Package ‘RLadyBug’

August 3, 2006

Type Package

Title Analysis of infectious diseases using stochastic epidemic models

Version 0.4

Date 2006-07-21

Author Michael Hoehle

Depends methods,MASS,boa,coda,rJava

Maintainer Michael Hoehle <hoehle@stat.uni-muenchen.de>

Description Analysis of small scale infectious diseases possibly with a spatial grid component using stochastic Susceptible-Exposed-Infectious-Recovered models. The R package wraps functionality of an Java program, i.e. a java virtual machine has to be installed on your computer.

License GPL version 2 (http://www.gnu.org/licenses/gpl.html)

URL http://www.stat.uni-muenchen.de/~hoehle/software/RLadyBug

LazyLoad yes

R topics documented:

- LBExperiment-class .................................................. 2
- LBInference-class ..................................................... 3
- LBInferenceMCMC-class ............................................... 4
- LBInferenceML-class ................................................... 5
- LBLayout-class ......................................................... 6
- LBOptions-class ........................................................ 7
- LBOptionsMCMC-class .................................................. 9
- LBOptionsML-class ..................................................... 11
- RLadyBug-package .................................................... 13
- abakaliki ............................................................. 14
- csfv .................................................................. 14
- ladybugExample ......................................................... 15
- laevens ................................................................. 15
- oneill ................................................................. 16
- readSpecFile ............................................................ 17
- seir ................................................................. 18
LBExperiment-class  

Class "LBExperiment"

Description

S4 class containing the data and the layout of the infectious disease data

Slots

data: Data Frame with six columns: x, y, E, I, R, D
layout: Object of class "LBLayout"

Methods

data2events signature(object = "LBExperiment"): convert the data.frame of events for each individual to a time order data.frame of events for the entire Experiment. The information about each individual is lost. This function is used internally.
show signature(object = "LBExperiment"): shows all slots of the LBExperiment object.
setLayout<- signature(object = "LBExperiment"): sets the Layout.
plot signature(signature(x="LBExperiment", y="missing"), function(x, y,...): The type argument should be a formula specifying the desired type of plot. By providing an additional options list individual parameters for the plots are provided. Valid formulae are
   state ~ time | position The number of susceptible, infectious and recovered as a function of time for each unit. Warning: in case there are many units this plot might be rather useless.
   state ~ time shows the total number of susceptible, infectious and recovered (i.e. summed over all units) as a function of time. Individual options are
   stacked boolean whether stacked boxplots or just time-series are shown.
   state ~ 1|position illustrates the three multivariate time series (susceptible, exposed, infected) as a "film" with noOfPics pictures. Individual options are
   chart Either "pie" or "bar", where the latter is default.
   justInf if FALSE pie charts with the number of S(t),E(t),I(t) are shown, otherwise only the number of infectious is shown, where the radius shows the proportion.
   noOfPics How many pictures in the animation. If not saved set the "History" attribute of the X11.
   PDF If TRUE the results are saved in PDF Files with the base name name.
   name Base name of the generated PDF Files. The actual files are then names name-addstr-number.pdf
   addstr This is added to the base name.
   individual ~ time shows all events for each individual
   individual ~ time | position show individual histories of each individual aligned to the same time axis

Author(s)

U. Feldmann and M. Höhle
**LBInference-class**

**Examples**

```r
sim.layout <- new("LBLayout", S0=matrix(c(13,rep(14,7)),ncol=4),
E0=matrix(c(1,rep(0,7)),ncol=4))
sim.opts <- new("LBOptions", seed=2006,
LBmodel=c("gamma","gamma","gamma",FALSE),
ignoreData=c(FALSE,FALSE,FALSE),
initBeta=list(init=0.125,
gamma=0.001, delta=0.001),
initBetaN=list(init=0.018,
gamma=0.001, delta=0.001),
initIncu=list(asis=FALSE, const=FALSE,
g=6.697, g.gamma=0.001, g.delta=0.001,
d=0.84, d.gamma=0.001,d.delta=0.001),
initInf=list(1.772, 0.001, 0.001, 0.123, 0.001, 0.001),
nitDia=list(149.126, 0.001, 0.001, 8.737, 0.001, 0.001))
exp <- simulate(sim.opts, layout=sim.layout)
plot(exp,type = state ~ time)
plot(exp,type = state ~ time, options=list(stacked=FALSE))
```

---

**LBInference-class**  
*Class "LBInference" – captures results for SEIR inference*

**Description**

This class contains results from inference by the LadyBug program.

**Objects from the Class**

Usually, there is no need to create objects of this class by hand.

**Slots**

- **paramHat**: Object of class "numeric" A vector with point estimates for the model parameters.
- **paramSe**: Object of class "numeric" Point estimates for the standard error.
- **aic**: Object of class "numeric" Aikakes Information Criterion.
- **loglik**: Object of class "numeric" Value of the log likelihood.

**Methods**

- **infValues** signature(object = "LBInference"): Fetches a list with all slots.
- **infValues<-** signature(object = "LBInference"): ...
- **show** signature(object = "LBInference"): ...
- **summary** signature(object = "LBInference"): ...

**See Also**

`LBInferenceML-class` and `LBInferenceMCMC-class`
Examples

data(oneill)
mcmc <- seir(oneill, oneill.opts)
## Not run: infValues(mcmc)

LBInferenceMCMC-class

Class "LBInferenceMCMC" – results from MCMC inference in SEIR models

Description

This class holds the results from MCMC inference for SEIR models, i.e. sample paths and provides routines to calculate $R_0$

Objects from the Class

Objects can be created by calls of the form `new("LBInferenceMCMC", paramHat, paramSe, aic, loglik, samplePaths)`

Slots

- `samplePaths`: Object of class "data.frame" A data frame containing the va
- `paramHat`: Object of class "numeric"
- `paramSe`: Object of class "numeric"
- `aic`: Object of class "numeric"
- `loglik`: Object of class "numeric"

Extends

Class "LBInference", directly.

Methods

- `infValues` signature(object = "LBInferenceMCMC"): ...
- `infValues<-` signature(object = "LBInferenceMCMC"): ...
- `initialize` signature(.Object = "LBInferenceMCMC"): ...
- `plot` signature(x = "LBInferenceMCMC", y = "missing"): Important is the which argument
  - `beta` CODA diagnost for the $\beta$ parameter
  - `betabetaN` Provides a diagnostic plot and HPD interval for the $\frac{\beta}{\beta_0}$ ratio.
- `R0` signature(object = "LBInferenceMCMC"): Compute the basic reproduction ratio for each sample. Mean, median, etc. are then computed.
- `samplePaths` signature(object = "LBInferenceMCMC"): get the sample paths
- `show` signature(object = "LBInferenceMCMC"): as usual
- `summary` signature(object = "LBInferenceMCMC"): as usual
See Also

LBInference-class

Examples

data(laevens)
inf.mcmc <- seir(laevens,laevens.opts)

# Algo part of the Options
algo(laevens.opts)

# Results
inf.mcmc

# Analysis through coda (library coda is called when starting RLadyBug)
samples <- mcmc(samplePaths(inf.mcmc))
plot(samples[,"beta"])

# Look at the \beta/\beta_n ratio
ratio <- plot(inf.mcmc,which = "betabetaN")
c(mean=ratio$mean,ratio$hpD)

# R0
quantile(R0(inf.mcmc,laevens),c(0.025,0.5,0.975))

---

LBInferenceML-class

Class "LBInferenceML" – results from ML inference in SEIR models

Description

Results from a maximum likelihood inference for SEIR models

Objects from the Class

Objects can be created by calls of the form new("LBInferenceML", ...). describe objects here

Slots

cov: Object of class "matrix" giving the covariance matrix of all parameters, i.e. this is the inverse negative Hessian matrix evaluated at the MLE.
corr: Object of class "numeric"
paramHat: Object of class "numeric" containing the MLE of all parameters
paramSe: Object of class "numeric" containing the standard error of all parameters
aic: Object of class "numeric" AIC of the fitted model
loglik: Object of class "numeric" containing the loglik at the MLE

Extends

Class "LBInference", directly.
Methods

- `infValues` signature(object = "LBInferenceML"): get all slots
- `infValues<-` signature(object = "LBInferenceML"): set a list of slots
- `show` signature(object = "LBInferenceML"): as usual
- `summary` signature(object = "LBInferenceML"): as usual

See Also

LBInference-class

Examples

data(laevensML)
seir(laevensML, laevensML.opts)

---

**LBLayout-class**

Class "LBLayout" – grid layout structure

Description

This class is used to specify the spatial (or structural) arrangement of the populations. Currently only a grid layout is handled.

Objects from the Class

Objects can be created by calls of the form `new("LBLayout", ...)`.

Slots

- **S0**: Object of class "matrix" A matrix specifying the number of initially susceptible in each unit.
- **E0**: Object of class "matrix" A matrix specifying the number of initially exposed in each unit.

Methods

- `layoutAsDataFrame` signature(object = "LBLayout"): Returns a data frame containing the columns "u", "x", "y", "S" and "E"
- `layoutMatrixes` signature(object = "LBLayout"): provides a list with S0 and E0 in matrix form
- `show` signature(object = "LBLayout"): as usual
- `summary` signature(object = "LBLayout"): as usual

Note

Currently, LadyBug is not able to handle more than one initially exposed. This will change in the near future.

See Also

See also LBExperiment-class.
Examples
data(csfv)

Description
Specification of LadyBug SEIR models

Objects from the Class
Objects can be created by calls of the form `new("LBOptions", seed, LBmodel, ignoreData, initBeta, initBetaN, initIncu, initInf, initDia, algo, randomWalk)

Slots
seed: Object of class "numeric" The seed value to use when calling the Java program
LBmodel: Object of class "vector" Contains a specification of the SEIR model, i.e. a vector with names

ignoreData: Object of class "vector" Booleans

initBeta: Object of class "list" Initial values:

initBetaN: Object of class "list" Initial values:

initIncu: Object of class "list" Initial values:

initInf: Object of class "list" Initial values:

initDia: Object of class "list" Initial values:
Methods

ignoreData signature(object = "LBOptions"): returns value of slot ignoreData

ignoreData<- signature(object = "LBOptions", value = "vector"): assigns value to slot ignoreData

initBeta signature(object = "LBOptions"): returns value of slot initBeta

initBeta<- signature(object = "LBOptions", value = "list"): assigns value to slot initBeta

initBetaN signature(object = "LBOptions"): returns value of slot initBetaN

initBetaN<- signature(object = "LBOptions", value = "list"): assigns value to slot initBetaN

initDia signature(object = "LBOptions"): returns value of slot initDia

initDia<- signature(object = "LBOptions", value = "list"): assigns value to slot initDia

initialize signature(.Object = "LBOptions"): does initializing of the slots when new("LBOptions", ... ) is called

initIncu signature(object = "LBOptions"): returns value of slot initIncu

initIncu<- signature(object = "LBOptions", value = "list"): assigns value to slot initIncu

initInf signature(object = "LBOptions"): returns value of slot initInf

initInf<- signature(object = "LBOptions", value = "list"): assigns value to slot initInf

initsAsDataFrame signature(object = "LBOptions"): returns initial values in a dataframe format

LBInits signature(object = "LBOptions"): returns all initial values (as there are initBeta, initBetaN, initIncu, initInf, initDia)

LBInits<- signature(object = "LBOptions", value = "list"): assigns value to all initial value slots (as there are initBeta, initBetaN, initIncu, initInf, initDia)

LBModel signature(object = "LBOptions"): returns value of slot LBModel

LBModel<- signature(object = "LBOptions", value = "vector"): assigns value to slot LBModel

LBOptions signature(object = "LBOptions", value = "vector"): returns values of real option slots (as there are seed, LBModel, ignoreData)

LBOptions<- signature(object = "LBOptions", value = "list"): assigns value to real option slots (as there are seed, LBModel, ignoreData)

optionsAsDataFrame signature(object = "LBOptions"): returns real option values in a dataframe format

seed signature(object = "LBOptions"): returns value of slot seed

seed<- signature(object = "LBOptions", value = "numeric"): assigns value to slot seed

show signature(object = "LBOptions"): shows the object

simulate signature(object = "LBOptions", layout = "Layout"): simulates data according to the specified model and init values and the given layout structure

summary signature(object = "LBOptions"): gives a summary of the object (at the moment no difference to show)
LBOptionsMCMC-class

Author(s)
M. Hoehle and U. Feldmann

See Also
See also LBOptionsMCMC-class and LBOptionsML-class

Examples

```r
opts <- new("LBOptions", seed=2003,
  LBmodel=c("gamma", "gamma", "gamma", FALSE ),
  ignoreData=c( TRUE, FALSE, FALSE ),
  initBeta=list( 0.125, 0.001, 0.001 ),
  initBetaN=list( init=0.018, gamma=0.001, delta=0.001 ),
  initIncu=list( g=6.697, g.gamma=0.001, g.delta=0.001,
               d=0.840, d.gamma=0.001, d.delta=0.001 ),
  initInf=list( 1.772, 0.001, 0.001, 0.123, 0.001, 0.001 ),
  initDia=list( 149.126, 0.001, 0.001, 8.737, 0.001, 0.001 ) )
layout <- new("LBLayout", S0=matrix( c( 14, 14 ), ncol=2 ),
            E0=matrix( c( 0, 1 ), ncol=2 ) )
exp <- simulate( opts, layout=layout )
```

LBOptionsMCMC-class

Class "LBOptionsMCMC" — Specification of MCMC estimation in SEIR models.

Description

Specification of MCMC estimation in SEIR models.

Objects from the Class

Objects can be created by calls of the form `new("LBOptionsMCMC", seed, LBmodel, ignoreData, initBeta, initBetaN, initIncu, initInf, initDia, algo, randomWalk)`.

Slots

`algo`: Object of class "vector". Contains a specification of the MCMC algorithm, i.e. a vector with names

- `samples` & how many? (without burnin)
- `thin` & how to thin the random numbers
- `burnin` & the first x random numbers will be ignored

`randomWalk`: Object of class "vector". Contains a specification of the random walk, i.e. a vector with names

- `betaRWsigma` & sigma concerning parameter $\beta$
- `betaNRWsigma` & sigma concerning parameter $\beta_n$
- `gammaERWsigma` & sigma concerning parameter $\gamma$
- `gammaIRWsigma` & sigma concerning parameter $\gamma$
- `gammaDRWsigma` & sigma concerning parameter $\gamma$
- `deltaERWsigma` & sigma concerning parameter $\delta$
- `deltaIRWsigma` & sigma concerning parameter $\delta$
- `deltaDRWsigma` & sigma concerning parameter $\delta$
- `ERWsigma` & sigma concerning unknown exposure times

`seed`: Object of class "numeric". The seed value to use when calling the Java program

`LBmodel`: Object of class "vector". Contains a specification of the SEIR model, i.e. a vector

See Also

LBOptionsMCMC-class and LBOptionsML-class

Examples

```r
opts <- new("LBOptions", seed=2003,
  LBmodel=c("gamma", "gamma", "gamma", FALSE ),
  ignoreData=c( TRUE, FALSE, FALSE ),
  initBeta=list( 0.125, 0.001, 0.001 ),
  initBetaN=list( init=0.018, gamma=0.001, delta=0.001 ),
  initIncu=list( g=6.697, g.gamma=0.001, g.delta=0.001,
               d=0.840, d.gamma=0.001, d.delta=0.001 ),
  initInf=list( 1.772, 0.001, 0.001, 0.123, 0.001, 0.001 ),
  initDia=list( 149.126, 0.001, 0.001, 8.737, 0.001, 0.001 ) )
layout <- new("LBLayout", S0=matrix( c( 14, 14 ), ncol=2 ),
            E0=matrix( c( 0, 1 ), ncol=2 ) )
exp <- simulate( opts, layout=layout )
```
with names

incuTimePDF & distribution of incubation time
infTimePDF & distribution of the infectious time
diagTimePDF & distribution of the seroconversion time
meanVar & mean variance representation of periods (TRUE/FALSE)

ignoreData: Object of class "vector". Booleans

ignoreE & Ignore the specified exposure (E) event times
ignoreI & Ignore the specified infective (I) event times
ignoreD & Ignore the specified diagnose (D) event time

initBeta: Object of class "list". Initial values:

init & for \( \beta \) \( \gamma \) \( \delta \) & for the priori parameter \( \gamma \) \( \delta \) & for the priori parameter \( \delta \)

initBetaN: Object of class "list". Initial values:

init & for \( \beta_n \) \( \gamma \) \( \delta \) & for the priori parameter \( \gamma \) \( \delta \) & for the priori parameter \( \delta \)

initIncu: Object of class "list". Initial values:

g & for parameter \( \gamma \) of the gamma distribution of the incubation time
\( g \cdot \gamma \) & for the parameter \( \gamma \) of the distribution of \( g \)
\( g \cdot \delta \) & for the parameter \( \delta \) of the distribution of \( g \)

or choose asis or constant:

asis & TRUE/FALSE
const & TRUE/FALSE
const.val & value of constant if \( \text{const} == \text{TRUE} \)

initInf: Object of class "list". Initial values:

\( g \) & for parameter \( \gamma \) of the gamma distribution of the infectious time
\( g \cdot \gamma \) & for the parameter \( \gamma \) of the distribution of \( g \)

initDia: Object of class "list". Initial values:

\( g \) & for parameter \( \gamma \) of the gamma distribution of the seroconversion time
\( g \cdot \gamma \) & for the parameter \( \gamma \) of the distribution of \( g \)

Extends

Class "LBOptions", directly.

Methods

algo signature(object = "LBOptionsMCMC"):
returns value of slot algo

algo<- signature(object = "LBOptionsMCMC", value = "vector"):
assigns value to slot algo

LBOptions signature(object = "LBOptionsMCMC"):
returns values of real option slots (as there are seed, LBModel, ignoreData, algo, randomWalk)

LBOptions<- signature(object = "LBOptionsMCMC", value = "list"):
assigns value to real option slots (as there are seed, LBModel, ignoreData, algo, randomWalk)

optionsAsDataFrame signature(object = "LBOptionsMCMC"):
returns real option values in a dataframe format

randomWalk signature(object = "LBOptionsMCMC"):
returns value of slot randomWalk
randomWalk <- signature(object = "LBOptionsMCMC", value = "vector"): assigns value to slot randomWalk

show signature(object = "LBOptionsMCMC"): \ shows the object

summary signature(object = "LBOptionsMCMC"): \ gives a summary of the object
(at the moment no difference to show)

writeOptionFile signature(object = "LBOptionsMCMC", filename="vector"): \ writes a file containing all options as input for java

Author(s)

M. Hoehle and U. Feldmann

See Also

See also LBOptions-class and LBOptionsML-class

Examples

opts <- new("LBOptionsMCMC", algo=c( samples=2500, thin=25, burnin=50000 ),
  randomWalk=c( "betaRWSigma"= 0.1,
    "betaNRWSigma"=0.1,
    "gammaERWSigma"=3,
    "deltaERWSigma"=1,
    "gammaIRWSigma"=1,
    "deltaIRWSigma"=1,
    "gammaDRWSigma"=3,
    "deltaDRWSigma"=1,
    "ERWSigma"=6 ),
  seed=2003,
  LBmodel=c( "gamma", "gamma", "gamma", FALSE ),
  ignoreData=c( TRUE, FALSE, TRUE ),
  initBeta=list( 0.4, 0.001, 0.001 ),
  initBetaN=list( init=0.005, gamma=0.001, delta=0.001 ),
  initIncu=list( g=1, g.gamma=0.001, g.delta=0.001,
    d=0.11, d.gamma=0.001, d.delta=0.001 ),
  initInf=list( 1, 0.001, 0.001, 0.11, 0.001, 0.001 ),
  initDia=list( 8, 0.001, 0.001, 0.8, 0.001, 0.001 ) )

---

LBOptionsML-class

Class "LBOptionsML" – maximum likelihood inference in SEIR models

Description

Specification of LadyBug SEIR models using maximum likelihood inference

Objects from the Class

Objects can be created by calls of the form new("LBOptionsML", seed, LBmodel, ignoreData, initBeta, initBetaN, initIncu, initInf, initDia, algo, randomWalk).
Slots

- **seed**: Object of class "numeric" The seed value to use when calling the Java program
- **LBmodel**: Object of class "vector" Contains a specification of the SEIR model, i.e. a vector with names
  - `incuTimePDF& distribution of incubation time`
  - `infTimePDF & distribution of the infectious time`
  - `diagTimePDF & distribution of the seroconversion time`
  - `meanVar & mean variance representation of periods (TRUE/FALSE)`
- **ignoreData**: Object of class "vector" Booleans
  - `ignoreE` & Ignore the specified exposure (E) event times
  - `ignoreI` & Ignore the specified infective (I) event times
  - `ignoreD` & Ignore the specified diagnose (D) event time
- **initBeta**: Object of class "list" Initial values:
  - `init & for β`
  - `{gamma & for the priori parameter γ, delta & for the priori parameter δ}`
- **initBetaN**: Object of class "list" Initial values:
  - `init & for β_n`
  - `{gamma & for the priori parameter γ, delta & for the priori parameter δ}`
- **initIncu**: Object of class "list" Initial values:
  - `{g & for parameter γ of the gamma distribution of the incubation time, g.gamma & for the parameter gamma of the distribution of g, g.delta & for the parameter delta of the distribution of g}`
  - or choose asis or constant:
    - `asis & TRUE/FALSE`
    - `const & TRUE/FALSE`
    - `const.val & value of constant if const == TRUE`
- **initInf**: Object of class "list" Initial values:
  - `{g & for parameter γ of the gamma distribution of the infectious time, g.gamma & for the parameter gamma of the distribution of g}`
- **initDia**: Object of class "list" Initial values:
  - `{g & for parameter γ of the gamma distribution of the seroconversion time, g.gamma & for the parameter gamma of the distribution of g}`

Extends

Class "LBOptions", directly.

Methods

- **show** signature(object = "LBOptionsML"):: shows the object
- **summary** signature(object = "LBOptionsML"):: gives a summary of the object (at the moment no difference to show)
- **writeOptionFile** signature(object = "LBOptionsML", filename = "vector"):: writes a file containing all options as input for java

Author(s)

M. Hoehle and U. Feldmann
See Also

See also LBOptions-class and LBOptionsMCMC-class

Examples

opts <- new( "LBOptionsML", seed=2003,
        LBmodel=c( "constant", "gamma", "none", FALSE ),
        ignoreData=c( FALSE, FALSE, FALSE ),
        initBeta=list( 0.4, 0.001, 0.001 ),
        initBetaN=list( init=0.005, gamma=0.001, delta=0.001 ),
        initIncu=list( asis=TRUE ),
        initInf=list( 1, 0.001, 0.001, 0.11, 0.001, 0.001 ),
        initDia=list( 8, 0.001, 0.001, 0.8, 0.001, 0.001 ) )

RLadyBug-package  Analysis of infectious diseases using stochastic epidemic models

Description

RLadyBug is an S4-package for the simulation, visualization and estimation of stochastic epidemic models in R. Utilizing the Susceptible-Exposed-Infected-Recovered (SEIR) Model as mathematical framework, maximum likelihood and Bayesian inference can be performed to estimate the parameters in a single outbreak of an infectious disease. With the package one step is taken towards statistical software supporting parameter estimation, the calculation of confidence intervals and hypothesis testing for transmission models.

Details

Package: RLadyBug
Type: Package
Version: 1.0
Date: 2006-07-03
License: GPL version 2 (http://www.gnu.org/licenses/gpl.html)

Author(s)

Ulrike Feldmann and Michael Hoehle
Maintainer: Michael Hoehle <hoehle@stat.uni-muenchen.de>

References


Examples

## Not run: demo(csda)
abakaliki  Smallpox epidemic in Abakaliki, Nigeria

Description
Use MCMC to estimate parameters in the smallpox epidemic of Abakaliki, Nigeria also treated in the article by ONeill and Roberts.

Usage
data(abakaliki)

Source

Examples
## Not run: data(abakaliki)
## Not run: seir(abakaliki,abakaliki.opts)

---

csfv  CSFV Transmission Experiment

Description
Analysis of the transmission rates in the classical swine fever virus transmission experiment in the Dewulf et al. (2001) article.

Usage
data(csfv)

Details
The csfvML dataset is a version of csfv, where the exposure time is specified with an artificially assumed fixed incubation time of 6 days, except for the inoculated individual which has an incubation time of three days. This is rather ad hoc but allows us to calculate ML estimates.

The csfvTDprior dataset is a version of csfv, where rather strong priori distributions are assumed for the waiting time from exposure until diagnosis.

Note that respective csfv.opts, csfvML.opts and csfvTD.opts objects are loaded which provide an appropriate estimation method.

Source
An experimental infection with classical swine fever in E2 sub-unit marker-vaccine vaccinated and in non-vaccinated pigs, Vaccine 19, pages 475-482.
**ladybugExample**

Examples

```r
## Not run: data(csfv)
## Not run: seir(csfv,csfv.opts)
```

---

### ladybugExample

Access files in the LadyBug directory examples directory

---

**Description**

A small helper function to access files in the LadyBug examples/ directory.

**Usage**

```r
ladybugExample(exp.file)
```

**Arguments**

- `exp.file` The filename relative to <LADYBUG>/examples/

**Value**

The complete filename, where <LADYBUG> is replaced by options("ladybugPath").

**Author(s)**

M. Höhle

**Examples**

```r
## Not run: ladybugExample( "/csfv/mcmc.sir" )
```

---

**laeuvens**

CSFV Experiment by Laevens et. al

---

**Description**

In this experiment the spread of CSFV was investigated in a $1 \times 3$ layout with $S(0) = (5, 5, 6)$ and $E(0) = (0, 1, 0)$ slaughter pigs. Every second day all pigs still alive were investigated using a virus isolation test based on blood plasma.

**Usage**

```r
data(laeuvens)
```
Format

The format is:

Formal class ‘LBExperiment’ [package "RLadyBug"] with 2 slots
..@ data :`data.frame`: 15 obs. of 6 variables:
  .. ..$ x: int [1:15] 1 1 1 1 1 1 1 1 1 1 ...
  .. ..$ y: int [1:15] 2 1 1 1 2 2 2 2 2 2 2 ...
  .. ..$ E: int [1:15] 0 18 14 20 10 10 10 6 20 ...
  .. ..$ I: int [1:15] 6 24 20 26 16 16 16 12 26 ...
  .. ..$ R: int [1:15] 12 34 28 34 34 34 34 28 30 28 ...
  .. ..$ D: int [1:15] 12 34 28 34 34 34 34 28 30 28 ...
..@ layout:Formal class 'LBLayout' [package "RLadyBug"] with 2 slots
  .. .. ..@ S0: num [1, 1:3] 5 5 5
  .. .. ..@ E0: num [1, 1:3] 0 1 0

Details

Together with an object laevens also an object laevens.opts is loaded which is an object of
class LBInferenceMCMC-class suitable for MCMC inference

The data(laevensML) contains a version of the data, where a constant incubation time of c=6 is
assumed. Here laevens.opts contains the necessary object for maximum likelihood inference.

Source

H. Laevens, F. Koenen, H. Deluyker and A. de Kruif, Experimental infection of slaughter pigs with
classical swine fever virus: transmission of the virus, course of the disease and antibody response,

Examples

data(laevens)

oneill

Simulated data of 5 recovery times from O'Neill et. al

Description

Use MCMC to estimate parameters in the 5 point epidemic in the 5 removal times epidemic (p.126)
of the article by O'Neill and Roberts.

Usage

data(oneill)

Details

n.a.

Source

**readSpecFile**

---

**Read LadyBug files and create corresponding S4 RLadyBug objects**

**Examples**

```r
## Not run: data(oneill)
## Not run: seir(oneill, oneill.opts)
```

### Description

Data and specification files in the LadyBug files are read and converted to S4 RLadyBug objects.

### Usage

```r
readSpecFile(options, data)
```

### Arguments

- `options`: Filename of the LadyBug options (i.e. the .sir) file
- `data`: Filename of the Data in LadyBug format

### Details

n.a.

### Value

A list containing

- `options`: The information relevant to the Options
- `experiment`: An object of class `Object` containing the layout and the event times of the data

### Author(s)

U. Feldmann and M. Höhle

### Examples

```r
## Not run: csfv <- readSpecFile( ladybugExample( "/csfv/mcmc.sir"), ladybugExample( "/csfv/csfv.data" )
#Show the MCMC options
## Not run: csfv$options
#Show the layout and initial configuration
## Not run: csfv$experiment
```
seir

Parameter estimation in SEIR-Models based on ML or MCMC

Description

Inference is perform for the parameters in an SEIR-model based on the data in experiment. The actual class of options (OptionsML or OptionsMCMC) decides what type of inference is performed.

Usage

seir(experiment, options)

Arguments

- experiment: Data corresponding to an Experiment
- options: An object of class Options. The specific action (ML or MCMC estimation) is determined by the subclass of options.

Details

Estimation is performed by calling LadyBug using a .jcall to the appropriate method in the Java class sir.estimate.LadyBug. Output is read from file and converted into an appropriate object of class LBInference-class.

Currently the method branches on the appropriate method using an if. Should become a generic method as some point.

Value

An object of class Inference

Author(s)

U. Feldmann and M. Höhle

See Also

LBOptions-class, LBOptionsML-class, LBOptionsMCMC-class

Examples

data(csfvML)
ml <- seir(csfvML, csfvML.opts)
ml

#MCMC Inference for the data from the Laevens experiment
data(laevens)
infmcmc <- seir(laevens, laevens.opts)
#Show some results
infmcmc

#Analysis through coda (library coda is called when starting RLadyBug)
samples <- mcmc(samplePaths(inf.mcmc))
plot(samples[,"beta"])