Result ~ Cov1 + Cov2 + (1|Village/Site), then getPrevalence will be a list of three data frames: estimates for every combination of covariates, estimates for every combination of covariates and village, and estimates for every combination of covariates, village, and site.
See Also

[PoolReg()] and [PoolRegBayes()]

Examples

# Perform logistic-type regression modelling for a synthetic dataset consisting
# of pools (sizes 1, 5, or 10) taken from 4 different regions and 3 different
# years. Within each region specimens are collected at 4 different villages,
# and within each village specimens are collected at 8 different sites.

### Models in a frequentist framework

#### ignoring hierarchical sampling frame within each region

```r
Mod <- PoolReg(Result ~ Region + Year,
               data = SimpleExampleData,
               poolSize = NumInPool)
summary(Mod)
```

#### accounting hierarchical sampling frame within each region

```r
HierMod <- PoolReg(Result ~ Region + Year + (1|Village/Site),
                     data = SimpleExampleData,
                     poolSize = NumInPool)
summary(HierMod)
```

# Extract fitted prevalence for each combination of region and year and then at
# each level of the hierarchical sampling frame (i.e. for each village in each
# region and each site in each village)

```r
getPrevalence(HierMod)
```

### Models in a Bayesian framework with default (non-informative) priors

#### ignoring hierarchical sampling frame within each region

```r
BayesMod <- PoolRegBayes(Result ~ Region + Year,
                           data = SimpleExampleData,
                           poolSize = NumInPool)
summary(BayesMod)
getPrevalence(BayesMod) # Extract fitted prevalence for each combination of region and year
```

#### accounting hierarchical sampling frame within each region

```r
BayesHierMod <- PoolRegBayes(Result ~ Region + Year + (1|Village/Site),
                               data = SimpleExampleData,
                               poolSize = NumInPool)
summary(BayesHierMod)
getPrevalence(BayesHierMod)
```

### Calculate adjusted estimates of prevalence

# We use the same function for all four models, but the outputs are slightly different

# Extract fitted prevalence for each combination of region and year

```r
getPrevalence(Mod)
```
HierPoolPrev

Estimation of prevalence based on presence/absence tests on pooled samples in a hierarchical sampling frame

Description

Estimation of prevalence based on presence/absence tests on pooled samples in a hierarchical sampling frame

Usage

HierPoolPrev(
  data,
  result,
  poolSize,
  hierarchy,
  ...
  prior.alpha = 0.5,
  prior.beta = 0.5,
  prior.absent = 0,
  alpha = 0.05,
  verbose = FALSE,
HierPoolPrev

cores = NULL

Arguments

data A data.frame with one row for each pooled sampled and columns for the size of the pool (i.e. the number of specimens / isolates / insects pooled to make that particular pool), the result of the test of the pool. It may also contain additional columns with additional information (e.g. location where pool was taken) which can optionally be used for splitting the data into smaller groups and calculating prevalence by group (e.g. calculating prevalence for each location)

result The name of column with the result of each test on each pooled sample. The result must be stored with 1 indicating a positive test result and 0 indicating a negative test result.

poolSize The name of the column with number of specimens/isolates/insects in each pool

hierarchy The name of column(s) indicating the group membership. In a nested sampling design with multiple levels of grouping the lower-level groups must have names/numbers that differentiate them from all other groups at the same level. E.g. If sampling was performed at 200 sites across 10 villages (20 site per village), then there should be 200 unique names for the sites. If, for instance, the sites are instead numbered 1 to 20 within each village, the village identifier (e.g. A, B, C...) should be combined with the site number to create unique identifiers for each site (e.g. A-1, A-2... for sites in village A and B-1, B-2... for the sites in village B etc.)

... Optional name(s) of columns with variables to stratify the data by. If omitted the complete dataset is used to estimate a single prevalence. If included prevalence is estimated separately for each group defined by these columns

prior.alpha, prior.beta, prior.absent

The prior on the prevalence in each group takes the form of beta distribution (with parameters alpha and beta). The default is prior.alpha = prior.beta = 1/2 i.e. the uninformative "Jeffrey's" prior. Another popular uninformative choice is prior.alpha = prior.beta = 1 i.e. a uniform prior. prior.absent is included for consistency with PoolPrev, but is currently ignored

alpha The confidence level to be used for the confidence and credible intervals. Defaults to 0.5% (i.e. 95% intervals)

verbose Logical indicating whether to print progress to screen. Defaults to false (no printing to screen)

cores The number of CPU cores to be used. By default one core is used

Value

A data.frame with columns:

- PrevMLE (the Maximum Likelihood Estimate of prevalence)
- CILow and CILow (Lower and Upper Confidence intervals using the Likelihood Ratio method)
- Bayesian Posterior Expectation
• CrILow and CrIHigh
• Number of Pools
• Number Positive

If grouping variables are provided in ... there will be an additional column for each stratifying variable. When there are no stratifying variables (supplied in ...) then the dataframe has only one row with the prevalence estimates for the whole dataset. When stratifying variables are supplied, then there is a separate row for each group.

Examples

# Calculate prevalence for a synthetic dataset consisting of pools (sizes 1, 5, # or 10) taken from 4 different regions and 3 different years. Within each # region specimens are collected at 4 different villages, and within each # village specimens are collected at 8 different sites.

# Prevalence for each combination of region and year:
# ignoring hierarchical sampling frame within each region
PoolPrev(SimpleExampleData, Result, NumInPool, Region, Year)
# accounting hierarchical sampling frame within each region
HierPoolPrev(SimpleExampleData, Result, NumInPool, c("Village","Site"), Region, Year)

---

**PoolLink**

*Link Function for Logistic Regression with Presence/Absence Tests on Pooled Samples*

**Description**

A custom link function for the `binomial` family to be used with `glm`

**Usage**

`PoolLink(PoolSize = 1)`

**Arguments**

- **PoolSize**: The number of specimens/isolates/insects in each pool. When used with `glm`, the length must either be 1 if all the pools are the same size, but the same length as the data otherwise.

**Value**

An object of class `link-glm`
Examples

# Perform logistic-type regression modelling for a synthetic dataset consisting
# of pools (sizes 1, 5, or 10) taken from 4 different regions and 3 different
# years. Within each region specimens are collected at 4 different villages,
# and within each village specimens are collected at 8 different sites.

### Models in a frequentist framework
# ignoring hierarchical sampling frame within each region
Mod <- PoolReg(Result ~ Region + Year,
    data = SimpleExampleData,
    poolSize = NumInPool)
summary(Mod)

# accounting hierarchical sampling frame within each region
HierMod <- PoolReg(Result ~ Region + Year + (1|Village/Site),
    data = SimpleExampleData,
    poolSize = NumInPool)
summary(HierMod)

# Extract fitted prevalence for each combination of region and year
getPrevalence(HierMod)

### Models in a Bayesian framework with default (non-informative) priors
# ignoring hierarchical sampling frame within each region
BayesMod <- PoolRegBayes(Result ~ Region + Year,
    data = SimpleExampleData,
    poolSize = NumInPool)
summary(BayesMod)

getPrevalence(BayesMod) # Extract fitted prevalence for each combination of region and year

# accounting hierarchical sampling frame within each region
BayesHierMod <- PoolRegBayes(Result ~ Region + Year + (1|Village/Site),
    data = SimpleExampleData,
    poolSize = NumInPool)
summary(BayesHierMod)

getPrevalence(BayesHierMod)

### Calculate adjusted estimates of prevalence
# We use the same function for all four models, but the outputs are slightly different

# Extract fitted prevalence for each combination of region and year
getPrevalence(Mod)

getPrevalence(BayesMod)

# Extract fitted prevalence for each combination of region and year
getPrevalence(HierMod)

getPrevalence(BayesHierMod)
#each level of the hierarchical sampling frame (i.e. for each village in each
#region and each site in each village)
gePrevalence(HierMod)

gPrevalence(BayesHierMod)

# You can also use getPrevalence to predict at prevalence for other values of
# the covariates (e.g. predict prevalence in year 4)

#Making a data frame containing data make predict on
DataFuture <- unique(data.frame(Region = SimpleExampleData$Region,
    Village = SimpleExampleData$Village,
    Site = SimpleExampleData$Site,
    Year = 4))

gPrevalence(Mod, newdata = DataFuture)
gPrevalence(HierMod, newdata = DataFuture)

gPrevalence(BayesMod, newdata = DataFuture)
gPrevalence(BayesHierMod, newdata = DataFuture)

---

**PoolPrev**

*Estimation of prevalence based on presence/absence tests on pooled samples*

---

**Description**

Estimation of prevalence based on presence/absence tests on pooled samples

**Usage**

PoolPrev(
  data,
  result,
  poolSize,
  ...
  prior.alpha = NULL,
  prior.beta = NULL,
  prior.absent = 0,
  alpha = 0.05,
  verbose = FALSE,
  cores = NULL
)
**PoolPrev**

Arguments

- **data**
  A data.frame with one row for each pooled sampled and columns for the size of the pool (i.e. the number of specimens / isolates / insects pooled to make that particular pool), the result of the test of the pool. It may also contain additional columns with additional information (e.g. location where pool was taken) which can optionally be used for stratifying the data into smaller groups and calculating prevalence by group (e.g. calculating prevalence for each location).

- **result**
  The name of column with the result of each test on each pooled sample. The result must be stored with 1 indicating a positive test result and 0 indicating a negative test result.

- **poolSize**
  The name of the column with number of specimens/isolates/insects in each pool

- **...**
  Optional name(s) of columns with variables to stratify the data by. If omitted the complete dataset is used to estimate a single prevalence. If included, prevalence is estimated separately for each group defined by these columns

- **prior.alpha, prior.beta, prior.absent**
  The default prior for the prevalence is the uninformative Jeffrey’s prior, however you can also specify a custom prior with a beta distribution (with parameters prior.alpha and prior.beta) modified to have a point mass of zero i.e. allowing for some prior probability that the true prevalence is exactly zero (prior.absent). Another popular uninformative choice is prior.alpha = 1, prior.beta = 1, prior.absent = 0, i.e. a uniform prior.

- **alpha**
  Defines the confidence level to be used for the confidence and credible intervals. Defaults to 0.05 (i.e. 95% intervals)

- **verbose**
  Logical indicating whether to print progress to screen. Defaults to false (no printing to screen).

- **cores**
  The number of CPU cores to be used. By default one core is used

Value

A data.frame with columns:

- PrevMLE (the Maximum Likelihood Estimate of prevalence)
- CI_Low and CI_High (Lower and Upper Confidence intervals using the Likelihood Ratio method)
- Bayesian Posterior Expectation
- CrILow and CrIHigh
- Number of Pools
- Number Positive

If grouping variables are provided in ... there will be an additional column for each grouping variable. When there are no grouping variables (supplied in ...) then the data frame has only one row with the prevalence estimates for the whole dataset. When grouping variables are supplied, then there is a separate row for each group.
Examples

# Try out on a synthetic dataset consisting of pools (sizes 1, 5, or 10) taken from 4 different regions and 3 different years. Within each region specimens are collected at 4 different villages, and within each village specimens are collected at 8 different sites.

# Prevalence across the whole (synthetic) dataset
PoolPrev(SimpleExampleData, Result, NumInPool)

# Prevalence in each Region
PoolPrev(SimpleExampleData, Result, NumInPool, Region)

# Prevalence for each year
PoolPrev(SimpleExampleData, Result, NumInPool, Year)

# Prevalence for each combination of region and year
PoolPrev(SimpleExampleData, Result, NumInPool, Region, Year)

---

**PoolReg**

Frequentist Mixed or Fixed Effect Logistic Regression with Presence/Absence Tests on Pooled Samples

Description

It can be useful to do mixed effects logistic regression on the presence/absence results from pooled samples, however one must adjust for the size of each pool to correctly identify trends and associations. This can be done by using a custom link function [PoolTestR::PoolLink()], defined in this package, in conjunction with using glm from the stats package (fixed effect models) or glmer from the lme4 package (mixed effect models).

Usage

PoolReg(formula, data, poolSize, link = "logit", ...)

Arguments

- **formula**: A formula of the kind used to define models in lme4, which are generalisation of the formulae used in lm or glm that allow for random/group effects. The left-hand side of the formula should be the name of column in data with the result of the test on the pooled samples. The result must be encoded with 1 indicating a positive test result and 0 indicating a negative test result.
- **data**: A data.frame with one row for each pooled sample and columns for the size of the pool (i.e. the number of specimens/isolates/insects pooled to make that particular pool), the result of the test of the pool and any number of columns to be used as the dependent variables in the logistic regression.
- **poolSize**: The name of the column with number of specimens/isolates/insects in each pool.
- **link**: Link function. There are two options ‘logit’ (logistic regression, the default) and ‘cloglog’ (complementary log log regression).
- **...**: Arguments to be passed on to stats::glm or lme4::glmer.
Value

An object of class glmerMod (or glm if there are no random/group effects)

Examples

# Perform logistic-type regression modelling for a synthetic dataset consisting
# of pools (sizes 1, 5, or 10) taken from 4 different regions and 3 different
# years. Within each region specimens are collected at 4 different villages,
# and within each village specimens are collected at 8 different sites.

### Models in a frequentist framework

# ignoring hierarchical sampling frame within each region
Mod <- PoolReg(Result ~ Region + Year,
               data = SimpleExampleData,
               poolSize = NumInPool)
summary(Mod)

# accounting hierarchical sampling frame within each region
HierMod <- PoolReg(Result ~ Region + Year + (1|Village/Site),
                    data = SimpleExampleData,
                    poolSize = NumInPool)
summary(HierMod)

# Extract fitted prevalence for each combination of region and year and then at
# each level of the hierarchical sampling frame (i.e. for each village in each
# region and each site in each village)
getPrevalence(HierMod)

### Models in a Bayesian framework with default (non-informative) priors

# ignoring hierarchical sampling frame within each region
BayesMod <- PoolRegBayes(Result ~ Region + Year,
                          data = SimpleExampleData,
                          poolSize = NumInPool)
summary(BayesMod)

getPrevalence(BayesMod) # Extract fitted prevalence for each combination of region and year

# accounting hierarchical sampling frame within each region
BayesHierMod <- PoolRegBayes(Result ~ Region + Year + (1|Village/Site),
                              data = SimpleExampleData,
                              poolSize = NumInPool)
summary(BayesHierMod)

getPrevalence(BayesHierMod)

### Calculate adjusted estimates of prevalence
# We use the same function for all four models, but the outputs are slightly different

# Extract fitted prevalence for each combination of region and year
getPrevalence(Mod)
getPrevalence(BayesMod)

# Extract fitted prevalence for each combination of region and year and then at
# each level of the hierarchical sampling frame (i.e. for each village in each
# region and each site in each village)
getPrevalence(HierMod)
getPrevalence(BayesHierMod)

# You can also use getPrevalence to predict at prevalence for other values of
# the covariates (e.g. predict prevalence in year 4)

# Making a data frame containing data make predict on
DataFuture <- unique(data.frame(Region = SimpleExampleData$Region,
                               Village = SimpleExampleData$Village,
                               Site = SimpleExampleData$Site,
                               Year = 4))

getPrevalence(Mod, newdata = DataFuture)
getPrevalence(HierMod, newdata = DataFuture)
getPrevalence(BayesMod, newdata = DataFuture)
getPrevalence(BayesHierMod, newdata = DataFuture)

---

PoolRegBayes  Bayesian Mixed or Fixed Effect Logistic Regression with Presence/Absence Tests on Pooled Samples

Description

It can be useful to do mixed effects logistic regression on the presence/absence results from pooled samples, however one must adjust for the size of each pool to correctly identify trends and associations.

Usage

PoolRegBayes(
  formula,
  data,
  poolSize,
  link = "logit",
  prior = NULL,
  cores = NULL,
  ...
)
Arguments

- **formula**: A formula of the kind used to define models in `brms`, which are generalisation of the formulae used in `lm`, `glm` or `lme4`. The left-hand side of the formula should be the name of column in `data` with the result of the test on the pooled samples. The result must be stored with 1 indicating a positive test result and 0 indicating a negative test result.

- **data**: A `data.frame` with one row for each pooled sampled and columns for the size of the pool (i.e. the number of specimens / isolates / insects pooled to make that particular pool), the result of the test of the pool and any number of columns to be used as the dependent variables in the logistic regression.

- **poolSize**: The name of the column with number of specimens / isolates / insects in each pool.

- **link**: Link function. There are two options ‘‘logit’’ (logistic regression, the default) and ‘‘cloglog’’ (complementary log log regression).

- **prior**: The priors to be used for the regression parameters. Defaults to a non-informative (normal(0,100)) prior on linear coefficients and a zero-truncated student-t prior on the group effect standard deviations. Custom priors must `brms` objects produced by `[brms::set_prior()]`.

- **cores**: The number of CPU cores to be used. By default one cores is used

- **...**: Additional arguments to be passed to `brms::brms`.

Value

An object of class `brms` with the regression outputs.

See Also

- `[PoolTestR::PoolReg()]`, `[PoolTestR::getPrevalence()]`

Examples

# Perform logistic-type regression modelling for a synthetic dataset consisting # of pools (sizes 1, 5, or 10) taken from 4 different regions and 3 different # years. Within each region specimens are collected at 4 different villages, # and within each village specimens are collected at 8 different sites.

```r
### Models in a frequentist framework
#ignoring hierarchical sampling frame within each region
Mod <- PoolReg(Result ~ Region + Year,
data = SimpleExampleData,
poolSize = NumInPool)
summary(Mod)

#accounting hierarchical sampling frame within each region
HierMod <- PoolReg(Result ~ Region + Year + (1|Village/Site),
data = SimpleExampleData,
poolSize = NumInPool)
```
summary(HierMod)
# Extract fitted prevalence for each combination of region and year and then at
# each level of the hierarchical sampling frame (i.e. for each village in each
# region and each site in each village)
getPrevalence(HierMod)

### Models in a Bayesian framework with default (non-informative) priors
# ignoring hierarchical sampling frame within each region
BayesMod <- PoolRegBayes(Result ~ Region + Year,
data = SimpleExampleData,
poolSize = NumInPool)
summary(BayesMod)
getPrevalence(BayesMod) # Extract fitted prevalence for each combination of region and year

# accounting hierarchical sampling frame within each region
BayesHierMod <- PoolRegBayes(Result ~ Region + Year + (1|Village/Site),
data = SimpleExampleData,
poolSize = NumInPool)
summary(BayesHierMod)
getPrevalence(BayesHierMod)

### Calculate adjusted estimates of prevalence
# We use the same function for all four models, but the outputs are slightly different

# Extract fitted prevalence for each combination of region and year
getPrevalence(Mod)
getPrevalence(BayesMod)

# Extract fitted prevalence for each combination of region and year and then at
# each level of the hierarchical sampling frame (i.e. for each village in each
# region and each site in each village)
getPrevalence(HierMod)
getPrevalence(BayesHierMod)

# You can also use getPrevalence to predict at prevalence for other values of
# the covariates (e.g. predict prevalence in year 4)

# Making a data frame containing data make predict on
DataFuture <- unique(data.frame(Region = SimpleExampleData$Region,
Village = SimpleExampleData$Village,
Site = SimpleExampleData$Site,
Year = 4))

getPrevalence(Mod, newdata = DataFuture)
getPrevalence(HierMod, newdata = DataFuture)
getPrevalence(BayesMod, newdata = DataFuture)
getPrevalence(BayesHierMod, newdata = DataFuture)

---

SimpleExampleData  A synthetic dataset for pooled testing

Description

The simple synthetic dataset consisting of pools (sizes 1, 5, or 10) taken from 4 different regions and 3 different years. Within each region specimens are collected at 4 different villages, and within each village specimens are collected at 8 different sites.

Usage

SimpleExampleData

Format

A data frame with 1152 rows and 6 variables:

- **NumInPool**  Number of specimens in pool. Takes values 1, 5, or 10.
- **Region**  ID of the region the pool was taken from. "A", "B", "C", or "D"
- **Village**  ID of village pool was taken from. Includes name of region e.g. "B-3" is village 3 from region B
- **Site**  ID of village pool was taken from. Includes name of region and village e.g. "B-3-7" is site 7 from village 3 from region B
- **Result**  Result of test on pool; 0 = negative, 1 = positive
- **Year**  Year of sampling. Years are 0, 1, or 2

---

TruePrev  A synthetic dataset for pooled testing

Description

This dataframe contains the 'true' values of prevalence for each, site, village, region and year used to generate the synthetic dataset ExampleData

Usage

TruePrev
Format

A data frame with 900 rows and 7 variables:

- **Region**: ID of the region the pool was taken from. "A", "B", or "C"
- **Village**: ID of village pool was taken from. Includes name of region e.g. "B-3" is village 3 from region B
- **Site**: ID of village pool was taken from. Includes name of region and village e.g. "B-3-7" is site 7 from village 3 from region B
- **Year**: Year of sampling. Years are 0, 1, or 2
- **PrevalenceRegion**: 'True' average prevalence in the region (in that year)
- **PrevalenceVillage**: 'True' average prevalence in the village (in that year)
- **PrevalenceSite**: 'True' prevalence at that site (in that year)

Details

The 'true' model can be summarised in formula notation as:

```
Result ~ Region + Year + (1|Village) + (0 + Year|Village) + (1|Site)
```

where the coefficient for Year is log(0.8), the standard deviation for intercept random effects for village and site are both 0.5, the standard deviation for the year random effect for village is 0.2 and the random effects are all uncorrelated/independent.
Index

* datasets
  ExampleData, 3
  SimpleExampleData, 17
  TruePrev, 17

binomial, 8
ExampleData, 3
getPrevalence, 4
glm, 8
HierPoolPrev, 6
PoolLink, 8
PoolPrev, 10
PoolReg, 12
PoolRegBayes, 14
PoolTestR(PoolTestR-package), 2
PoolTestR-package, 2

SimpleExampleData, 17
TruePrev, 17