Package 'Boom'

February 19, 2015

Version 0.2

Date 2014-11-25

Title Bayesian Object Oriented Modeling

Author Steven L. Scott is the sole author and creator of the BOOM project. Some code in the BOOM libraries has been modified from other open source projects. These include Cephes (obtained from Netlib, written by Stephen L. Moshier), NEWUOA (M.J.D Powell, obtained from Powell's web site), and a modified version of the R math libraries (R core development team). Original copyright notices have been maintained in all source files. In these cases, copyright claimed by Steven L. Scott is limited to modifications made to the original code.

Maintainer Steve Scott <stevescott@google.com>

Description A C++ library for Bayesian modeling, with an emphasis on Markov chain Monte Carlo. Although boom contains a few R utilities (mainly plotting functions), its primary purpose is to install the BOOM C++ library on your system so that other packages can link against it.

License LGPL-2.1 | file LICENSE

Depends MASS, R(>= 3.1.0)

LinkingTo BH (>= 1.15.0-2)

SystemRequirements GNU Make, C++11

NeedsCompilation yes

Repository CRAN

Date/Publication 2014-12-03 09:51:05

R topics documented:

`add.segments` *Function to add horizontal line segments to an existing plot*

Description

Adds horizontal line segments to an existing plot. The segments are centered at `x` with height `y`. The `x` values are assumed to be equally spaced, so that `diff(x)` is a constant 'dx'. The line segments go from `x +/- half.width.factor *dx`, so if `half.width.factor=.5` there will be no gaps between segments. The default is to leave a small gap.

This function was originally used to add reference lines to side-by-side boxplots.

Usage

```
AddSegments(x, y, half.width.factor = 0.45, ...)
```

Arguments

x	A numeric vector giving the midpoints of the line segments.
y	A numeric vector of the same length as x giving the vertical position of the line segments
half.width.factor	See 'description' above.
...	graphical parameters controlling the type of lines used in the line segments

Value

Called for its side effect.

Author(s)

Steven L. Scott

See Also

boxplot.true

Examples

```
x <- rnorm(100)
y <- rnorm(100, 1)
boxplot(list(x=x,y=y))
AddSegments(1:2, c(0, 1))  ## add segments to the boxplot
```

---

ar1.coefficient.prior *Normal prior for an AR1 coefficient*

Description

A (possibly truncated) Gaussian prior on the autoregression coefficient in an AR1 model.

Usage

```
Ar1CoefficientPrior(mu = 0, sigma = 1, force.stationary = TRUE,
    force.positive = FALSE, initial.value = mu)
```

Arguments

mu	The mean of the prior distribution.
sigma	The standard deviation of the prior distribution.
force.stationary	Logical. If TRUE then the prior support for the AR1 coefficient will be truncated to (-1, 1).
force.positive	Logical. If TRUE then the prior for the AR1 coefficient will be truncated so that zero support is given to values less than zero.
initial.value	The initial value of the parameter being modeled in the MCMC algorithm.

Details

The Ar1CoefficientPrior() syntax is preferred, as it more closely matches R's syntax for other constructors.

Author(s)

Steven L. Scott <stevescott@google.com>

References

Gelman, Carlin, Stern, Rubin (2003), "Bayesian Data Analysis", Chapman and Hall.

---

beta.prior *Beta prior for a binomial proportion*

Description

Specifies beta prior distribution for a binomial probability parameter.

Usage

```
BetaPrior(a = 1, b = 1, mean = NULL, sample.size = NULL)
```

Arguments

a	A positive real number interpretable as a prior success count.
b	A positive real number interpretable as a prior failure count.
mean	A positive real number representing a/(a+b).
sample.size	A positive real number representing a+b.

Details

The distribution should be specified either with a and b, or with mean and sample.size.

Author(s)

Steven L. Scott <stevescott@google.com>

References

Gelman, Carlin, Stern, Rubin (2003), "Bayesian Data Analysis", Chapman and Hall.

`boxplot.mcmc.matrix` *Plot the distribution of a matrix*

Description

Plot the marginal distribution of each element in the Monte Carlo distribution of a matrix (e.g. a variance matrix or transition probability matrix). Rows and columns in the boxplots correspond to rows and columns in the matrix being plotted.

Usage

```
BoxplotMcmcMatrix(X, ylim = range(X), col.names,
                  row.names, truth, colors = NULL,
                  las = 0, ...)
```

Arguments

`X`	3 dimensional array. The first dimension is the Monte Carlo index (e.g. MCMC iteration). The second and third dimensions are the row and column of the matrix being plotted. E.g. `X[i,j,k]` is Monte Carlo draw `i` of matrix element `j,k`.
`ylim`	2-vector giving the lower and upper limits of the vertical axis.
`col.names`	(optional) character vector giving the names of matrix columns (third dimension of `X`).
`row.names`	(optional) character vector giving the names of matrix rows (second dimension of `X`).
`truth`	(optional) scalar or matrix giving the values of reference lines to be plotted on each boxplot. If a scalar then the same value will be used for each boxplot. If a matrix then the rows and columns of the matrix correspond to the second and third dimension of `X`.
`colors`	A vector of colors to use for the boxplots. Each row uses the same color scheme.
`las`	Controls the orientation of axis labels. See the `las` section in the help page for `par`.
`...`	Extra arguments passed to `boxplot`

Value

Called for its side effect, which is to draw a set of side-by-side boxplots on the current graphics device.

Author(s)

Steven L. Scott

See Also

boxplot.true, boxplot

Examples

```
X <- array(rnorm(1000 * 3 * 4), dim=c(1000, 3, 4))
dimnames(X)[[2]] <- paste("row", 1:3)
dimnames(X)[[3]] <- paste("col", 1:4)
BoxplotMcmcMatrix(X)

truth <- 0
BoxplotMcmcMatrix(X, truth=truth)

truth <- matrix(rnorm(12), ncol=4)
BoxplotMcmcMatrix(X, truth=truth)
```

---

boxplot.true *side-by-side boxplots from a matrix, with optional reference values*

Description

Plots side-by-side boxplots of the columns of the matrix x. Each boxplot can have its own reference line (truth) and standard error lines se.truth, if desired. This function was originally written to display MCMC output, where the reference lines were true values used to test an MCMC simulation.

Usage

```
BoxplotTrue(x, truth = NULL, vnames = NULL, center = FALSE,
            se.truth = NULL, color = "white", ...)
```

Arguments

x	The matrix whose columns are to be plotted.
truth	(optional) A vector of reference values with length equal to ncol(x).
vnames	(optional) character vector giving the column names of x.
center	(optional) logical. If truth is supplied then center=TRUE will center each column of x around truth to show the variation around the reference line.
se.truth	(optional) numeric vector of length ncol(x). If truth is supplied then additional reference lines will be drawn at truth +/- 2*se.truth.
color	(optional) vector of colors for each boxplot.
...	additional arguments to boxplot.

Value

called for its side effect

Author(s)

Steven L. Scott

See Also

boxplot.matrix, boxplot,

Examples

```
x <- t(matrix(rnorm(5000, 1:5, 1:5), nrow=5))
BoxplotTrue(x, truth=1:5, se.truth=1:5, col=rainbow(5), vnames =
  c("EJ", "TK", "JT", "OtherEJ", "TJ") )
```

---

check.data *Checking data formats*

Description

Checks that data matches a concept

Usage

```
check.scalar.probability(x)
check.positive.scalar(x)
check.nonnegative.scalar(x)
check.probability.distribution(x)
```

Arguments

x	An object to be checked.

Details

If the object does not match the concept being checked, stop is called. Otherwise TRUE is returned.

Author(s)

Steven L. Scott <stevescott@google.com>

compare.den *Compare several density estimates.*

Description

Produces multiple density plots on a single axis, to compare the columns of a matrix or the elements of a list.

Usage

```
CompareDensities(x,
                 legend.text = NULL,
                 legend.location = "topright",
                 legend.title = NULL,
                 xlim = NULL,
                 ylim = NULL,
                 xlab = "parameter",
                 ylab = "density",
                 main = "",
                 lty = NULL,
                 col = "black",
                 axes = TRUE,
                 na.rm = TRUE,
                 ...)
```

Arguments

x	matrix or list of numeric vectors. A density plot is produced for each column of the matrix or element of the list.
legend.text	(optional) character vector giving names of each density plot.
legend.location	Entry that can be passed to legend.
legend.title	The legend title.
xlim	(optional) horizonal range of the plotting region. If omitted the region will be sized to fit all the observations in x.
ylim	(optional) vertical range of the plotting region. If omitted the region will be sized to fit all empirical density plots.
xlab	label to be placed on the horizontal axis
ylab	label to be placed on the vertical axis
main	main title for the plot
lty	The line types to use for the different densities. See par. If NULL then a different line type will be used for each density.
col	vector of colors for the densities to be plotted.
axes	Logical. Should axes and a box be drawn around the figure?

na.rm	Logical value indicating whether NA's should be removed.
...	Other graphical parameters passed to plot.density, and lines.

Value

Called for its side effect, which is to produce multiple density plots on the current graphics device.

Author(s)

Steven L. Scott

See Also

density

Examples

```
x <- t(matrix(rnorm(5000, 1:5, 1:5), nrow=5))
CompareDensities(x, legend.text=c("EJ", "TK", "JT", "OtherEJ", "TJ"),
                 col=rainbow(5), lwd=2)
```

---

compare.many.densities

Compare several density estimates.

Description

Produce a plot that compares the kernel density estimates for each element in a series of Monte Carlo draws of a vector or matrix.

Usage

```
CompareManyDensities(list.of.arrays,
                     style = c("density", "box"),
                     main = "",
                     color = NULL,
                     gap = 0,
                     burn = 0,
                     supress.labels = FALSE,
                     x.same.scale = TRUE,
                     y.same.scale = FALSE,
                     xlim = NULL,
                     ylim = NULL,
                     legend.location = c("top", "right"),
                     legend.cex = 1,
                     reflines = NULL,
                     ...)
```

Arguments

list.of.arrays	A list of arrays representing the MCMC draws of the vector or matrix in question. Each list element represents a different group. The first index in each list list element represents the Monte Carlo draw number (or iteration). The remaining indices represent the variables to be plotted. If the first list element has variable names assigned to its indices, these will be used to label the plots.
style	The style of plot to use for comparing distributions.
main	The main title of the plot.
color	A vector of colors to be used for representing the groups.
gap	The gap (in lines) between plots.
burn	The number of MCMC iterations to be discarded as burn-in.
supress.labels	Logical. If FALSE then the dimnames (if any) of the first element in list.of.arrays will be used to annotate the plot. If TRUE then no labels will be used.
x.same.scale	Logical indicating whether the same horizontal scale should be used for all the plots.
y.same.scale	Logical indicating whether the same vertical scale should be used for all the plots. This argument is ignored if style == "box".
xlim	Either NULL, or a pair of numbers giving limits for the horizontal axis. If xlim is set then the same xlim values will be used for all plots and the x.same.scale argument will be ignored.
ylim	Either NULL, or a pair of numbers giving limits for the vertical axis. If ylim is set then the same ylim values will be used for all plots and the y.same.scale argument will be ignored. This argument is ignored if style == "box".
legend.location	The location of the legend, either on top or at the right. It can also be NULL in which case no legend will appear. The legend names will be taken from names(list.of.arrays). If it does not have names, then no legend will be produced.
legend.cex	The relative scale factor to use for the legend text.
reflines	This can be NULL, in which case no reference lines are drawn, it can be a single real number in which case a reference line will be drawn at that value in each panel, or it can be a vector with length equal to the number of panels, in which case a reference line will be drawn at each panel-specific value.
...	Extra arguments passed to CompareDen.

Author(s)

Steven L. Scott

See Also

density, CompareManyTs

Examples

```
x <- array(rnorm(9000), dim = c(1000, 3, 3))
dimnames(x) <- list(NULL, c("Larry", "Moe", "Curly"), c("Larry", "Eric", "Sergey"))
y <- array(rnorm(9000), dim = c(1000, 3, 3))
z <- array(rnorm(9000), dim = c(1000, 3, 3))
data <- list(x = x, y = y, z = z)
CompareManyDensities(data, color = c("red", "blue", "green"))
CompareManyDensities(data, style = "box")

x <- matrix(rnorm(5000), nrow = 1000)
colnames(x) <- c("Larry", "Moe", "Curly", "Shemp", "???")
y <- matrix(rnorm(5000), nrow = 1000)
z <- matrix(rnorm(5000), nrow = 1000)
data <- list(x = x, y = y, z = z)
CompareManyDensities(data, color = c("red", "blue", "green"))
CompareManyDensities(data, style = "box")
```

`compare.many.ts` *Compares several density estimates.*

Description

Produce a plot that compares the kernel density estimates for each element in a series of Monte Carlo draws of a vector or matrix.

Usage

```
CompareManyTs(list.of.ts, burn = 0, type = "l", gap = 0,
              boxes = TRUE, thin = 1, labels = NULL,
              same.scale = TRUE, ylim = NULL, refline = NULL,
              color = NULL, ...)
```

Arguments

`list.of.ts`	A list of time series matrices, data.frames or 3-dimensional arrays, all of the same size. The list elements correspond to groups. The first index of the array in each list element corresponds to time. The subsequent indices correspond to variables to be plotted.
`burn`	The number of initial observations to be discarded as burn-in (when plotting MCMC output).
`type`	The plotting type to use when plotting the time series. See `plot`.
`gap`	The amount of space to put between plots.
`boxes`	Logical. Should boxes be drawn around the plots?
`thin`	Plot every thin'th observation. This can reduce the amount of time it takes to make the plot if there are many long time series.
`labels`	A character vector to use as labels for individual plots.

same.scale	Logical. If TRUE then all plots are shown on the same verical scale, and vertical axes are drawn. If FALSE then each plot gets its own scale.
ylim	The scale of the vertical axis. If non-NULL then same.scale will be set to TRUE.
refline	The scalar value at which a thin dotted horizontal line should be plotted in each panel. This is useful for highlighting zero, for example.
color	A vector of colors to use for the plots.
...	Extra arguments passed to 'plot' and 'axis'.

Author(s)

Steven L. Scott

See Also

PlotManyTs, CompareManyDensities

Examples

```
x <- array(rnorm(9000), dim = c(1000, 3, 3))
dimnames(x) <- list(NULL, c("Larry", "Moe", "Curly"), c("Larry", "Eric", "Sergey"))
y <- array(rnorm(9000), dim = c(1000, 3, 3))
z <- array(rnorm(9000), dim = c(1000, 3, 3))
data <- list(x = x, y = y, z = z)
CompareManyTs(data, color = c("red", "blue", "green"))

x <- matrix(rnorm(5000), nrow = 1000)
colnames(x) <- c("Larry", "Moe", "Curly", "Shemp", "???")
y <- matrix(rnorm(5000), nrow = 1000)
z <- matrix(rnorm(5000), nrow = 1000)
data <- list(x = x, y = y, z = z)
CompareManyTs(data, color = c("red", "blue", "green"))
```

---

compare.vector.distribution

Boxplots to compare distributions of vectors

Description

Uses boxplots to compare distributions of vectors.

Usage

```
CompareVectorBoxplots(draws, main = NULL, colors = NULL, burn = 0)
```

Arguments

draws	A list of MCMC draws. Each list element is a matrix with rows corresponding to MCMC iterations and columns to variables. The matrices can have different numbers of rows, but should have the same numbers of columns.
main	Main title of the plot.
colors	Colors to use for the boxplots. The length must match the number entries in draws.
burn	The number of initial MCMC iterations to discard before making the plot.

Details

Creates side-by-side boxplots with the dimensions of each vector gropued together.

Examples

```
x <- matrix(rnorm(300, mean = 1:3, sd = .4), ncol = 3, byrow = TRUE)
y <- matrix(rnorm(600, mean = 3:1, sd = .2), ncol = 3, byrow = TRUE)
CompareVectorBoxplots(list(x = x, y = y), colors = c("red", "blue"))
```

---

dirichlet.prior *Dirichlet prior for a multinomial distribution*

Description

Specifies Dirichlet prior for a discrete probability distribution.

Usage

```
DirichletPrior(prior.counts, initial.value = NULL)
```

Arguments

prior.counts	A vector of positive numbers representing prior counts.
initial.value	The initial value in the MCMC algorithm of the distribution being modeled.

Author(s)

Steven L. Scott <stevescott@google.com>

References

Gelman, Carlin, Stern, Rubin (2003), "Bayesian Data Analysis", Chapman and Hall.

discrete-uniform-prior

Discrete prior distributions

Description

Prior distributions over a discrete quantities.

Usage

```
PointMassPrior(location)
PoissonPrior(mean, lower.limit = 0, upper.limit = Inf)
DiscreteUniformPrior(lower.limit, upper.limit)
```

Arguments

location	The location of the point mass.
mean	The mean of the Poisson distribution.
lower.limit	The smallest value within the support of the distribution. The prior probability for numbers less than lower.limit is zero.
upper.limit	The largest value within the support of the distribution. The prior probability for numbers greater than upper.limit is zero.

Value

Each function returns a prior object whose class is the same as the function name. All of these inherit from "DiscreteUniformPrior" and from "Prior".

The PoissonPrior assumes a potentially truncated Poisson distribution with the given mean.

Author(s)

Steven L. Scott <stevescott@google.com>

Examples

```
## Specify an exact number of trees in a Bart model (see the BoomBart
## package).

ntrees <- PointMassPrior(200)

## Uniform prior between 50 and 100 trees, including the endpoints.
ntrees <- DiscreteUniformPrior(50, 100)

## Truncated Poisson prior, with a mean of 20, a lower endpoint of 1,
## and an upper endpoint of 50.
ntrees <- PoissonPrior(20, 1, 50)
```

double.model *Prior distributions for a real valued scalar*

Description

A DoubleModel is a class of prior distributions for real valued scalar parameters. A DoubleModel is sometimes used by a probability model that does not have a conjugate prior.

Author(s)

Steven L. Scott <stevescott@google.com>

See Also

UniformPrior, BetaPrior, NormalPrior, and GammaPrior.

external.legend *Add an external legend to an array of plots.*

Description

ExternalLegendLayout sets up a plotting region to plot a regular grid of points, with an optional legend on the top or right of the grid.

AddExternalLegend adds a legend to a grid of plots that was set up using ExternalLegendLayout.

Usage

```
ExternalLegendLayout(nrow,
                     ncol,
                     legend.labels,
                     legend.location = c("top", "right"),
                     outer.margin.lines = rep(4, 4),
                     gap.between.plots = rep(0, 4),
                     legend.cex = 1,
                     x.axis = TRUE,
                     y.axis = TRUE)

AddExternalLegend(legend.labels,
                  legend.location = c("top", "right"),
                  legend.cex =1,
                  bty = "n",
                  ...)
```

Arguments

nrow	The number of rows in the array of plots.
ncol	The number of columns in the array of plots.
legend.labels	The labels to be used in the legend.
legend.location	Specifies whether the legend should appear on the top or the right hand side. It can also be NULL, indicating that no legend is desired.
outer.margin.lines	A vector of length four giving the number of lines of text desired for the outer margins of the plot. See the oma argument of par. This can also be specified as a single number, to be repeated 4 times.
gap.between.plots	A vector of length 4 giving the number of lines of text to leave between grid panels. See the mar argument of par. This can also be specified as a single number, to be repeated 4 times.
legend.cex	The scale factor that will be used for legend text. This must match the scale factor used in add.external.legend.
x.axis	Will any plots have a horizontal axis?
y.axis	Will any plots have a vertical axis?
bty	Type of box to draw around the legend. Can be "n" (for no box) or "o" for a box. See legend.
...	Extra arguments passed to legend.

Value

ExternalLegendLayout returns the original graphical parameters, intended for use with on.exit.

AddExternalLegend returns invisible NULL.

Author(s)

Steven L. Scott

See Also

legend layout

Examples

```
example.plot <- function() {
  x <- rnorm(100)
  y <- rnorm(100)
  scale <- range(x, y)
  opar <- ExternalLegendLayout(nrow = 2,
                               ncol = 2,
                               legend.labels = c("foo", "bar"))
```

```
  on.exit({par(opar); layout(1)})
  hist(x, xlim = scale, axes = FALSE, main = "")
  mtext("X", side = 3, line = 1)
  box()
  plot(x, y, xlim = scale, ylim = scale, axes = FALSE)
  box()
  axis(3)
  axis(4)
  plot(y, x, xlim = scale, ylim = scale, axes = FALSE, pch = 2, col = 2)
  box()
  axis(1)
  axis(2)
  hist(y, xlim = scale, axes = FALSE, main = "")
  mtext("Y", side = 1, line = 1)
  box()
  AddExternalLegend(legend.labels = c("foo", "bar"),
                    pch = 1:2,
                    col = 1:2,
                    legend.cex = 1.5)
}

## Now call example.plot().
example.plot()

## Because of the call to on.exit(), in example.plot,
## the original plot layout is restored.
hist(1:10)
```

`gamma.prior` *Gamma prior distribution*

Description

Specifies gamma prior distribution.

Usage

```
GammaPrior(a = NULL, b = NULL, prior.mean = NULL, initial.value = NULL)
```

Arguments

`a`	The shape parameter in the Gamma(a, b) distribution.
`b`	The scale paramter in the Gamma(a, b) distribution.
`prior.mean`	The mean the Gamma(a, b) distribution, which is a/b.
`initial.value`	The initial value in the MCMC algorithm of the variable being modeled.

Details

The mean of the Gamma(a, b) distribution is a/b and the variance is a/b^2. If `prior.mean` is not `NULL`, then one of either `a` or `b` must be non-`NULL` as well.

GammaPrior is the conjugate prior for a Poisson mean or an exponential rate. For a Poisson mean `a` corresponds to a prior sum of observations and `b` to a prior number of observations. For an exponential rate the roles are reversed `a` represents a number of observations and `b` the sum of the observed durations. The gamma distribution is a generally useful for parameters that must be positive.

The gamma distribution is the conjugate prior for the reciprocal of a Guassian variance, but `SdPrior` should usually be used in that case.

Author(s)

Steven L. Scott <stevescott@google.com>

References

Gelman, Carlin, Stern, Rubin (2003), "Bayesian Data Analysis", Chapman and Hall.

`GenerateFactorData` *Generate a data frame of all factor data*

Description

This function is mainly intended for example code and unit testing. It generates a `data.frame` containing all factor data.

Usage

```
GenerateFactorData(factor.levels.list, sample.size)
```

Arguments

`factor.levels.list`	A list of character vectors giving factor level names. The names attribute of this list becomes the set of variables names for the return data frame.
`sample.size`	The desired number of rows in the returned data frame.

Author(s)

Steven L. Scott <stevescott@google.com>

Examples

```
foo <- GenerateFactorData(list(a = c("foo", "bar", "baz"),
                               b = c("larry", "moe", "curly", "shemp")),
                          50)

head(foo)
#     a     b
# 1 bar curly
# 2 foo curly
# 3 bar   moe
# 4 bar   moe
# 5 baz curly
# 6 bar curly
```

`is.even` *Check whether a number is even or odd.*

Description

Check whether a number is even or odd.

Usage

```
IsEven(x)
IsOdd(x)
```

Arguments

`x`	An integer or vector of integers.

Value

Logical indicating whether the argument is even (or odd).

Author(s)

Steven L. Scott

Examples

```
IsEven(2) ## TRUE
IsOdd(2)  ## FALSE
```

markov.prior *Prior for a Markov chain*

Description

The conjugate prior distribution for the parameters of a homogeneous Markov chain. The rows in the transition probability matrix modeled with independent Dirichlet priors. The distribution of the initial state is modeled with its own independent Dirichlet prior.

Usage

```
MarkovPrior(prior.transition.counts = NULL,
            prior.initial.state.counts = NULL,
            state.space.size = NULL,
            uniform.prior.value = 1)
```

Arguments

`prior.transition.counts`
: A matrix of the same dimension as the transition probability matrix being modeled. Entry (i, j) represents the "prior count" of transitions from state `i` to state `j`.

`prior.initial.state.counts`
: A vector of positive numbers representing prior counts of initial states.

`state.space.size`
: If both prior.transition.counts and prior.initial.state.counts are missing, then they will be filled with an object of dimension state.space.size where all entries are set to `uniform.prior.value`.

`uniform.prior.value`
: The default value to use for entries of `prior.transition.counts` and `prior.initial.state.counts`, when they are not supplied by the user.

Author(s)

Steven L. Scott <stevescott@google.com>

References

Gelman, Carlin, Stern, Rubin (2003), "Bayesian Data Analysis", Chapman and Hall.

match_data_frame *MatchDataFrame*

Description

Given two data frames with the same data, but with rows and columns in potentially different orders, produce a pair of permutations such that data2[row.permutation, column.permutation] matches data1.

Usage

```
MatchDataFrame(data.to.match, data.to.permute)
```

Arguments

data.to.match	The data frame to be matched.
data.to.permute	The data frame to be permuted.

Value

Returns a list with two elements.

column.permutation	A vector of indices such that the columns of data2[, column.permutation] match the columns of data1. The matching is based on column names.
row.permutation	A vector of indices such that the rows of data2[row.permutation, column.permutation] match the rows of data1. The matching is done by converting rows to strings, and matching the strings.

Author(s)

Steven L. Scott <stevescott@google.com>

Examples

```
x1 <- data.frame(larry = rnorm(10), moe = 1:10, curly = rpois(10, 2))
x2 <- x1[c(1:5, 10:6), c(3, 1, 2)]

m <- MatchDataFrame(x1, x2)
x2[m$row.permutation, m$column.permutation] == x1  ## all TRUE
```

mvn.diagonal.prior *diagonal MVN prior*

Description

A multivariate normal prior distribution formed by the product of independent normal margins.

Usage

```
MvnDiagonalPrior(mean.vector, sd.vector)
```

Arguments

mean.vector	A vector giving the mean of the prior distribution.
sd.vector	The standard deviations of the components in the distribution. I.e. the square root of the diagonal of the variance matrix.

Author(s)

Steven L. Scott <stevescott@google.com>

References

Gelman, Carlin, Stern, Rubin (2003), "Bayesian Data Analysis", Chapman and Hall.

mvn.independent.sigma.prior

Independence prior for the MVN

Description

A prior for the parameters of the multivariate normal distribution that assumes Sigma to be a diagonal matrix with elements modeled by independent inverse Gamma priors.

Usage

```
MvnIndependentSigmaPrior(mvn.prior, sd.prior.list)
```

Arguments

mvn.prior	An object of class MvnPrior that is the prior distribution for the multivariate normal mean parameter.
sd.prior.list	A list of SdPrior objects modeling the diagonal elements of the multivariate normal variance matrix. The off-diagonal elements are assumed to be zero.

Author(s)

Steven L. Scott <stevescott@google.com>

References

Gelman, Carlin, Stern, Rubin (2003), "Bayesian Data Analysis", Chapman and Hall.

---

mvn.prior *Multivariate normal prior*

Description

A multivariate normal prior distribution.

Usage

```
MvnPrior(mean, variance)
```

Arguments

mean	A vector giving the mean of the prior distribution.
variance	A symmetric positive definite matrix giving the variance of the prior distribution.

Author(s)

Steven L. Scott <stevescott@google.com>

References

Gelman, Carlin, Stern, Rubin (2003), "Bayesian Data Analysis", Chapman and Hall.

---

normal.inverse.gamma.prior *Normal inverse gamma prior*

Description

The NormalInverseGammaPrior is the conjugate prior for the mean and variance of the scalar normal distribution. The model says that

$$\frac{1}{\sigma^2} \sim Gamma(df/2, ss/2)\mu|\sigma \sim N(\mu_0, \sigma^2/\kappa)$$

Usage

```
NormalInverseGammaPrior(mu.guess, mu.guess.weight = .01,
       sigma.guess, sigma.guess.weight = 1, ...)
```

Arguments

`mu.guess`	The mean of the prior distribution. This is μ_0 in the description above.
`mu.guess.weight`	The number of observations worth of weight assigned to `mu.guess`. This is κ in the description above.
`sigma.guess`	A prior estimate at the value of `sigma`. This is $\sqrt{ss/df}$.
`sigma.guess.weight`	The number of observations worth of weight assigned to `sigma.guess`. This is df.
`...`	blah

Author(s)

Steven L. Scott <stevescott@google.com>

References

Gelman, Carlin, Stern, Rubin (2003), "Bayesian Data Analysis", Chapman and Hall.

---

normal.inverse.wishart.prior

Normal inverse Wishart prior

Description

The NormalInverseWishartPrior is the conjugate prior for the mean and variance of the multivariate normal distribution. The model says that

$$\Sigma^{-1} \sim Wishart(\nu, S)\mu|\sigma \sim N(\mu_0, \Sigma/\kappa)$$

The $Wishart(S, \nu)$ distribution is parameterized by `S`, the *inverse* of the sum of squares matrix, and the scalar degrees of freedom parameter `nu`.

The distribution is improper if $\nu < dim(S)$.

Usage

```
NormalInverseWishartPrior(mean.guess,
                          mean.guess.weight = .01,
                          variance.guess,
                          variance.guess.weight = nrow(variance.guess) + 1)
```

Arguments

`mean.guess`	The mean of the prior distribution. This is μ_0 in the description above.
`mean.guess.weight`	The number of observations worth of weight assigned to `mean.guess`. This is κ in the description above.
`variance.guess`	A prior estimate at the value of Σ. This is S^{-1}/ν in the notation above.
`variance.guess.weight`	The number of observations worth of weight assigned to `variance.guess`. This is df.

Author(s)

Steven L. Scott <stevescott@google.com>

References

Gelman, Carlin, Stern, Rubin (2003), "Bayesian Data Analysis", Chapman and Hall.

`normal.prior` *Normal (scalar Gaussian) prior distribution*

Description

Specifies an inverse Gamma prior for a variance parameter, but inputs are defined in terms of a standard deviation.

Usage

```
NormalPrior(mu, sigma, initial.value = mu)
```

Arguments

`mu`	The mean of the prior distribution.
`sigma`	The standard deviation of the prior distribution.
`initial.value`	The initial value of the parameter being modeled in the MCMC algorithm.

Author(s)

Steven L. Scott <stevescott@google.com>

References

Gelman, Carlin, Stern, Rubin (2003), "Bayesian Data Analysis", Chapman and Hall.

pairs.density *Pairs plot for posterior distributions.*

Description

A pairs plot showing the posterior distribution of the given list of Monte Carlo draws. Plots above the diagonal show the posterior distribution on a scale just wide enough to fit the plots. The diagonal shows a marginal density plot, and the subdiagonal shows the distribution with all plots on a common scale.

Usage

```
PairsDensity(draws,
             nlevels = 20,
             lty = NULL,
             color = NULL,
             subset = NULL,
             labels,
             legend.location = "top",
             legend.cex = 1,
             label.cex = 1,
             ...)
```

Arguments

draws	Either a matrix or a list of matrices. If a list is provided then each list element is plotted as a separate set of contours, and all matrices must have the same number of columns (though the number of rows can differ).
nlevels	The number of contour levels to plot.
lty	The line types to use for the different elements in draws.
color	The color to use for different elements in draws.
subset	If draws is a list, then this can be a numerical vector. If draws has names, then subset can be a character vector naming which elements to include. If NULL then all elements of draws are plotted.
labels	If labels is missing and the first element of draws has non-NULL colnames then these will be used to label the pairs plot. If a character vector of length ncol(draws[[1]]) then this character vector will be used in place of the colnames. If NULL then no labels will be used.
legend.location	Either "top", or "right" specifying the location for the legend, or NULL, indicating that no legend is desired. if draws is a matrix or a singleton list then no legend is produced.
legend.cex	Scale factor to use for the legend labels.
label.cex	Scale factor to use for the row and column labels.
...	Extra arguments (graphical parameters), passed to plot, PlotDensityContours, axis, and AddExternalLegend.

Author(s)

Steven L. Scott

See Also

pairs, CompareDensities, CompareManyDensities

Examples

```
## You can see the pairs plot for a single set of draws.
y <- matrix(rnorm(5000, mean = 1:5), ncol = 5, byrow = TRUE)
PairsDensity(y)

## You can also compare two or more sets of draws.
z <- matrix(rnorm(2500, mean = 2:6), ncol = 5, byrow = TRUE)
PairsDensity(list("first set" = y, "second set" = z))
```

---

plot.density.contours *Contour plot of a bivariate density.*

Description

Contour plot of one ore more bivariate densities. This function was originally created to implement PairsDensity, but might be useful on its own.

Usage

```
PlotDensityContours(draws,
                    x.index = 1,
                    y.index = 2,
                    xlim = NULL,
                    ylim = NULL,
                    nlevels = 20,
                    subset = NULL,
                    color = NULL,
                    lty = NULL,
                    axes = TRUE,
                    ...)
```

Arguments

draws	Either a matrix or a list of matrices. If a list is provided then each list element is plotted as a separate set of contours, and all matrices must have the same number of columns (though the number of rows can differ).
x.index	The index of the parameter to plot on the horizonal axis.

y.index	The index of the beta coefficient to plot on the vertical axis.
xlim	Limits on the horizontal axis. If NULL then the plot is just wide enough to fit the contours.
ylim	Limits on the vertical axis. If NULL then the plot is just tall enough to fit the contours.
nlevels	The number of contour levels to plot.
subset	If draws is a list, then this can be a numerical vector. If draws has names, then subset can be a character vector naming which elements to include. If NULL then all elements of draws are plotted.
color	The color to use for different elemetns in draws.
lty	The line types to use for the different elements in draws.
axes	Logical. Should x and y axies be drawn?
...	Extra arguments passed to contour.

Author(s)

Steven L. Scott

See Also

contour, kde2d

Examples

```
## You can see the pairs plot for a single set of draws.
y <- matrix(rnorm(5000, mean = 1:5), ncol = 5, byrow = TRUE)
PlotDensityContours(y, 3, 1)

## You can also compare two or more sets of draws.
z <- matrix(rnorm(2500, mean = 2:6), ncol = 5, byrow = TRUE)
PlotDensityContours(list("first set" = y, "second set" = z), 3, 1)
```

---

plot.dynamic.distribution

Plots the pointwise evolution of a distribution over an index set.

Description

Produces an dynamic distribution plot where gray scale shading is used to show the evolution of a distribution over an index set. This function is particularly useful when the index set is too large to do side-by-side boxplots.

Usage

```
PlotDynamicDistribution(curves,
                        x = 1:ncol(curves),
                        quantile.step=.01,
                        xlim = range(x),
                        xlab = "X",
                        ylim = range(curves),
                        ylab = "distribution",
                        add=FALSE,
                        ...)
```

Arguments

curves	A matrix where each row represents a curve (e.g. a simulation of a time series from a posterior distribution) and columns represent different points in the index set. For example, a long time series would be a wide matrix.
x	An optional vector of ordinates 'curves' will be plotted against. Good choices for x are numeric, Date (see as.Date).
quantile.step	Each color step in the plot corresponds to this difference in quantiles. Smaller values make prettier plots, but the plots take longer to produce.
xlim	The x limits (x1, x2) of the plot. Note that x1 > x2 is allowed and leads to a "reversed axis".
xlab	Label for the horzontal axis.
ylim	The y limits (y1, y2) of the plot. Note that y1 > y2 is allowed and leads to a "reversed axis".
ylab	Label for the vertical axis.
add	Logical. If true then add the plot to the current plot. Otherwise a fresh plot will be created.
...	Extra arguments to pass on to plot

Details

The function works by passing many calls to polygon. Each polygon is associated with a quantile level, with darker shading near the median.

Value

This function is called for its side effect, which is to produce a plot on the current graphics device.

Author(s)

Steven L. Scott <stevescott@google.com>

Examples

```
x <- t(matrix(rnorm(1000 * 100, 1:100, 1:100), nrow=100))
## x has 1000 rows, and 100 columns.  Column i is N(i, i^2) noise.

PlotDynamicDistribution(x)
time <- as.Date("2010-01-01", format = "%Y-%m-%d") + (0:99 - 50)*7
PlotDynamicDistribution(x, time)
```

`plot.macf` *Plots individual autocorrelation functions for many-valued time series*

Description

Produces individual autocorrelation functions for many-valued time series such as those produced by highly multivariate MCMC output. Cross-correlations such as those produced by `acf` are not shown.

Usage

```
PlotMacf(x, lag.max = 40, gap = 0.5, main = NULL, boxes = TRUE,
         xlab = "lag", ylab = "ACF", type = "h")
```

Arguments

`x`	matrix or 3-way array of MCMC output (or other time series). The first dimension represents discrete time.
`lag.max`	maximum lag to use when computing ACF's.
`gap`	non-negative scalar. gap between plots
`main`	character. main title for the plot
`boxes`	logical. Should boxes be drawn around the plots
`xlab`	character label for horizontal axis.
`ylab`	character label for vertical axis.
`type`	type of line plot to show. Defaults to "h". See `plot.default` for other options.

Value

Called for its side effect

Author(s)

Steven L. Scott

See Also

`acf`, `plot.many.ts`.

Examples

```
x <- matrix(rnorm(1000), ncol=10)
PlotMacf(x)
```

`plot.many.ts` *Multiple time series plots*

Description

Plots many time series plots on the same graphical device. Each plot gets its own frame. Scales can be adjusted to see variation in each plot (each plot gets its own scale), or variation between plots (common scale).

Usage

```
PlotManyTs(x, type = "l", gap = 0, boxes = TRUE, truth = NULL,
           thin = 1, labs, same.scale = TRUE, ylim = NULL,
           refline = NULL, color = NULL, ...)
```

Arguments

`x`	matrix or 3-dimensional array to be plotted
`type`	type of line plots to produce. See `plot.default` for other options.
`gap`	number of lines of space to put between plots.
`boxes`	logical. Should boxes be drawn around each plot?
`truth`	A vector or matrix of reference values to be added to each plot as a horizontal line. The dimension should match `dim(x)[-1]`
`thin`	Frequency of observations to plot. E.g. `thin=10` means plot every 10'th observation. Thinning can speed things up when plotting large amounts MCMC output.
`labs`	(optional) character vector giving the title (e.g. variable name) for each plot.
`same.scale`	logical. If true then plots are drawn with a common vertical axis, which is displayed on alternating rows of the plot. If false each plot gets its own axis, which is not displayed.
`ylim`	
`refline`	a vector or scalar value to use as a reference line. This is a supplement to the `truth` argument. It can be useful when comparing true values (used in a simulation), estimated values (e.g. point estimates of parameters) and MCMC output.
`color`	Vector of colors to use in the plots.
`...`	extra arguments to be passed to `plot` and `axis`.

Value

Called for its side effect.

Author(s)

Steven L. Scott

See Also

plot.ts (for plotting a small number of time series) plot.macf

Examples

```
x <- matrix(rnorm(1000), ncol=10)
PlotManyTs(x)

PlotManyTs(x, same = FALSE)
```

---

sd.prior *Prior for a standard deviation or variance*

Description

Specifies an inverse Gamma prior for a variance parameter, but inputs are defined in terms of a standard deviation.

Usage

```
SdPrior(sigma.guess, sample.size = .01, initial.value = sigma.guess,
        fixed = FALSE, upper.limit = Inf)
```

Arguments

sigma.guess	A prior guess at the value of the standard deviation.
sample.size	The weight given to sigma.guess. Interpretable as a prior observation count.
initial.value	The initial value of the paramter in the MCMC algorithm.
fixed	Logical. Some algorithms allow you to fix sigma at a particular value. If TRUE then sigma will remain fixed at initial.value, if supported.
upper.limit	If positive, this is the upper limit on possible values of the standard deviation parameter. Otherwise the upper limit is assumed infinite. Not supported by all MCMC algorithms.

Author(s)

Steven L. Scott <stevescott@google.com>

References

Gelman, Carlin, Stern, Rubin (2003), "Bayesian Data Analysis", Chapman and Hall.

thin *Thin the rows of a matrix*

Description

Systematic sampling of every thin'th row of a matrix or vector. Useful for culling MCMC output or denoising a plot.

Usage

```
thin(x, thin)
```

Arguments

x	The array to be thinned. The first dimension is the one sampled over.
thin	The frequency of observations to keep. With thin=10 you will keep every 10th observation.

Value

The thinned vector, matrix, or array is returned.

Author(s)

Steven L. Scott

Examples

```
x <- rnorm(100)
thin(x, 10)
# returns a 10 vector

y <- matrix(rnorm(200), ncol=2)
thin(y, 10)
# returns a 10 by 2 matrix
```

`uniform.prior` *Uniform prior distribution*

Description

Specifies a uniform prior distribution on a real-valued scalar parameter.

Usage

```
UniformPrior(lo = 0, hi = 1, initial.value = NULL)
```

Arguments

`lo`	The lower limit of support.
`hi`	The upper limit of support.
`initial.value`	The initial value of the parameter in question to use in the MCMC algorithm. If `NULL` then the mean `(lo + hi)/2` is used.

Author(s)

Steven L. Scott <stevescott@google.com>

References

Gelman, Carlin, Stern, Rubin (2003), "Bayesian Data Analysis", Chapman and Hall.

Index