

# Package ‘jagshelper’

July 30, 2024

**Type** Package

**Title** Extracting and Visualizing Output from 'jagsUI'

**Version** 0.3.1

**Date** 2024-07-30

**Author** Matt Tyers [aut, cre]

**Maintainer** Matt Tyers <matattyersstat@gmail.com>

**Description** Tools are provided to streamline Bayesian analyses in 'JAGS' using the 'jagsUI' package. Included are functions for extracting output in simpler format, functions for streamlining assessment of convergence, and functions for producing summary plots of output. Also included is a function that provides a simple template for running 'JAGS' from 'R'. Referenced materials can be found at <[DOI:10.1214/ss/1177011136](https://doi.org/10.1214/ss/1177011136)>.

**License** GPL-2

**Encoding** UTF-8

**LazyData** true

**LazyDataCompression** xz

**RoxygenNote** 7.3.1

**Depends** R (>= 3.5.0)

**Imports** jagsUI, MASS

**Suggests** knitr, rmarkdown, testthat (>= 3.0.0)

**Config/testthat/edition** 3

**URL** <https://github.com/mbtyers/jagshelper>

**BugReports** <https://github.com/mbtyers/jagshelper/issues>

**VignetteBuilder** knitr

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2024-07-30 17:20:02 UTC

## Contents

jagshelper-package . . . . .	3
asdf_jags_out . . . . .	4
asdf_prior_jags_out . . . . .	4
caterpillar . . . . .	5
chaindens_df . . . . .	7
chaindens_jags . . . . .	8
chaindens_line . . . . .	9
check_neff . . . . .	10
check_Rhat . . . . .	10
comparecat . . . . .	11
comparedens . . . . .	13
comparepriors . . . . .	14
cor_jags . . . . .	15
crossplot . . . . .	16
envelope . . . . .	19
expit . . . . .	21
jags_df . . . . .	22
jags_plist . . . . .	23
logit . . . . .	24
nbyname . . . . .	24
nparam . . . . .	25
overlayenvelope . . . . .	26
pairstrace_jags . . . . .	28
plotcor_jags . . . . .	29
plotdens . . . . .	31
plotRhats . . . . .	33
plot_postpred . . . . .	34
pull_post . . . . .	37
qq_postpred . . . . .	38
rcolors . . . . .	39
skeleton . . . . .	40
SS_data . . . . .	40
SS_out . . . . .	41
tracedens_jags . . . . .	41
traceworstRhat . . . . .	42
trace_df . . . . .	44
trace_jags . . . . .	45
trace_line . . . . .	46
ts_postpred . . . . .	47

## Index

49

---

jagshelper-package      *Functions for Extracting and Visualizing Output from 'jagsUI'*

---

## Description

Functions are provided to help run Bayesian analyses in JAGS using the 'jagsUI' package. Included are functions for extracting output in simpler format, functions for streamlining assessment of convergence, and functions for producing summary plots of output. Also included is a function that provides a simple template for running JAGS from R.

## Details

Package: jagshelper  
Type: Package  
Version: 0.3.1  
Date: 2024-07-30  
License: GPL-2

The jagshelper package is intended to extend and streamline Bayesian analysis using the 'jagsUI' package.

The [skeleton](#) function prints a template JAGS model with associated R code to the console, which can easily be copied & pasted to an R script and modified as needed.

Functions are also provided for visually assessing model convergence. In particular, [tracedens\\_jags](#) gives a relatively simple syntax for trace plots of a collection or subset of parameter nodes, and overlays by-chain kernel densities for visual assessment of marginal posterior shapes as well as overlap between MCMC chains. Another function that could be particularly useful to users is [plotRhats](#), which gives a visual representation of the values of the Gelman-Rubin convergence diagnostic  $R_{hat}$  (or alternately effective sample size  $n_{eff}$ ) for all saved parameters. This may be particularly useful in the case where a model has many saved parameters. Additionally, function [traceworstRhat](#) is a wrapper for [tracedens\\_jags](#), but only produces trace plots for the parameter nodes with the worst (largest) values of  $R_{hat}$  or  $n_{eff}$ . Functions [qq\\_postpred](#), [ts\\_postpred](#), and [plot\\_postpred](#) provide some posterior predictive checks of a vector of data and corresponding vector (matrix, in output form) of posterior predictive samples.

Functions are also provided for visualizing posterior densities; in particular, the case of a vector of parameter nodes (one-dimensional in the JAGS model, giving a two-dimensional matrix of MCMC iterations). Notably, the [envelope](#) function is intended for a sequence of nodes (as in a time series), and the [caterpillar](#) function is intended for cases in which order may not matter (as in a collection of random effects). The [crossplot](#) function provides methods for bivariate plotting of two parameters, or for overlaying paired nodes of two parameter vectors.

Wrapper functions are also given for overlay of multiple such plots, as [overlayenvelope](#) and [comparecat](#), and [comparedens](#) giving plots as vertically-oriented left- and right-facing kernel densities.

**Author(s)**

Matt Tyers

Maintainer: Matt Tyers <matttyersstat@gmail.com>

---

asdf\_jags\_out

*Example data: asdf jags out*

---

**Description**

A simple model, equivalent to that produced by the output produced by [\link{skeleton}](#).

**Usage**

asdf\_jags\_out

**Format**

An object of class jagsUI of length 24.

---

asdf\_prior\_jags\_out

*Example data: asdf prior jags out*

---

**Description**

A simple model, equivalent to that produced by the output produced by [\link{skeleton}](#), with the addition of prior samples for all parameters.

**Usage**

asdf\_prior\_jags\_out

**Format**

An object of class jagsUI of length 24.

caterpillar

*Caterpillar plot***Description**

Caterpillar plot of the posterior densities of a vector of parameter nodes, in which the sequential order of nodes might not be important, such as vector of random effects.

This produces a set of overlaid interval bars (default values are 50 percent and 95 percent), with overlaid median markings, for each of a vector of parameter nodes.

**Usage**

```
caterpillar(
  df,
  p = NULL,
  x = NA,
  row = NULL,
  column = NULL,
  median = TRUE,
  mean = FALSE,
  ci = c(0.5, 0.95),
  lwd = 1,
  col = 4,
  add = FALSE,
  xlab = "",
  ylab = "",
  main = NULL,
  ylim = NULL,
  xax = NA,
  transform = c("none", "exp", "expit"),
  medlwd = lwd,
  medwd = 1,
  ...
)
```

**Arguments**

df	Output object returned from <code>jagsUI::jags()</code> ; or alternately, two-dimensional <code>data.frame</code> or matrix in which parameter node element is given by column and MCMC iteration is given by row. A vector may also be used, that expresses MCMC iterations of a single parameter node.
p	Parameter name, if input to <code>df</code> is a <code>jagsUI</code> output object.
x	Vector of X-coordinates for plotting.
row	Row to subset, in the case of a 2-d matrix of parameter nodes in-model.
column	Column to subset, in the case of a 2-d matrix of parameter nodes in-model.

median	Whether to include medians
mean	Whether to include means
ci	Vector of intervals to overlay. Defaults to 50 percent and 95 percent.
lwd	Base line width for plotting. Defaults to 1.
col	Color for plotting
add	Whether to add to existing plot
xlab	X-axis label
ylab	Y-axis label
main	Plot title. If the default (NULL) is accepted and argument p is used, p will be used for the title.
ylim	Y-axis limits. If the default (NULL) is accepted, the limits will be determined automatically.
xax	Vector of possible x-axis tick labels. Defaults to the data.frame column names.
transform	Should the y-axis be (back)transformed? Options are "exp", indicating exponential, or "expit", indicating inverse-logit. Defaults to "none", indicating no transformation. Note: if transform="exp" is used, consider adding additional plotting argument log="y".
medlwd	Line width of median line
medwd	Relative width of median line. Defaults to 1, perhaps smaller numbers will look better?
...	additional plotting arguments

**Value**

NULL

**Author(s)**

Matt Tyers

**See Also**

[envelope](#), [crossplot](#)

**Examples**

```
## usage with input data.frame
a <- jags_df(asdf_jags_out, p="a")

caterpillar(a)
caterpillar(a, ci=seq(.1,.9,by=.1))
caterpillar(a, lwd=2)
caterpillar(a, xax=c("effect 1", "effect 2", "effect 3"))

## usage with input as jagsUI object
```

```

caterpillar(asdf_jags_out, p="a")
caterpillar(SS_out, p="rate")

## usage with a 2-d parameter matrix
caterpillar(SS_out, p="cycle_s", column=1)
caterpillar(SS_out, p="cycle_s", column=2)

## usage with an exponential transformation
caterpillar(SS_out, p="trend", transform="exp", ylab="exp transform")
caterpillar(SS_out, p="trend", transform="exp", ylab="exp transform", log="y")
caterpillar(SS_out, p="trend", transform="expit", ylab="expit (inv logit) transform")

```

---

chaindens\_df                      *By-chain kernel density of each column of a data.frame.*

---

### Description

By-chain kernel density plot of each column of a posterior data.frame.

### Usage

```
chaindens_df(df, nline, parmflow = NULL, ...)
```

### Arguments

df	Posterior data.frame
nline	Number of chains
parmflow	Optional call to par(mfrow) for the number of rows & columns of plot window. Returns the graphics device to previous state afterward.
...	additional plotting arguments or arguments

### Value

NULL

### Author(s)

Matt Tyers

### See Also

[tracedens\\_jags](#), [trace\\_jags](#), [trace\\_line](#)

### Examples

```

a <- jags_df(asdf_jags_out, p="a")

chaindens_df(a, nline=3, parmflow=c(3,1))

```

---

chaindens\_jags      *By-chain kernel densities of jagsUI object*

---

### Description

By-chain kernel densities of a whole jagsUI object, or optional subset of parameter nodes.

### Usage

```
chaindens_jags(x, p = NULL, exact = FALSE, parmflow = NULL, lwd = 1, ...)
```

### Arguments

x	Posterior jagsUI object
p	Parameter name for subsetting: if this is specified, only parameters with names beginning with this string will be plotted.
exact	Whether p should be an exact match (TRUE) or just match the beginning of the string (FALSE). Defaults to FALSE.
parmflow	Optional call to par(mfrow) for the number of rows & columns of plot window. Returns the graphics device to previous state afterward.
lwd	Line width for plotting. Defaults to 1.
...	additional plotting arguments

### Value

NULL

### Author(s)

Matt Tyers

### See Also

[tracedens\\_jags](#), [trace\\_jags](#), [chaindens\\_line](#), [chaindens\\_df](#)

### Examples

```
chaindens_jags(asdf_jags_out, parmflow=c(4,2))
chaindens_jags(x=asdf_jags_out, p="a", parmflow=c(3,1))
```



---

chaindens_line	<i>Simple by-chain kernel density plot</i>
----------------	--

---

**Description**

By-chain kernel density plot of a single parameter node.

**Usage**

```
chaindens_line(x, nline, lwd = 1, main = "", ...)
```

**Arguments**

x	Posterior vector
nline	Number of chains
lwd	Line width
main	Plot title
...	additional plotting arguments

**Value**

NULL

**Author(s)**

Matt Tyers

**See Also**

[tracedens\\_jags](#), [chaindens\\_jags](#), [chaindens\\_df](#)

**Examples**

```
b1 <- jags_df(asdf_jags_out, p="b1")
chaindens_line(b1, nline=3, main="b1")
```

---

check\_neff                      *Quick summary of n.eff values by parameter name*

---

### Description

Returns the mean number of n.eff values (by each parameter) that are greater than a specified threshold criterion.

n.eff is calculated within 'JAGS', and may be interpreted as a crude measure of effective sample size for a given parameter node.

### Usage

```
check_neff(x, thresh = 500)
```

### Arguments

x	Output object from <code>jagsUI::jags()</code>
thresh	Threshold value (defaults to 500)

### Value

Numeric (named) giving the proportion of n.eff values above the given threshold.

### Author(s)

Matt Tyers

### See Also

[check\\_Rhat](#), [traceworstRhat](#), [plotRhats](#), [qq\\_postpred](#), [ts\\_postpred](#)

### Examples

```
check_neff(SS_out)
```

---

check\_Rhat                      *Quick summary of Rhat values by parameter name*

---

### Description

Returns the mean number of Rhat values for each parameter (by each parameter) that are less than a specified threshold criterion.

Rhat (Gelman-Rubin Convergence Diagnostic, or Potential Scale Reduction Factor) is calculated within 'JAGS', and is commonly used as a measure of convergence for a given parameter node. Values close to 1 are seen as evidence of adequate convergence.

**Usage**

```
check_Rhat(x, thresh = 1.1)
```

**Arguments**

x                    Output object from `jagsUI::jags()`  
thresh              Threshold value (defaults to 1.1)

**Value**

Numeric (named) giving the proportion of Rhat values below the given threshold.

**Author(s)**

Matt Tyers

**References**

Gelman, A., & Rubin, D. B. (1992). Inference from Iterative Simulation Using Multiple Sequences. *Statistical Science*, 7(4), 457–472. <http://www.jstor.org/stable/2246093>

**See Also**

[check\\_neff](#), [traceworstRhat](#), [plotRhats](#), [qq\\_postpred](#), [ts\\_postpred](#)

**Examples**

```
check_Rhat(SS_out)
```

---

comparecat

*Compare Caterpillar Plots*

---

**Description**

Interleaved caterpillar plots for all parameters (or a specified subset) from a list of `jagsUI` output objects or `data.frames`. The intent of this function is easy comparison of inferences from multiple comparable models.

Here a [caterpillar](#) plot is defined as a set of overlaid interval bars (default values are 50 percent and 95 percent), with overlaid median markings, for each of a vector of parameter nodes.

**Usage**

```
comparecat(
  x,
  p = NULL,
  ci = c(0.5, 0.95),
  ylim = NULL,
  col = NULL,
  xlab = "",
  ylab = "",
  transform = c("none", "exp", "expit"),
  ...
)
```

**Arguments**

<code>x</code>	List of output objects returned from <code>jagsUI</code> or <code>data.frames</code>
<code>p</code>	Optional vector of parameters to subset. All parameters with names matching the beginning of the string supplied will be returned. If the default (NULL) is accepted, all parameters will be plotted.
<code>ci</code>	Credible intervals widths to plot. Defaults to 50% and 95%.
<code>ylim</code>	Y-axis limits for plotting. If the default (NULL) is accepted, limits will be automatically determined.
<code>col</code>	Vector of colors for plotting. If the default (NULL) is accepted, colors will be automatically drawn.
<code>xlab</code>	X-axis label
<code>ylab</code>	Y-axis label
<code>transform</code>	Should the y-axis be (back)transformed? Options are "exp", indicating exponential, or "expit", indicating inverse-logit. Defaults to "none", indicating no transformation. Note: if <code>transform="exp"</code> is used, consider adding additional plotting argument <code>log="y"</code> .
<code>...</code>	additional plotting arguments

**Value**

NULL

**Author(s)**

Matt Tyers

**See Also**

[caterpillar](#), [crossplot](#), [comparedens](#), [comparepriors](#)

**Examples**

```
## This is the same output object three times, but shows functionality.
comparecat(x=list(asdf_jags_out, asdf_jags_out, asdf_jags_out),
           p=c("a", "b", "sig"))

## Transformed
comparecat(x=list(asdf_jags_out, asdf_jags_out, asdf_jags_out),
           p=c("sig"), transform="exp")
comparecat(x=list(asdf_jags_out, asdf_jags_out, asdf_jags_out),
           p=c("sig"), transform="exp", log="y")
```

---

comparedens

*Compare Density*


---

**Description**

Side-by-side kernel density plots for all parameters (or a specified subset) from two `jagsUI` output objects or `data.frames`. The intent of this function is easy comparison of inferences from two comparable models.

Kernel densities are plotted vertically, either left- or right-facing. Parameters with the same name are plotted facing one another.

**Usage**

```
comparedens(
  x1,
  x2,
  p = NULL,
  minCI = 0.99,
  ylim = NULL,
  legendnames = NULL,
  legendpos = "topleft",
  col = c(4, 2),
  ...
)
```

**Arguments**

<code>x1</code>	Output object returned from <code>jagsUI</code> ; or alternately a <code>data.frame</code>
<code>x2</code>	Second output object returned from <code>jagsUI</code> ; or alternately a <code>data.frame</code>
<code>p</code>	Optional vector of parameters to subset. All parameters with names matching the beginning of the string supplied will be returned. If the default ( <code>NULL</code> ) is accepted, all parameters will be plotted.
<code>minCI</code>	Minimum CI width for plotting. This is intended as a method for excluding far-outlying MCMC samples when determining the appropriate y-axis limits for plotting. Defaults to 99%.

ylim	Y-axis limits for plotting. If the default (NULL) is accepted, limits will be automatically determined.
legendnames	Names for optional legend. If the default NULL is accepted, no legend will be drawn.
legendpos	Position for optional legend. Defaults to "topleft".
col	Colors for kernel density plots. Defaults to colors 4 and 2 (blue and red).
...	additional plotting arguments

**Value**

NULL

**Author(s)**

Matt Tyers

**See Also**

[comparecat](#), [comparepriors](#)

**Examples**

```
## This is the same output object twice, but shows functionality.
comparedens(x1=asdf_jags_out, x2=asdf_jags_out, p=c("a","b","sig"),
            legendnames=c("Model 1", "Model 2"))
```

---

comparepriors

*Compare Priors*

---

**Description**

Side-by-side kernel density plots for all parameters with parameter names ending in "\_prior", and corresponding parameters without. It should be noted that these parameters must be specified in JAGS as well as the corresponding parameters, and this is left to the user.

This function is a wrapper of [comparedens](#).

Kernel densities are plotted vertically, either left- or right-facing. Parameters with the same name are plotted facing one another.

**Usage**

```
comparepriors(x, parmflow = NULL, ...)
```

**Arguments**

x	Output object returned from <code>jagsUI::jags()</code>
parmflow	Optional call to <code>par(mfrow)</code> for the number of rows & columns of plot window. Returns the graphics device to previous state afterward.
...	additional arguments to <a href="#">comparedens</a>

**Value**

NULL

**Author(s)**

Matt Tyers

**See Also**[comparecat](#), [comparedens](#), [plodens](#)**Examples**

```
## a look at what parameters exist in the input object
nbyname(asdf_prior_jags_out)

## then, showing the function usage
comparepriors(asdf_prior_jags_out, parmflow=c(2, 3))
```

---

cor\_jags

*Correlation matrix from a JAGS object*


---

**Description**

Computes a correlation matrix of all MCMC samples from an object returned by 'jagsUI', or an optional subset of parameter nodes.

**Usage**

```
cor_jags(x, p = NULL, exact = FALSE)
```

**Arguments**

x	Output object returned from jagsUI
p	Optional string to begin posterior names. If NULL is used, all parameters will be used
exact	Whether name must be an exact match (TRUE) or with initial sub-string matching only supplied characters (FALSE). Defaults to FALSE.

**Value**

A 2-dimensional correlation matrix (n X n, where n is the number of parameter nodes)

**Author(s)**

Matt Tyers

**See Also**[plotcor\\_jags](#)**Examples**

```
cor_jags(asdf_jags_out)
```

---

 crossplot

*Bivariate Plot of Posterior Densities*


---

**Description**

Bivariate plot of the posterior densities of corresponding vectors of parameter nodes. Three plotting methods are provided, that may be overlaid if desired.

- If `drawcross == TRUE`, [caterpillar](#)-like plots will be produced, with quantile intervals in the x- and y- directions.
- If `drawx == TRUE`, [caterpillar](#)-like plots will be produced, but rotated along the standardized principal component axes. This may be useful to draw if correlation is present.
- If `drawblob == TRUE`, smoothed polygons will be produced, each containing approximately `ci = x100%` of the associated MCMC samples.

All methods can overlay multiple bars or polygons, depending on the length of `ci`.

**Usage**

```
crossplot(
  dfx,
  dfy = NULL,
  p = NULL,
  col = 4,
  drawcross = TRUE,
  drawx = FALSE,
  drawblob = FALSE,
  blobres = NULL,
  blobsmooth = NULL,
  outline = FALSE,
  ci = c(0.5, 0.95),
  lwd = 1,
  mean = FALSE,
  link = FALSE,
  linklwd = 1,
  labels = FALSE,
  labelpos = NULL,
  labelcex = 0.7,
  whichx = NULL,
  rowx = NULL,
```



```

    columnx = NULL,
    whichy = NULL,
    rowy = NULL,
    columny = NULL,
    xlab = NULL,
    ylab = NULL,
    main = NULL,
    xlim = NULL,
    ylim = NULL,
    transformx = c("none", "exp", "expit"),
    transformy = c("none", "exp", "expit"),
    add = FALSE,
    ...
)

```

### Arguments

dfx	Output object returned from <code>jagsUI::jags()</code> ; or alternately, two-dimensional data.frame or matrix in which parameter node element is given by column and MCMC iteration is given by row. A vector may also be used, that expresses MCMC iterations of a single parameter node. If used with <code>dfy=</code> , this will be plotted in the x-direction.
dfy	Optionally, a two-dimensional data.frame or matrix in which parameter node element is given by column and MCMC iteration is given by row. A vector may also be used, that expresses MCMC iterations of a single parameter node. If used, this will be plotted in the y-direction.
p	Vector of parameter names, if input to <code>dfx</code> is a <code>jagsUI</code> output object. If used, this must be of length 2.
col	Color for plotting, or recyclable vector of colors. Defaults to 4. If <code>col == "random"</code> , <code>rcolors</code> will be used to generate a random vector of colors.
drawcross	Whether to draw quantile bars in the x- and y-directions. Defaults to TRUE.
drawx	Whether to draw quantile bars along the standardized principal component axes. Defaults to FALSE.
drawblob	Whether to draw smoothed quantile polygons. Defaults to FALSE.
blobres	Optional tuning parameter for drawing quantile polygons, and corresponds to the number of polygon vertices. If the default NULL is accepted, the function will supply a value based on the number of MCMC samples.
blobsmooth	Optional tuning parameter for drawing quantile polygons, and corresponds to half the number of polygon vertices used for local smoothing. If the default NULL is accepted, the function will supply a value based on the number of MCMC samples and the number of vertices.
outline	Whether to draw quantile polygons as lines rather than filled regions. Defaults to FALSE.
ci	Vector of intervals to overlay. Defaults to 50 percent and 95 percent.
lwd	Base line width for plotting. Defaults to 1.

mean	Whether to include points for means. Defaults to FALSE.
link	Whether to link medians in sequence. Defaults to FALSE.
linklwd	Line width to use for linking. Defaults to 1.
labels	Whether to add labels, or a vector of labels to add. Defaults to FALSE.
labelpos	Optionally, an argument to pos in <a href="#">text</a> for labels. Defaults to NULL.
labelcex	Optional character expansion for labels. Defaults to 0.7.
whichx	Element to subset for x, if only one element of a vector of parameter nodes is desired for plotting.
rowx	Row to subset for x, in the case of a 2-d matrix of parameter nodes in-model.
columnx	Column to subset for x, in the case of a 2-d matrix of parameter nodes in-model.
whichy	Element to subset for y, if only one element of a vector of parameter nodes is desired for plotting.
rowy	Row to subset for y, in the case of a 2-d matrix of parameter nodes in-model.
columny	Column to subset for y, in the case of a 2-d matrix of parameter nodes in-model.
xlab	X-axis label. If the default NULL is accepted, this will be drawn automatically.
ylab	Y-axis label. If the default NULL is accepted, this will be drawn automatically.
main	Plot title.
xlim	X-axis limits. If the default (NULL) is accepted, the limits will be determined automatically.
ylim	Y-axis limits. If the default (NULL) is accepted, the limits will be determined automatically.
transformx	Should the x-axis be (back)transformed? Options are "exp", indicating exponential, or "expit", indicating inverse-logit. Defaults to "none", indicating no transformation. Note: if transformx="exp" is used, consider adding additional plotting argument log="x" or log="xy".
transformy	Should the y-axis be (back)transformed? Options are "exp", indicating exponential, or "expit", indicating inverse-logit. Defaults to "none", indicating no transformation. Note: if transformy="exp" is used, consider adding additional plotting argument log="y" or log="xy".
add	Whether to add to existing plot
...	additional plotting arguments

**Value**

NULL

**Author(s)**

Matt Tyers

**See Also**[caterpillar](#), [pairstrace\\_jags](#)

**Examples**

```

## basic functionality with cross geometry
crossplot(SS_out, p=c("trend","rate"))

## default labels
crossplot(SS_out, p=c("trend","cycle"), labels=TRUE)

## showing:
## - link lines
## - blob geometry (smoothed confidence polygons)
## - random colors with col="random"
crossplot(SS_out, p=c("trend","cycle"),
          labels=SS_data$x, labelpos=1, link=TRUE, drawblob=TRUE,
          col="random")

## adding x geometry and showing usage with a single vector element (41)
crossplot(SS_out, p=c("trend","cycle"),
          whichx=41, whichy=41,
          drawblob=TRUE, drawx=TRUE)

## single vectors (or data.frames or 2d matrices) can also be used
xx <- SS_out$sims.list$trend[,41]
yy <- SS_out$sims.list$cycle[,41]

par(mfrow = c(2, 2))
plot(xx, yy, col=adjustcolor(1, alpha.f=.1), pch=16, main="cross geometry")
crossplot(xx, yy, add=TRUE, col=1)
plot(xx, yy, col=adjustcolor(1, alpha.f=.1), pch=16, main="x geometry")
crossplot(xx, yy, add=TRUE, col=1,
          drawcross=FALSE, drawx=TRUE)
plot(xx, yy, col=adjustcolor(1, alpha.f=.1), pch=16, main="blob geometry")
crossplot(xx, yy, add=TRUE, col=1,
          drawcross=FALSE, drawblob=TRUE)
plot(xx, yy, col=adjustcolor(1, alpha.f=.1), pch=16, main="blob outlines")
crossplot(xx, yy, add=TRUE, col=1,
          drawcross=FALSE, drawblob=TRUE, outline=TRUE)

```

---

envelope

*Envelope plot*


---

**Description**

Envelope plot of the posterior densities of a vector of parameter nodes, in which the sequential order of nodes is important, such as a time series.

This produces a plot of overlaid shaded strips, each corresponding to a given interval width (defaults to 50 percent and 95 percent), with an overlaid median line.

**Usage**

```
envelope(
  df,
  p = NULL,
  x = NA,
  row = NULL,
  column = NULL,
  median = TRUE,
  ci = c(0.5, 0.95),
  col = 4,
  add = FALSE,
  dark = 0.3,
  outline = FALSE,
  xlab = "",
  ylab = "",
  main = NULL,
  ylim = NULL,
  transform = c("none", "exp", "expit"),
  ...
)
```

**Arguments**

<code>df</code>	Output object returned from <code>jagsUI::jags()</code> ; or alternately, two-dimensional data. <code>frame</code> or matrix in which parameter node element is given by column and MCMC iteration is given by row.
<code>p</code>	Parameter name, if input to <code>df</code> is a <code>jagsUI</code> output object.
<code>x</code>	Vector of X-coordinates for plotting.
<code>row</code>	Row to subset, in the case of a 2-d matrix of parameter nodes in-model.
<code>column</code>	Column to subset, in the case of a 2-d matrix of parameter nodes in-model.
<code>median</code>	Whether to include median line
<code>ci</code>	Vector of intervals to overlay. Defaults to 50 percent and 95 percent.
<code>col</code>	Color for plotting
<code>add</code>	Whether to add to existing plot
<code>dark</code>	Opacity (0-1) for envelopes. Note that multiple overlapping intervals will darken the envelope.
<code>outline</code>	Whether to just envelope outlines
<code>xlab</code>	X-axis label
<code>ylab</code>	Y-axis label
<code>main</code>	Plot title. If the default (NULL) is accepted and argument <code>p</code> is used, <code>p</code> will be used for the title.
<code>ylim</code>	Y-axis limits for plotting. If the default (NULL) is accepted, these will be determined automatically.

transform      Should the y-axis be (back)transformed? Options are "exp", indicating exponential, or "expit", indicating inverse-logit. Defaults to "none", indicating no transformation. Note: if transform="exp" is used, consider adding additional plotting argument log="y".

...              additional plotting arguments or arguments to lines()

**Value**

NULL

**Author(s)**

Matt Tyers

**See Also**

[overlayenvelope](#), [caterpillar](#)

**Examples**

```
## usage with input data.frame
trend <- jags_df(SS_out, p="trend")
envelope(trend, x=SS_data$x)

## usage with jagsUI object
envelope(SS_out, p="trend")

## usage with 2-d jagsUI object
envelope(SS_out, p="cycle_s", column=1, main="cycle")
envelope(SS_out, p="cycle_s", column=2, col=2, add=TRUE) ## overlay

## scale transformation
envelope(SS_out, p="trend", transform="exp", ylab="exp transform")
envelope(SS_out, p="trend", transform="exp", ylab="exp transform", log="y")
```

---

expit

*Expit, or inverse logit*

---

**Description**

Inverse logit, where logit is defined as  $\log(x/(1-x))$ .

Expit (inverse logit) is defined as  $\exp(x)/(1+\exp(x))$ .

**Usage**

expit(x)

**Arguments**

x                    Numeric vector

**Value**

Numeric vector

**Author(s)**

Matt Tyers

**See Also**

[logit](#)

**Examples**

```
expit(0)
```

---

jags\_df

*Extract data.frame*

---

**Description**

Extracts the posterior samples from `jagsUI` output in the form of a `data.frame`. This simpler construction has a few benefits: operations may be more straightforward, and posterior objects will be smaller files and can be written to an external table or `.csv`, etc.

**Usage**

```
jags_df(x, p = NULL, exact = FALSE)
```

**Arguments**

x                    Output object from `jagsUI::jags()`

p                    Optional string to begin posterior names. If `NULL` is used, all parameters will be returned.

exact                Whether name must be an exact match (`TRUE`) or with initial sub-string matching only supplied characters (`FALSE`). Defaults to `FALSE`.

**Value**

A `data.frame` with a column associated with each parameter and a row associated with each MCMC iteration.

**Author(s)**

Matt Tyers

**See Also**[pull\\_post](#)**Examples**

```
out_df <- jags_df(asdf_jags_out)
```

---

`jags_plist`*Plist*

---

**Description**

Extracts a list of matrices, one for each saved parameter node. Each list element will be all posterior samples from that parameter node, arranged in a matrix with a column associated with each MCMC chain and a row for each MCMC iteration.

**Usage**

```
jags_plist(x, p = NULL, exact = FALSE)
```

**Arguments**

<code>x</code>	jagsUI output object
<code>p</code>	String to subset parameter names, if a subset is desired
<code>exact</code>	Whether <code>p</code> should be an exact match (TRUE) or just match the beginning of the string (FALSE). Defaults to FALSE.

**Value**

A list with an element associated with each parameter. Each element will be a matrix with a column associated with each MCMC chain and a row for each MCMC iteration.

**Note**

It is unlikely that a user will need this function; it is included primarily as a helper function used by other functions in this package.

**Author(s)**

Matt Tyers

**Examples**

```
out_plist <- jags_plist(asdf_jags_out)
str(out_plist)

a_plist <- jags_plist(asdf_jags_out, p=c("a", "sig_a"))
str(a_plist)
```

---

logit	<i>Logit</i>
-------	--------------

---

**Description**

Logit  $\log(x/(1-x))$

**Usage**

logit(x)

**Arguments**

x                    Numeric vector

**Value**

Numeric vector

**Author(s)**

Matt Tyers

**See Also**

[expit](#)

**Examples**

```
logit(0.5)
```

---

nbyname	<i>Number of parameter nodes by parameter name</i>
---------	--

---

**Description**

Returns a list of the numbers of parameter nodes saved in jagsUI output, by parameter name. As a default, what is returned for each list element is a vector of the array dimensions within the JAGS model (that is, excluding the dimension associated with the number of MCMC samples for each parameter node), or alternately, just the total number of parameter nodes.

**Usage**

```
nbyname(x, justtotal = FALSE)
```



**Arguments**

`x` Output object from `jagsUI::jags()`  
`justtotal` Whether to just report the total number of parameters, as opposed to dimensions.

**Value**

A list with an element associated with each parameter. Each element can be interpreted as the vector length or array dimension associated with the given parameter.

**Author(s)**

Matt Tyers

**See Also**

[nparam](#)

**Examples**

```
head(jags_df(asdf_jags_out))
nbyname(asdf_jags_out)

nparam(SS_out)
nbyname(SS_out)
```

---

nparam	<i>Number of parameters</i>
--------	-----------------------------

---

**Description**

Total number of individual parameter nodes saved in `jagsUI` output.

**Usage**

```
nparam(x)
```

**Arguments**

`x` Output object from `jagsUI::jags()`

**Value**

A single numeric value giving the number of parameter nodes.

**Author(s)**

Matt Tyers

**See Also**[nbyname](#)**Examples**

```
head(jags_df(asdf_jags_out))  
  
nparam(asdf_jags_out)
```

---

overlayenvelope	<i>Overlay envelope plots</i>
-----------------	-------------------------------

---

**Description**

Overlays multiple envelope plots of posterior data.frames, or outputs returned from jagsUI. This would be best suited to a set of posterior data.frames or 2-d matrices representing sequential vectors of parameter nodes.

Here a single [envelope](#) plot is defined as a set of overlaid shaded strips, each corresponding to a given interval width (defaults to 50 percent and 95 percent), with an overlaid median line.

**Usage**

```
overlayenvelope(  
  df,  
  p = NULL,  
  x = NA,  
  row = NULL,  
  column = NULL,  
  median = TRUE,  
  ci = c(0.5, 0.95),  
  col = NULL,  
  add = FALSE,  
  dark = 0.3,  
  outline = FALSE,  
  xlab = "",  
  ylab = "",  
  main = NULL,  
  ylim = NULL,  
  legend = TRUE,  
  legendnames = NULL,  
  legendpos = "topleft",  
  transform = c("none", "exp", "expit"),  
  ...  
)
```

**Arguments**

<code>df</code>	Primary input can be specified in a number of ways: either a <code>list()</code> of posterior <code>data.frames</code> or matrices, a list of output objects returned from <code>jagsUI::jags()</code> , a 3-dimensional array in which the input matrices to plot are separated according to the 3rd array dimension, or a single output object returned from <code>jagsUI::jags()</code> with multiple arguments passed to <code>p</code> , following.
<code>p</code>	Parameter name, if input to <code>df</code> is a list of <code>jagsUI</code> output objects; or a vector of parameter names, if input to <code>df</code> is a single <code>jagsUI</code> output object.
<code>x</code>	Optional vector of X-coordinates for plotting.
<code>row</code>	Row to subset, in the case of a 2-d matrix of parameter nodes in-model.
<code>column</code>	Column to subset, in the case of a 2-d matrix of parameter nodes in-model.
<code>median</code>	Whether to include median line
<code>ci</code>	Vector of intervals to overlay. Defaults to 50 percent and 95 percent.
<code>col</code>	Vector of colors for plotting
<code>add</code>	Whether to add to existing plot
<code>dark</code>	Opacity (0-1) for envelopes. Note that multiple overlapping intervals will darken the envelope. Defaults to 0.3.
<code>outline</code>	Whether to just envelope outlines
<code>xlab</code>	X-axis label
<code>ylab</code>	Y-axis label
<code>main</code>	Plot title. If the default (NULL) is accepted and argument <code>p=</code> is used, <code>p</code> will be used for the title.
<code>ylim</code>	Y-axis limits for plotting. If the default (NULL) is accepted, these will be determined automatically.
<code>legend</code>	Whether to automatically try to add a legend. Defaults to TRUE.
<code>legendnames</code>	Optional vector of names for a legend.
<code>legendpos</code>	Position for optional legend. Defaults to "topleft".
<code>transform</code>	Should the y-axis be (back)transformed? Options are "exp", indicating exponential, or "expit", indicating inverse-logit. Defaults to "none", indicating no transformation. Note: if <code>transform="exp"</code> is used, consider adding additional plotting argument <code>log="y"</code> .
<code>...</code>	additional plotting arguments or arguments to <code>lines()</code>

**Value**

NULL

**Author(s)**

Matt Tyers

**See Also**[envelope](#), [crossplot](#)

**Examples**

```
## usage with list of input data.frames
overlayenvelope(df=list(SS_out$sims.list$cycle_s[,1],
                        SS_out$sims.list$cycle_s[,2]))

## usage with a 3-d input array
overlayenvelope(df=SS_out$sims.list$cycle_s)

## usage with a jagsUI output object and parameter name (2-d parameter)
overlayenvelope(df=SS_out, p="cycle_s")

## usage with a single jagsUI output object and multiple parameters
overlayenvelope(df=SS_out, p=c("trend","rate"))

## exponential transformation
overlayenvelope(df=SS_out, p="cycle_s", transform="exp",
                ylab="exp transform")
overlayenvelope(df=SS_out, p="cycle_s", transform="exp",
                ylab="exp transform", log="y")
```

---

pairtrace\_jags

*Pairs trace plot*


---

**Description**

Two-dimensional trace plots (or alternately, scatter plots or contour plots) of each possible pair of parameters from a possible subset. May be useful in assessing correlation between parameter nodes, or problematic posterior surfaces.

**Usage**

```
pairtrace_jags(
  x,
  p = NULL,
  points = FALSE,
  contour = FALSE,
  lwd = 1,
  alpha = 0.2,
  parmflow = NULL,
  ...
)
```

**Arguments**

x	Output object returned from jagsUI
p	Optional vector of parameters to subset
points	Whether to plot as scatter plots instead. Defaults to FALSE.

contour	Whether to plot as contour plots instead. Defaults to FALSE.
lwd	Line width for trace plots. Defaults to 1.
alpha	Opacity of lines (or points, when points=TRUE). Defaults to 0.2.
parmfrow	Optional call to par(mfrow) for the number of rows & columns of plot window. Returns the graphics device to previous state afterward.
...	additional plotting arguments or arguments to tracedens_jags()

**Value**

NULL

**Author(s)**

Matt Tyers

**See Also**[tracedens\\_jags](#), [crossplot](#)**Examples**

```

pairstrace_jags(SS_out, p="sig", parmfrow=c(2,3), lwd=2)
pairstrace_jags(SS_out, p="sig", parmfrow=c(2,3), points=TRUE)
pairstrace_jags(SS_out, p="sig", parmfrow=c(2,3), contour=TRUE)

pairstrace_jags(asdf_jags_out, parmfrow=c(3,3))
pairstrace_jags(asdf_jags_out, parmfrow=c(3,3), points=TRUE)
pairstrace_jags(asdf_jags_out, parmfrow=c(3,3), contour=TRUE)

```

---

`plotcor_jags`*Plot a correlation matrix from a JAGS object*

---

**Description**

Plots a correlation matrix of all MCMC samples from an object returned by 'jagsUI', or an optional subset of parameter nodes. Correlation is plotted as shades of red (positive) or blue (negative).

In the case of vectors or arrays of nodes for each parameter name, a single axis tick will be used for all nodes with a single name. This has the effect of giving greater visual weight to single parameters, and reducing plot clutter.

Values of correlation are overlaid for all parameters with few nodes, with character size scaled according to the absolute correlation.

**Usage**

```
plotcor_jags(  
  x,  
  p = NULL,  
  exact = FALSE,  
  mincor = 0,  
  maxn = 4,  
  maxcex = 1,  
  legend = TRUE,  
  ...  
)
```

**Arguments**

x	Output object returned from <code>jagsUI</code> , or a <code>data.frame</code> with MCMC output
p	Optional string to begin posterior names. If <code>NULL</code> is used, all parameters will be used
exact	Whether name must be an exact match ( <code>TRUE</code> ) or with initial sub-string matching only supplied characters ( <code>FALSE</code> ). Defaults to <code>FALSE</code> .
mincor	Minimum (absolute) correlation to use for text labels. Defaults to 0 (all will be plotted)
maxn	Maximum number of nodes per parameter name for text labels, to prevent plot clutter. Defaults to 4.
maxcex	Maximum character expansion factor for text labels. Defaults to 1.
legend	Whether to produce a plot legend. Defaults to <code>TRUE</code> .
...	Optional plotting arguments

**Value**

`NULL`

**Author(s)**

Matt Tyers

**See Also**

[cor\\_jags](#)

**Examples**

```
plotcor_jags(asdf_jags_out, maxcex=0.7)  
  
plotcor_jags(SS_out, p=c("trend", "rate", "sig"))
```

---

plotdens	<i>Plot kernel densities of single parameter nodes</i>
----------	--

---

### Description

Produces a kernel density plot of a single or multiple parameter nodes (overlaid).

Input can be of multiple possible formats: either a single or list of output objects from `jagsUI` with an associated vector of parameter names, or a vector or `data.frame` of posterior samples.

### Usage

```
plotdens(
  df,
  p = NULL,
  exact = FALSE,
  add = FALSE,
  col = NULL,
  shade = TRUE,
  lwd = 2,
  minCI = 0.99,
  legend = TRUE,
  legendpos = "topleft",
  legendnames = NULL,
  main = NULL,
  xlab = "",
  ylab = "Density",
  ...
)
```

### Arguments

<code>df</code>	Input object for plotting. See examples below.
<code>p</code>	Vector of parameter names, if <code>df</code> is given as a single or list of output objects from <code>jagsUI</code>
<code>exact</code>	Whether the <code>p=</code> argument should match the parameter name exactly. See <a href="#">jags_df</a> for details.
<code>add</code>	Whether to add to an existing plot ( <code>TRUE</code> ) or produce a new plot. Defaults to <code>FALSE</code> .
<code>col</code>	Vector of colors for plotting. If the default ( <code>NULL</code> ) is accepted, colors will be automatically selected.
<code>shade</code>	Whether to shade the regions below the kernel density curve(s). Defaults to <code>TRUE</code> .
<code>lwd</code>	Line width for kernel density curves. Defaults to 2. Note: setting this to 0 (or <code>FALSE</code> ) will suppress lines.

minCI	Minimum CI width to include for all density curves. Defaults to 99%.
legend	Whether to plot a legend. Defaults to TRUE.
legendpos	Position for automatic legend. Defaults to "topleft".
legendnames	Names for legend
main	Plot title. Defaults to "".
xlab	X-axis label. Defaults to "".
ylab	Y-axis label. Defaults to "Density".
...	Optional plotting arguments

**Value**

NULL

**Author(s)**

Matt Tyers

**See Also**

[comparedens](#), [comparecat](#), [comparepriors](#)

**Examples**

```
## jagsUI object with a single parameter
plotdens(asdf_jags_out, p="b1")

## jagsUI object with multiple nodes of a parameter
plotdens(asdf_jags_out, p="a")

## jagsUI object with multiple parameter nodes
plotdens(asdf_jags_out, p=c("a[1]","a[2]","a[3]"))

## data.frame with multiple columns
plotdens(jags_df(asdf_jags_out, p="a"))

## list of jagsUI objects with a single parameter name
plotdens(list(asdf_jags_out,asdf_jags_out,asdf_jags_out), p="b1")

## list of jagsUI objects with a vector of parameter names
plotdens(list(asdf_jags_out,asdf_jags_out,asdf_jags_out), p=c("a[1]","a[2]","a[3]"))
```



---

plotRhats

*Plotting all Rhat values*


---

### Description

Plotting all values of Rhat (or alternately `n.eff`) from an output object returned by `jagsUI`, or perhaps a subset of parameters. This function is intended as a quick graphical check of which parameters have adequately converged.

Rhat (Gelman-Rubin Convergence Diagnostic, or Potential Scale Reduction Factor) is calculated within 'JAGS', and is commonly used as a measure of convergence for a given parameter node. Values close to 1 are seen as evidence of adequate convergence. `n.eff` is also calculated within 'JAGS', and may be interpreted as a crude measure of effective sample size for a given parameter node.

### Usage

```
plotRhats(
  x,
  p = NULL,
  n.eff = FALSE,
  fence = NULL,
  plotsequence = FALSE,
  splitarr = FALSE,
  margin = NULL,
  ...
)
```

### Arguments

<code>x</code>	Output object returned from <code>jagsUI</code>
<code>p</code>	Optional vector of parameters to subset
<code>n.eff</code>	Optionally, whether to plot <code>n.eff</code> instead of Rhat. Defaults to <code>FALSE</code> .
<code>fence</code>	Value of horizontal lines to overlay as reference. Accepting the default value ( <code>NULL</code> ) will give fence values of 1.1 (a commonly used value) and 1.01 for Rhat, or 100 and 500 for <code>n.eff</code> .
<code>plotsequence</code>	Whether to plot parameter vectors (or matrices) in a sequence, running left to right, which may be useful for time series models, etc. If the default ( <code>FALSE</code> ) is used, a vertical cluster will be plotted for each parameter, resulting in a simpler plot if there are many parameters. Note that the Rhat values will still be plotted in sequence if the default ( <code>FALSE</code> ) is used.
<code>splitarr</code>	Whether to split 2+ dimensional parameter arrays by a given dimension, rather than plotting the full array in one vertical cluster or continuous sequence. This may be recommended in the case of large arrays. Defaults to <code>FALSE</code> .

margin      If splitarr= is set to TRUE, which array margin to split by. In the case of a 2-dimensional array, setting margin=2 will separate the array by column. If the default (NULL) is accepted, the function will split by the smallest dimension, therefore splitting into the fewest groups.

...          additional plotting arguments

**Value**

NULL

**Author(s)**

Matt Tyers

**References**

Gelman, A., & Rubin, D. B. (1992). Inference from Iterative Simulation Using Multiple Sequences. *Statistical Science*, 7(4), 457–472. <http://www.jstor.org/stable/2246093>

**See Also**

[tracworstRhat](#), [check\\_Rhat](#), [qq\\_postpred](#), [ts\\_postpred](#), [plot\\_postpred](#)

**Examples**

```
## plotting everything
plotRhats(SS_out)
str(SS_out$Rhat) # the associated values

plotRhats(SS_out, n.eff=TRUE)
str(SS_out$n.eff) # the associated values

## behavior of splitarr and margin are shown
plotRhats(SS_out)
plotRhats(SS_out, splitarr=TRUE)
str(SS_out$Rhat) # the associated values

## plotsequence may be useful in the case of a sequence of values
plotRhats(SS_out, p=c("trend", "cycle_s"), splitarr=TRUE, plotsequence=TRUE)
```

## Description

This is a wrapper function that produces a sequence of plots illustrating the posterior predictive distribution. Optional plots are:

- An [envelope](#) plot of the posterior predictive distribution as a time series, overlaid with the data values (if `plot_data=TRUE` is used)
- The centered posterior predictive distributions, as plotted by [ts\\_postpred](#), and overlaid with the data residuals (if `plot_residuals=TRUE` is used)
- The approximate residual standard deviation, calculated from a moving window of 10 data points in sequence. (if `plot_sd=TRUE` is used)

These three plots are repeated, for a sequence of different variables expressed on the x-axis, potentially highlighting different features of the dataset or model structure:

- The data sequence (if `whichplots=` contains 1)
- The `x=` variable supplied (if `whichplots=` contains 2)
- The `y=` variable supplied (if `whichplots=` contains 3)
- The fitted values, as estimated by the posterior predictive median (if `whichplots=` contains 4)

While not an omnibus posterior predictive check, this plot can be useful for detecting an overparameterized model, or else improper specification of observation error.

It should be noted that this function will only produce meaningful results with a vector of data, as opposed to a single value.

The posterior predictive distribution can be specified in two possible ways: either a single output object from `jagsUI` with an associated parameter name, or as a matrix or `data.frame` of posterior samples.

## Usage

```
plot_postpred(
  ypp,
  y,
  p = NULL,
  x = NULL,
  whichplots = c(1, 2, 4),
  plot_data = TRUE,
  plot_residuals = TRUE,
  plot_sd = TRUE,
  pch = 1,
  pointcol = 1,
  lines = FALSE,
  ...
)
```

## Arguments

`ypp` Either a matrix or `data.frame` of posterior samples, or an output object returned from `jagsUI` and a supplied parameter name

y	The associated data vector
p	A character name, if a jagsUI object is passed to ypp
x	The time measurements associated with time series y. If the default NULL is accepted, associated plots will be suppressed.
whichplots	A vector of which sets of plots to produce (that is, with respect to which variables on the x-axis). See above for details. Defaults to c(1, 2, 4).
plot_data	Whether to produce plots associated with the data (y=) time series and untransformed posterior predictive distribution. Defaults to TRUE.
plot_residuals	Whether to produce plots associated with the residual time series and posterior predictive residuals. Defaults to TRUE.
plot_sd	Whether to produce plots of the moving-window standard deviation of the residuals. Defaults to TRUE.
pch	Plotting character for points, which will accept a vector input. See <a href="#">points</a> . Defaults to 1.
pointcol	Plotting color for points. Defaults to 1.
lines	Whether to add a line linking data time series points. Defaults to FALSE.
...	Additional arguments to <a href="#">envelope</a>

**Value**

NULL

**Note**

This function assumes the existence of a matrix of posterior predictive samples corresponding to a data vector, the construction of which must be left to the user. This can be accomplished within JAGS, or using appropriate simulation from the posterior samples.

**Author(s)**

Matt Tyers

**See Also**

[qq\\_postpred](#), [ts\\_postpred](#), [check\\_Rhat](#), [check\\_neff](#), [traceworstRhat](#), [plotRhats](#)

**Examples**

```
# first, a quick look at the example data...
str(SS_data)
str(SS_out$sims.list$ypp)

# recommended usage
parmfrow <- par("mfrow") # storing graphics state
par(mfcol = c(3,3)) # a recommended setting to organize plots

plot_postpred(ypp=SS_out, p="ypp", y=SS_data$y, x=SS_data$x)

par(mfrow = parmfrow) # resetting graphics state
```

---

pull_post	<i>Subset from posterior data.frame</i>
-----------	---

---

### Description

Extracts a subset vector or `data.frame` from a `data.frame` consisting of more columns, such that column names match a name given in the `p`= argument. This may be useful in creating smaller objects consisting of MCMC samples.

### Usage

```
pull_post(x, p = NULL, exact = FALSE)
```

### Arguments

<code>x</code>	Posterior <code>data.frame</code>
<code>p</code>	String to begin posterior names. If <code>NULL</code> is used, all parameters will be returned.
<code>exact</code>	Whether name must be an exact match ( <code>TRUE</code> ) or with initial sub-string matching only supplied characters ( <code>FALSE</code> ). Defaults to <code>FALSE</code> .

### Value

A `data.frame` with a column associated with each (subsetting) parameter and a row associated with each MCMC iteration.

### Author(s)

Matt Tyers

### See Also

[jags\\_df](#)

### Examples

```
out_df <- jags_df(asdf_jags_out)

b <- pull_post(out_df, p="b")
str(b)
a <- pull_post(out_df, p=c("a","sig_a"))
str(a)
sigs <- pull_post(out_df, p="sig")
str(sigs)
justsig <- pull_post(out_df, p="sig", exact=TRUE)
str(justsig)
```

---

 qq\_postpred

---

*Quantile-quantile plot from posterior predictive distribution*


---

### Description

Produces a quantile-quantile plot, calculated from the quantiles of a vector of data (most likely a time series), with respect to the matrix of associated posterior predictive distributions.

While not an omnibus posterior predictive check, this plot can be useful for detecting an overparameterized model, or else improper specification of observation error. Like a traditional Q-Q plot, a well-specified model will have points that lie close to the  $x=y$  line. In the case of this function, an overparametrized model will typically produce a plot with a much shallower slope, possibly with many associated posterior predictive quantiles close to 0.5.

It should be noted that this function will only produce meaningful results with a vector of data, as opposed to a single value.

The posterior predictive distribution can be specified in two possible ways: either a single output object from jagsUI with an associated parameter name, or as a matrix or data.frame of posterior samples.

### Usage

```
qq_postpred(ypp, y, p = NULL, add = FALSE, ...)
```

### Arguments

ypp	Either a matrix or data.frame of posterior samples, or an output object returned from jagsUI and a supplied parameter name
y	The associated data vector
p	A character name, if a jagsUI object is passed to ypp
add	Whether to add the plot to an existing plot. Defaults to FALSE.
...	Optional plotting arguments

### Value

NULL

### Note

This function assumes the existence of a matrix of posterior predictive samples corresponding to a data vector, the construction of which must be left to the user. This can be accomplished within JAGS, or using appropriate simulation from the posterior samples.

### Author(s)

Matt Tyers

**See Also**

[ts\\_postpred](#), [plot\\_postpred](#), [check\\_Rhat](#), [check\\_neff](#), [traceworstRhat](#), [plotRhats](#)

**Examples**

```
# first, a quick look at the example data...
str(SS_data)
str(SS_out$sims.list$ypp)

# plotting the example posterior predictive distribution with the data
# points overlaid. Note the overdispersion in the posterior predictive.
caterpillar(SS_out, p="ypp")
points(SS_data$y)

# using a jagsUI object as ypp input
qq_postpred(ypp=SS_out, p="ypp", y=SS_data$y)

# using a matrix as ypp input
qq_postpred(ypp=SS_out$sims.list$ypp, y=SS_data$y)
```

---

rcolors

*Random Colors*

---

**Description**

Creates a vector of randomly-generated colors.

**Usage**

```
rcolors(n)
```

**Arguments**

n                      Vector length

**Value**

A vector of colors

**Author(s)**

Matt Tyers

**Examples**

```
n <- 1000
cols <- rcolors(n)
x <- runif(n)
y <- runif(n)
plot(x,y, col=cols, pch=16)
```

---

skeleton	<i>Skeleton</i>
----------	-----------------

---

**Description**

Prints an example 'JAGS' model and associated 'jagsUI' code to the console, along with code to simulate a corresponding dataset. This is intended to serve as a template that can be altered as needed by the user.

**Usage**

```
skeleton(NAME = "NAME")
```

**Arguments**

NAME                    Name to append to JAGS model object, etc.

**Value**

NULL

**Note**

The printed code will use the `cat()` function to write the model code to an external text file. It may be desirable to use a call to `\link{tempfile}()` instead, to eliminate creation of unneeded files.

**Author(s)**

Matt Tyers

**Examples**

```
skeleton("asdf")
```

---

SS_data	<i>Example data: Time series associated with SS JAGS out</i>
---------	--

---

**Description**

The time series and time measurements associated with the time series model `\link{SS_out}`.

**Usage**

```
SS_data
```

**Format**

An object of class `data.frame` with 41 rows and 2 columns.



---

SS_out	<i>Example data: SS JAGS out</i>
--------	----------------------------------

---

**Description**

A time series model with multiple observations of a single time series, and with two stochastic cycle components.

**Usage**

```
SS_out
```

**Format**

An object of class `jagsUI` of length 24.

**Details**

This model is included partly to show a model with vectors or 2-dimensional matrices of parameter nodes, and also to give an example of poor model convergence.

---

tracedens_jags	<i>Combination of trace plots and by-chain kernel densities of jagsUI object</i>
----------------	--

---

**Description**

Combination of trace plots and by-chain kernel densities of a whole `jagsUI` object, or optional subset of parameter nodes.

**Usage**

```
tracedens_jags(  
  x,  
  p = NULL,  
  exact = FALSE,  
  parmflow = NULL,  
  lwd = 1,  
  shade = TRUE,  
  ...  
)
```

**Arguments**

x	Posterior jagsUI object
p	Parameter name for subsetting: if this is specified, only parameters with names beginning with this string will be plotted.
exact	Whether p should be an exact match (TRUE) or just match the beginning of the string (FALSE). Defaults to FALSE.
parmfrow	Optional call to par(mfrow) for the number of rows & columns of plot window. Returns the graphics device to previous state afterward.
lwd	Line width for plotting. Defaults to 1.
shade	Whether to add semi-transparent shading to by-chain kernel densities. Defaults to TRUE.
...	additional plotting arguments

**Value**

NULL

**Author(s)**

Matt Tyers

**See Also**[trace\\_jags](#), [chaindens\\_jags](#), [pairstrace\\_jags](#)**Examples**

```
tracedens_jags(asdf_jags_out, parmfrow=c(4,2))
tracedens_jags(asdf_jags_out, p="a", parmfrow=c(3,1))
```

---

 traceworstRhat

---

*Trace plots corresponding to the worst values of Rhat*


---

**Description**

Trace plots with kernel densities will be created for parameters with the largest (worst) associated values of Rhat. This function is primarily intended for parameters with a vector (or array) of values. Rhat (Gelman-Rubin Convergence Diagnostic, or Potential Scale Reduction Factor) is calculated within 'JAGS', and is commonly used as a measure of convergence for a given parameter node. Values close to 1 are seen as evidence of adequate convergence. n.eff is also calculated within 'JAGS', and may be interpreted as a crude measure of effective sample size for a given parameter node.

**Usage**

```
traceworstRhat(x, p = NULL, n.eff = FALSE, margin = NULL, parmfrow = NULL, ...)
```

**Arguments**

x	Output object returned from jagsUI
p	Optional vector of parameters to subset
n.eff	Whether to plot parameters with the smallest associated values of n.eff instead. Defaults to FALSE.
margin	In the case of a 2+ dimensional array associated with a given parameter, this will have the effect of plotting the worst Rhat corresponding to each margin specified. For example, specifying margin=2 (column) will plot the parameter with the worst Rhat value from each column. In contrast, specifying margin=NULL (the default) will cause the function to plot the single array element with the largest Rhat value.
parmfrow	Optional call to par(mfrow) for the number of rows & columns of plot window. Returns the graphics device to previous state afterward.
...	additional plotting arguments or arguments to tracedens_jags()

**Value**

NULL

**Author(s)**

Matt Tyers

**References**

Gelman, A., & Rubin, D. B. (1992). Inference from Iterative Simulation Using Multiple Sequences. *Statistical Science*, 7(4), 457–472. <http://www.jstor.org/stable/2246093>

**See Also**

[plotRhats](#), [check\\_Rhat](#), [qq\\_postpred](#), [ts\\_postpred](#), [plot\\_postpred](#)

**Examples**

```
## plotting everything
traceworstRhat(SS_out, parmfrow=c(3,2))
SS_out$Rhat # the associated values

traceworstRhat(SS_out, parmfrow=c(3,2), n.eff=TRUE)
SS_out$n.eff # the associated values

## in the case of a 2-D array, setting margin=2 gives the max Rhat
## associated with each column, rather than the global max
traceworstRhat(x=SS_out, p="cycle_s", margin=2, parmfrow=c(2,2))
SS_out$Rhat
traceworstRhat(x=SS_out, p="cycle_s", margin=2, parmfrow=c(2,2), n.eff=TRUE)
SS_out$n.eff
```

---

trace_df	<i>Trace plot of each column of a data.frame.</i>
----------	---

---

**Description**

Trace plot of each column of a posterior 'data.frame'.

**Usage**

```
trace_df(df, nline, parmflow = NULL, ...)
```

**Arguments**

df	Posterior data.frame
nline	Number of chains
parmflow	Optional call to <code>par(mfrow)</code> for the number of rows & columns of plot window. Returns the graphics device to previous state afterward.
...	additional plotting arguments or arguments to <code>trace_line()</code>

**Value**

NULL

**Author(s)**

Matt Tyers

**See Also**

[tracedens\\_jags](#), [trace\\_jags](#), [trace\\_line](#)

**Examples**

```
a <- jags_df(asdf_jags_out, p="a")
trace_df(a, nline=3, parmflow=c(3,1))
```

---

trace_jags	<i>Trace plot of jagsUI object</i>
------------	------------------------------------

---

**Description**

Trace plot of a whole jagsUI object, or optional subset of parameter nodes.

**Usage**

```
trace_jags(x, p = NULL, exact = FALSE, parmfrow = NULL, lwd = 1, ...)
```

**Arguments**

x	Posterior jagsUI object
p	Parameter name for subsetting: if this is specified, only parameters with names beginning with this string will be plotted.
exact	Whether p should be an exact match (TRUE) or just match the beginning of the string (FALSE). Defaults to FALSE.
parmfrow	Optional call to par(mfrow) for the number of rows & columns of plot window. Returns the graphics device to previous state afterward.
lwd	Line width for plotting. Defaults to 1.
...	additional plotting arguments

**Value**

NULL

**Author(s)**

Matt Tyers

**See Also**

[tracedens\\_jags](#), [pairstrace\\_jags](#), [trace\\_df](#), [trace\\_line](#)

**Examples**

```
trace_jags(asdf_jags_out, parmfrow=c(4,2))
trace_jags(asdf_jags_out, p="a", parmfrow=c(3,1))
```

---

trace_line	<i>Simple trace plot</i>
------------	--------------------------

---

**Description**

Trace plot of a single parameter node.

**Usage**

```
trace_line(x, nline, lwd = 1, main = "", ...)
```

**Arguments**

x	Posterior vector
nline	Number of MCMC chains
lwd	Line width
main	Plot title
...	additional plotting arguments

**Value**

NULL

**Author(s)**

Matt Tyers

**See Also**

[tracedens\\_jags](#), [trace\\_jags](#), [trace\\_df](#), [chaindens\\_line](#)

**Examples**

```
b1 <- jags_df(asdf_jags_out, p="b1")
trace_line(b1, nline=3, main="b1")
```

ts\_postpred

*Time series plot of centered posterior predictive distribution***Description**

Produces a plot of centered posterior predictive distributions associated with a vector of data (most likely a time series), defined as the difference between posterior predictive and posterior predictive median.

Also overlays the posterior predictive residuals, defined as the differences between data values and their respective posterior predictive medians.

While not an omnibus posterior predictive check, this plot can be useful for detecting an overparameterized model, or else improper specification of observation error.

It should be noted that this function will only produce meaningful results with a vector of data, as opposed to a single value.

The posterior predictive distribution can be specified in two possible ways: either a single output object from `jagsUI` with an associated parameter name, or as a matrix or `data.frame` of posterior samples.

**Usage**

```
ts_postpred(
  ypp,
  y,
  p = NULL,
  x = NULL,
  lines = FALSE,
  pch = 1,
  pointcol = 1,
  transform = c("none", "exp", "expit"),
  ...
)
```

**Arguments**

<code>ypp</code>	Either a matrix or <code>data.frame</code> of posterior samples, or an output object returned from <code>jagsUI</code> and a supplied parameter name
<code>y</code>	The associated data vector
<code>p</code>	A character name, if a <code>jagsUI</code> object is passed to <code>ypp</code>
<code>x</code>	The time measurements associated with time series <code>y</code> . If the default <code>NULL</code> is accepted, equally-spaced integer values will be used.
<code>lines</code>	Whether to add a line linking data time series points. Defaults to <code>FALSE</code> .
<code>pch</code>	Plotting character for points, which will accept a vector input. See <a href="#">points</a> . Defaults to 1.
<code>pointcol</code>	Plotting color for points. Defaults to 1.

transform      Should the y-axis be (back)transformed? Options are "exp", indicating exponential, or "expit", indicating inverse-logit. Defaults to "none", indicating no transformation. Note: if transform="exp" is used, consider adding additional plotting argument log="y".

...              Additional arguments to [envelope](#)

**Value**

NULL

**Note**

This function assumes the existence of a matrix of posterior predictive samples corresponding to a data vector, the construction of which must be left to the user. This can be accomplished within JAGS, or using appropriate simulation from the posterior samples.

**Author(s)**

Matt Tyers

**See Also**

[qq\\_postpred](#), [plot\\_postpred](#), [check\\_Rhat](#), [check\\_neff](#), [traceworstRhat](#), [plotRhats](#)

**Examples**

```
# first, a quick look at the example data...
str(SS_data)
str(SS_out$sims.list$ypp)

# plotting the example posterior predictive distribution with the data
# points overlayed. Note the overdispersion in the posterior predictive.
caterpillar(SS_out, p="ypp")
points(SS_data$y)

# using a jagsUI object as ypp input
ts_postpred(ypp=SS_out, p="ypp", y=SS_data$y)

# using a matrix as ypp input
ts_postpred(ypp=SS_out$sims.list$ypp, y=SS_data$y)

# exp transformation
ts_postpred(ypp=SS_out, p="ypp", y=SS_data$y, transform="exp")
ts_postpred(ypp=SS_out, p="ypp", y=SS_data$y, transform="exp", log="y")
```



# Index

## \* datasets

- asdf\_jags\_out, 4
- asdf\_prior\_jags\_out, 4
- SS\_data, 40
- SS\_out, 41

asdf\_jags\_out, 4  
asdf\_prior\_jags\_out, 4

caterpillar, 3, 5, 11, 12, 16, 18, 21  
chaindens\_df, 7, 8, 9  
chaindens\_jags, 8, 9, 42  
chaindens\_line, 8, 9, 46  
check\_neff, 10, 11, 36, 39, 48  
check\_Rhat, 10, 10, 34, 36, 39, 43, 48  
comparecat, 3, 11, 14, 15, 32  
comparedens, 3, 12, 13, 14, 15, 32  
comparepriors, 12, 14, 14, 32  
cor\_jags, 15, 30  
crossplot, 3, 6, 12, 16, 27, 29

envelope, 3, 6, 19, 26, 27, 35, 36, 48  
expit, 21, 24

jags\_df, 22, 31, 37  
jags\_plist, 23  
jagshelper (jagshelper-package), 3  
jagshelper-package, 3

logit, 22, 24

nbyname, 24, 26  
nparam, 25, 25

overlayenvelope, 3, 21, 26

pairstrace\_jags, 18, 28, 42, 45  
plot\_postpred, 3, 34, 34, 39, 43, 48  
plotcor\_jags, 16, 29  
plotdens, 15, 31  
plotRhats, 3, 10, 11, 33, 36, 39, 43, 48

points, 36, 47  
pull\_post, 23, 37

qq\_postpred, 3, 10, 11, 34, 36, 38, 43, 48

rcolors, 17, 39

skeleton, 3, 40  
SS\_data, 40  
SS\_out, 41

text, 18  
trace\_df, 44, 45, 46  
trace\_jags, 7, 8, 42, 44, 45, 46  
trace\_line, 7, 44, 45, 46  
tracedens\_jags, 3, 7–9, 29, 41, 44–46  
tracworstRhat, 3, 10, 11, 34, 36, 39, 42, 48  
ts\_postpred, 3, 10, 11, 34–36, 39, 43, 47