**[小默的博客](http://xmuxiaomo.github.io/)**

Top of Form



Bottom of Form

* [首页](http://xmuxiaomo.github.io/)
* [分类](http://xmuxiaomo.github.io/categories)
* [归档](http://xmuxiaomo.github.io/archives)
* [标签](http://xmuxiaomo.github.io/tags)
* [关于](http://xmuxiaomo.github.io/about)

R Programming Assignment 2: Lexical Scoping

发表于 2015-06-14   |   分类于 [技术分享](http://xmuxiaomo.github.io/categories/技术分享/)   |

**Introduction**

This second programming assignment will require you to write an R function is able to cache potentially time-consuming computations. For example, taking the mean of a numeric vector is typically a fast operation. However, for a very long vector, it may take too long to compute the mean, especially if it has to be computed repeatedly (e.g. in a loop). If the contents of a vector are not changing, it may make sense to cache the value of the mean so that when we need it again, it can be looked up in the cache rather than recomputed. In this Programming Assignment will take advantage of the scoping rules of the R language and how they can be manipulated to preserve state inside of an R object.

**Example: Caching the Mean of a Vector**

In this example we introduce the <<- operator which can be used to assign a value to an object in an environment that is different from the current environment. Below are two functions that are used to create a special object that stores a numeric vector and caches its mean.

The first function, makeVector creates a special “vector”, which is really a list containing a function to

1. set the value of the vector
2. get the value of the vector
3. set the value of the mean
4. get the value of the mean

|  |
| --- |
| makeVector <- function(x = numeric()) {  m <- NULL  set <- function(y) {  x <<- y  m <<- NULL  }  get <- function() x  setmean <- function(mean) m <<- mean  getmean <- function() m  list(set = set, get = get,  setmean = setmean,  getmean = getmean) } |

The following function calculates the mean of the special “vector” created with the above function. However, it first checks to see if the mean has already been calculated. If so, it gets the mean from the cache and skips the computation. Otherwise, it calculates the mean of the data and sets the value of the mean in the cache via the setmean function.

|  |
| --- |
| cachemean <- function(x, ...) {  m <- x$getmean()  if(!is.null(m)) {  message("getting cached data")  return(m)  }  data <- x$get()  m <- mean(data, ...)  x$setmean(m)  m } |

**Assignment: Caching the Inverse of a Matrix**

Matrix inversion is usually a costly computation and there may be some benefit to caching the inverse of a matrix rather than compute it repeatedly (there are also alternatives to matrix inversion that we will not discuss here). Your assignment is to write a pair of functions that cache the inverse of a matrix.

Write the following functions:

1. makeCacheMatrix: This function creates a special “matrix” object that can cache its inverse.
2. cacheSolve: This function computes the inverse of the special “matrix” returned by makeCacheMatrix above. If the inverse has already been calculated (and the matrix has not changed), then the cacheSolve should retrieve the inverse from the cache.

Computing the inverse of a square matrix can be done with the solve function in R. For example, if X is a square invertible matrix, then solve(X) returns its inverse.

For this assignment, assume that the matrix supplied is always invertible.

**My Solution**

**Functions**

cachematrix.R:

|  |
| --- |
| ## Caching the Inverse of a Matrix: ## Matrix inversion is usually a costly computation and there may be some  ## benefit to caching the inverse of a matrix rather than compute it repeatedly. ## Below are a pair of functions that are used to create a special object that  ## stores a matrix and caches its inverse.  ## This function creates a special "matrix" object that can cache its inverse.  makeCacheMatrix <- function(x = matrix()) {  inv <- NULL  set <- function(y) {  x <<- y  inv <<- NULL  }  get <- function() x  setInverse <- function(inverse) inv <<- inverse  getInverse <- function() inv  list(set = set,  get = get,  setInverse = setInverse,  getInverse = getInverse) }   ## This function computes the inverse of the special "matrix" created by  ## makeCacheMatrix above. If the inverse has already been calculated (and the  ## matrix has not changed), then it should retrieve the inverse from the cache.  cacheSolve <- function(x, ...) {  ## Return a matrix that is the inverse of 'x'  inv <- x$getInverse()  if (!is.null(inv)) {  message("getting cached data")  return(inv)  }  mat <- x$get()  inv <- solve(mat, ...)  x$setInverse(inv)  inv } |

**Testing My Functions**

|  |
| --- |
| > source("ProgrammingAssignment2/cachematrix.R") > my\_matrix <- makeCacheMatrix(matrix(1:4, 2, 2)) > my\_matrix$get()  [,1] [,2] [1,] 1 3 [2,] 2 4 > my\_matrix$getInverse() NULL > cacheSolve(my\_matrix)  [,1] [,2] [1,] -2 1.5 [2,] 1 -0.5 > cacheSolve(my\_matrix) getting cached data  [,1] [,2] [1,] -2 1.5 [2,] 1 -0.5 > my\_matrix$getInverse()  [,1] [,2] [1,] -2 1.5 [2,] 1 -0.5 > my\_matrix$set(matrix(c(2, 2, 1, 4), 2, 2)) > my\_matrix$get()  [,1] [,2] [1,] 2 1 [2,] 2 4 > my\_matrix$getInverse() NULL > cacheSolve(my\_matrix)  [,1] [,2] [1,] 0.6666667 -0.1666667 [2,] -0.3333333 0.3333333 > cacheSolve(my\_matrix) getting cached data  [,1] [,2] [1,] 0.6666667 -0.1666667 [2,] -0.3333333 0.3333333 > my\_matrix$getInverse()  [,1] [,2] [1,] 0.6666667 -0.1666667 [2,] -0.3333333 0.3333333 |

**Another Useful Example**

A good example for getting a better understanding of differences amount **formal parameters, local variables and free variables**. And it is also helpful for learning the **<<-** operator.

open.account.R:

|  |
| --- |
| open.account <- function(total) {  list(  deposit = function(amount) {  if(amount <= 0)  stop("Deposits must be positive!\n")  total <<- total + amount  cat(amount, "deposited. Your balance is", total, "\n\n")  },  withdraw = function(amount) {  if(amount > total)  stop("You don't have that much money!\n")  total <<- total - amount  cat(amount, "withdrawn. Your balance is", total, "\n\n")  },  balance = function() {  cat("Your balance is", total, "\n\n")  }  ) } |

Simple tests for open.account function:

|  |
| --- |
| > peter <- open.account(100) > mm <- open.account(200) > peter$withdraw(30) 30 withdrawn. Your balance is 70   > peter$balance() Your balance is 70  > mm$balance() Your balance is 200   > peter$deposit(50) 50 deposited. Your balance is 120   > peter$balance() Your balance is 120   > peter$withdraw(500) Error in peter$withdraw(500) : You don't have that much money! > mm$balance() Your balance is 200  > mm$withdraw(12) 12 withdrawn. Your balance is 188   > mm$balance() Your balance is 188   > peter$balance() Your balance is 120 |

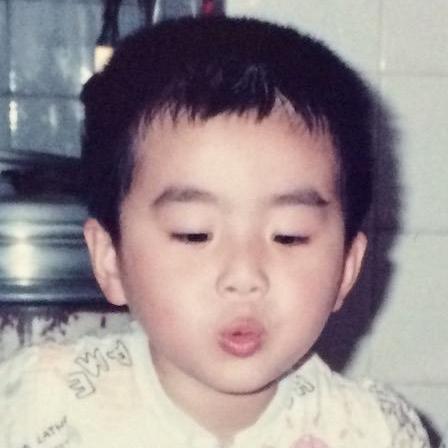
More information about [Scope](http://cran.r-project.org/doc/manuals/R-intro.html#DOCF23).

[#R](http://xmuxiaomo.github.io/tags/R/)

[烈日下的沉醉](http://xmuxiaomo.github.io/2015/06/15/烈日下的沉醉/)

[听俞敏洪讲座的思与悟](http://xmuxiaomo.github.io/2015/06/13/听俞敏洪讲座的思与悟/)

* 文章目录
* 站点概览



xmuxiaomo

匆忙世间的闲人。

[244日志](http://xmuxiaomo.github.io/archives)

[11分类](http://xmuxiaomo.github.io/categories)

[52标签](http://xmuxiaomo.github.io/tags)

[RSS](http://xmuxiaomo.github.io/atom.xml)

[github](https://github.com/xmuxiaomo) [weibo](http://weibo.com/u/2457084784) [1. Introduction](http://xmuxiaomo.github.io/2015/06/14/R-Programming-Assignment-2/#Introduction)

1. [2. Example: Caching the Mean of a Vector](http://xmuxiaomo.github.io/2015/06/14/R-Programming-Assignment-2/#Example:_Caching_the_Mean_of_a_Vector)
2. [3. Assignment: Caching the Inverse of a Matrix](http://xmuxiaomo.github.io/2015/06/14/R-Programming-Assignment-2/#Assignment:_Caching_the_Inverse_of_a_Matrix)
3. [4. My Solution](http://xmuxiaomo.github.io/2015/06/14/R-Programming-Assignment-2/#My_Solution)
   1. [4.1. Functions](http://xmuxiaomo.github.io/2015/06/14/R-Programming-Assignment-2/#Functions)
   2. [4.2. Testing My Functions](http://xmuxiaomo.github.io/2015/06/14/R-Programming-Assignment-2/#Testing_My_Functions)

Top of Form





Bottom of Form

This site is still being indexed. Please try your search again in a few minutes.

[search by swiftype](http://swiftype.com/?ref=pbo)