Package ‘mcmcensemble’

March 20, 2024

Title Ensemble Sampler for Affine-Invariant MCMC

Version 3.1.0

Description Provides ensemble samplers for affine-invariant Monte Carlo Markov Chain, which allow a faster convergence for badly scaled estimation problems. Two samplers are proposed: the 'differential.evolution' sampler from ter Braak and Vrugt (2008) <doi:10.1007/s11222-008-9104-9> and the 'stretch' sampler from Goodman and Weare (2010) <doi:10.2140/camcos.2010.5.65>.

License GPL-2

URL `https://hugogruson.fr/mcmcensemble/`, `https://github.com/Bisaloo/mcmcensemble`

BugReports `https://github.com/Bisaloo/mcmcensemble/issues`

Depends R (>= 3.5)

Imports future.apply, progressr

Suggests bayesplot, coda, mockery, testthat (>= 3.0.0), knitr, rmarkdown

Encoding UTF-8

RoxygenNote 7.3.1

Config/testthat/edition 3

Config/Needs/website r-for-educators/flair, spacefillr

VignetteBuilder knitr

NeedsCompilation no

Author Hugo Gruson [cre, aut, cph] (<`https://orcid.org/0000-0002-4094-1476`>), Sanda Dejanic [aut, cph], Andreas Scheidegger [aut, cph] (<`https://orcid.org/0000-0003-2575-2172`>)

Maintainer Hugo Gruson <`hugo.gruson+R@normalesup.org`>

Repository CRAN

Date/Publication 2024-03-20 16:50:02 UTC

R topics documented:

`MCMCEnsemble` *MCMC ensemble sampler*

Description

Ensemble Markov Chain Monte Carlo sampler with different strategies to generate proposals. Either the *stretch move* as proposed by Goodman and Weare (2010), or a *differential evolution jump move* similar to Braak and Vrugt (2008).

Usage

```
MCMCEnsemble(
  f,
  inits,
  max.iter,
  n.walkers = 10 * ncol(inits),
  method = c("stretch", "differential.evolution"),
  coda = FALSE,
  ...
)
```

Arguments

`f`	function that returns a single scalar value proportional to the log probability density to sample from.
`inits`	A matrix (or data.frame) containing the starting values for the walkers. Each column is a variable to estimate and each row is a walker
`max.iter`	maximum number of function evaluations
`n.walkers`	number of walkers (ensemble size). An integer greater or equal than 2.
`method`	method for proposal generation, either `"stretch"`, or `"differential.evolution"`. This argument will be saved as an attribute in the output (see examples).
`coda`	logical. Should the samples be returned as coda::mcmc.list object? (defaults to `FALSE`)
`...`	further arguments passed to `f`

Value

- if `coda = FALSE` a list with:
  - *samples*: A three dimensional array of samples with dimensions `walker` x `generation` x `parameter`

    - *log.p*: A matrix with the log density evaluate for each walker at each generation.

- if coda = TRUE a list with:
    - *samples*: A object of class coda::mcmc.list containing all samples.
    - *log.p*: A matrix with the log density evaluate for each walker at each generation.

In both cases, there is an additional attribute (accessible via `attr(res, "ensemble.sampler")`) recording which ensemble sampling algorithm was used.

References

- ter Braak, C. J. F. and Vrugt, J. A. (2008) Differential Evolution Markov Chain with snooker updater and fewer chains. Statistics and Computing, 18(4), 435–446, doi:10.1007/s11222008-91049
- Goodman, J. and Weare, J. (2010) Ensemble samplers with affine invariance. Communications in Applied Mathematics and Computational Science, 5(1), 65–80, doi:10.2140/camcos.2010.5.65

Examples

```
## a log-pdf to sample from
p.log <- function(x) {
    B <- 0.03                              # controls 'bananacity'
    -x[1]^2/200 - 1/2*(x[2]+B*x[1]^2-100*B)^2
}

## set options and starting point
n_walkers <- 10
unif_inits <- data.frame(
  "a" = runif(n_walkers, 0, 1),
  "b" = runif(n_walkers, 0, 1)
)


## use stretch move
res1 <- MCMCEnsemble(p.log, inits = unif_inits,
                     max.iter = 300, n.walkers = n_walkers,
                     method = "stretch")

attr(res1, "ensemble.sampler")

str(res1)


## use stretch move, return samples as 'coda' object
res2 <- MCMCEnsemble(p.log, inits = unif_inits,
                     max.iter = 300, n.walkers = n_walkers,
                     method = "stretch", coda = TRUE)

attr(res2, "ensemble.sampler")

summary(res2$samples)
plot(res2$samples)
```

```
## use different evolution move, return samples as 'coda' object
res3 <- MCMCEnsemble(p.log, inits = unif_inits,
                     max.iter = 300, n.walkers = n_walkers,
                     method = "differential.evolution", coda = TRUE)

attr(res3, "ensemble.sampler")

summary(res3$samples)
plot(res3$samples)
```

Index