Package ‘LCMCR’

February 3, 2020

Type Package
Title Bayesian Non-Parametric Latent-Class Capture-Recapture
Version 0.4.11
Date 2020-02-03
Author Daniel Manrique-Vallier
Description Bayesian population size estimation using non parametric latent-class models.
Maintainer Daniel Manrique-Vallier <dmanriqu@indiana.edu>
License GPL (>= 2)
Depends R (>= 3.0.1)
Imports methods
        Lcm_CR_Strat_fn.R
SystemRequirements Gnu Scientific Library version >= 1.12
NeedsCompilation yes
Repository CRAN
Date/Publication 2020-02-03 19:40:09 UTC

R topics documented:

  LCMCR-package ..................................................  2
  kosovo_aggregate .............................................  3
  lcmCR ..........................................................  4
  lcmCR_PostSampl .............................................  6
  lcm_CR_Basic-class ..........................................  7
  lcm_CR_Basic_generator ....................................  8
  MCMCenviron-class ..........................................  9

Index 11
Description

This package implements a fully Bayesian multiple-recapture method for estimating the unknown size of a population using non-parametric latent class models. This is an implementation of the method described in Manrique-Vallier (2016). The estimation algorithm is based on Markov Chain Monte Carlo sampling.

Details

<table>
<thead>
<tr>
<th>Package:</th>
<th>LCMCR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type:</td>
<td>Package</td>
</tr>
<tr>
<td>Version:</td>
<td>0.4.11</td>
</tr>
<tr>
<td>Date:</td>
<td>2020-02-03</td>
</tr>
<tr>
<td>License:</td>
<td>GPL &gt;= 2</td>
</tr>
</tbody>
</table>

Author(s)

Daniel Manrique-Vallier <dmanriqu@indiana.edu>

References


Examples

```r
library('LCMCR')

###Using Kosovo data.###
data(kosovo_aggregate)

###Example 1: Capture-Recapture estimation using convenience functions###
#Create and initialize an LCMCR object for MCMC sampling#
## Not run:
sampler <- lcmCR(captures = kosovo_aggregate, tabular = FALSE, in_list_label = '1',
                 not_in_list_label = '0', K = 10, a_alpha = 0.25, b_alpha = 0.25,
                 seed = 'auto', buffer_size = 10000, thinning = 100)
#Obtain 1000 samples from the posterior distribution of N#
N <- lcmCR_PostSampl(sampler, burnin = 10000, samples = 1000, thinning = 100, output = FALSE)
#Posterior quantiles#
```
### Example 2: Capture-Recapture estimation using the lcm_CR_Basic object directly ###

Create and initialize an LCMCR object for MCMC sampling:

```
sampler <- lcmCR(captures = kosovo_aggregate, tabular = FALSE, in_list_label = '1',
                 not_in_list_label = '0', K = 10, a_alpha = 0.25, b_alpha = 0.25,
                 seed = 'auto', buffer_size = 1000, thinning = 100)
```

Run 10000 iterations as burn-in:

```
sampler$Update(10000, output = FALSE)
```

List all parameters from the model:

```
sampler$Get_Param_List()
```

Set parameter 'n0' for tracing:

```
sampler$Set_Trace('n0')
```

List currently traced parameters:

```
sampler$Get_Trace_List()
```

Activate tracing:

```
sampler$Activate_Tracing()
```

Run the sampler 100000 times:

```
sampler$Update(100000, output = FALSE)
```

Get the 1000 samples from the posterior distribution of N:

```
N <- sampler$Get_Trace('n0') + sampler$n
```

Plot the trace of N:

```
plot(N, type = 'l')
```

Compute posterior quantiles:

```
quantile(N, c(0.025, 0.5, 0.975))
```

## End(Not run)
**Format**

A data frame with 4400 observations on the following 4 variables.

- **EXH** a factor with levels 0 1
- **ABA** a factor with levels 0 1
- **OSCE** a factor with levels 0 1
- **HRW** a factor with levels 0 1

**Details**

This data set was analyzed by Ball et al. (2002).

**References**


**Examples**

```r
data(kosovo_aggregate)
```

---

**1cmCR**

Bayesian Nonparametric Latent Class Capture-Recapture

**Description**

Create and initialize an object of class `1cm_CR_Basic`.

**Usage**

```r
1cmCR(captures, tabular = FALSE, in_list_label = "1", not_in_list_label = "0", K = 5, a_alpha = 0.25, b_alpha = 0.25, buffer_size = 10000, thinning = 10, seed = "auto", verbose = TRUE)
```

**Arguments**

- **captures** input dataset. A data frame with the multiple-recapture data. See 'Details' for input formats.
- **tabular** a logical value indicating whether or not the data is tabulated. See 'Details'.
- **in_list_label** factor label that indicates that individual is in list (e.g. 'Yes')
- **not_in_list_label** factor label that indicates that individual is in not list (e.g. 'No')
- **K** maximum number of latent classes. Indicates the truncation level of the stick-breaking process.
- **a_alpha** shape parameter of the prior distribution of concentration parameter of the stick-breaking process.
**Details**

Input data must be provided as a data frame. The first J columns are two-level factors representing the multiple-recapture lists. Arguments `in_list_label` and `not_in_list_label` indicate the labels that represent inclusion and exclusion from the lists. This function supports two input formats:

- When `tabular=FALSE` each row represents a single individual’s capture history. The number of rows must match the size of the observed population. Rows indicating no capture in all list simultaneously are illegal.
- When `tabular=TRUE` each row represents a unique capture pattern. This format requires an additional numeric column at the right, called "Freq", indicating the count corresponding to such pattern.

**Value**

An object of class `lcm_CR_Basic` initialized and ready to use.

**Author(s)**

Daniel Manrique-Vallier

**See Also**

`lcm_CR_Basic`, `lcm_CR_Basic_generator`

**Examples**

```r
require('LCMCR')
data(kosovo_aggregate)
sampler <- lcmCR(captures = kosovo_aggregate, tabular = FALSE, in_list_label = '1', not_in_list_label = '0', K = 10, a_alpha = 0.25, b_alpha = 0.25, seed = 'auto', buffer_size = 10000, thinning = 100)
sampler
N <- lcmCR_PostSampl(sampler, burnin = 10000, samples = 1000, thinning = 100, output = FALSE)
quantile(N, c(0.025, 0.5, 0.975))
```
**Description**

Convenience function for generate samples from the posterior distribution of the population size using an initialized `lcm_CR_Basic` object.

**Usage**

```r
lcmCR_PostSampl(object, burnin = 10000, samples = 1000, thinning = 10,
                 clear_buffer = FALSE, output = TRUE)
```

**Arguments**

- `object`: an initialized `lcm_CR_Basic` object.
- `burnin`: number of burn in iterations.
- `samples`: number of samples to be generated. Note that this is not the same as the number of iterations for the sampler. Samples are saved one every `thinning` iterations.
- `thinning`: subsampling interval. Samples are saved one every `thinning` iterations.
- `clear_buffer`: logical. Clear the tracing buffer before sampling?
- `output`: logical. Print messages?

**Value**

A vector with the samples posterior samples of the population size parameter.

**Warning**

Invoking this function deletes the content of the object’s tracing buffer.

**Note**

To create and initialize the `lcm_CR_Basic` object use `lcmCR` or `lcm_CR_Basic_generator`. The user is responsible to check whether the chain has reached the stationary distribution or not.

**Author(s)**

Daniel Manrique-Vallier

**Examples**

```r
data(kosovo_aggregate)
sampler <- lcmCR(captures = kosovo_aggregate, tabular = FALSE, in_list_label = '1',
                 not_in_list_label = '0', K = 10, a_alpha = 0.25, b_alpha = 0.25, seed = 'auto')
N <- lcmCR_PostSampl(sampler, burnin = 10000, samples = 1000, thinning = 100, output = FALSE)
quantile(N, c(0.025, 0.5, 0.975))
```
Description

MCMC sampler for the Bayesian non-parametric latent class capture-recapture model.

Extends

Class "MCMCenviron", directly. All reference classes extend and inherit methods from "envRefClass".

Fields

All fields are read-only.

- **pointer**: external pointer to the C++ object.
- **blobsize**: size (in bytes) of the raw object data for serialization. (currently not implemented.)
- **local_seed**: seed of the internal random number generator.
- **J**: number of lists in the Capture-Recapture data.
- **K**: maximum number of latent classes in the model (truncation level of the stick-breaking process).
- **n**: observed number of individuals.
- **Captures**: original provided data.

Methods

initialize(data_captures, K, a_alpha, b_alpha, in_list_symbol, len_buffer, subsamp):

Class constructor.

**Arguments:**
- **data_captures**: input dataset. A data frame with the multiple-recapture data.
- **K**: maximum number of latent classes. Indicates the truncation level of the stick-breaking process.
- **a_alpha**: shape parameter of the prior distribution of concentration parameter of the stick-breaking process.
- **b_alpha**: inverse scale parameter of the prior distribution of concentration parameter of the stick-breaking process.
- **in_list_symbol**: factor label that indicates that individual is in list (e.g. 'Yes')
- **buffer_size**: Size of the tracing buffer.
- **subsamp**: thinning interval for the tracing buffer.
- **verbose**: logical. Generate progress messages?

The following methods are inherited (from the corresponding class): Change_SubSamp ("MCMCenviron"), Set_Trace ("MCMCenviron"), Change_Trace_Length ("MCMCenviron"), initialize ("MCMCenviron"), Get_Iteration ("MCMCenviron"), Get_Param ("MCMCenviron"), Reset_Traces ("MCMCenviron"), Get_Status ("MCMCenviron"), Update ("MCMCenviron"), Get_Trace_Size ("MCMCenviron"), Get_Trace ("MCMCenviron"), Get_Trace_List ("MCMCenviron"), Get_Param_List ("MCMCenviron"), Init_Model ("MCMCenviron"), Activate_Tracing ("MCMCenviron"), Deactivate_Tracing ("MCMCenviron"), Set_Seed ("MCMCenviron"), show ("MCMCenviron")
Note
Use the convenience function `lcmCR` to create objects of this class. This class inherits most of its functionality from "MCMCenviron".

Author(s)
Daniel Manrique-Vallier

See Also
`lcmCR`, `MCMCenviron`.

Examples
```r
showClass("lcm_CR_Basic")
```

---

**lcm_CR_Basic_generator**

*Generator for Class lcm_CR_Basic*

**Description**
Generator function for class `lcm_CR_Basic`.

**Usage**
`lcm_CR_Basic_generator(...)`

**Arguments**
`...` arguments to be passed to `lcm_CR_Basic` constructor.

**Value**
An object of class `lcm_CR_Basic`.

**Note**
The convenience function `lcmCR` provides a simpler mechanism to create `lcm_CR_Basic` objects.

**Author(s)**
Daniel Manrique-Vallier.

**See Also**
`lcmCR`
Examples

```r
data(kosovo_aggregate)
x <- lcm_CR_Basic_generator(data_captures=kosovo_aggregate, K=10, a_alpha=0.25, b_alpha=0.25,
len_buffer=10000, subsamp=500, in_list_symbol = '1')
x$Get_Status()
```

---

**MCMCenviron-class**

### Class "MCMCenviron"

#### Description

A generic interface for MCMC sampler objects implementing Bayesian models. Methods provide access to underlying functionality implemented in C++. The underlying implementation provides basic functionality for controlling the chain, and a 'tracing buffer' for storing and retrieving the samples.

#### Extends

All reference classes extend and inherit methods from "envRefClass".

#### Fields

(All fields are read-only.)

- `pointer`: external pointer to the C++ object
- `blobsize`: size (in bytes) of the raw object data for serialization. (currently not implemented.)
- `seed`: seed of the internal random number generator.

#### Methods

**GENERAL METHODS**

- **Init_Model(output = TRUE, seed=c('auto', 'r.seed'))**: Initializes the sampler.
  
  **Arguments:**
  
  - `output`: logical. Print messages to the screen?
  - `seed`: integer. Seed of the internal RNG. Additionally, `seed='auto'` autogenerates the seed from the internal clock; `seed='r.seed'` autogenerates the seed from the current state of the .Random.seed variable.

- **Update(num_iter, output = TRUE)**: Runs `num_iter` iterations of the sampler. Set `output = FALSE` to suppress console output.

- **Get_Iteration()**: Retrieves the current number of iterations the sampler.

- **Get_Param_List()**: Retrieves the names of the parameters of the model.

- **Get_Param(param)**: Retrieves the current value of the parameter `param`.

- **Set_Seed(seed)**: Seeds the internal random number generator. It does not affect R’s internal RNG.
Get_Status(): Retrieves the current state of the chain

Value:
iteration numeric. Current iteration
initialized logical. Is the sampler initialized?
buffer_size numeric. Capacity (in samples) of the tracing buffer.
buffer_used numeric. Number of samples currently stored in the tracing buffer.
tracing character. Names of the variables currently traced.
thinning numeric. Thinning interval of the tracing buffer.

METHODS FOR CONTROLLING THE TRACING BUFFER

Get_Trace_List(): Retrieves the names of the parameters being currently traced.

Activate_Tracing(): Activates the tracing buffer. Traced variables will be stored in the buffer when generated with Update().

Deactivate_Tracing(): Deactivates the tracing buffer. Calls to Update() will not store samples in the buffer.

Set_Trace(traces): Adds parameters to tracer.

Arguments:
param: character vector. Names of the parameters to trace. To list the available parameters for tracing use the Get_Param_List() method.

Get_Trace(param): Retrieves samples stored in the tracing buffer.

Arguments:
param: character. Name of the parameter to retrieve.

Value: An array. The first dimension indexes the sample; the rest correspond to the original dimensions of the parameter as defined in the model.

Reset_Traces(): Deletes the content of the tracing buffer.

Change_SubSamp(new_subsamp): Changes the sub-sampling period (thinning) of the tracing buffer.

Warning: This operation deletes the current content of the tracing buffer.

Get_Trace_Size(): Retrieves the size (in number of samples) of the trace buffer.

Change_Trace_Length(new_length): Changes the size (in number of samples) of the tracing buffer.

Warning: This operation deletes the current content of the tracing buffer.

Note
This class is not designed to be used directly, but as a generic interface for samplers implementing specific models.

Author(s)
Daniel Manrique-Vallier

Examples
showClass("MCMCenviron")
Index

*Topic capture-recapture
  lcmCR, 4
*Topic classes
  lcm_CR_Basic-class, 7
*Topic datasets
  kosovo_aggregate, 3

envRefClass, 7, 9

kosovo_aggregate, 3

lcm_CR_Basic, 5, 6, 8
lcm_CR_Basic (lcm_CR_Basic-class), 7
lcm_CR_Basic-class, 7
lcm_CR_Basic_generator, 5, 6, 8
LCMCR (LCMCR-package), 2
lcmCR, 4, 6, 8
LCMCR-package, 2
lcmCR_PostSampl, 6

MCMCenviron, 7, 8
MCMCenviron-class, 9