Quick Guide for `coga`

Chaoran Hu

In this vignette, we give a quick guide with R package `coga`. The purpose of `coga` is evaluation of density and distribution function for convolution of independent gamma variables. Let $X_1, \dots, X_n$ be n mutually independent random variables that have gamma distributions with shape parameter $\alpha_i \geq 0$ and rate parameters $\lambda_i > 0$, $i = 1, \dots, n$. Then, the random variables,

$$Y = X_1 + \cdots + X_n,$$

is defined as the convolution of independent gamma variables. In this package, two exact methods (Mathai, 1982 and Moschopoulos, 1985) and one approximate method (Barnabani, 2017) are implemented. The *exact* here means the true value will be evaluated, which is opposite to *approximate*.

```
## load coga in R
library(coga)
```

A quick summary is given here for convenience, which can help you to choose the better method. For details, please read the following sections.

	speed	accuracy	#variables (n)	parameter recycling
`dcoga`, `pcoga`	slow	exact	>=2	yes
`dcoga2dim`, `pcoga2dim`	quick	exact	=2	no
`dcoga_approx`, `pcoga_approx`	medium	approximate	>=3	yes

Exact evaluation of convolution of gamma variables

Let us start from Moschopoulos(1985), which is implemented as the R function `dcoga` and `pcoga`. By this two functions, we can calculate density and distribution function of convolution of gamma variables with $n \geq 2$. For example, we have $Y = X_1 + X_2 + X_3$, with $X_1 \sim Gamma(2, 3)$, $X_2 \sim Gamma(5, 2)$ and $X_3 \sim Gamma(7, 4)$. Then, the density and distribution function of Y at a grid, $1, 2, \dots, 10$, can be evaluated by the following code.

```
dcoga(1:10, c(2, 5, 7), c(3, 2, 4))
```

```
##  [1] 3.363844e-05 1.295149e-02 1.254798e-01 2.783394e-01 2.820258e-01
##  [6] 1.772706e-01 8.124817e-02 2.985823e-02 9.341621e-03 2.589356e-03
```

```
pcoga(1:10, c(2, 5, 7), c(3, 2, 4))
```

```
##  [1] 3.014360e-06 3.036587e-03 6.107646e-02 2.697602e-01 5.641851e-01
##  [6] 7.970735e-01 9.228391e-01 9.749704e-01 9.928284e-01 9.981365e-01
```

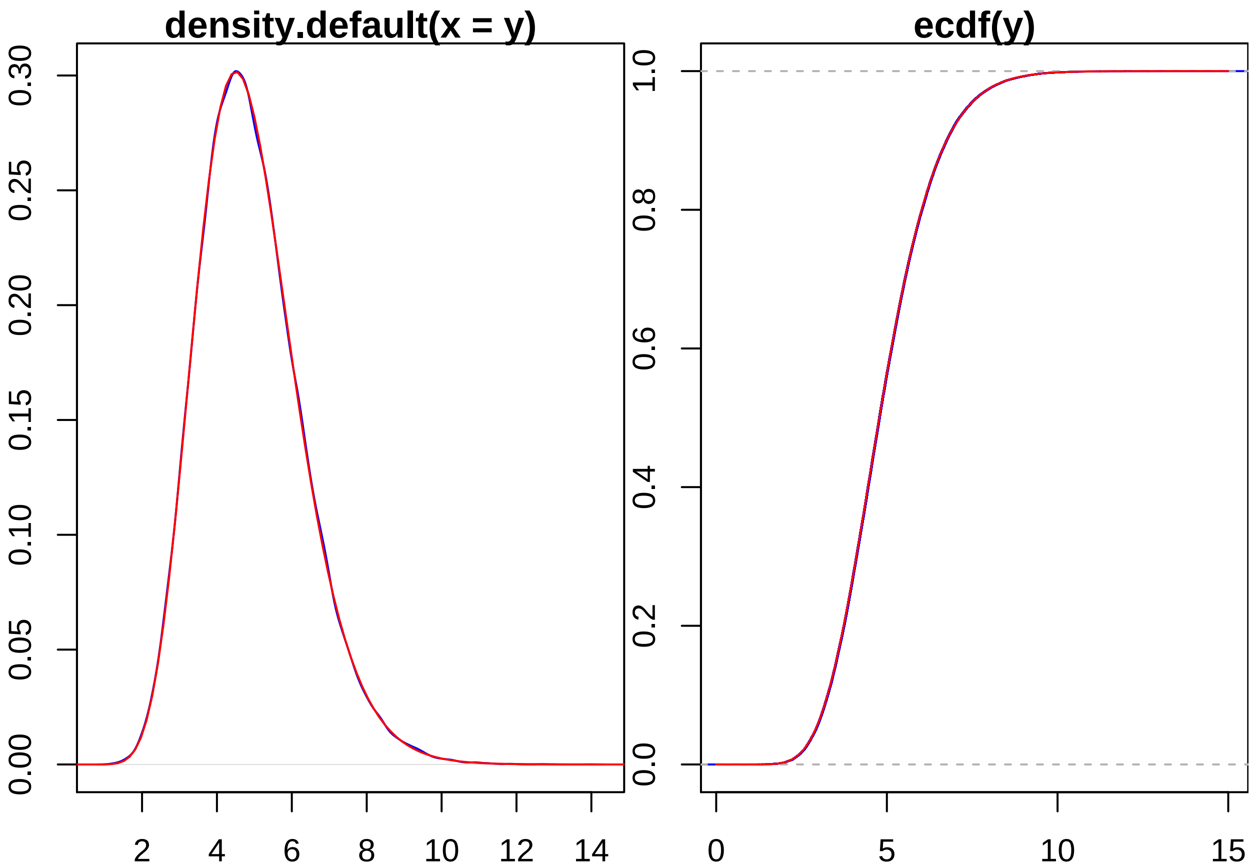
We also show the correctness of these methods by following plot. The left plot is for density and the right plot is for distribution function. The blue lines in these plots is from simulation work and the red lines is from `dcoga` and `pcoga`.

```
set.seed(123)
## do grid
y <- rcoga(100000, c(2, 5, 7), c(3, 2, 4))
grid <- seq(0, 15, length.out=100)
## calculate pdf and cdf
```

```r
pdf <- dcoga(grid, c(2, 5, 7), c(3, 2, 4))
cdf <- pcoga(grid, c(2, 5, 7), c(3, 2, 4))

par(mfrow = c(1, 2), mar = c(2,2,1,0))
## plot pdf
plot(density(y), col="blue")
lines(grid, pdf, col="red")

## plot cdf
plot(ecdf(y), col="blue")
lines(grid, cdf, col="red")
```




```r
par(mfrow = c(1, 1))
```

By these two function, `dcoga` and `pcoga`, we can gain the precise value of density and distribution function of convolution of gamma variables. But, the speed of computation can be improved further under convolution of two gamma variables case ($n = 2$). Mathai(1982) gives us this method, which is implemented as `dcoga2dim` and `pcoga2dim`. For example, we have $Y = X_1 + X_2$, with $X_1 \sim Gamma(2, 3)$ and $X_2 \sim Gamma(5, 2)$ and we calculate the density and distribution function by following code. Note that `dcoga2dim` and `pcoga2dim` give us the same result as `dcoga` and `pcoga`.

```r
dcoga2dim(1:10, 2, 5, 3, 2)
```

```
##  [1] 0.0411859988 0.2777396118 0.3403093599 0.2095050875 0.0890108622
##  [6] 0.0300338151 0.0086685609 0.0022361577 0.0005301785 0.0001177327
```

```r
dcoga(1:10, c(2, 5), c(3, 2))
```

```
##  [1] 0.0411859988 0.2777396118 0.3403093599 0.2095050875 0.0890108622
##  [6] 0.0300338151 0.0086685609 0.0022361577 0.0005301785 0.0001177327
```

```
pcoga2dim(1:10, 2, 5, 3, 2)
```

```
##  [1] 0.00808714 0.16253645 0.49639106 0.77705516 0.92117138 0.97629377
##  [7] 0.99367213 0.99845685 0.99964924 0.99992462
```

```
pcoga(1:10, c(2, 5), c(3, 2))
```

```
##  [1] 0.00808714 0.16253645 0.49639106 0.77705516 0.92117138 0.97629377
##  [7] 0.99367213 0.99845685 0.99964924 0.99992462
```

Now, let's take a look at the difference of computation time between these two methods, which reveals the huge computation speed advantage of `dcoga2dim` and `pcoga2dim`.

```
microbenchmark::microbenchmark(dcoga2dim(1:10, 2, 5, 3, 2),
                               dcoga(1:10, c(2, 5), c(3, 2)),
                               pcoga2dim(1:10, 2, 5, 3, 2),
                               pcoga(1:10, c(2, 5), c(3, 2)))
```

```
## Unit: microseconds
##                           expr      min       lq       mean    median
##    dcoga2dim(1:10, 2, 5, 3, 2)    5.486    7.020   10.68313   10.6305
##  dcoga(1:10, c(2, 5), c(3, 2)) 2396.970 2893.606 4221.80184 3182.2485
##    pcoga2dim(1:10, 2, 5, 3, 2)  369.208  374.189  376.63538  375.7105
##  pcoga(1:10, c(2, 5), c(3, 2)) 2611.657 3083.739 5000.83262 3332.0945
##         uq       max neval cld
##    12.4815    24.344   100  a
##  5989.0650  9480.951   100   b
##   377.8360   419.705   100  a
##  6888.5525 44222.856   100   b
```

Approximate evaluation of convolution of gamma variables

The approximate method is given by Barnabani(2017) and is implemented as `dcoga_approx` and `pcoga_approx`, which only give us the approximate result but also give us the benefit of computation speed. We mention that this method only work for $n \geq 3$. For example, we have $Y = X_1 + X_2 + X_3$, with $X_1 \sim Gamma(2, 3)$, $X_2 \sim Gamma(5, 2)$ and $X_3 \sim Gamma(7, 4)$. Then, the density and distribution function of Y at a grid, $1, 2, \ldots, 10$, can be evaluated by the following code.

```
dcoga_approx(1:10, c(2, 5, 7), c(3, 2, 4))
```

```
##  [1] 3.882054e-05 1.370563e-02 1.264513e-01 2.757307e-01 2.805617e-01
##  [6] 1.786373e-01 8.258924e-02 3.013126e-02 9.117437e-03 2.360774e-03
```

```
pcoga_approx(1:10, c(2, 5, 7), c(3, 2, 4))
```

```
##  [1] 3.509682e-06 3.261758e-03 6.255803e-02 2.703705e-01 5.622780e-01
##  [6] 7.952981e-01 9.226225e-01 9.755455e-01 9.933694e-01 9.984289e-01
```

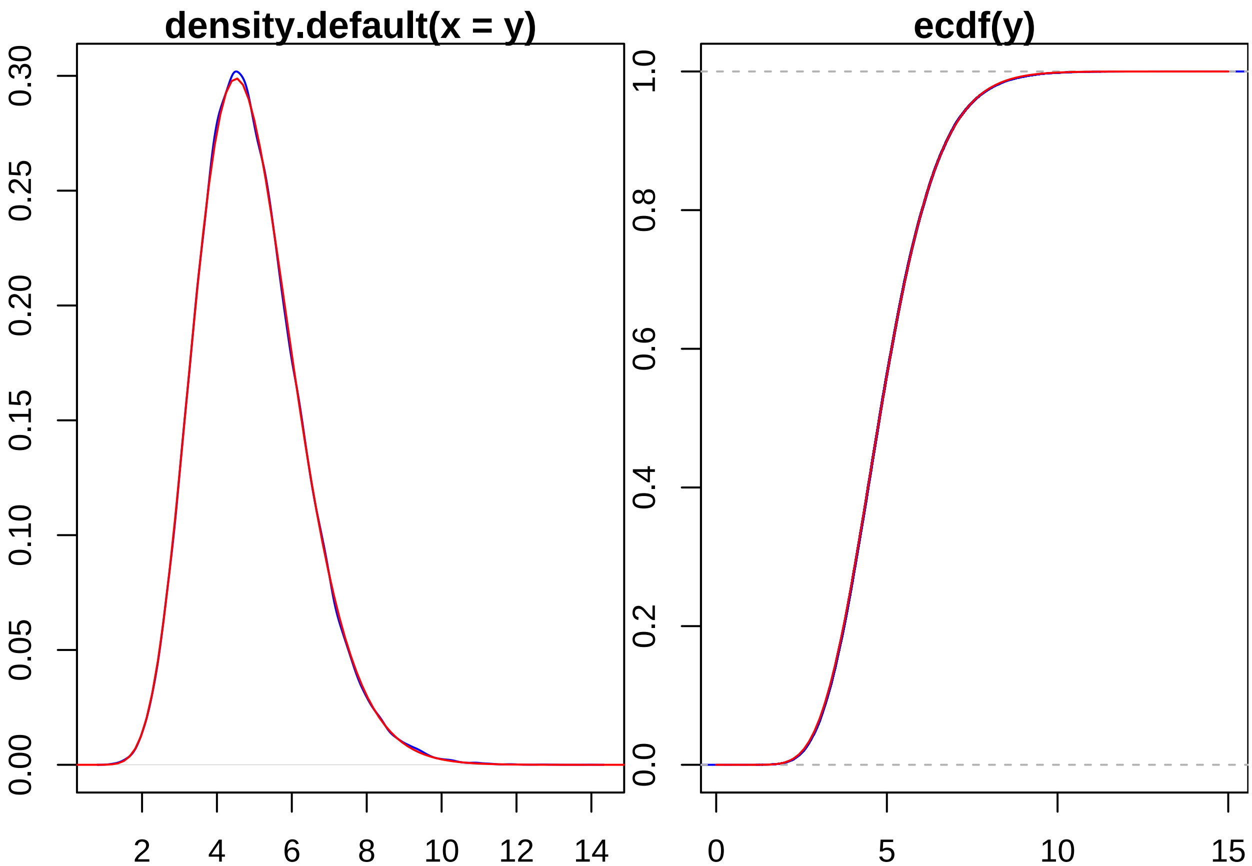
The veracity of approximation method is shown in the following plot.

```
set.seed(123)
## do grid
y <- rcoga(100000, c(2, 5, 7), c(3, 2, 4))
grid <- seq(0, 15, length.out=100)
```

```
## calculate pdf and cdf
pdf <- dcoga_approx(grid, c(2, 5, 7), c(3, 2, 4))
cdf <- pcoga_approx(grid, c(2, 5, 7), c(3, 2, 4))

par(mfrow = c(1, 2), mar = c(2,2,1,0))
## plot pdf
plot(density(y), col="blue")
lines(grid, pdf, col="red")

## plot cdf
plot(ecdf(y), col="blue")
lines(grid, cdf, col="red")
```




```
par(mfrow = c(1, 1))
```

Parameter recycling

The parameter recycling means if the input of `shape` and `rate` have different length, the function will make up the shorter one to the longer one. For example, these two pairs of code will give us the same result.

```
dcoga(1:5, c(1, 2), c(1, 3, 4, 2, 5))
```

```
## Warning in dcoga(1:5, c(1, 2), c(1, 3, 4, 2, 5)): number of rate is not a
## multiple of shape.

## [1] 0.06312673 0.30528954 0.31055854 0.18197777 0.08348673
```

```r
dcoga(1:5, c(1, 2, 1, 2, 1), c(1, 3, 4, 2, 5))
```

```
## [1] 0.06312673 0.30528954 0.31055854 0.18197777 0.08348673
```

```r
pcoga(1:5, c(1, 3, 5, 2, 2), c(3, 5))
```

```
## Warning in pcoga(1:5, c(1, 3, 5, 2, 2), c(3, 5)): number of shape is not a
## multiple of rate.

## [1] 0.0001059771 0.0322652799 0.2792503481 0.6584623021 0.8928924252
```

```r
pcoga(1:5, c(1, 3, 5, 2, 2), c(3, 5, 3, 5, 3))
```

```
## [1] 0.0001059771 0.0322652799 0.2792503481 0.6584623021 0.8928924252
```

Within this package, `dcoga`, `pcoga`, `rcoga`, `dcoga_approx`, `pcoga_approx` have this future.

References

[1] Moschopoulos, Peter G. "The distribution of the sum of independent gamma random variables." Annals of the Institute of Statistical Mathematics 37.1 (1985): 541-544.

[2] Mathai, A.M.: Storage capacity of a dam with gamma type inputs. Ann. Inst. Statist.Math. 34, 591-597 (1982).

[3] Barnabani, M. (2017). An approxmation to the convolution of gamma distributions. Communications in Statistics - Simulation and Computation 46(1), 331-343.