R: Performing Gradient Descent

Asked 1 year, 6 months ago Modified today Viewed 714 times Part of R Language Collective

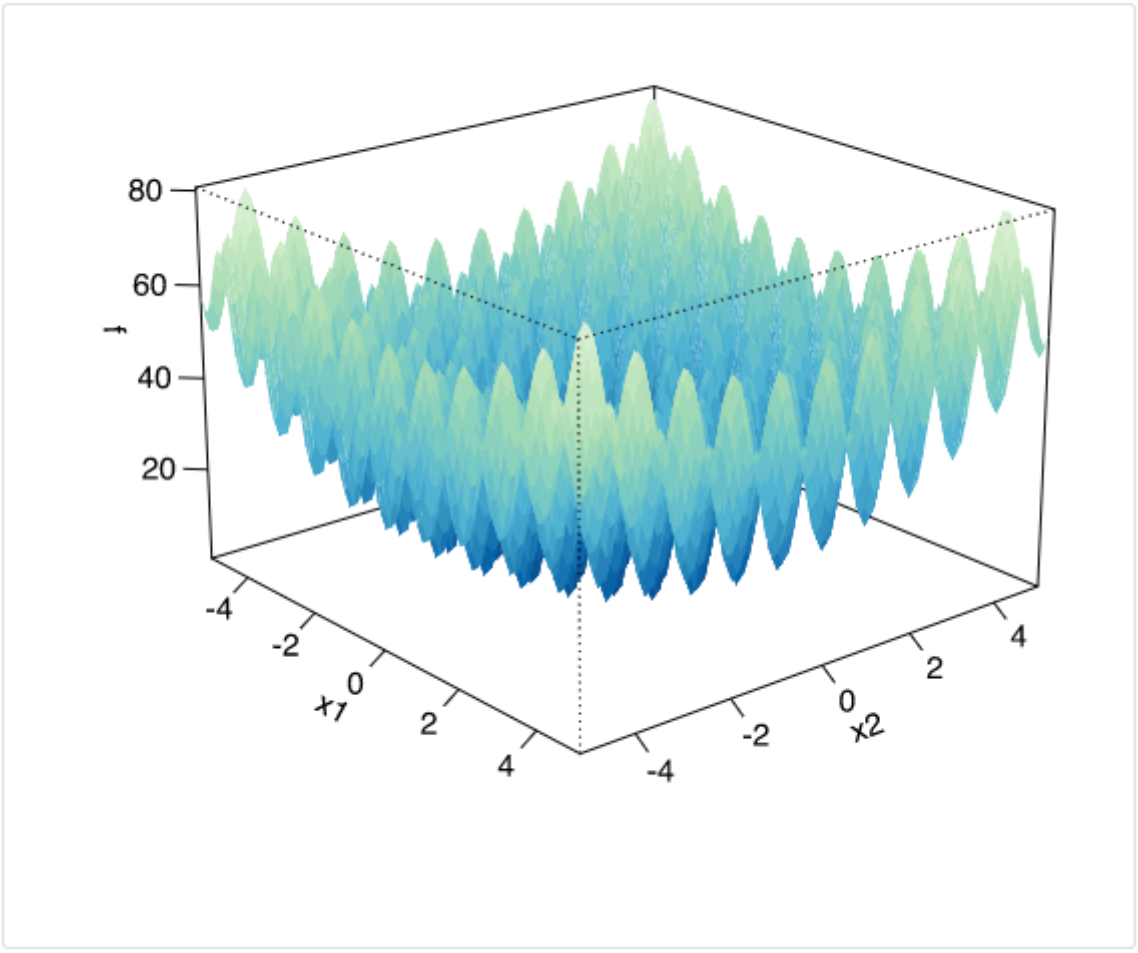
1

I am working with the R programming language.

I am trying to learn more about optimization algorithms, and as a learning exercise - I would like to try an **optimize a mathematical function using the (famous) gradient descent algorithm using the R programming language.**

For instance, I would like to try and "optimize" (i.e. find out the values of "x1 and x2" that produce the smallest possible value of "y") the following function (this function is called the *Rastrign Function*, and is a popular function to test optimization algorithms on due to its irregular and complicated shape):

$$f(x_1, x_2) = 20 + x_1^2 + x_2^2 - 10(\cos(2\pi x_1) + \cos(2\pi x_2)),$$



I first defined this function in R:

```
Rastrigin <- function(x)
 {
     return(20 + x[1]^2 + x[2]^2 - 10*(cos(2*pi*x[1]) + cos(2*pi*x[2])))
 }
```

Then, I tried to do some research and see if there are any standard and common implementations of gradient descent in R. For example, I found out about the "optim()" function in (base) R, which provides many choices of popular optimization algorithms such as "BFGS", "Simulated Annealing" and "Nelder-Meade". For instance, below I used a variant of the "BFGS" algorithm to optimize the Rastrign Function:

```
#run BFGS optimization algorithm:

optim(par = c(2,2), Rastrigin, lower = c(-5,-5), upper = c(5,5), method = "L-BFGS-B")


$par
[1] 5.453531e-15 5.453531e-15

$value
[1] 0

$counts
function gradient
       7        7

$convergence
[1] 0

$message
[1] "CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH"
```

Based on the results of the above code, it seems like the BFGS algorithm was able to successfully find the minimum of this function by returning values of x1 and x2 that are very close to the true minimum (using trigonometry, we can see that if "x1 = x2 = 0", `f(x1,x2) = 20 + 0 + 0 - 10*(cos(0) + cos(0)) = 20 - 10*2 = 20 - 20 = 0`).

My Question: I tried looking for a standard function in R that would allow you to perform gradient descent optimization, but I could not find anything.

Does anyone know if there are any standard functions in R for gradient descent optimization? Can someone please show me how to do this?

Thanks!

References:

- https://stat.ethz.ch/R-manual/R-devel/library/stats/html/optim.html

r algorithm optimization Edit tags

Share Edit Follow Close Flag

asked Jan 19, 2022 at 17:11

stats_noob
5,467 3 27 80

Use google and check out cran.r-project.org/web/views/Optimization.html Note that package recommendations are regarded as opinion baesd and so off topic for SO. – G. Grothendieck Jan 19,

2022 at 18:38

Thank you for your reply! My question is not so much of a package recommendation - but rather, how can this be done in R? I looked at the link you posted and there dosent seem to be an exact gradient descent algorithm... unless it goes by a different name on the list? Thank you so much! – stats_noob Jan 19, 2022 at 19:45

Steepest descent is a special case of gradient descent so you can also look for that. Also search SO's R tag. – G. Grothendieck Jan 19, 2022 at 19:58

2 Answers

Sorted by:
Reset to default

Date modified (newest first)

0

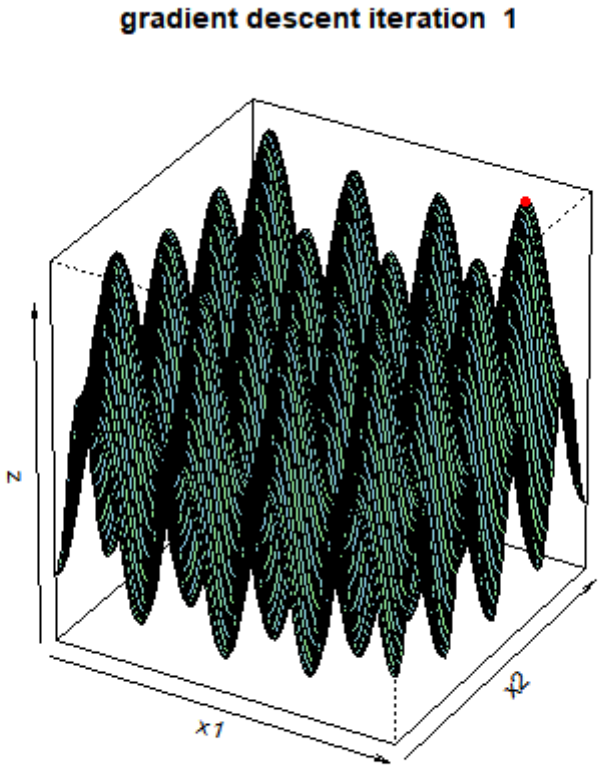
You can implement gradient descent too with a couple of lines of code as below, only thing is that the right learning rate needs to be found.

```
f <- function(x) {
  x1 <- x[1]
  x2 <- x[2]
  return (20 + x1**2 + x2**2 - 10*(cos(2*pi*x1)+cos(2*pi*x2)))
}

grad_f <- function(x) { # could be computed with numerical approximation too
  x1 <- x[1]
  x2 <- x[2]
  return (c(2*x1 + 10*(2*pi*sin(2*pi*x1)), 2*x2 + 10*(2*pi*sin(2*pi*x2))))
}

grad_descent <- function(f, grad_f, x, lr, niter) {
  for (i in 1:niter) {
    x <- x - lr*grad_f(x)
  }
  return(x)
}

x <- rep(1.5, 2)
grad_descent(f, grad_f, x, lr=0.001, niter=25)
# [1] 0.9952904 0.9952904
```



The next animation shows how GD converges to (local) minimum of the function.

Share Edit Delete Flag

answered just now

Sandipan Dey
21.3k 2 49 62

0

As indicated in the comments, (I just learned that) "gradient descent" is the same as "steepest descent":

```
library(pracma)

> steep_descent(c(1, 1), Rastrigin)

$xmin
[1] 0.9949586 0.9949586

$fmin
[1] 1.989918

$niter
[1] 3
```

Share Edit Follow Flag

answered Jan 20, 2022 at 17:13

S

stats_noob
5,467 3 27 80
