Package 'Rtwalk'

February 2, 2026

Type Package

Title An MCMC Sampler Using the t-Walk Algorithm

Version 2.0.0

Maintainer Rodrigo Fonseca Villa <rodrigo03.villa@gmail.com>

Description Implements the t-walk algorithm, a general-purpose, self-adjusting Markov Chain Monte Carlo (MCMC) sampler for continuous distributions as described by Christen & Fox (2010) <doi:10.1214/10-BA603>. The t-walk requires no tuning and is robust for a wide range of target distributions, including high-dimensional and multimodal problems. This implementation includes an option for running multiple chains in parallel to accelerate sampling and facilitate convergence diagnostics.

License GPL-3

Encoding UTF-8

RoxygenNote 7.3.2

Imports parallel, stats, utils

Suggests mvtnorm, coda, devtools, roxygen2, knitr, rmarkdown, ellipse, ggplot2, ggthemes, gridExtra, reshape2, viridis

VignetteBuilder knitr

URL https://github.com/rodrigosqrt3/Rtwalk

BugReports https://github.com/rodrigosqrt3/Rtwalk/issues

NeedsCompilation no

Author Rodrigo Fonseca Villa [aut, cre]

Repository CRAN

Date/Publication 2026-02-02 10:50:02 UTC

Contents

`twalk` *Run the t-walk MCMC Algorithm*

Description

This function implements the t-walk algorithm by Christen & Fox (2010), a general-purpose MCMC sampler that does not require manual tuning. The function can run multiple independent MCMC chains in parallel to accelerate execution and facilitate convergence diagnostics.

Usage

```
twalk(log_posterior, n_iter, x0, xp0, n_chains = 1, n_cores = NULL, ...)
```

Arguments

`log_posterior`	A function that takes a parameter vector as its first argument and returns the scalar log posterior density. Additional arguments can be passed to this function via '...'.
`n_iter`	The number of iterations to run for each chain.
`x0`	A numeric vector with the initial values for the first point ('x').
`xp0`	A numeric vector with the initial values for the second point ('x'').
`n_chains`	The number of independent MCMC chains to run. Defaults to '1', which runs a single chain sequentially. If greater than 1, parallel mode is activated.
`n_cores`	The number of CPU cores to use in parallel mode. If 'NULL' (default), it will attempt to use all available cores minus one.
`...`	Additional arguments to be passed to the 'log_posterior' function.

Value

A list containing:

`all_samples`	A matrix with the combined samples from all chains.
`acceptance_rate`	The average acceptance rate across all chains.
`total_iterations`	The total number of samples generated (n_iter * n_chains).
`n_dim`	The dimension of the parameter space.
`individual_chains`	If 'n_chains > 1', a list containing the raw results from each separate chain, useful for diagnostics like R-hat.

Examples

```
# Example 1: Sampling from a Bivariate Normal (sequential mode)
# The 'mvtnorm' package is required for this example
if (requireNamespace("mvtnorm", quietly = TRUE)) {
  log_post <- function(x) {
   mvtnorm::dmvnorm(x, mean = c(0, 0), sigma = matrix(c(1, 0.8, 0.8, 1), 2, 2), log = TRUE)
  }

  # Run with fewer iterations for a quick example
  # Set a seed for reproducibility
  set.seed(123)
  result_seq <- twalk(log_posterior = log_post, n_iter = 5000,
                          x0 = c(-1, 1), xp0 = c(1, -1))

  plot(result_seq$all_samples, pch = '.', main = "t-walk Samples (Sequential)")
}


# Example 2: The same problem in parallel (will run faster)
# Using 2 chains. n_iter is now per chain.
if (requireNamespace("mvtnorm", quietly = TRUE)) {
  set.seed(123)
  result_par <- twalk(log_posterior = log_post, n_iter = 2500,
                          x0 = c(-1, 1), xp0 = c(1, -1), n_chains = 2)

  plot(result_par$all_samples, pch = '.', main = "t-walk Samples (Parallel)")
}
```

Index