Package 'madgrad'

April 29, 2026

Title 'MADGRAD' Method for Stochastic Optimization

Version 0.2.0

Description A Momentumized, Adaptive, Dual Averaged Gradient Method for Stochastic Optimization algorithm. MADGRAD is a 'best-of-both-worlds' optimizer with the generalization performance of stochastic gradient descent and at least as fast convergence as that of Adam, often faster. A drop-in optim_madgrad() implementation is provided based on Defazio et al (2020) <doi:10.48550/arXiv.2101.11075>.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

Imports torch (>= 0.3.0), rlang

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

NeedsCompilation no

Author Daniel Falbel [aut, cre, cph],
Posit Software, PBC [cph],
MADGRAD original implementation authors. [cph]

Maintainer Daniel Falbel <dfalbel@gmail.com>

Repository CRAN

Date/Publication 2026-04-29 09:10:02 UTC

Contents

`optim_madgrad` *A Momentumized, Adaptive, Dual Averaged Gradient Method for Stochastic Optimization.*

Description

MADGRAD is a general purpose optimizer that can be used in place of SGD or Adam may converge faster and generalize better. Currently GPU-only. Typically, the same learning rate schedule that is used for SGD or Adam may be used. The overall learning rate is not comparable to either method and should be determined by a hyper-parameter sweep.

Usage

```
optim_madgrad(params, lr = 0.01, momentum = 0.9, weight_decay = 0, eps = 1e-06)
```

Arguments

`params`	(list): List of parameters to optimize.
`lr`	(float): Learning rate (default: 1e-2).
`momentum`	(float): Momentum value in the range [0,1) (default: 0.9).
`weight_decay`	(float): Weight decay, i.e. a L2 penalty (default: 0).
`eps`	(float): Term added to the denominator outside of the root operation to improve numerical stability. (default: 1e-6).

Details

MADGRAD requires less weight decay than other methods, often as little as zero. Momentum values used for SGD or Adam's beta1 should work here also.

On sparse problems both weight_decay and momentum should be set to 0. (not yet supported in the R implementation).

Value

An optimizer object implementing the `step` and `zero_grad` methods.

Examples

```
if (torch::torch_is_installed()) {
library(torch)
x <- torch_randn(1, requires_grad = TRUE)
opt <- optim_madgrad(x)
for (i in 1:100) {
  opt$zero_grad()
  y <- x^2
  y$backward()
  opt$step()
}
```

```
    all.equal(x$item(), 0, tolerance = 1e-9)
    }
```

Index