Package ‘LangevinFlow’

May 29, 2026

Type Package

Title Langevin Diffusion Samplers with a C++ Backend

Version 0.1.0

Description Provides lightweight, dependency-minimal implementations of Langevin diffusion based Markov chain Monte Carlo samplers, including the Unadjusted Langevin Algorithm (ULA) and the Metropolis-Adjusted Langevin Algorithm (MALA). The core sampling loops are written in C++ via 'Rcpp' and 'RcppArmadillo' for performance, while exposing a simple R-level interface where the user supplies the gradient of the negative log-density (and, for MALA, the negative log-density itself). Intended as a building block for Bayesian inference and stochastic optimization rather than a full probabilistic programming framework. Methods follow Roberts and Tweedie (1996) <doi:10.2307/3318418> and Roberts and Rosenthal (1998) <doi:10.1111/1467-9868.00123>.

License MIT + file LICENSE

Encoding UTF-8

Depends R (>= 3.5.0)

Imports Rcpp (>= 1.0.0), stats, graphics

LinkingTo Rcpp, RcppArmadillo

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

VignetteBuilder knitr

RoxygenNote 7.3.3

URL https://github.com/BehroozMoosavi/LangevinFlow

BugReports https://github.com/BehroozMoosavi/LangevinFlow/issues

Config/testthat/edition 3

NeedsCompilation yes

Author Behrooz Moosavi [aut, cre] (ORCID: <https://orcid.org/0000-0003-4378-3167>)

Maintainer Behrooz Moosavi <bem159@pitt.edu>

Repository CRAN

Date/Publication 2026-05-29 12:30:03 UTC

Contents

mala 2
plot.langevin_chain 3
summary.langevin_chain 3
ula 4

Index **6**

`mala` *Metropolis-Adjusted Langevin Algorithm*

Description

Asymptotically exact sampler for $\pi(x) \propto \exp(-\beta U(x))$. At each step a Langevin proposal is generated as in `ula` and then accepted or rejected by the Metropolis-Hastings rule so that π is exactly invariant. Optimal acceptance rates are typically near 0.574 in high dimension (Roberts & Rosenthal, 1998).

Usage

```
mala(init_x, U, grad_u, step_size, n_iter, beta = 1, burn_in = 0L)
```

Arguments

`init_x`	Numeric vector. Starting state of the chain.
`U`	Function. Takes a numeric vector of length `length(init_x)` and returns the scalar potential $U(x) = -\log \pi(x)$ (additive constants do not matter).
`grad_u`	Function. Returns the gradient of `U`; same shape convention as for `ula`.
`step_size`	Positive numeric scalar. Discretization step γ.
`n_iter`	Positive integer. Number of iterations.
`beta`	Positive numeric scalar. Inverse temperature; defaults to 1.
`burn_in`	Non-negative integer, strictly less than `n_iter`. Defaults to 0.

Value

An object of class `"langevin_chain"` (see `ula` for structure). The `acceptance_rate` component is populated and `accepted` is a logical vector indicating per-iteration outcomes (post-burn-in).

References

Roberts, G. O., & Rosenthal, J. S. (1998). Optimal scaling of discrete approximations to Langevin diffusions. *Journal of the Royal Statistical Society, Series B*, 60(1), 255-268.

See Also

`ula`

Examples

```
# Standard 2D Gaussian: U(x) = 0.5 * ||x||^2.
set.seed(1)
U      <- function(x) 0.5 * sum(x^2)
grad_u <- function(x) x
fit <- mala(init_x = c(3, -3), U = U, grad_u = grad_u,
            step_size = 0.3, n_iter = 2000, burn_in = 500)
summary(fit)
```

---

plot.langevin_chain *Plot a Langevin Chain*

Description

Trace plot for each dimension, plus a marginal histogram. For high-dimensional chains, only the first `max_dim` coordinates are shown.

Usage

```
## S3 method for class 'langevin_chain'
plot(x, max_dim = 4L, ...)
```

Arguments

`x`	An object of class "`langevin_chain`".
`max_dim`	Maximum number of coordinates to display. Defaults to 4.
`...`	Passed to `plot`.

Value

Called for side effect. Returns `x` invisibly.

---

summary.langevin_chain *Summarize a Langevin Chain*

Description

Computes per-coordinate posterior summaries (mean, standard deviation, and selected quantiles) from a fitted Langevin chain.

Usage

```
## S3 method for class 'langevin_chain'
summary(object, probs = c(0.025, 0.5, 0.975), ...)
```

Arguments

`object`	An object of class "langevin_chain".
`probs`	Numeric vector of probabilities for quantile computation.
`...`	Currently unused.

Value

Invisibly, a data frame of summaries. Printed by default.

`ula` *Unadjusted Langevin Algorithm*

Description

Draws a Markov chain whose stationary distribution approximates $\pi(x) \propto \exp(-\beta U(x))$ using the Euler-Maruyama discretization of the overdamped Langevin diffusion. The discretization introduces a bias that vanishes as `step_size` tends to zero; for an asymptotically exact sampler, use `mala`.

Usage

```
ula(init_x, grad_u, step_size, n_iter, beta = 1, burn_in = 0L)
```

Arguments

`init_x`	Numeric vector. Starting state of the chain. Its length defines the dimension.
`grad_u`	Function. Takes a numeric vector of length `length(init_x)` and returns the gradient of $U(x) = -\log \pi(x)$, as a numeric vector of the same length. See the sign convention in `?LangevinFlow`.
`step_size`	Positive numeric scalar. Discretization step γ.
`n_iter`	Positive integer. Number of iterations to run.
`beta`	Positive numeric scalar. Inverse temperature; defaults to 1.
`burn_in`	Non-negative integer. Number of initial samples to discard. Must be strictly less than `n_iter`. Defaults to 0.

Value

An object of class "langevin_chain", which is a list with components:

samples Numeric matrix of post-burn-in samples; rows index iterations and columns index dimensions.

algorithm Character string "ULA".

step_size, beta, n_iter, burn_in, dimension Echoed inputs.

acceptance_rate `NA` for ULA (no accept/reject step).

elapsed_secs Wall-clock runtime of the sampler in seconds.

References

Roberts, G. O., & Tweedie, R. L. (1996). Exponential convergence of Langevin distributions and their discrete approximations. *Bernoulli*, 2(4), 341-363.

See Also

mala

Examples

```
# Standard 2D Gaussian target: U(x) = 0.5 * ||x||^2, grad_U(x) = x.
set.seed(1)
grad_u <- function(x) x
fit <- ula(init_x = c(3, -3), grad_u = grad_u,
           step_size = 0.05, n_iter = 2000, burn_in = 500)
summary(fit)
```

Index

`mala`, 2, *4*, *5*

`plot`, *3*
`plot.langevin_chain`, 3

`summary.langevin_chain`, 3

`ula`, *2*, 4