

# Package: IsingSampler (via r-universe)

July 18, 2024

**Type** Package

**Title** Sampling Methods and Distribution Functions for the Ising Model

**Version** 0.2.3

**Maintainer** Sacha Epskamp <mail@sachaepskamp.com>

**Description** Sample states from the Ising model and compute the probability of states. Sampling can be done for any number of nodes, but due to the intractability of the Ising model the distribution can only be computed up to ~10 nodes.

**License** GPL-2

**Imports** plyr, magrittr, nnet, dplyr

**Depends** Rcpp (>= 0.10.4), R (>= 3.0.0)

**LinkingTo** Rcpp

**URL** [github.com/SachaEpskamp/IsingSampler](https://github.com/SachaEpskamp/IsingSampler)

**NeedsCompilation** yes

**Author** Sacha Epskamp [aut, cre], Jesse Boot [ctb]

**Repository** CRAN

**Date/Publication** 2023-08-21 09:42:33 UTC

## Contents

IsingSampler-package . . . . .	2
EstimateIsing . . . . .	3
IsingEntropy . . . . .	5
IsingLikelihood . . . . .	6
IsingPL . . . . .	7
IsingSampler . . . . .	7
IsingStateProb . . . . .	9
IsingSumLikelihood . . . . .	9
LinTransform . . . . .	10

<b>Index</b>	<b>12</b>
--------------	-----------

---

IsingSampler-package    *Sampling methods and distribution functions for the Ising model*

---

## Description

This package can be used to sample states from the Ising model and compute the probability of states. Sampling can be done for any number of nodes, but due to the intractability of the Ising model the distribution can only be computed up to ~10 nodes.

## Author(s)

Sacha Epskamp

Maintainer: Sacha Epskamp <mail@sachaepskamp.com>

## Examples

```
## This code compares the different sampling algorithms to the expected
## distribution of states in a tractible number of nodes.

## In the end are examples on how to obtain the distribution.

# Input:
N <- 5 # Number of nodes
nSample <- 5000 # Number of samples

# Ising parameters:
Graph <- matrix(sample(0:1,N^2,TRUE,prob = c(0.7, 0.3)),N,N) * rnorm(N^2)
Graph <- pmax(Graph,t(Graph)) / N
diag(Graph) <- 0
Thresh <- -(rnorm(N)^2)
Beta <- 1

# Response options (0,1 or -1,1):
Resp <- c(0L,1L)

# All possible states:
AllStates <- do.call(expand.grid,lapply(1:N,function(x)Resp))

# Simulate with metropolis:
MetData <- IsingSampler(nSample, Graph, Thresh, Beta, 1000/N,
  responses = Resp, method = "MH")

# Simulate exact samples (CFTP):
ExData <- IsingSampler(nSample, Graph, Thresh, Beta, 100,
  responses = Resp, method = "CFTP")

# Direct simulation:
DirectData <- IsingSampler(nSample, Graph, Thresh, Beta, method = "direct")
```

```

# State distributions:
MetDist <- apply(AllStates,1,function(x)sum(colSums(t(MetData) == x)==N))
ExDist <- apply(AllStates,1,function(x)sum(colSums(t(ExData) == x)==N))
DirectDist <- apply(AllStates,1,function(x)sum(colSums(t(DirectData) == x)==N))
ExpDist <- exp(- Beta * apply(AllStates,1,function(s)IsingSampler:::H(Graph,s,Thresh)))
ExpDist <- ExpDist/sum(ExpDist) * nSample

## Plot to compare distributions:
plot(MetDist, type="l", col="blue", pch=16, xlab="State", ylab="Freq",
     ylim=c(0,max(MetDist,DirectDist,ExDist)))
points(DirectDist,type="l",col="red",pch=16)
points(ExpDist,type="l",col="green",pch=16)
points(ExDist,type="l",col="purple",pch=16)
legend("topright", col=c("blue","red","purple","green"),
     legend=c("Metropolis","Direct","Exact","Expected"),lty=1,bty='n')

## Likelihoods:

# Sumscores:
IsingSumLikelihood(Graph, Thresh, Beta, Resp)

# All states:
IsingLikelihood(Graph, Thresh, Beta, Resp)

# Single state:
IsingStateProb(rep(Resp[1],N),Graph, Thresh, Beta, Resp)

```

---

EstimateIsing

*non-regularized estimation methods for the Ising Model*


---

## Description

This function can be used for several non-regularized estimation methods of the Ising Model. See details.

## Usage

```

EstimateIsing(data, responses, beta = 1, method = c("uni", "pl",
           "bi", "ll"), adj = matrix(1, ncol(data), ncol(data)),
           ...)
EstimateIsingUni(data, responses, beta = 1, adj = matrix(1, ncol(data),
           ncol(data)), min_sum = -Inf, thresholding = FALSE, alpha = 0.01,
           AND = TRUE, ...)
EstimateIsingBi(data, responses, beta = 1, ...)
EstimateIsingPL(data, responses, beta = 1, ...)
EstimateIsingLL(data, responses, beta = 1, adj = matrix(1, ncol(data),
           ncol(data)), ...)

```

**Arguments**

data	Data frame with binary responses to estimate the Ising model over
responses	Vector of length two indicating the response coding (usually $c(0L, 1L)$ or $c(-1L, 1L)$ )
beta	Inverse temperature parameter
method	The method to be used. <code>pl</code> uses pseudolikelihood estimation, <code>uni</code> sequential univariate regressions, <code>bi</code> bivariate regressions and <code>ll</code> estimates the Ising model as a loglinear model.
adj	Adjacency matrix of the Ising model.
min_sum	The minimum sum score that is artificially possible in the dataset. Defaults to <code>-Inf</code> . Set this only if you know a lower sum score is not possible in the data, for example due to selection bias.
thresholding	Logical, should the model be thresholded for significance?
alpha	Alpha level used in thresholding
AND	Logical, should an AND-rule (both regressions need to be significant) or OR-rule (one of the regressions needs to be significant) be used?
...	Arguments sent to estimator functions

**Details**

The following algorithms can be used (see Epskamp, Maris, Waldorp, Borsboom; in press).

- `pl` Estimates the Ising model by maximizing the pseudolikelihood (Besag, 1975).
- `uni` Estimates the Ising model by computing univariate logistic regressions of each node on all other nodes. This leads to a single estimate for each threshold and two estimates for each network parameter. The two estimates are averaged to produce the final network. Uses [glm](#).
- `bi` Estimates the Ising model using multinomial logistic regression of each pair of nodes on all other nodes. This leads to a single estimate of each network parameter and  $\$p\$$  estimates of each threshold parameter. Uses [multinom](#).
- `ll` Estimates the Ising model by phrasing it as a loglinear model with at most pairwise interactions. Uses [loglin](#).

**Value**

A list containing the estimation results:

graph	The estimated network
thresholds	The estimated thresholds
results	The results object used in the analysis

**Author(s)**

Sacha Epskamp (mail@sachaepskamp.com)

## References

Epskamp, S., Maris, G., Waldorp, L. J., and Borsboom, D. (in press). Network Psychometrics. To appear in: Irwing, P., Hughes, D., and Booth, T. (Eds.), *Handbook of Psychometrics*. New York: Wiley.

Besag, J. (1975), Statistical analysis of non-lattice data. *The statistician*, 24, 179-195.

## Examples

```
# Input:
N <- 5 # Number of nodes
nSample <- 500 # Number of samples

# Ising parameters:
Graph <- matrix(sample(0:1,N^2,TRUE,prob = c(0.7, 0.3)),N,N) * rnorm(N^2)
Graph <- Graph + t(Graph)
diag(Graph) <- 0
Thresholds <- rep(0,N)
Beta <- 1

# Response options (0,1 or -1,1):
Resp <- c(0L,1L)
Data <- IsingSampler(nSample,Graph, Thresholds)

# Pseudolikelihood:
resPL <- EstimateIsing(Data, method = "pl")
cor(Graph[upper.tri(Graph)], resPL$graph[upper.tri(resPL$graph)])

# Univariate logistic regressions:
resUni <- EstimateIsing(Data, method = "uni")
cor(Graph[upper.tri(Graph)], resUni$graph[upper.tri(resUni$graph)])

# bivariate logistic regressions:
resBi <- EstimateIsing(Data, method = "bi")
cor(Graph[upper.tri(Graph)], resBi$graph[upper.tri(resBi$graph)])

# Loglinear model:
resLL <- EstimateIsing(Data, method = "ll")
cor(Graph[upper.tri(Graph)], resLL$graph[upper.tri(resLL$graph)])
```

---

IsingEntropy

*Entropy of the Ising Model*

---

## Description

Returns (marginal/conditional) Shannon information of the Ising model.

## Usage

```
IsingEntropy(graph, thresholds, beta = 1, conditional = numeric(0),
             marginalize = numeric(0), base = 2, responses = c(0L, 1L))
```

**Arguments**

graph	Weights matrix
thresholds	Thresholds vector
beta	Inverse temperature
conditional	Indices of nodes to condition on
marginalize	Indices of nodes to marginalize over
base	Base of the logarithm
responses	Vector of outcome responses.

**Author(s)**

Sacha Epskamp <mail@sachaepskamp.com>

---

IsingLikelihood      *Likelihood of all states from tractible Ising model.*

---

**Description**

This function returns the likelihood of all possible states. Is only tractible up to roughly 10 nodes.

**Usage**

```
IsingLikelihood(graph, thresholds, beta, responses = c(0L, 1L),
                potential = FALSE)
```

**Arguments**

graph	Square matrix indicating the weights of the network. Must be symmetrical with 0 as diagonal.
thresholds	Vector indicating the thresholds, also known as the external field.
beta	Scalar indicating the inverse temperature.
responses	Response options. Typically set to c(-1L, 1L) or c(0L, 1L) (default). Must be integers!
potential	Logical, return the potential instead of the probability of each state?

**Author(s)**

Sacha Epskamp

---

 IsingPL

*Pseudo-likelihood*


---

**Description**

Computes the pseudolikelihood of a dataset given an Ising Model.

**Usage**

```
IsingPL(x, graph, thresholds, beta, responses = c(0L, 1L))
```

**Arguments**

x	A binary dataset
graph	Square matrix indicating the weights of the network. Must be symmetrical with 0 as diagonal.
thresholds	Vector indicating the thresholds, also known as the external field.
beta	Scalar indicating the inverse temperature.
responses	Response options. Typically set to c(-1L, 1L) or c(0L, 1L) (default). Must be integers!

**Value**

The pseudolikelihood

**Author(s)**

Sacha Epskamp (mail@sachaepskamp.com)

---

 IsingSampler

*Sample states from the Ising model*


---

**Description**

This function samples states from the Ising model using one of three methods. See details.

**Usage**

```
IsingSampler(n, graph, thresholds, beta = 1, nIter = 100, responses = c(0L, 1L),
  method = c("MH", "CFTP", "direct"), CFTPretry = 10, constrain)
```

**Arguments**

n	Number of states to draw
graph	Square matrix indicating the weights of the network. Must be symmetrical with 0 as diagonal.
thresholds	Vector indicating the thresholds, also known as the external field.
beta	Scalar indicating the inverse temperature.
nIter	Number of iterations in the Metropolis and exact sampling methods.
responses	Response options. Typically set to c(-1L, 1L) or c(0L, 1L) (default). Must be integers!
method	The sampling method to use. Must be "MH", "CFTP" or "direct". See details.
CFTPretry	The amount of times a sample from CFTP may be retried. If after 100 couplings from the past the chain still results in NA values the chain is reset with different random numbers. Be aware that data that requires a lot of CFTP resets might not resemble exact samples anymore.
constrain	A (number of samples) by (number of nodes) matrix with samples that need be constrained; NA indicates that the sample is unconstrained. Defaults to a matrix of NAs.

**Details**

This function uses one of three sampling methods. "MH" can be used to sample using a Metropolis-Hastings algorithm. The chain is initiated with random values from the response options, then for each iteration for each node a node is set to the second response option with the probability of that node being in the second response option given all other nodes and parameters. Typically, 100 of such iterations should suffice for the chain to converge.

The second method, "CFTP" enhances the Metropolis-Hastings algorithm with Coupling from the Past (CFTP; Murray, 2007) to draw exact samples from the distribution. This is slower than the default Metropolis-Hastings but guarantees exact samples. However, it does depend on the graph structure and the number of nodes if these exact samples can be obtained in feasible time.

The third option, "direct", simply computes for every possibly state the probability and draws samples directly from the distribution of states by using these probabilities. This also guarantees exact samples, but quickly becomes intractible (roughly above 10 nodes).

**Value**

A matrix containing samples of states.

**Author(s)**

Sacha Epskamp (mail@sachaepskamp.com)

**References**

Murray, I. (2007). Advances in Markov chain Monte Carlo methods.



**See Also**

[IsingSampler-package](#) for examples

**Examples**

```
## See IsingSampler-package help page
```

---

IsingStateProb	<i>Likelihood of single state from tractible Ising model.</i>
----------------	---

---

**Description**

This function returns the likelihood of a single possible state. Is only tractible up to roughly 10 nodes.

**Usage**

```
IsingStateProb(s, graph, thresholds, beta, responses = c(0L, 1L))
```

**Arguments**

s	Vector containing the state to evaluate.
graph	Square matrix indicating the weights of the network. Must be symmetrical with 0 as diagonal.
thresholds	Vector indicating the thresholds, also known as the external field.
beta	Scalar indicating the inverse temperature.
responses	Response options. Typically set to c(-1L, 1L) or c(0L, 1L) (default). Must be integers!

**Author(s)**

Sacha Epskamp (mail@sachaepskamp.com)

---

IsingSumLikelihood	<i>Likelihood of sumscores from tractible Ising model.</i>
--------------------	--

---

**Description**

This function returns the likelihood of all possible sumscores. Is only tractible up to roughly 10 nodes.

**Usage**

```
IsingSumLikelihood(graph, thresholds, beta, responses = c(0L, 1L))
```

**Arguments**

graph	Square matrix indicating the weights of the network. Must be symmetrical with 0 as diagonal.
thresholds	Vector indicating the thresholds, also known as the external field.
beta	Scalar indicating the inverse temperature.
responses	Response options. Typically set to $c(-1L, 1L)$ or $c(0L, 1L)$ (default). Must be integers!

**Author(s)**

Sacha Epskamp (mail@sachaepskamp.com)

---

LinTransform	<i>Transform parameters for linear transformations on response categories</i>
--------------	---

---

**Description**

This function is mainly used to translate parameters estimated with response options set to 0 and 1 to a model in which the response options are -1 and 1, but can be used for any linear transformation of response options.

**Usage**

```
LinTransform(graph, thresholds, from = c(0L, 1L), to = c(-1L, 1L), a, b)
```

**Arguments**

graph	A matrix containing an Ising graph
thresholds	A vector containing thresholds
from	The original response encoding
to	The response encoding to transform to
a	The slope of the transformation. Overwrites to.
b	The intercept of the transformation. Overwrites to.

**Author(s)**

Sacha Epskamp <sacha.epskamp@gmail.com>

**Examples**

```
N <- 4 # Number of nodes

# Ising parameters:
Graph <- matrix(sample(0:1,N^2,TRUE,prob = c(0.7, 0.3)),N,N) * rnorm(N^2)
Graph <- pmax(Graph,t(Graph)) / N
diag(Graph) <- 0
Thresh <- -(rnorm(N)^2)
Beta <- 1

p1 <- IsingLikelihood(Graph, Thresh, Beta, c(0,1))

a <- 2
b <- -1

# p2 <- IsingLikelihood(Graph/(a^2), Thresh/a - (b*rowSums(Graph))/a^2, Beta, c(-1,1))

p2 <- IsingLikelihood(LinTransform(Graph,Thresh)$graph,
                      LinTransform(Graph,Thresh)$thresholds ,
                      Beta, c(-1,1))

LinTransform

round(cbind(p1[,1],p2[,1]),5)

plot(p1[,1],p2[,1])
abline(0,1)
```

# Index

EstimateIsing, [3](#)  
EstimateIsingBi (EstimateIsing), [3](#)  
EstimateIsingLL (EstimateIsing), [3](#)  
EstimateIsingPL (EstimateIsing), [3](#)  
EstimateIsingUni (EstimateIsing), [3](#)

glm, [4](#)

IsingEntropy, [5](#)  
IsingLikelihood, [6](#)  
IsingPL, [7](#)  
IsingSampler, [7](#)  
IsingSampler-package, [2](#)  
IsingStateProb, [9](#)  
IsingSumLikelihood, [9](#)

LinTransform, [10](#)  
loglin, [4](#)

multinom, [4](#)