R_and_C_tester

Luke Yee

3/20/2020

```
library(Rcpp)
```

```
## Warning: package 'Rcpp' was built under R version 3.6.2
```

```
library(bench)
```

```
## Warning: package 'bench' was built under R version 3.6.2
```

```
sourceCpp("c_script.cpp")



testdata1 <- rnorm(1e5,50,10)

bench::mark(sample(testdata1, 100))
```

```
## # A tibble: 1 x 6
##   expression                  min   median `itr/sec` mem_alloc `gc/sec`
##   <bch:expr>             <bch:tm> <bch:tm>     <dbl> <bch:byt>    <dbl>
## 1 sample(testdata1, 100)    103us    138us     6273.     411KB     43.8
```

```
bench::mark(sammy(testdata1, 100))
```

```
## # A tibble: 1 x 6
##   expression                min   median `itr/sec` mem_alloc `gc/sec`
##   <bch:expr>           <bch:tm> <bch:tm>     <dbl> <bch:byt>    <dbl>
## 1 sammy(testdata1, 100)   17.1us   22.7us    27946.    44.8KB     19.6
```

```
print("Rcpp version uses less memory and is faster")
```

```
## [1] "Rcpp version uses less memory and is faster"
```

```
bench::mark(sample(testdata1, 1000))
```

```
## # A tibble: 1 x 6
##   expression                   min   median `itr/sec` mem_alloc `gc/sec`
##   <bch:expr>              <bch:tm> <bch:tm>     <dbl> <bch:byt>    <dbl>
## 1 sample(testdata1, 1000)    146us    220us     3667.     405KB     28.0
```

```
bench::mark(sammy(testdata1, 1000))
```

```
## # A tibble: 1 x 6
##   expression                   min   median `itr/sec` mem_alloc `gc/sec`
##   <bch:expr>              <bch:tm> <bch:tm>     <dbl> <bch:byt>    <dbl>
## 1 sammy(testdata1, 1000)    770us    865us     1036.    3.87MB     82.5
```

```
print("For larger sample sizes, the standard R version starts to become faster and more memory efficien
```

```
## [1] "For larger sample sizes, the standard R version starts to become faster and more memory efficie
```

```
bench::mark(rnorm(1000, 100, 10))
```

```
## # A tibble: 1 x 6
```

```
##   expression                min   median `itr/sec` mem_alloc `gc/sec`
##   <bch:expr>           <bch:tm> <bch:tm>     <dbl> <bch:byt>    <dbl>
## 1 rnorm(1000, 100, 10)   62.5us   82.1us    10691.    10.4KB     2.01
```

```
bench::mark(cnorm(1000, 100, 10))
```

```
## # A tibble: 1 x 6
##   expression                min   median `itr/sec` mem_alloc `gc/sec`
##   <bch:expr>           <bch:tm> <bch:tm>     <dbl> <bch:byt>    <dbl>
## 1 cnorm(1000, 100, 10)    749us    820us     1102.    3.87MB     83.9
```

```
print("cnorm appears to be faster")
```

```
## [1] "cnorm appears to be faster"
```

```
bench::mark(rnorm(4000, 100, 10))
```

```
## # A tibble: 1 x 6
##   expression                min   median `itr/sec` mem_alloc `gc/sec`
##   <bch:expr>           <bch:tm> <bch:tm>     <dbl> <bch:byt>    <dbl>
## 1 rnorm(4000, 100, 10)    252us    260us     3231.    33.8KB     2.01
```

```
bench::mark(cnorm(4000, 100, 10))
```

```
## Warning: Some expressions had a GC in every iteration; so filtering is
## disabled.
```

```
## # A tibble: 1 x 6
##   expression                min   median `itr/sec` mem_alloc `gc/sec`
##   <bch:expr>           <bch:tm> <bch:tm>     <dbl> <bch:byt>    <dbl>
## 1 cnorm(4000, 100, 10)   14.8ms   18.7ms      53.3    61.2MB     63.2
```

```
print("right around sample sizes of 4000, cnorm starts to take up too much memory and is slower")
```

```
## [1] "right around sample sizes of 4000, cnorm starts to take up too much memory and is slower"
```

```
bench::mark(rgamma(100, 10, 5))
```

```
## # A tibble: 1 x 6
##   expression             min   median `itr/sec` mem_alloc `gc/sec`
##   <bch:expr>        <bch:tm> <bch:tm>     <dbl> <bch:byt>    <dbl>
## 1 rgamma(100, 10, 5)    9.4us   12.5us    76865.    10.9KB        0
```

```
bench::mark(cgamma(100, 10, 5))
```

```
## # A tibble: 1 x 6
##   expression             min   median `itr/sec` mem_alloc `gc/sec`
##   <bch:expr>        <bch:tm> <bch:tm>     <dbl> <bch:byt>    <dbl>
## 1 cgamma(100, 10, 5)   41.5us     44us    17698.    44.8KB     13.8
```

```
print("the base r version handles this distribution sampling better than the c++ version")
```

```
## [1] "the base r version handles this distribution sampling better than the c++ version"
```