Package ‘bayesvl’

May 22, 2019

Type Package

Title Visually Learning the Graphical Structure of Bayesian Networks and Performing MCMC with 'Stan'

Version 0.8.4

Date 2019-05-20

Author Viet-Phuong La [aut, cre], Quan-Hoang Vuong [aut]

Maintainer Viet-Phuong La <lvphuong@gmail.com>

Imports coda, bnlearn, ggplot2, bayesplot, viridis, reshape2, dplyr

Suggests loo (>= 2.0.0)

Depends R (>= 3.4.0), rstan (>= 2.10.0), StanHeaders (>= 2.18.0), stats, graphics, methods

Description Provides users with its associated functions for pedagogical purposes in visually learning Bayesian networks and Markov chain Monte Carlo (MCMC) computations. It enables users to: a) Create and examine the (starting) graphical structure of Bayesian networks; b) Create random Bayesian networks using a dataset with customized constraints; c) Generate Stan code for structures of Bayesian networks for sampling the data and learning parameters; d) Plot the network graphs; e) Perform Markov chain Monte Carlo computations and produce graphs for posteriors checks. The package refers to one reference item, which describes the methods and algorithms: Vuong, Quan-Hoang and La, Viet-Phuong (2019) <doi:10.31219/osf.io/w5dx6> The 'bayesvl' R package. Open Science Framework (May 18).

License GPL (>= 3)

BugReports `https://github.com/sshpa/bayesvl/issues`

URL `https://github.com/sshpa/bayesvl`

NeedsCompilation no

R topics documented:

`bayesvl-package` *BayesVL package for Bayesian statistical analyses in R*

Description

The R package for visually learning the graphical structures of Bayesian networks, and performing Hamiltonian MCMC with Stan through `bvl_model2Stan`, `bvl_modelFit`

Details

Package:	bayesvl
Type:	Package
Version:	0.8.0
Date:	13 May 2019
License:	GPL-3
Website:	Bayesvl

Author(s)

Quan-Hoang Vuong, Viet-Phuong La

References

For documentation, case studies and worked examples, and other tutorial information visit the References section on our Github:

- github.com/sshpa/bayesvl/master/References/bvl_ug_en08.pdf

See Also

`bayesvl-class`, `bvl_modelFit`, `bvl_model2Stan`

Examples

```
# Design the model in directed acyclic graph
model <- bayesvl()

# add observed data nodes to the model
model <- bvl_addNode(model, "Lie", "binom")
model <- bvl_addNode(model, "B", "binom")
model <- bvl_addNode(model, "C", "binom")
model <- bvl_addNode(model, "T", "binom")

# add path between nodes
model <- bvl_addArc(model, "B", "Lie", "slope")
model <- bvl_addArc(model, "C", "Lie", "slope")
model <- bvl_addArc(model, "T", "Lie", "slope")
```

```
summary(model)
```

`bayesvl bnlearn utilities` *bnlearn interface for bayesvl objects*

Description

Provides the interface to the functions in the bnlearn package for network diagnostics of an object of class `bayesvl`.

Usage

```
# Interface to bn.fit function to fit the parameters of
# a Bayesian network conditional on its structure.
bvl_bnBayes(dag, data = NULL, method = "bayes", iss = 10, ...)

# Interface to bnlearn score function to compute the score of the Bayesian network.
bvl_bnScore(dag, data = NULL, ...)

# Interface to arc.strength function to measure the strength of the probabilistic
# relationships expressed by the arcs of a Bayesian network.
bvl_bnStrength(dag, data = NULL, criterion = "x2", ...)

# Interface to bn.fit.barchart function to plot fit
# the parameters of a Bayesian network conditional on its structure.
bvl_bnBarchart(dag, data = NULL, method = "bayes", iss = 10, ...)
```

Arguments

`dag`	an object of class `bayesvl`
`data`	a data frame containing the variables in the model.
`method`	a character string, either mle for Maximum Likelihood parameter estimation or bayes for Bayesian parameter estimation (currently implemented only for discrete data).
`iss`	a numeric value, the imaginary sample size used by the bayes method to estimate the conditional probability tables associated with discrete nodes
`criterion`	a character string, the method using for measuring
`...`	extra arguments from the generic method

Value

`bvl_bnScore()` return a number, value of score.

Author(s)

La Viet-Phuong, Vuong Quan-Hoang

References

For documentation, case studies and worked examples, and other tutorial information visit the References section on our Github:

- github.com/sshpa/bayesvl/master/References/bvl_ug_en08.pdf

`bayesvl graph utilities` *Utilities to manipulate graphs*

Description

Manipulate directed acyclic graph of an object of class `bayesvl`.

Usage

```
# added a new node to the graph.
bvl_addNode(dag, name, dist = "norm", priors = NULL, fun = NULL, out_type = NULL,
  lower = NULL, upper=NULL, test = NULL)

# added a new path between nodes to the graph.
bvl_addArc(dag, from, to, type = "slope", priors = NULL, fun = NULL)

# added a new path between nodes to the graph.
bvl_addArc(dag, from, to, type = "slope", priors = NULL, fun = NULL)
```

Arguments

`dag`	an object of class `bayesvl`
`name`	a character string, the name of a node.
`dist`	a character string, distribution code of the node (`norm`, `binom`).
`priors`	a vector of string, the priors of the node or path.
`fun`	a character string, the transform function of the node.
`out_type`	a character string, the variable data type (`int`, `real`, ...).
`lower`	integer or real, the lower bound of variable data type (`int` or `real`).
`upper`	integer or real, the upper bound of variable data type (`int` or `real`).
`test`	a vector of testing values for variable.
`from`	a character string, the name of node the path connect from.
`to`	a character string, the name of node the path connect to.
`type`	a character string, the path type between nodes (`slope`, `varint`, ...).

Value

`bvl_addNode()`, `bvl_addArc()` return object class `bayesvl`.

Author(s)

La Viet-Phuong, Vuong Quan-Hoang

References

For documentation, case studies and worked examples, and other tutorial information visit the References section on our Github:

- github.com/sshpa/bayesvl/master/References/bvl_ug_en08.pdf

Examples

```
dag = bayesvl()

# add nodes to dag
dag = bvl_addNode(dag, "node1")
dag = bvl_addNode(dag, "node2")

# add the path between two nodes
dag = bvl_addArc(dag, "node1", "node2")

summary(dag)
```

---

bayesvl plot utilities

Plot utilities for bayesvl objects

Description

Provides plot methods and the interface to the MCMC module in the bayesplot package for plotting MCMC draws and diagnostics for an object of class `bayesvl`.

Usage

```
# Plot network diagram to visualize the model
bvl_bnPlot(dag, ...)

# Plots historgram of regression parameters computed from posterior draws in grid layout
bvl_plotParams (dag, row = 2, col = 2, credMass = 0.89, params = NULL)

# The interface to mcmc_intervals for plotting uncertainty intervals
# computed from posterior draws
bvl_plotIntervals (dag, params = NULL, fun = "stat", stat = "mean",
  prob = 0.8, prob_outer = 0.95,
  color_scheme = "blue", labels = NULL)

# The interface to mcmc_intervals for plotting density computed from posterior draws
bvl_plotAreas (dag, params = NULL, fun = "stat", stat = "mean",
  prob = 0.8, prob_outer = 0.95,
  color_scheme = "blue", labels = NULL)
```

```
bvl_plotPairs (dag, params = NULL, fun = "stat", stat = "mean",
  prob = 0.8, prob_outer = 0.95,
  color_scheme = "blue", labels = NULL)

bvl_plotDensity (dag, params = NULL, size = 1, labels = NULL)

bvl_plotDensity2d(dag, x, y, color = NULL, color_scheme = "red", labels = NULL)

bvl_plotTrace (dag, params = NULL)

bvl_plotDiag (dag)

bvl_plotGelman (dag, params = NULL)

bvl_plotGelmans (dag, params = NULL, row = 2, col = 2)

bvl_plotAcfs ( dag, params = NULL, row = 2, col = 2)

bvl_plotTest (dag, y_name, test_name, n = 200, color_scheme = "blue")
```

Arguments

`dag`	an object of class `bayesvl`
`params`	Optional: character vector of parameter names.
`fun`	Optional: statistic function.
`stat`	Optional: the plotting function to call.
`prob`	Optional: the probability mass to include in the inner interval. Default is 0.8.
`prob_outer`	Optional: the probability mass to include in the outer interval. Default is 0.95.
`row`	Optional: number of rows of grid layout.
`col`	Optional: number of columns of grid layout.
`credMass`	Optional: specifying the mass within the credible interval. Default is 0.89.
`size`	Optional: the size of line width.
`color_scheme`	Optional: color scheme. Default is "blue"
`...`	extra arguments from the generic method
`y_name`	a character string. Name of outcome variable
`test_name`	a character string. Name of test variable and test value
`n`	number of yrep values to plot
`x`	a character string. Name of x parameter to pair with
`y`	a character string. Name of y parameter to pair with
`color`	a character string. Variable for color of points on density plot
`labels`	Optional: character vector of parameter labels.

Value

`bvl_plotIntervals(), bvl_plotPairs()` return a ggplot object that can be further customized using the ggplot2 package.

Author(s)

La Viet-Phuong, Vuong Quan-Hoang

References

For documentation, case studies and worked examples, and other tutorial information visit the References section on our Github:

- github.com/sshpa/bayesvl/master/References/bvl_ug_en08.pdf

Examples

```
## create network model
model <- bayesvl()
## add the observed data nodes
model <- bvl_addNode(model, "O", "binom")
model <- bvl_addNode(model, "Lie", "binom")
model <- bvl_addNode(model, "Viol", "binom")
model <- bvl_addNode(model, "VB", "binom")
model <- bvl_addNode(model, "VC", "binom")
model <- bvl_addNode(model, "VT", "binom")
model <- bvl_addNode(model, "Int1", "binom")
model <- bvl_addNode(model, "Int2", "binom")

## add the tranform data nodes and arcs as part of the model
model <- bvl_addNode(model, "B_and_Viol", "trans")
model <- bvl_addNode(model, "C_and_Viol", "trans")
model <- bvl_addNode(model, "T_and_Viol", "trans")
model <- bvl_addArc(model, "VB",         "B_and_Viol", "*")
model <- bvl_addArc(model, "Viol",       "B_and_Viol", "*")
model <- bvl_addArc(model, "VC",         "C_and_Viol", "*")
model <- bvl_addArc(model, "Viol",       "C_and_Viol", "*")
model <- bvl_addArc(model, "VT",         "T_and_Viol", "*")
model <- bvl_addArc(model, "Viol",       "T_and_Viol", "*")
model <- bvl_addArc(model, "B_and_Viol",  "O", "slope")
model <- bvl_addArc(model, "C_and_Viol",  "O", "slope")
model <- bvl_addArc(model, "T_and_Viol",  "O", "slope")

model <- bvl_addArc(model, "Viol",   "O", "slope")

model <- bvl_addNode(model, "B_and_Lie", "trans")
model <- bvl_addNode(model, "C_and_Lie", "trans")
model <- bvl_addNode(model, "T_and_Lie", "trans")
model <- bvl_addArc(model, "VB",        "B_and_Lie", "*")
model <- bvl_addArc(model, "Lie",       "B_and_Lie", "*")
model <- bvl_addArc(model, "VC",        "C_and_Lie", "*")
model <- bvl_addArc(model, "Lie",       "C_and_Lie", "*")
model <- bvl_addArc(model, "VT",        "T_and_Lie", "*")
model <- bvl_addArc(model, "Lie",       "T_and_Lie", "*")
model <- bvl_addArc(model, "B_and_Lie",  "O", "slope")
model <- bvl_addArc(model, "C_and_Lie",  "O", "slope")
model <- bvl_addArc(model, "T_and_Lie",  "O", "slope")

model <- bvl_addArc(model, "Lie",   "O", "slope")
model <- bvl_addNode(model, "Int1_or_Int2", "trans")
model <- bvl_addArc(model, "Int1", "Int1_or_Int2", "+")
model <- bvl_addArc(model, "Int2", "Int1_or_Int2", "+")

model <- bvl_addArc(model, "Int1_or_Int2", "O", "varint")

## Plot network diagram to visualize the model
bvl_bnPlot(model)
```

bayesvl stan utilities

Build RStan models from directed acyclic graph

Description

Build Stan models from directed acyclic graph of an object of class `bayesvl`.

Usage

```
# build Stan models from directed acyclic graph.
bvl_model2Stan(dag, ppc = "")

# compile and simulate samples from the model.
bvl_modelFit(dag, data, warmup = 1000, iter = 5000, chains = 2, ppc = "", ...)

# summarize the stan priors used for the model.
bvl_stanPriors(dag)

# summarize the stan parameters used for the model.
bvl_stanParams(dag)

# summarize the generated formula at the node.
bvl_formula(dag, nodeName, outcome = T, re = F)
```

Arguments

`dag`	an object of class `bayesvl`
`data`	a data frame or list containing the data
`warmup`	Optional: Number of warmup iterations. By default, half of iter
`iter`	Optional: Number of iterations of sampling. Default is 5000
`chains`	Optional: Number of independent chains to sample from. Default is 2
`ppc`	Optional: a character string contains posterior predictive check scripts
`...`	extra arguments from the generic method
`nodeName`	A character string contains the node name
`outcome`	Optional: Whether show out distribution
`re`	Optional: Whether run recursive for all up-level nodes

Value

`bvl_model2Stan()` return character string of rstan code generated from the model.

`bvl_modelFit()` return an object class `bayesvl` which contains result with the following slots.

`model`	Stan model code
`stanfit`	`stanfit` object returned by `stan`
`standata`	The data
`pars`	Parameter names monitored in samples
`formula`	Generated formula from the model

`bvl_stanPriors()` return character string of rstan priors generated from the model.

`bvl_stanParams()` return character string of rstan parameters generated from the model.

Author(s)

La Viet-Phuong, Vuong Quan-Hoang

References

For documentation, case studies and worked examples, and other tutorial information visit the References section on our Github:

- github.com/sshpa/bayesvl/master/References/bvl_ug_en08.pdf

Examples

```
# Design the model in directed acyclic graph
model <- bayesvl()
model <- bvl_addNode(model, "Lie", "binom")
model <- bvl_addNode(model, "B", "binom")
model <- bvl_addNode(model, "C", "binom")
model <- bvl_addNode(model, "T", "binom")

model <- bvl_addArc(model, "B", "Lie", "slope")
model <- bvl_addArc(model, "C", "Lie", "slope")
model <- bvl_addArc(model, "T", "Lie", "slope")

# Generate the Stan model's code
model_string <- bvl_model2Stan(model)
cat(model_string)

# Show priors in generated Stan model
bvl_stanPriors(model)
```

bayesvl-class *Class* bayesvl*: object class of bayesvl model*

Description

This object contains an object of class bayesvl, return by bayesvl.

Slots

`call`: Original function call that produced the fit

`nodes`: The list of nodes in model

`arcs`: The list of arcs in model

`pars`: List of parameters

`stanfit`: Object of class `stanfit`

`rawdata`: Observed data frame

`standata`: Data used in sampling

`posterior`: Coerce to a Data Frame of object of class `stanfit`

`elapsed`: Elapsed time of MCMC simulation

Methods

`show signature(object = "bayesvl")`: print the default summary for the model.

`summary` Displays the `model` slot.

References

For documentation, case studies and worked examples, and other tutorial information visit the References section on our Github:

- github.com/sshpa/bayesvl/master/References/bvl_ug_en08.pdf

See Also

bayesvl

Examples

```
# Design the model in directed acyclic graph
model <- bayesvl()

# add observed data nodes to the model
model <- bvl_addNode(model, "Lie", "binom")
model <- bvl_addNode(model, "B", "binom")
model <- bvl_addNode(model, "C", "binom")
model <- bvl_addNode(model, "T", "binom")

# add path between nodes
model <- bvl_addArc(model, "B", "Lie", "slope")
model <- bvl_addArc(model, "C", "Lie", "slope")
model <- bvl_addArc(model, "T", "Lie", "slope")

  summary(model)
```

`Legends345` *Legends345 data*

Description

Legends345.

Usage

```
data(Legends345)
```

Format

1. O : Whether or not happy ending for main character
2. VB : Whether or not the main character behaves in accordance with the core values of Buddhism
3. VC : Whether or not the main character behaves in accordance with the core values of Confucianism
4. VT : Whether or not the main character behaves in accordance with the core values of Taoism
5. Lie : Whether or not the main character tells lie
6. Viol : Whether or not the main character commits acts of violence
7. Int1 : Whether there are interventions from the supernatural world
8. Int2 : Whether there are interventions from the human world

References

For documentation, case studies and worked examples, and other tutorial information visit the References section on our Github:

- github.com/sshpa/bayesvl/master/References/bvl_ug_en08.pdf

Examples

```
data(Legends345)

data1 <- Legends345
head(data1)
```

Index