

# Package ‘rbenchmark’

February 15, 2012

**Type** Package

**Title** Benchmarking routine for R

**Version** 0.3

**Date** 2010-03-05

**Author** Wacek Kusnierczyk <waku@idi.ntnu.no>

**Maintainer** Wacek Kusnierczyk <waku@idi.ntnu.no>

**Description** rbenchmark is inspired by the Perl module Benchmark, and is intended to facilitate benchmarking of arbitrary R code. The library consists of just one function, benchmark, which is a simple wrapper around system.time. Given a specification of the benchmarking process (counts of replications, evaluation environment) and an arbitrary number of expressions, benchmark evaluates each of the expressions in the specified environment, replicating the evaluation as many times as specified, and returning the results conveniently wrapped into a data frame.

**License** GPL (>= 2)

**LazyLoad** yes

**URL** <http://rbenchmark.googlecode.com>

**Repository** CRAN

**Date/Publication** 2010-03-07 17:10:02

## R topics documented:

benchmark . . . . .	2
rbenchmark . . . . .	5

<b>Index</b>	<b>6</b>
--------------	----------

---

benchmark                      *a simple routine for benchmarking R code*

---

### Description

benchmark is a simple wrapper around `system.time`.

Given a specification of the benchmarking process (counts of replications, evaluation environment) and an arbitrary number of expressions, benchmark evaluates each of the expressions in the specified environment, replicating the evaluation as many times as specified, and returning the results conveniently wrapped into a data frame.

### Usage

```
benchmark(
  ...,
  columns = c(
    "test", "replications", "elapsed", "relative", "user.self", "sys.self",
    "user.child", "sys.child"),
  order = "test",
  replications = 100,
  environment = parent.frame())
```

### Arguments

...	captures any number of unevaluated expressions passed to benchmark as named or unnamed arguments.
columns	a character or integer vector specifying which columns should be included in the returned data frame (see below).
order	a character or integer vector specifying which columns should be used to sort the output data frame. Any of the columns that can be specified for columns (see above) can be used, even if it is not included in columns and will not appear in the output data frame. If order=NULL, the benchmarks will appear sorted by the order of the expressions in the call to benchmark.
replications	a numeric vector specifying how many times an expression should be evaluated when the runtime is measured. If replications consists of more than one value, each expression will be benchmarked multiple times, once for each value in replications.
environment	the environment in which the expressions will be evaluated.

### Details

The parameters `replications`, `environment`, `columns`, and `order` are optional and have the following default values:

- `replications = 100`  
By default, each expression will be benchmarked once, and will be evaluated 100 times within the benchmark.

- `environment = parent.frame()`  
By default, all expressions will be evaluated in the environment in which the call to `benchmark` is made.
- `columns = c('test', 'replications', 'elapsed', 'relative', 'user.self', 'sys.self', 'user.child', 'sys.child')`  
By default, the returned data frame will contain all columns generated internally in `benchmark`. These named columns will contain the following data:
  - `test`: a character string naming each individual benchmark. If the corresponding expression was passed to `benchmark` in a named argument, the name will be used; otherwise, the expression itself converted to a character string will be used.
  - `replications`: a numeric vector specifying the number of replications used within each individual benchmark.
  - `elapsed`, `user.self`, `sys.self`, `user.child`, and `sys.child` are columns containing values reported by `system.time`; see Sec. 7.1 Operating system access in The R language definition, or see [system.time](#).
  - `relative`: a column containing benchmark values relative to the shortest benchmark value.
- `order = 'test'`  
By default, the data frame is sorted by the column `test` (the labels of the expressions or the expressions themselves; see above).

## Value

The value returned from a call to `benchmark` is a data frame with rows corresponding to individual benchmarks, and columns as specified above.

An individual benchmark corresponds to a unique combination (see below) of an expression from `...` and a replication count from `replications`; if there are  $n$  expressions in `...` and  $m$  replication counts in `replication`, the returned data frame will consist of  $n*m$  rows, each corresponding to an individual, independent (see below) benchmark.

If either `...` or `replications` contain duplicates, the returned data frame will contain multiple benchmarks for the involved expression-replication combinations. Note that such multiple benchmarks for a particular expression-replication pair will, in general, have different timing results, since they will be evaluated independently (unless the expressions perform side effects that can influence each other's performance).

## Note

Not all expressions, if passed as unnamed arguments, will be cast to character strings as you might expect:

```
benchmark({x = 5; 1:x^x})
# the benchmark will be named '{'
```

`benchmark` performs no smart argument-parameter matching. Any named argument whose name is not exactly `'replications'`, `'environment'`, `'columns'`, or `'order'` will be treated as an expression to be benchmarked:

```
benchmark(1:10^5, repl=1000)
# there will be a benchmark named 'repl'
```

See <<http://code.google.com/p/rbenchmark>> for more details.

### Author(s)

Wacek Kusnierczyk <<mailto:waku@idi.ntnu.no>>

### Examples

```
# example 1
# benchmark the allocation of one 10^6-element numeric vector,
# replicated 100 times
benchmark(1:10^6)

# Example 2
# A call to benchmark with two named expressions and three replication
# counts, output sorted by the replication counts and then by the elapsed time:
means.rep = function(n, m)
  mean(replicate(n, rnorm(m)))
means.pat = function(n, m)
  colMeans(array(rnorm(n*m), c(m, n)))
benchmark(
  rep=means.rep(100, 100),
  pat=means.pat(100, 100),
  replications=10^(1:3),
  order=c('replications', 'elapsed'))

# Example 3
# A call to benchmark with duplicate expressions and replication counts,
# output with selected columns, additional column computed afterwards:
means.pat = function(n, m)
  colMeans(array(rnorm(n*m), c(m, n)))
within(
  benchmark(
    replications=rep(100, 3),
    means.pat(100, 100),
    means.pat(100, 100),
    columns=c('test', 'elapsed', 'replications')),
  {average=elapsed/replications})

# Example 4
# A call to benchmark with a list of arbitrary predefined expressions:
means.rep = function(n, m)
  mean(replicate(n, rnorm(m)))
means.pat = function(n, m)
  colMeans(array(rnorm(n*m), c(m, n)))
tests = list(
  rep=expression(means.rep(100, 100)),
  pat=expression(means.pat(100, 100)))
do.call(benchmark,
```

```
c(tests,
  list(replications=100,
        columns=c('test', 'elapsed', 'replications'),
        order='elapsed'))
```

---

rbenchmark

*rbenchmark provides a simple routine for benchmarking R code.*

---

## Description

rbenchmark is inspired by the Perl module Benchmark, and is intended to facilitate benchmarking of arbitrary R code.

The library consists of just one function, benchmark, which is a simple wrapper around system.time.

Given a specification of the benchmarking process (counts of replications, evaluation environment) and an arbitrary number of expressions, benchmark evaluates each of the expressions in the specified environment, replicating the evaluation as many times as specified, and returning the results conveniently wrapped into a data frame.

## Details

Package:	benchmark
Type:	Package
Version:	0.3
Date:	2010-03-05
License: GPL-2 LazyLoad:	yes

## Author(s)

Wacek Kusnierczyk

Maintainer: Wacek Kusnierczyk <waku@idi.ntnu.no>

## Examples

```
library(rbenchmark)
benchmark(1:10^8, replications=5)
```

# Index

\*Topic **package**  
rbenchmark, 5

benchmark, 2

rbenchmark, 5

system.time, 3