

Package: GiRaF (via r-universe)

September 4, 2024

Type Package

Title Gibbs Random Fields Analysis

Version 1.0.1

Date 2016-02-13

Author Julien Stoehr, Pierre Pudlo and Nial Friel

Maintainer Julien Stoehr <julien.stoehr@ucd.ie>

Description Allows calculation on, and sampling from Gibbs Random Fields, and more precisely general homogeneous Potts model. The primary tool is the exact computation of the intractable normalising constant for small rectangular lattices. Beside the latter function, it contains method that give exact sample from the likelihood for small enough rectangular lattices or approximate sample from the likelihood using MCMC samplers for large lattices.

License GPL (>= 2)

Imports methods, Rcpp (>= 0.12.3)

LinkingTo Rcpp, RcppArmadillo, BH

Suggests knitr

VignetteBuilder knitr

NeedsCompilation yes

Repository CRAN

Date/Publication 2020-10-14 16:34:25 UTC

Contents

GiRaF-package	2
exact.mrf	4
NC.mrf	6
sampler.mrf	7

Index	10
--------------	-----------

Description

GiRaF is a package for calculations on, and sampling from Gibbs (or discrete Markov) random fields.

Details

GiRaF offers various tools for the analysis of Gibbs random fields and more precisely general homogeneous Potts model with possible anisotropy and potential on singletons (cliques composed of single vertex). **GiRaF** substantially lowers the barrier for practitioners aiming at analysing such Gibbs random fields. **GiRaF** contains exact methods for small lattices and several approximate methods for larger lattices that make the analysis easier for practitioners.

The “GiRaF-introduction” vignette gives a detailed introduction on the package.

For a complete list of functions, use `library(help = "GiRaF")`.

Author(s)

Julien Stoehr, Pierre Pudlo and Nial Friel.

Maintainer: Julien Stoehr <julien.stoehr@ucd.ie>

References

Friel, N. and Rue, H. (2007). Recursive computing and simulation-free inference for general factorizable models. *Biometrika*, **94(3)**:661–672.

Geman, S. and Geman, D. (1984). Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **6(6)**:721–741.

Reeves, R. and Pettitt, A. N. (2004). Efficient recursions for general factorisable models. *Biometrika*, **91(3)**:751–757.

Swendsen, R. H. and Wang, J.-S. (1987). Nonuniversal critical dynamics in Monte Carlo simulations. *Physical Review Letters*, **58(2)**:86–88.

See Also

The “GiRaF-introduction” vignette

Examples

```
# Dimension of the lattice
height <- 8
width <- 10

# Interaction parameter
```

```
Beta <- 0.6 # Isotropic configuration
# Beta <- c(0.6, 0.6) # Anisotropic configuration when nei = 4
# Beta <- c(0.6, 0.6, 0.6, 0.6) # Anisotropic configuration when nei = 8

# Number of colors
K <- 2
# Number of neighbors
G <- 4

# Optional potential on sites
potential <- runif(K,-1,1)
# Optional borders.
Top <- Bottom <- sample(0:(K-1), width, replace = TRUE)
Left <- Right <- sample(0:(K-1), height, replace = TRUE)
Corner <- sample(0:(K-1), 4, replace = TRUE)

# Partition function for the default setting
NC.mrf(h = height, w = width, param = Beta)

# When specifying the number of colors and neighbors
NC.mrf(h = height, w = width, ncolors = K, nei = G, param = Beta)

# When specifying an optional potential on sites
NC.mrf(h = height, w = width, ncolors = K, nei = G, param = Beta,
       pot = potential)

# When specifying possible borders. The users will omit to mention all
# the non-existing borders
NC.mrf(h = height, w = width, ncolors = K, nei = G, param = Beta,
       top = Top, left = Left, bottom = Bottom, right = Right, corner = Corner)

# Exact sampling for the default setting
exact.mrf(h = height, w = width, param = Beta, view = TRUE)

# When specifying the number of colors and neighbors
exact.mrf(h = height, w = width, ncolors = K, nei = G, param = Beta,
         view = TRUE)

# When specifying an optional potential on sites
exact.mrf(h = height, w = width, ncolors = K, nei = G, param = Beta,
         pot = potential, view = TRUE)

# When specifying possible borders. The users will omit to mention all
# the non-existing borders
exact.mrf(h = height, w = width, ncolors = K, nei = G, param = Beta,
         top = Top, left = Left, bottom = Bottom, right = Right, corner = Corner, view = TRUE)

# Algorithm settings
n <- 200
method <- "Gibbs"

# Sampling method for the default setting
sampler.mrf(iter = n, sampler = method, h = height, w = width,
```

```

        param = Beta, view = TRUE)

# Sampling using an existing configuration as starting point
sampler.mrf(iter = n, sampler = method, h = height, w = width,
            ncolors = K, nei = G, param = Beta,
            initialise = FALSE, view = TRUE)

# Specifying optional arguments. The users may omit to mention all
# the non-existing borders
sampler.mrf(iter = n, sampler = method, h = height, w = width,
            ncolors = K, nei = G, param = Beta,
            pot = potential, top = Top, left = Left, bottom = Bottom,
            right = Right, corner = Corner, view = TRUE)

# Gibbs sampler with sequential updates of the sites.
sampler.mrf(iter = n, sampler = "Gibbs", h = height, w = width,
            ncolors = K, nei = G, param = Beta,
            random = FALSE, view = TRUE)

```

exact.mrf

Exact sampler for Gibbs Random Fields

Description

exact.mrf gives exact sample from the likelihood of a general Potts model defined on a rectangular $h \times w$ lattice ($h \leq w$) with either a first order or a second order dependency structure and a small number of rows (up to 19 for 2-state models).

Usage

```

exact.mrf(h, w, param, ncolors = 2, nei = 4,
          pot = NULL, top = NULL, left = NULL,
          bottom = NULL, right = NULL, corner = NULL, view = FALSE)

```

Arguments

h	the number of rows of the rectangular lattice.
w	the number of columns of the rectangular lattice.
param	numeric entry setting the interaction parameter (edges parameter)
ncolors	the number of states for the discrete random variables. By default, ncolors = 2.
nei	the number of neighbors. The latter must be one of nei = 4 or nei = 8, which respectively correspond to a first order and a second order dependency structure. By default, nei = 4.
pot	numeric entry setting homogeneous potential on singletons (vertices parameter). By default, pot = NULL
top, left, bottom, right, corner	numeric entry setting constant borders for the lattice. By default, top = NULL, left = NULL, bottom = NULL, right = NULL, corner = NULL.

view Logical value indicating whether the draw should be printed. Do not display the optional borders.

References

Friel, N. and Rue, H. (2007). Recursive computing and simulation-free inference for general factorizable models. *Biometrika*, **94**(3):661–672.

See Also

The “GiRaF-introduction” vignette

Examples

```
# Dimension of the lattice
height <- 8
width <- 10

# Interaction parameter
Beta <- 0.6 # Isotropic configuration
# Beta <- c(0.6, 0.6) # Anisotropic configuration when nei = 4
# Beta <- c(0.6, 0.6, 0.6, 0.6) # Anisotropic configuration when nei = 8

# Number of colors
K <- 2
# Number of neighbors
G <- 4

# Optional potential on sites
potential <- runif(K,-1,1)
# Optional borders.
Top <- Bottom <- sample(0:(K-1), width, replace = TRUE)
Left <- Right <- sample(0:(K-1), height, replace = TRUE)
Corner <- sample(0:(K-1), 4, replace = TRUE)

# Exact sampling for the default setting
exact.mrf(h = height, w = width, param = Beta, view = TRUE)

# When specifying the number of colors and neighbors
exact.mrf(h = height, w = width, ncolors = K, nei = G, param = Beta,
          view = TRUE)

# When specifying an optional potential on sites
exact.mrf(h = height, w = width, ncolors = K, nei = G, param = Beta,
          pot = potential, view = TRUE)

# When specifying possible borders. The users will omit to mention all
# the non-existing borders
exact.mrf(h = height, w = width, ncolors = K, nei = G, param = Beta,
          top = Top, left = Left, bottom = Bottom, right = Right, corner = Corner, view = TRUE)
```

NC.mrf

*Normalising constant of a Gibbs Random Field***Description**

Partition function of a general Potts model defined on a rectangular $h \times w$ lattice ($h \leq w$) with either a first order or a second order dependency structure and a small number of rows (up to 25 for 2-state models).

Usage

```
NC.mrf(h, w, param, ncolors = 2, nei = 4,
       pot = NULL, top = NULL, left = NULL,
       bottom = NULL, right = NULL, corner = NULL)
```

Arguments

h	the number of rows of the rectangular lattice.
w	the number of columns of the rectangular lattice.
param	numeric entry setting the interaction parameter (edges parameter)
ncolors	the number of states for the discrete random variables. By default, ncolors = 2.
nei	the number of neighbors. The latter must be one of nei = 4 or nei = 8, which respectively correspond to a first order and a second order dependency structure. By default, nei = 4.
pot	numeric entry setting homogeneous potential on singletons (vertices parameter). By default, pot = NULL
top, left, bottom, right, corner	numeric entry setting constant borders for the lattice. By default, top = NULL, left = NULL, bottom = NULL, right = NULL, corner = NULL.

References

Friel, N. and Rue, H. (2007). Recursive computing and simulation-free inference for general factorizable models. *Biometrika*, **94**(3):661–672.

Reeves, R. and Pettitt, A. N. (2004). Efficient recursions for general factorisable models. *Biometrika*, **91**(3):751–757.

See Also

The “GiRaF-introduction” vignette

Examples

```

# Dimension of the lattice
height <- 8
width <- 10

# Interaction parameter
Beta <- 0.6 # Isotropic configuration
# Beta <- c(0.6, 0.6) # Anisotropic configuration when nei = 4
# Beta <- c(0.6, 0.6, 0.6, 0.6) # Anisotropic configuration when nei = 8

# Number of colors
K <- 2
# Number of neighbors
G <- 4

# Optional potential on sites
potential <- runif(K,-1,1)
# Optional borders.
Top <- Bottom <- sample(0:(K-1), width, replace = TRUE)
Left <- Right <- sample(0:(K-1), height, replace = TRUE)
Corner <- sample(0:(K-1), 4, replace = TRUE)

# Partition function for the default setting
NC.mrf(h = height, w = width, param = Beta)

# When specifying the number of colors and neighbors
NC.mrf(h = height, w = width, ncolors = K, nei = G, param = Beta)

# When specifying an optional potential on sites
NC.mrf(h = height, w = width, ncolors = K, nei = G, param = Beta,
      pot = potential)

# When specifying possible borders. The users will omit to mention all
# the non-existing borders
NC.mrf(h = height, w = width, ncolors = K, nei = G, param = Beta,
      top = Top, left = Left, bottom = Bottom, right = Right, corner = Corner)

```

sampler.mrf

MCMC samplers for Gibbs Random Fields

Description

sampler.mrf gives approximate sample from the likelihood of a general Potts model defined on a rectangular $h \times w$ lattice ($h \leq w$) with either a first order or a second order dependency structure. Available options are the Gibbs sampler (Geman and Geman (1984)) and the Swendsen-Wang algorithm (Swendsen and Wang (1987)).

Usage

```
sampler.mrf(iter, sampler = "Gibbs" , h, w,
            param, ncolors = 2, nei = 4, pot = NULL,
            top = NULL, left = NULL, bottom = NULL, right = NULL,
            corner = NULL, initialise = TRUE, random = TRUE, view = FALSE)
```

Arguments

<code>iter</code>	Number of iterations of the algorithm.
<code>sampler</code>	The method to be used. The latter must be one of "Gibbs" or "SW" corresponding respectively to the Gibbs sampler and the Swendsen-Wang algorithm.
<code>h</code>	the number of rows of the rectangular lattice.
<code>w</code>	the number of columns of the rectangular lattice.
<code>param</code>	numeric entry setting the interaction parameter (edges parameter)
<code>ncolors</code>	the number of states for the discrete random variables. By default, <code>ncolors = 2</code> .
<code>nei</code>	the number of neighbors. The latter must be one of <code>nei = 4</code> or <code>nei = 8</code> , which respectively correspond to a first order and a second order dependency structure. By default, <code>nei = 4</code> .
<code>pot</code>	numeric entry setting homogeneous potential on singletons (vertices parameter). By default, <code>pot = NULL</code>
<code>top, left, bottom, right, corner</code>	numeric entry setting constant borders for the lattice. By default, <code>top = NULL, left = NULL, bottom = NULL, right = NULL, corner = NULL</code> .
<code>initialise</code>	Logical value indicating whether initial guess should be randomly drawn.
<code>random</code>	Logical value indicating whether the sites should be updated sequentially or randomly. Used only with the "Gibbs" option.
<code>view</code>	Logical value indicating whether the draw should be printed. Do not display the optional borders.

References

Geman, S. and Geman, D. (1984). Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **6(6)**:721-741.

Swendsen, R. H. and Wang, J.-S. (1987). Nonuniversal critical dynamics in Monte Carlo simulations. *Physical Review Letters*, **58(2)**:86-88.

See Also

The "GiRaF-introduction" vignette

Examples

```
# Algorithm settings
n <- 200
method <- "Gibbs"

# Dimension of the lattice
height <- width <- 100

# Interaction parameter
Beta <- 0.6 # Isotropic configuration
# Beta <- c(0.6, 0.6) # Anisotropic configuration when nei = 4
# Beta <- c(0.6, 0.6, 0.6, 0.6) # Anisotropic configuration when nei = 8

# Number of colors
K <- 2
# Number of neighbors
G <- 4

# Optional potential on sites
potential <- runif(K,-1,1)
# Optional borders.
Top <- Bottom <- sample(0:(K-1), width, replace = TRUE)
Left <- Right <- sample(0:(K-1), height, replace = TRUE)
Corner <- sample(0:(K-1), 4, replace = TRUE)

# Sampling method for the default setting
sampler.mrf(iter = n, sampler = method, h = height, w = width,
            param = Beta, view = TRUE)

# Sampling using an existing configuration as starting point
sampler.mrf(iter = n, sampler = method, h = height, w = width,
            ncolors = K, nei = G, param = Beta,
            initialise = FALSE, view = TRUE)

# Specifying optional arguments. The users may omit to mention all
# the non-existing borders
sampler.mrf(iter = n, sampler = method, h = height, w = width,
            ncolors = K, nei = G, param = Beta,
            pot = potential, top = Top, left = Left, bottom = Bottom,
            right = Right, corner = Corner, view = TRUE)

# Gibbs sampler with sequential updates of the sites.
sampler.mrf(iter = n, sampler = "Gibbs", h = height, w = width,
            ncolors = K, nei = G, param = Beta,
            random = FALSE, view = TRUE)
```

Index

* **gibbs**

GiRaF-package, [2](#)

* **mrf**

GiRaF-package, [2](#)

* **potts**

GiRaF-package, [2](#)

exact.mrf, [4](#)

GiRaF (GiRaF-package), [2](#)

GiRaF-package, [2](#)

NC.mrf, [6](#)

sampler.mrf, [7](#)