Package 'psp'

June 7, 2021

Title Parameter Space Partitioning MCMC for Global Model Evaluation

Version 0.1

Date 2021-05-06

Description Implements an n-dimensional parameter space partitioning algorithm for evaluating the global behaviour of formal computational models as described by Pitt, Kim, Navarro and Myung (2006) <doi:10.1037/0033-295X.113.1.57>.

License GPL (>= 3)

Depends parallel

Encoding UTF-8

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation no

Author Lenard Dome [aut, cre],
Andy Wills [aut]

Maintainer Lenard Dome <lenarddome@protonmail.ch>

Repository CRAN

Date/Publication 2021-06-07 06:50:02 UTC

R topics documented:

psp-package *Parameter Space Partitioning MCMC for Global Model Evaluation*

Description

Implements global model evaluations for formal computational models in the cognitive sciences.

Author(s)

Lenard Dome

Maintainer: Lenard Dome <lenarddome@protonmail.ch>

psp_control *Control the behaviour of the psp_global implementation*

Description

psp_control allows users to define characteristics of the parameter space partitioning MCMC algorithm as implemented in psp_global.

Usage

```
psp_control(radius = 0.1, init = NULL, lower, upper,
            pop = 400, cl = NULL,
            param_names = NULL,
            parallel = TRUE,
            cluster_names = NULL,
            iterations = 1000)
```

Arguments

radius	The radius of the hypershere with n-dimensions to sample from. Must be a double. Default is 0.1.
init	A vector of parameters to use as the first jumping distribution. If NULL (default), parameter search starts from the center of the parameter space.
lower, upper	Vectors specifiying the lower and upper boundaries of the parameter space for each parameter. The i-th element of lower and upper bounds applies to the i-th parameter.
pop	The minimum population psp_global aims to find for each ordinal pattern discovered. This can stop the parameter search early in case the population of all ordinal pattern are equal to or larger than pop. If you do not want to use this option, set it to NULL or Inf. Default is 400.
parallel	If TRUE (default), uses the parallel package to run evaluations of jumping distributions for each chain parallel.

cl	If parallel is TRUE, the number of cores to use for makeCluster from the **parallel** package. If null (default), use all cores.
param_names	A character vector that includes the names of each parameter. If NULL (default), a character vector is generated with parameter_1, parameter_2, parameter_3", ...
cluster_names	A character vector that includes the list of functions to be loaded into each cluster. Default is NULL.
iterations	The number of global iterations for psp_global. Default is 1000.

Value

Returns a control list suitable for psp_global with the above elements.

See Also

psp_global.

Examples

```
# two parameter model
psp_control(lower = rep(0, 2), upper = rep(1, 2), init = rep(0.5, 2),
           radius = rep(0.25, 2), cluster_names = NULL,
           parallel = FALSE, iterations = 500)
```

---

psp_global *Parameter Space Partitioning*

Description

An all-purpose implementation of the Parameter Space Partitioning MCMC Algorithm described by Pitt, Kim, Navarro, Myung (2006).

Usage

```
psp_global(fn, control = psp_control())
```

Arguments

fn	The ordinal function. It should take a numeric vector (parameter set) as its argument, and return an ordinal response pattern as character (e.g. "A > B"). NA values are not currently allowed.
control	a list of control parameters, see psp_control

Details

This function implements the Parameter Space Partitioning algorithm desribed by Pitt et al. (2006). The algorithm is as follows:

0. Initialize parameter space.

0. Select first set of parameters, and evaluate the model on this set. Its ordinal output will become the first ordinal pattern and the first region in the parameter space.

1. Pick a random jumping distribution from for each ordinal pattern from the sampling region defined by a hypershere with a center of the last recorded parameter set for a given pattern.

2. Evaluate model on all new parameter sets.

3. Record new patterns and their corresponding parameter sets. If the parameter sets returns an already discovered pattern, add parameter set to their records. Return to Step 1.

This process runs in parallel for each discovered pattern.

Value

Return a list with the following items:

`ps_partitions`	A data frame containing coordinates from their regions and their corresponding ordinal response pattern output by `fn`. Columns include (in this order): parameter coordinates, their ordinal pattern output by `fn`, the global iteration of the MCMC. Each row corresponds with the evaluation of a single set of parameters.
`ps_patterns`	A table with the ordinal patterns discovered and the population of their corresponding region - the number of parameter sets discovered to produce the ordinal pattern.

References

Pitt, M. A., Kim, W., Navarro, D. J., & Myung, J. I. (2006). Global model analysis by parameter space partitioning. Psychological Review, 113(1), 57.

Weisstein, Eric W. "Hypersphere Point Picking." From MathWorld–A Wolfram Web Resource. https://mathworld.wolfram.com/HyperspherePointPicking.html

See Also

`psp_control`.

Examples

```
library(psp)

#' euclidean distance
#'
#' @param a vector coordinate 1
#' @param b vector coordinate 2
#' @return euclidean distance between coordinates
euclidean <- function(a, b) sqrt(sum((a - b)^2))
```

```
# define center points for the 10 regions in a two-dimensional space
positions <- NULL
for (i in seq_len(2)) positions <- cbind(positions, sample(500, 10))

#' dummy hypercube model to test the PSP function
#' The model takes in a set of coordinates, calculates its distance from all
#' all of available coordinates, then return closest region number.
#' This model generalizes to n-dimensions
#'
#' @param x a vector of coordinates
#' @return The number of the region as character
#' @examples
#' model(runif(5))
model <- function(par) {
    areas <- NULL
    for (i in seq_along(par)) {
        range <- c(1, 0)
        if (i %% 2 == 0) {
            range <- c(0, 1)
        }
        areas <- cbind(areas,
                       seq(range[1], range[2], length.out = 500)[positions[,i]])
    }
    dist <- apply(areas, 1, function(x) euclidean(par, x))
    return(as.character(which.min(dist)))
}

# run Parameter Space Partitioning with some default settings
# Here we run the MCMC for 400 iterations, but the partitioning
# will stop if the population of all regions reach 200.
# Note that we have to load our utility function into
# the clusters, because PSPglobal is currently parallelized.
out <- psp_global(model, psp_control(lower = rep(0, 2),
                                     upper = rep(1, 2),
                                     init = rep(0.5, 2),
                                     radius = rep(0.25, 2),
                                     pop = 100,
                                     parallel = FALSE,
                                     iterations = 100))

print(out)
```

Index