Using CharFunToolR package

Introduction

Application of the exact statistical inference frequently leads to a non-standard probability distributions of the considered estimators or test statistics. Frequently, evaluation of the PDF, CDF, and/or the quantile function (QF) is possible from the CF. In many important situations, derivation of the CFs is more simple than derivation of the PDFs and/or CDFs.

In particular, CF of a linear combination, $Y = c_1X_1 + \ldots + c_nX_n$, where X_j are independent RVs with known CFs, say $\mathrm{cf}_{X_j}(t)$, and coefficients c_j, is given by

$$\mathrm{cf}_Y(t) = \prod_{j=1}^{n} \mathrm{cf}_{X_j}(c_j t).$$

CF of a stochastic convolution (compound distribution), $Y = X_1 + \ldots + X_N$, where X_j are i.i.d. RVs with common $\mathrm{cf}_X(t)$ and N is a discrete RV defined on non-negative integers with $\mathrm{cf}_N(t)$, is given by

$$\mathrm{cf}_Y(t) = \mathrm{cf}_N(-i\log(\mathrm{cf}_X(t))).$$

CF of a weighted mixture of distributions, say $F = \sum_{j=1}^{n} w_j F_{X_j}$ with $\sum_{j=1}^{n} w_j = 1$ is

$$\mathrm{cf}_F(t) = \sum_{j=1}^{n} w_j \mathrm{cf}_{X_j}(t).$$

The empirical characteristic function (ECF), based on the random sample $X_1, \ldots, X_n$, where $X_j \sim F$, is defined as

$$\mathrm{cf}_{\hat{F}_n}(t) = \sum_{j=1}^{n} w_j \mathrm{cf}_{X_j}(t) = \frac{1}{n}\sum_{j=1}^{n} e^{itX_j},$$

i.e. equally weighted mixture of the CFs of the Dirac random variables, $\mathrm{cf}_{X_j}(t) = e^{itX_j}$.

However, analytical derivation of the PDF and/or CDF by using the inverse Fourier transform is available only in special cases. Thus, in most practical situations, a numerical derivation of the PDF/CDF from the CF is an indispensable tool.

The Characteristic Functions Package (**CharFunToolR**) consists of a set of algorithms for evaluating selected characteristic funcions (CF) and algorithms for numerical inversion of the (combined and/or compound) characteristic functions, used to evaluate the probability density function (PDF) and the cumulative distribution function (CDF).

The package includes inversion algorithms, including those based on simple trapezoidal rule for computing the integrals defined by the Gil-Pelaez formulae (Gil-Pelaez, 1951), and/or by using the Fast Fourier Transformation (FFT) algorithm for computing the Fourier transform integrals.

Numerical inversion of characteristic function

Let Y be a random variable with known CF $\mathrm{cf}_Y(t)$. Then, by the Fourier inversion theorem, the PDF of the random variable Y is given by

$$\mathrm{pdf}_Y(y) = \frac{1}{2\pi}\int_{-\infty}^{\infty} e^{-ity}\mathrm{cf}_Y(t)dt, \;\; y \in \mathbb{R}.$$

This exact inverse Fourier transform can be naturally approximated by the truncated integral form

$$\mathrm{pdf}_Y(y) = \frac{1}{2\pi}\int_{-T}^{T} e^{-ity}\mathrm{cf}_Y(t)dt,$$

where T is a sufficiently large (real) value, and the integrand is a complex (oscillatory) function, which is assumed to decay to zero reasonably fast for $|t| \to \infty$ (as is typical for continuous distributions used in metrological applications). Note that the selection of the integration limit and the speed of integrand function decay contribute significantly to the truncation error of this approximation.

In general, the required integral can be evaluated by any suitable numerical quadrature method. Frequently, a simple trapezoidal rule gives fast and satisfactory results. However, caution is necessary if the integrand is a highly oscillatory function, what is the case if $|y|$ is a large value from the tailarea of the distribution, as e.g. the 99% quantile. In such situations a more advanced quadrature method should be used, typically in combination with efficient root-finding algorithms and algorithms for accelerated computing of the limit values of the sums of alternating series (not considered here), see e.g., (Sidi, 1988), (Cohen, Villegas, & Zagier, 2000). Note that the selection of the integration method significantly contributes to the integration error of this approximation.

Let us briefly describe two approaches (see Viktor Witkovsky, 2016 for more details) for numerical inversion of CF implemented in our package. More precisely it is a method based on Gil-Pelaez formulae (Gil-Pelaez, 1951) and FFT algorithm.

The Gil-Pelaez inversion formulae

In (Gil-Pelaez, 1951), Gil-Pelaez derived the inversion formulae, suitable for numerical evaluation of the PDF and/or the CDF, which require integration of a real-valued function, only. The PDF is given by

$$\mathrm{pdf}_Y(y) = \frac{1}{\pi}\int_{0}^{\infty} \mathcal{R}\left(e^{-ity}\mathrm{cf}_Y(t)\right)dt.$$

Further, if y is a continuity point of the cumulative distribution function of Y, the CDF is given by

$$\mathrm{cdf}_Y(y) = \frac{1}{2} - \frac{1}{\pi}\int_{0}^{\infty} \mathcal{I}\left(\frac{e^{-ity}\mathrm{cf}_Y(t)}{t}\right)dt.$$

By $\mathcal{R}\left(f(t)\right)$ and $\mathcal{I}\left(f(t)\right)$ we denote the real and imaginary part of the complex function $f(t)$, respectively.

In general, the integrals in formulas introduced above can be computed by any numerical quadrature method, possibly in combination with efficient root-finding algorithms and accelerated computing of limits of the alternating series, as considered, e.g., in (Sidi, 1982) and (Waller, Turnbull, & Hardin, 1995). For more details about numerical computations, efficient approximations of integrals, precision and successful previous implementations of Gil-Pelaez inversion formulae, see (Viktor Witkovsky, 2016) or documentation of function `cf2DistGP`.

In our package function `cf2DistGP` implements this approach for numerical inversion of CF. You can see examples how to use the function in the following section.

Numerical inversion of the characteristic function by using the FFT algorithm

This approach for computing the PDF by numerical inversion of the characteristic function by using the FFT algorithm is based on the results by Hürlimann in (Hürlimann, 2013). Alternatively, for other applications based on using the fractional fast Fourier transform (FRFT), see (Bailey & Swarztrauber, 1991) and also (Carr & Madan, 1999, Chourdakis (2005), Held (2014), Kim, Rachev, Bianchi, & Fabozzi (2009)).

Here we shall approximate the continuous Fourier transform (CFT), say

$$F(y) = \int\limits_{-\infty}^{\infty} e^{-u2\pi uy}f(u)du,$$

by a discrete Fourier transform (DFT). The DFT can be efficiently evaluated by using the FFT algorithm that computes the same result as the DFT, but much faster.

For complex numbers $f_0, \ldots, f_{N-1}$ the DFT is defined as

$$F_k = \sum_{j=0}^{N-1} e^{-i2\pi k \frac{j}{N}} f_j, \;\; k \in 0, 1, \ldots, N-1.$$

Formally, here we shall use the following notation,

$$\boldsymbol{F}_N = FFT(\boldsymbol{f}_N),$$

where $\boldsymbol{f}_N = (f_0, \ldots, f_{N-1})$ and $\boldsymbol{F}_N = (F_0, \ldots, F_{N-1})$.

The relationship between the CF and the PDF is given by the (inverse) continuous Fourier transform defined by

$$\mathrm{pdf}_Y(y) = \frac{1}{2\pi} \int_{-\infty}^{\infty} e^{-ity} \mathrm{cf}_Y(t) dt, \;\; y \in \mathbb{R}.$$

For a sufficiently large interval $(-T, T)$ it is possible to approximate a PDF by

$$\mathrm{pdf}_Y(y) = \frac{1}{2\pi} \int_{-T}^{T} e^{-ity} \mathrm{cf}_Y(t) dt.$$

Here we shall consider the integral approximation, based on the mid-point integration rule, $\int\limits_a^b f(x)dx \approx \frac{f(a)+f(b)}{2}(b-a)$, which corresponds to the trapezoidal quadrature. For more alternative approaches based on more sophisticated integration rules see, e.g., (V. Witkovsky, Wimmer, & Duby, 2015). For more details about numerical inversion of the characteristic function by using the FFT algorithm, see (Viktor Witkovsky, 2016) or documentation of function `cf2DistFFT`.

The CharFunToolR package

In this section we present several classes of R functions from our package. We choose some functions as representatives, we give examples how to use particular functions in R and the other functions in the package from the same families of functions work similalrly.

Functions & examples

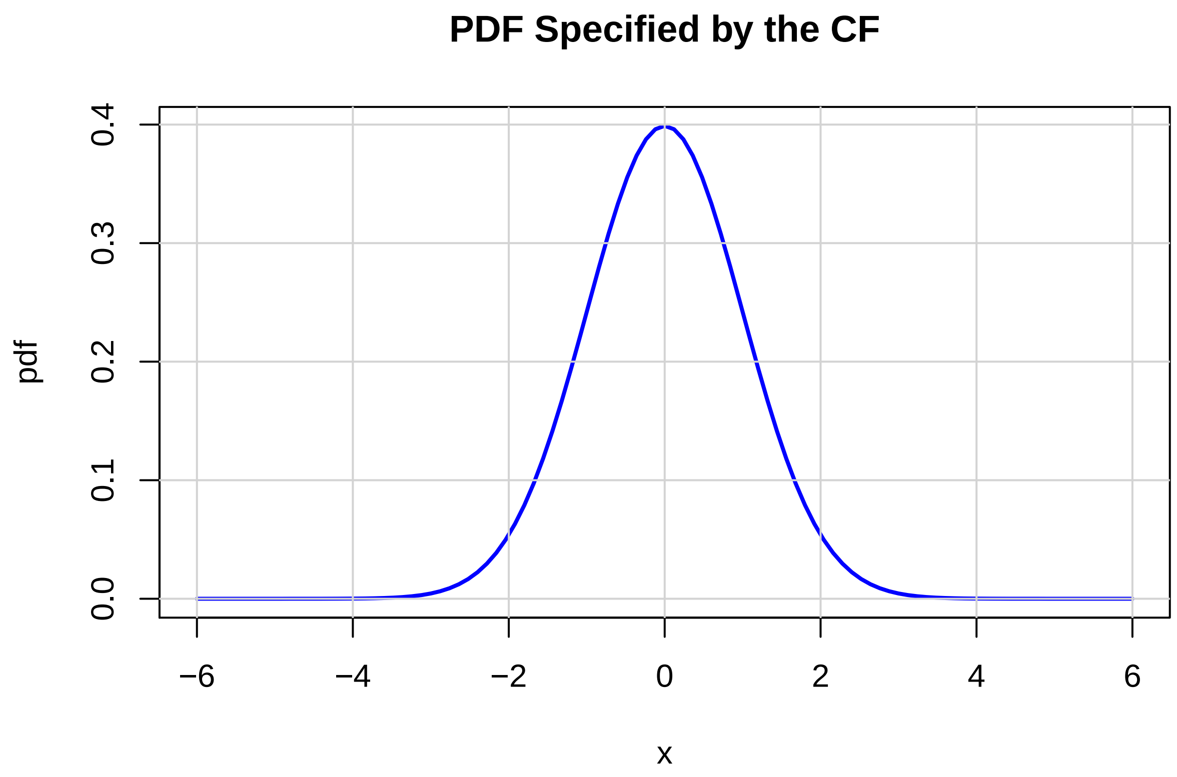
The first important class of functions in our package is *CF Inversion Algorithm*. Currently it involves two R functions: `cf2DistGP` and `cf2DistFFT` mentioned in introductory part of this manual. We have already explained shortly the main ideas of algorithms which these functions implement. So let us take a look at particular examples in R about how to use at first `cf2DistGP` and then `cf2DistFFT`.

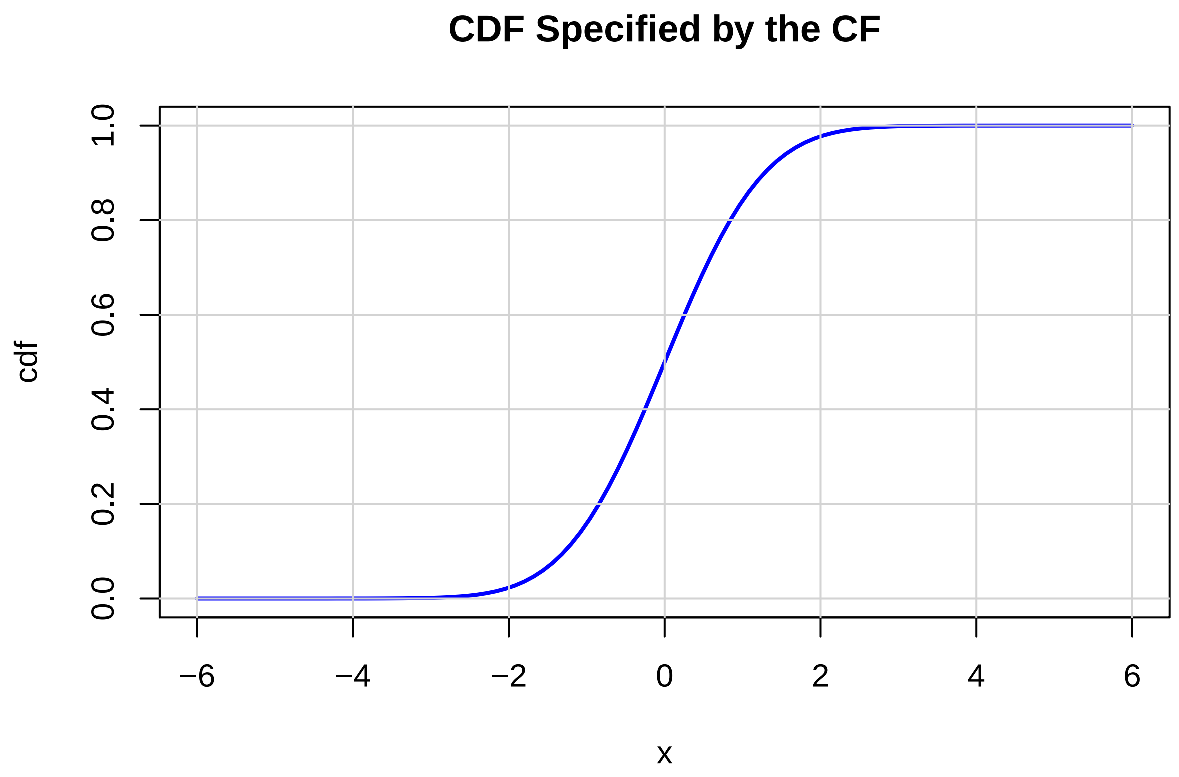
Imagine we would like to calculate CDF/PDF of $N(0,1)$ by inverting its CF. At least the `cf` input argument of `cf2DistGP` need to be specified. We connect package **CharFunToolR** and create a characteristic function for $N(0,1)$.

```
library(CharFunToolR)
cf <- function(t) exp(-t ^ 2 / 2)
```

The final step is to call `cf2DistGP` to calculate CDF/PDF.

```
result <- cf2DistGP(cf)
```


PDF Specified by the CF
pdf
0.0
0.1
0.2
0.3
0.4
−6
−4
−2
0
2
4
6
x



```
# to see the structure of the whole output you can run the following command
# str(result)
```

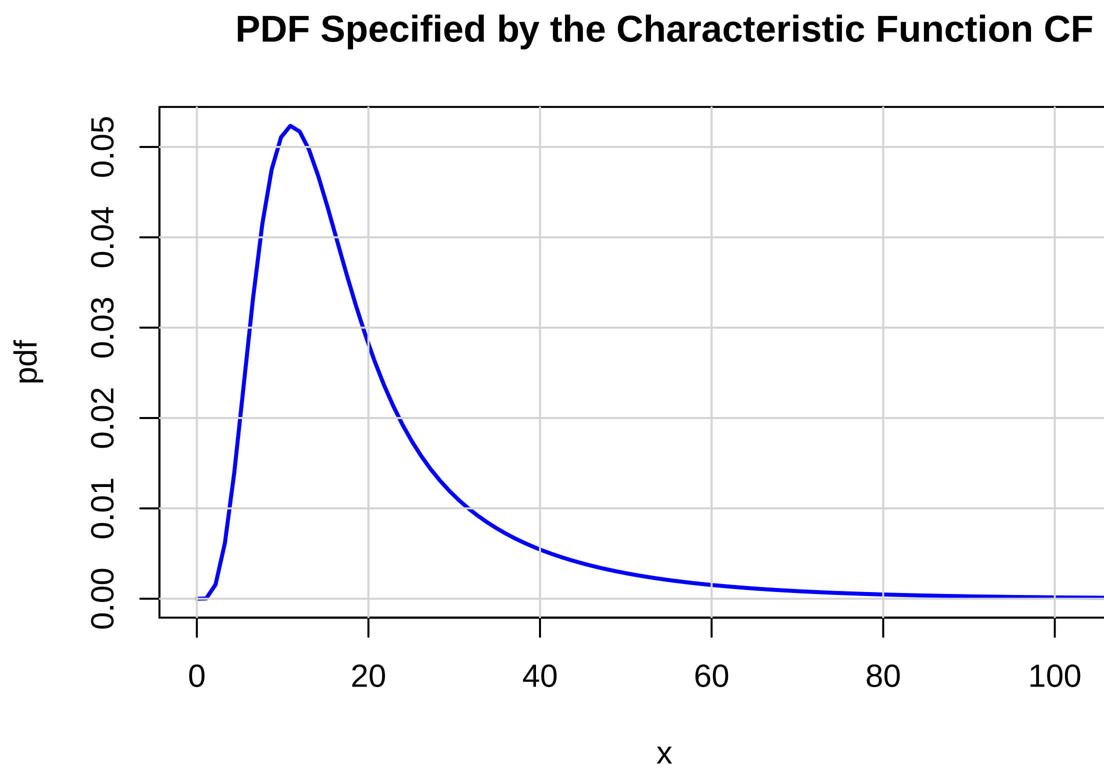
Note that argument `x` (vector of values where the CDF/PDF is computed) of `cf2DistGP(cf, x, prob, options)` is set by default. Similarly `options` object (list) is created with some default values. As `prob` has not been specified, the quantiles are not calculated and therefore `options$qf = NULL`.

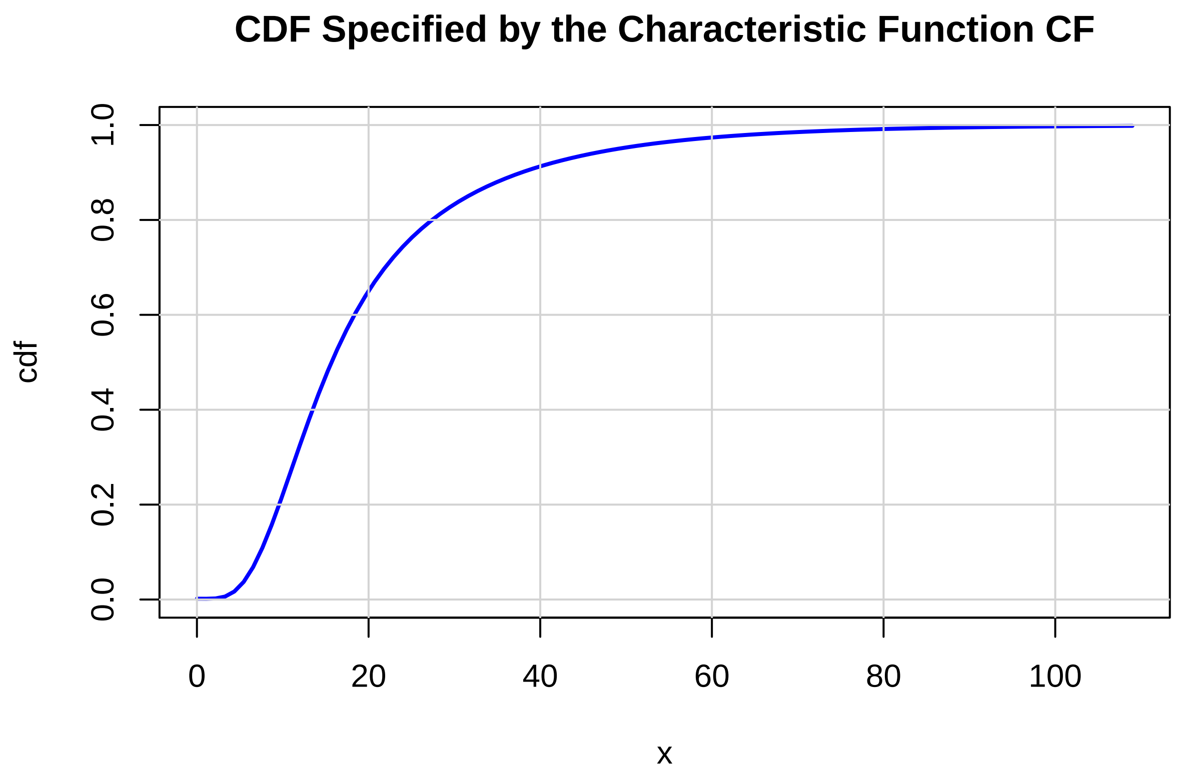
In the next example we would like to calculate CDF/PDF/CQ of linear combination of independent χ^2 random variables (RVs) (Chi-squared RVs with 1 and 10 degrees of freedom) $Y = 10X_{\chi^2_1} + X_{\chi^2_{10}}$. At first we specify degrees of freedom for $X_{\chi^2_1}$ and $X_{\chi^2_1}$, create their characteristic funtions and also the CF of their linear combination Y.

```
df1 <- 1
df2 <- 10
cfChi2_1 <- function(t) (1 - 2i * t)^(-df1 / 2)
cfChi2_10 <- function(t) (1 - 2i * t)^(-df2 / 2)
cf_Y <- function(t) cfChi2_1(10 * t) * cfChi2_10(t)
```

Again the `cf` argument of `cf2DistFFT(cf, x, prob, options)` is mandatory. The other input arguments can be specified or not and consequently the `cf2DistFFT` is executed.

```
options <- list()
options$xMin <- 0
result <- cf2DistFFT(cf = cf_Y, options = options)
```


PDF Specified by the Characteristic Function CF
pdf
0.00
0.01
0.02
0.03
0.04
0.05
0
20
40
60
80
100
x



```
# str(result)
```

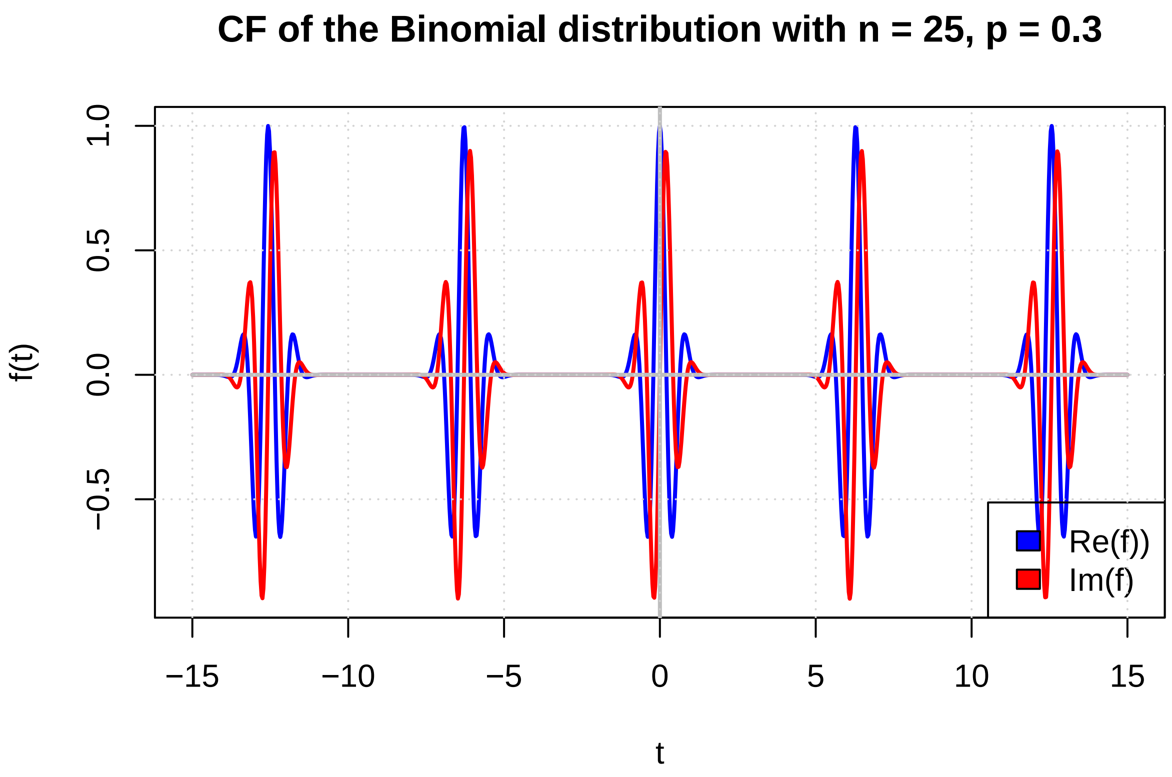
Note that argument `x` (vector of values where the CDF/PDF is computed) of `cf2DistFFT(cf, x, prob, options)` is set by default again. Despite the fact that `prob` has not been specified, this time the default values has been set `prob = c(0.9, 0.95, 0.975, 0.99, 0.995, 0.999)`. In turn the corresponding quantiles can be calculated.

The next family of R functions in our package is *Discrete Probability Distribution*. It includes characteristic functions of chosen discrete probability distributions. We give three examples to illustrate what can be done in **CharFunToolR** with discrete probability distributions. We demonstrate ideas with function `cfN_Binomial`. In first example we want to compute values of CF of the Binomial distribution with the following parameters: the number of trials `n = 25`, probability of success `p = 0.3`. We choose 1001 equally spaced points in interval $[-15, 15]$, in that we want to know the values of CF.

```
n <- 25
p <- 0.3
t <- seq(-15, 15, length.out = 1001)
```

Input arguments of `cfN_Binomial` have been set so we can execute the code below. You can see that we use an auxiliary function `plotReIm` from **CharFunToolR** to plot the real and imaginary part of CF.

```
plotReIm(function(t)
        cfN_Binomial(t, n, p),
        t,
        title = "CF of the Binomial distribution with n = 25, p = 0.3")
```

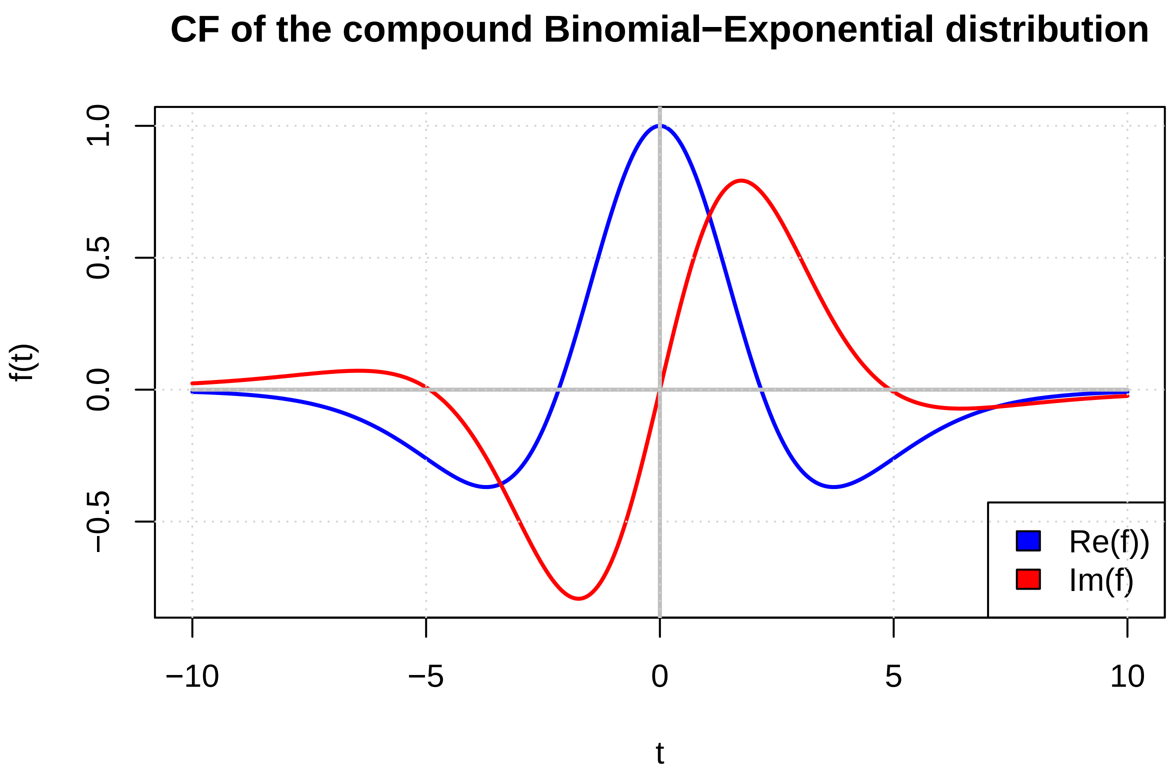


The next example is about calculation of values of CF of the compound Binomial-Exponential distribution. As in the previous example we set values of input parameters `n` and `p` for Binomial distribution. We also set λ parameter for Exponential distribution, define `cfX` as the characteristic function of Exponential distribution and choose 501 equally spaced points in $[-10, 10]$ in which we want to calculate values of CF of the compound Binomial-Exponential distribution. Note that we use `cfX_Exponential` function from **CharFunToolR**.

```
n <- 25
p <- 0.3
lambda <- 10
cfX <- function(t) cfX_Exponential(t, lambda)
t <- seq(-10, 10, length.out = 501)
```

All the input parameters have been set so we can run the final command to get the values of CF and to plot the CF.

```
plotReIm(function(t)
        cfN_Binomial(t, n, p, cfX),
        t,
        title = "CF of the compound Binomial-Exponential distribution")
```



In the third example we want to compute PDF/CDF of the compound Binomial-Exponential distribution using `cf2DistGP` function. We set the input parameters similarly as in the previous examples.

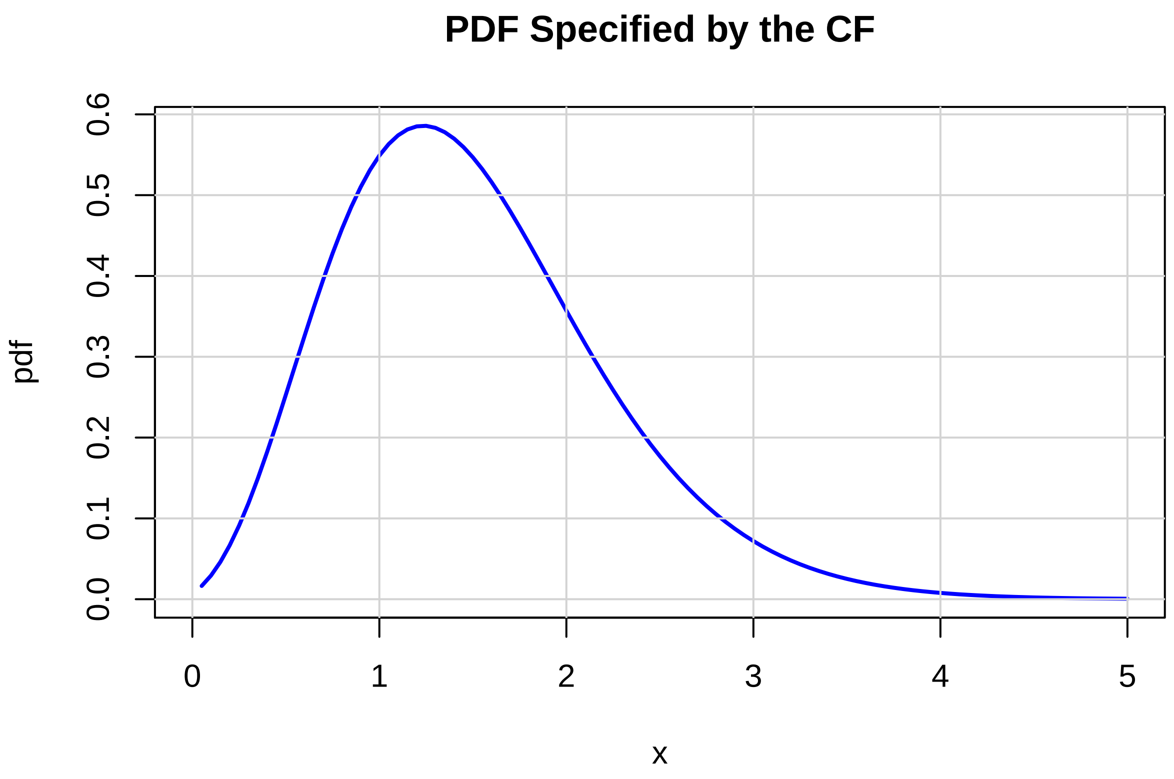
```
n <- 25
p <- 0.3
lambda <- 5
cfX <- function(t)
        cfX_Exponential(t, lambda)
cf <- function(t)
        cfN_Binomial(t, n, p, cfX)
```

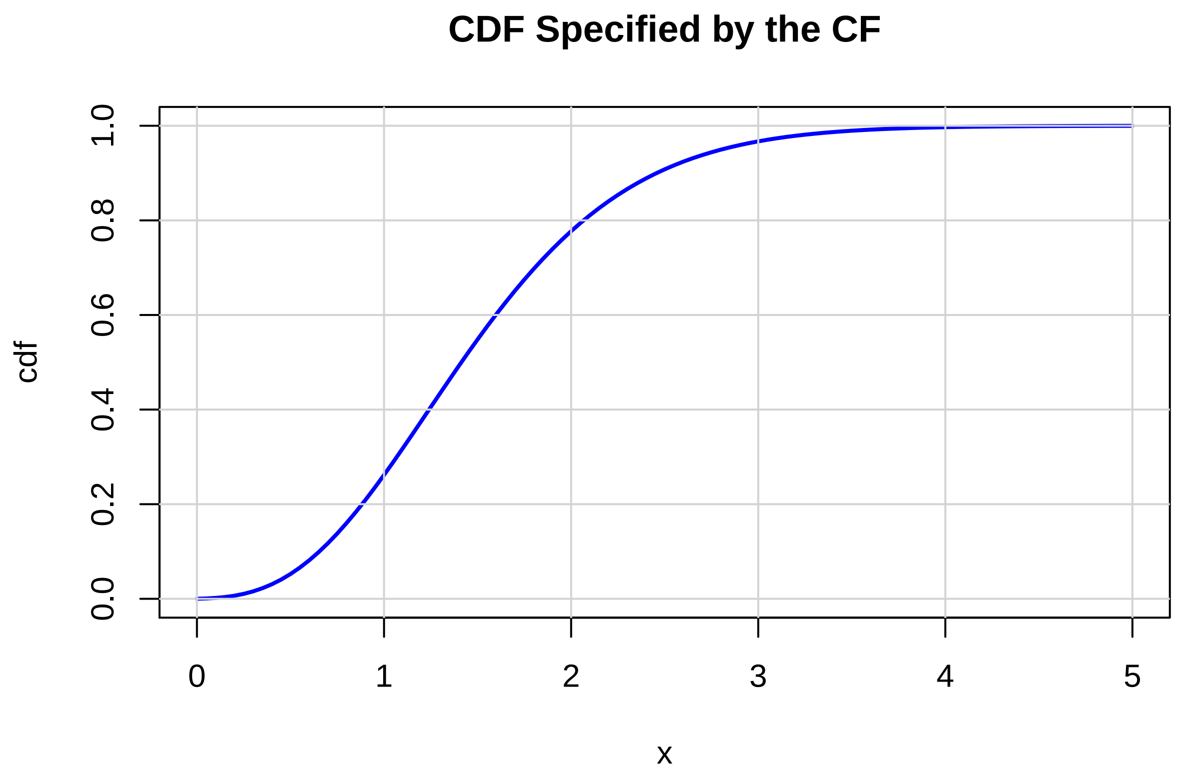
We choose 101 equally spaced points (`x`) in $[0, 5]$ to calculate PDF/CDF. Probabilities (`prob`) are specified to calculate corresponding quantiles (Quantile Function - QF). It is necessary to specify for `cf2DistGP` that considered distribution is compound.

```
x <- seq(0, 5, length.out = 101)
prob <- c(0.9, 0.95, 0.99)
options <- list()
options$isCompound <- TRUE
```

Finally we execute `cf2DistGP(cf, x, prob, options)` to calculate PDF/CDF/QF of the compound Binomial-Exponential distribution. As you can see, the plots for PDF and CDF are created by default.

```
result <- cf2DistGP(cf, x, prob, options)
```

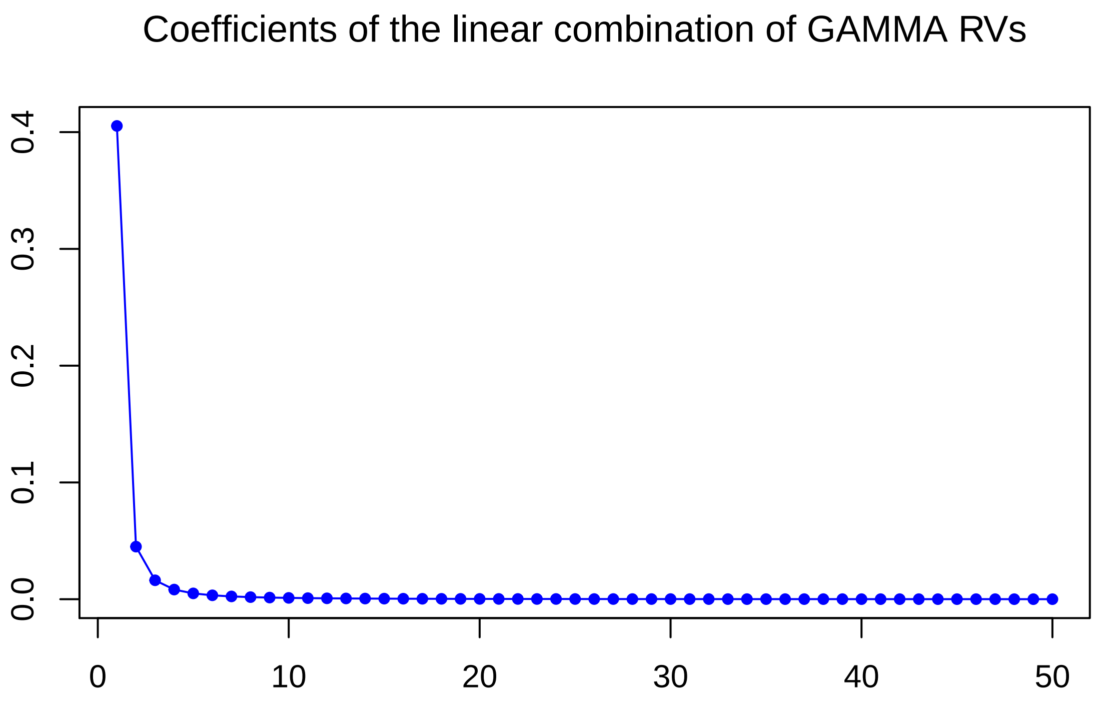

PDF Specified by the CF
pdf
x
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0
1
2
3
4
5



```r
# str(result)
```

Another family of R functions in our package is *Continuous Probability Distribution*. It includes characteristic functions of chosen continuous probability distributions. We give two examples to illustrate some applications of mentioned class of R functions in **CharFunToolR**. We demonstrate ideas with function `cf_Gamma`. In first example we would like to calculate values of CF of a linear combination of independent Gamma RVs. So we need to specify coefficients of particular linear combination. We can also plot the coefficients.

```r
coef <- 1 / (((1:50) - 0.5) * pi) ^ 2
plot(
        1:50,
        coef,
        xlab = "",
        ylab = "",
        type = "p",
        pch = 20,
        col = "blue",
        cex = 1,
        main = expression('Coefficients of the linear combination of GAMMA RVs')
)
lines(1:50, coef, col = "blue")
```

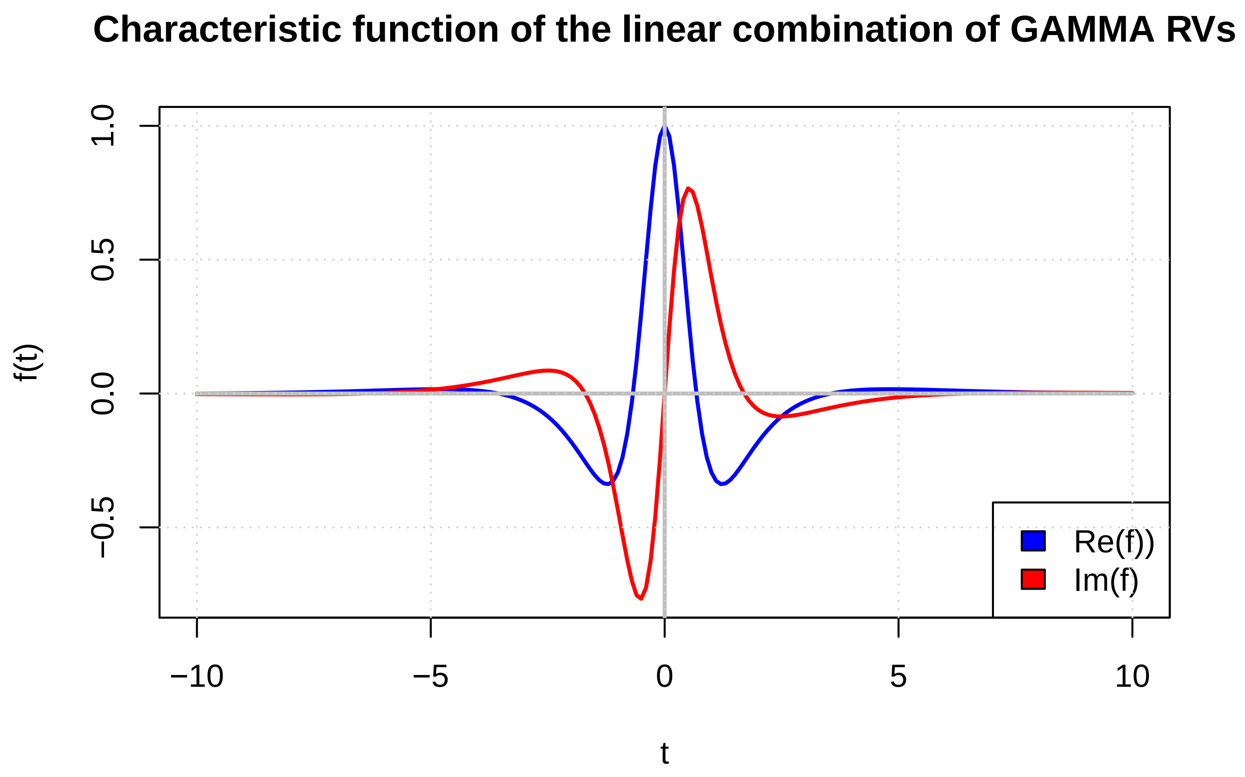


In the following step we specify parameters of Gamma RVs. Note that all RVS of considered linear combination have the same shape (`alpha`) and scale (`beta`) parameter. We also specify points (`t`) to calculate the values of CF.

```
alpha <- 5 / 2
beta <- 1 / 2
t <- seq(from = -10,
         to = 10,
         length.out = 201)
```

Now we can calculate the values of CF, plot the real and imaginary part of CF.

```
plotReIm(function(t)
        cf_Gamma(t, alpha, beta, coef),
        t,
        title = "Characteristic function of the linear combination of GAMMA RVs")
```



In the second example we want to compute PDF/CDF from the CF by `cf2DistGP`. To this purpose we consider the same linear combination of independent Gamma RVs as in the previous example.

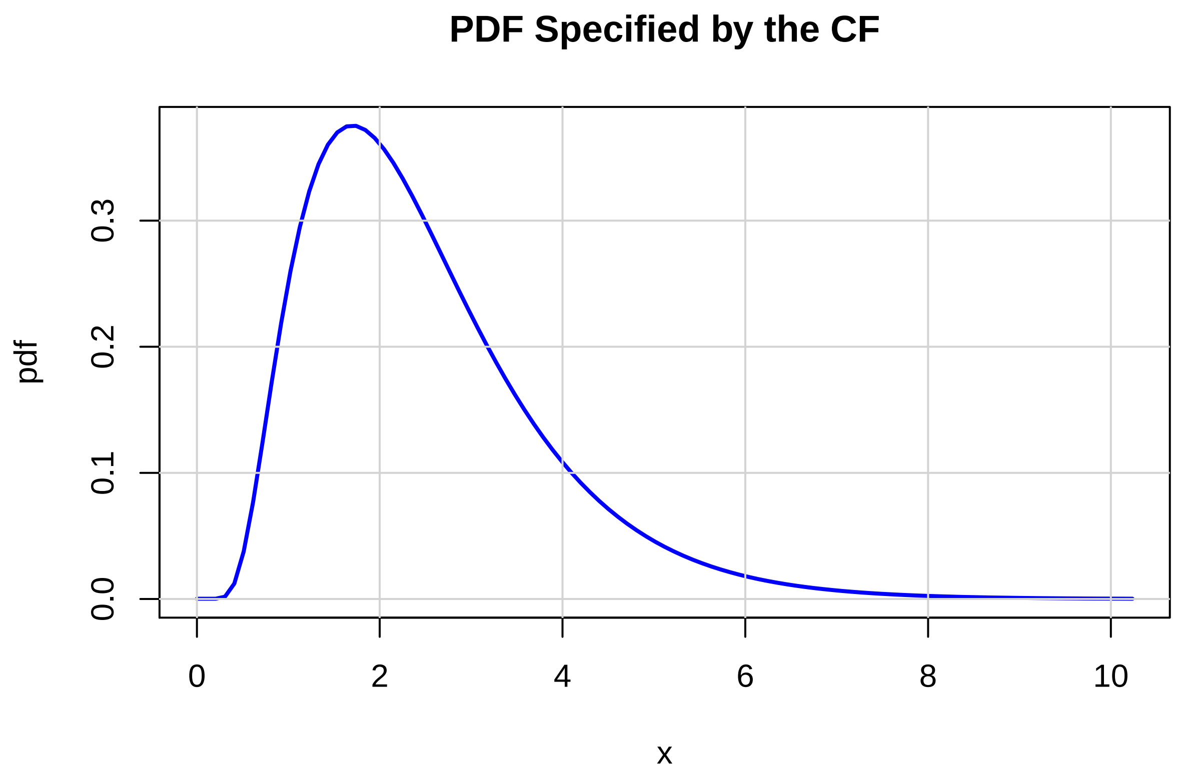
```
alpha <- 5 / 2
beta <- 1 / 2
coef <- 1 / (((1:50) - 0.5) * pi) ^ 2
cf <- function(t)
        cf_Gamma(t, alpha, beta, coef)
```

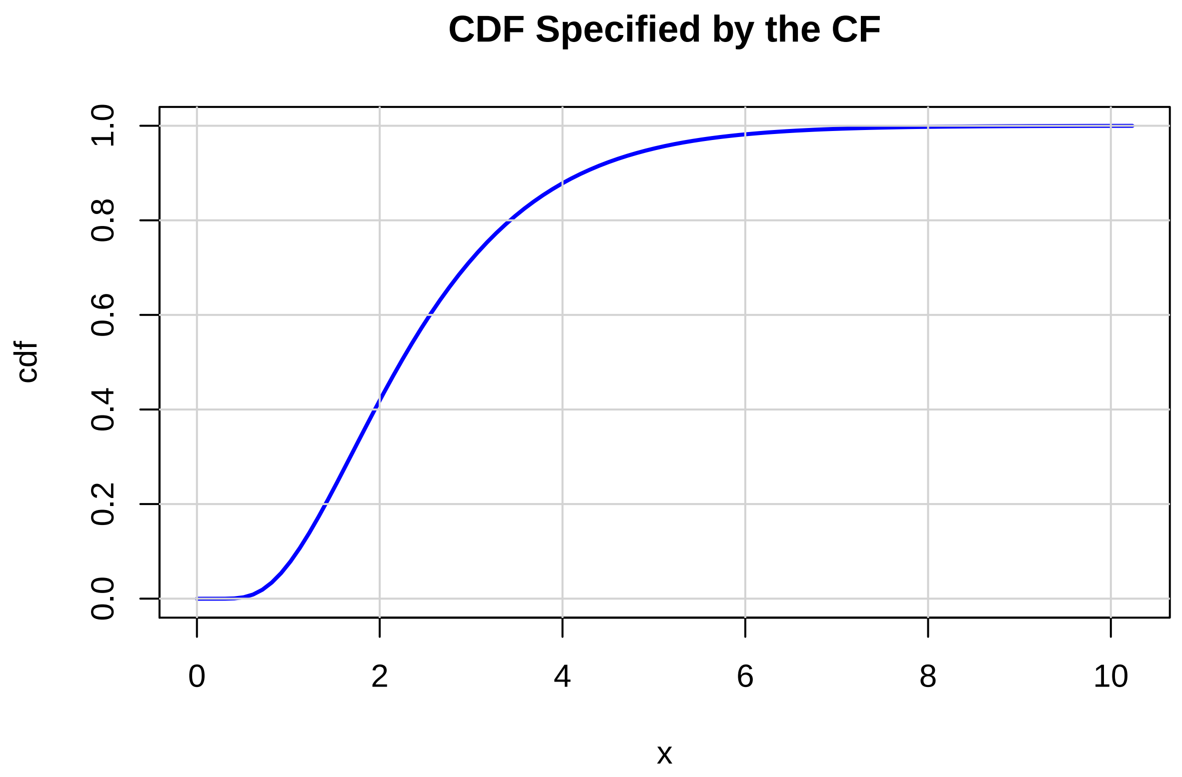
We can specify some advanced options. Realizing that support of PDF/CDF is a non-negative part of real line we set `options$xMin` to 0. We can controll the precision of numerical integration by setting `options$N`.

```
options <- list()
options$N <- 2 ^ 10
options$xMin <- 0
```

Finally we execute `cf2DistGP(cf = cf, options = options)` to get the numerical and graphical result of PDF/CDF.

```
result <- cf2DistGP(cf = cf, options = options)
```


PDF Specified by the CF
pdf
x
0.0
0.1
0.2
0.3
0
2
4
6
8
10



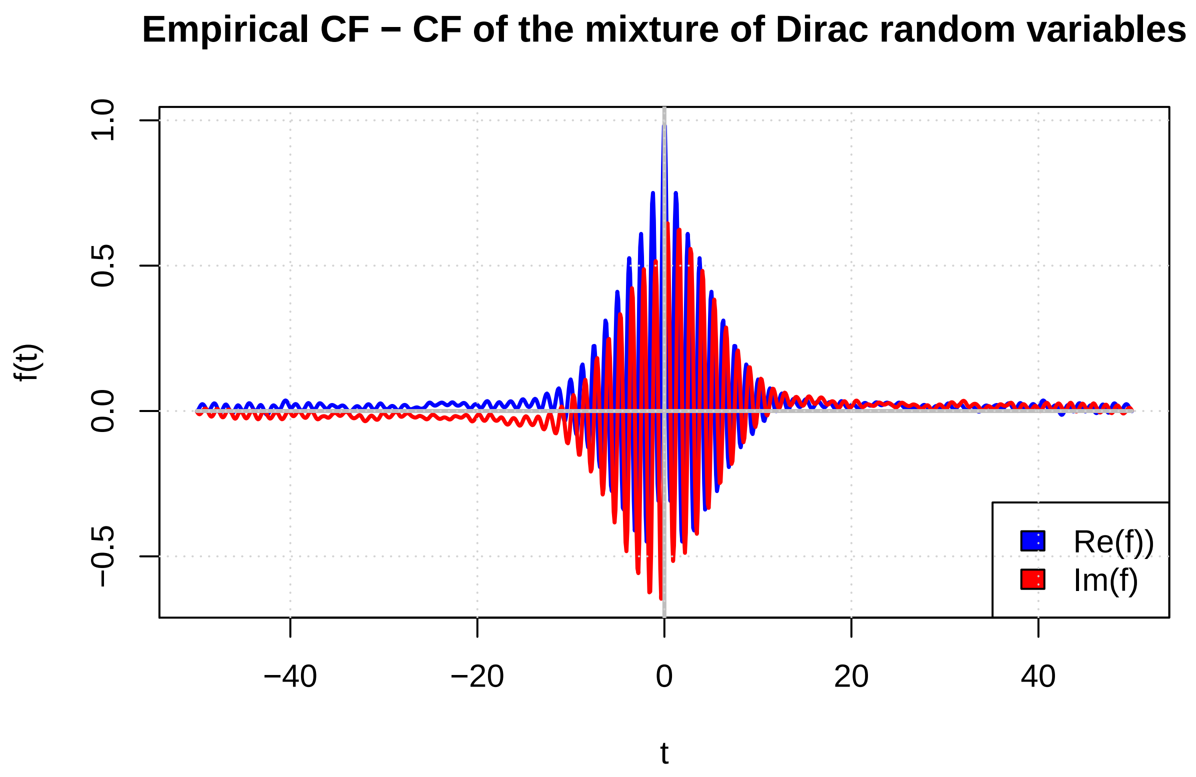
Note that funtion `cf_Gamma` has one more optional input argument denoted `niid`. It stands for scalar convolution coeficient. It is possible to calculate CF of convolution. See documentation of `cf_Gamma` for more details.

In this last part of tutorial we present examples with function `cfE_Empirical` from family denoted *Empirical Probability Distribution*. `cfE_Empirical` is based on the following principle. Let $X_1, \ldots, X_n$ be i.i.d. random variables with the common distribution function F_X. The empirical distribution based on the random sample $X_1, \ldots, X_n$ is a mixture distribution of equally weighted degenerate Dirac distributions, concentrated at $X_1, \ldots, X_n$. Hence, the observed empirical characteristic function (ECF) is equally weighted mixture of the characteristic functions of the Dirac random variables concentrated at the observed values x_j if X_j, i.e. mixture of CFs given by $cf_{x_j}(t) = e^{itx_j}$,

$$cf_{\hat{F}_X}(t) = \frac{1}{n}\sum_{j=1}^{n} e^{itx_j}.$$

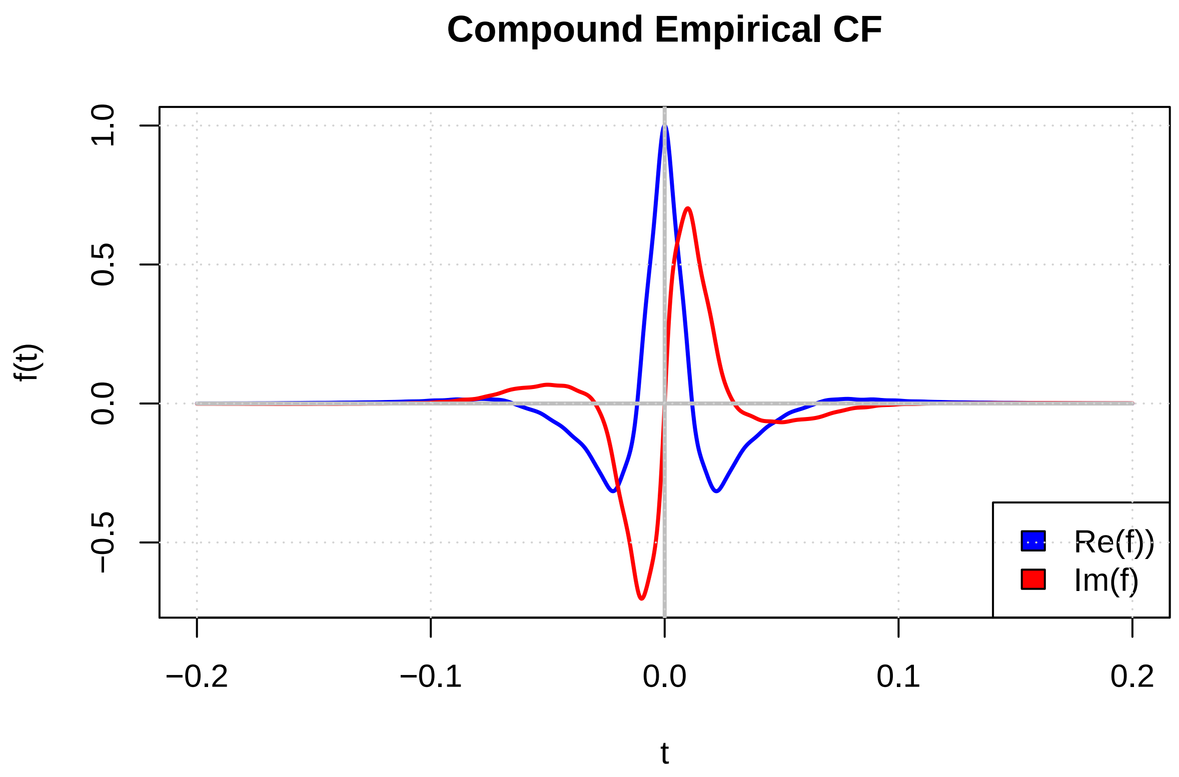
Let us first introduce an example, in which we calculate the ECF of weighted mixture of independent Dirac variables from particular Normal, Student's t and Chi-square distribution.

```r
set.seed(101)
n <- 1000
data <- c(rnorm(3 * n, 5, 0.2), rt(n, 3), rchisq(n, 1))
t <- seq(-50, 50, length.out = 2 ^ 10)
plotReIm(function(t)
        cfE_Empirical(t, data),
        t,
        title = "Empirical CF - CF of the mixture of Dirac random variables")
```



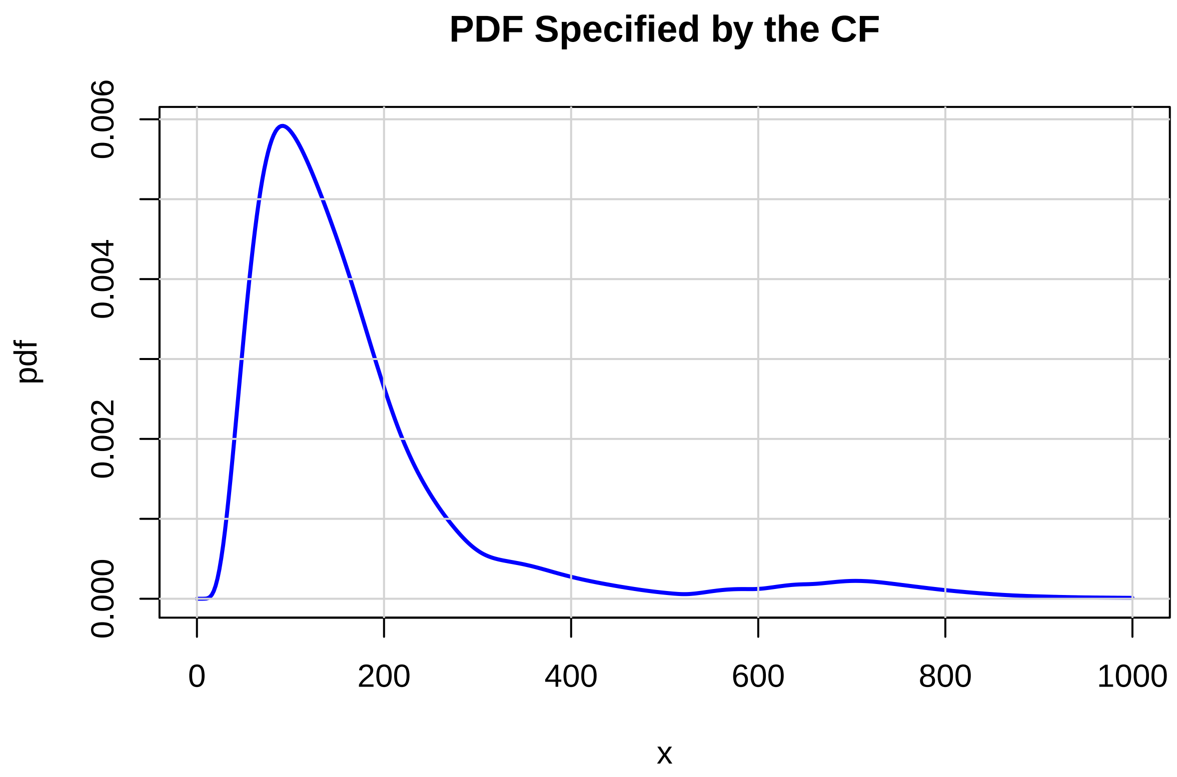
It is also possible to use `cfE_Empirical` to get the values of PDF/CDF of the compound empirical distribution. In the following example we compose the Binomial and Log Normal distibution. At first we calculate and plot the ECF of the compound empirical distribution.

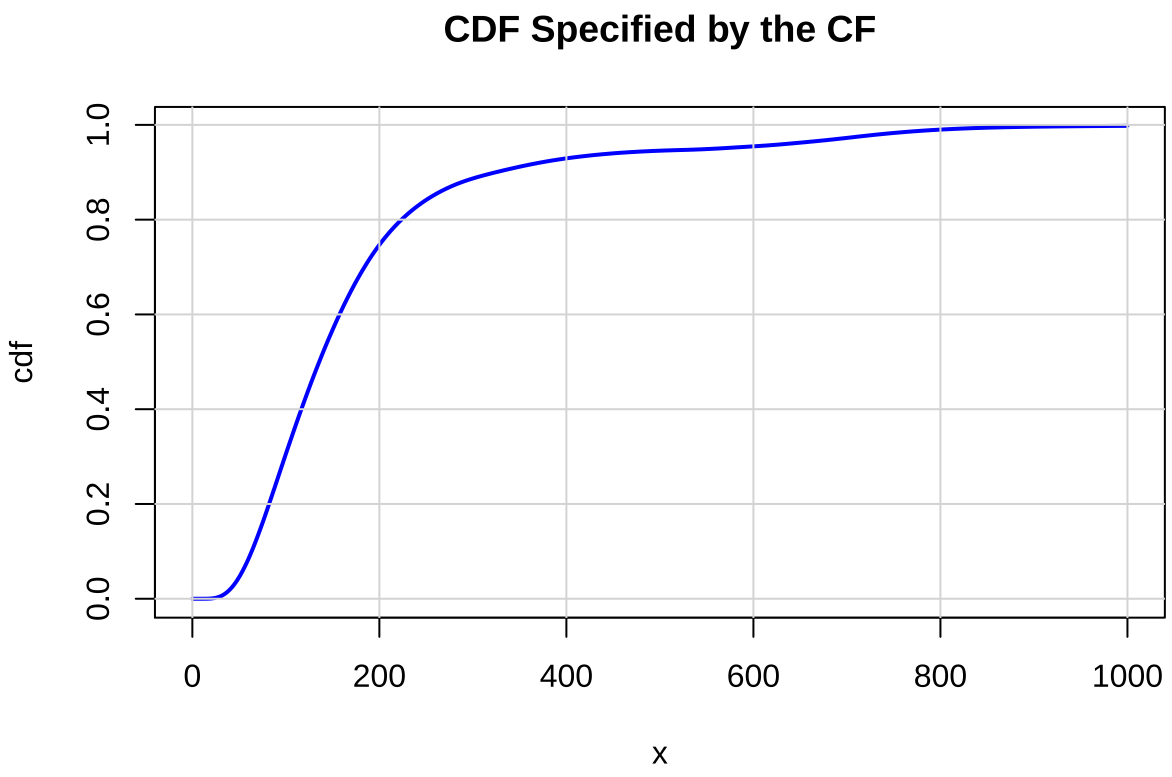
```r
set.seed(101)
lambda <- 25
nN <- 10
Ndata <- rpois(nN, lambda)

mu <- 0.1
sigma <- 2
nX <- 1500
Xdata <- rlnorm(nX, mu, sigma)
cfX <- function(t)
        cfE_Empirical(t, Xdata)
cf  <- function(t)
        cfE_Empirical(t, Ndata, cfX)
t <- seq(-0.2, 0.2, length.out = 2 ^ 10)
plotReIm(cf, t, title = "Compound Empirical CF")
```



In the final step we compute and plot PDF/CDF of the compound empirical distribution. We also get the values for chosen percentiles of that distribution.

```
x <- seq(0, 1000, length.out = 501)
prob <- c(0.9, 0.95)
options <- list()
options$N <- 2 ^ 10
options$SixSigmaRule <- 10
result <- cf2DistGP(cf, x, prob, options)
```


PDF Specified by the CF
pdf
0.000
0.002
0.004
0.006
0
200
400
600
800
1000
x



There some other families of R functions in **CharFunToolR**. For example *Symmetric Probability Distribution*, *Circular Probability Distribution*, but such functions belong to *Continuous Probability Distribution* family as well. You can find some auxiliar functions in our package. These are represented by families *Special Function* or *Graphic Function*. Such functions serve a special purpose. For example they are used in case of interpolation or when some specific (hypergeometric) function needs to be computed or they are used to create plots. We do not provide details of using such functions here, as they are beyond the main idea of **CharFunToolR** package. You can find some details in documentation of particular functions.

References

Bailey, D., & Swarztrauber, P. (1991). The Fractional Fourier Transform and Applications. *SIAM Rev.*, *33*(3), 389–404.

Carr, P., & Madan, D. B. (1999). Option Valuation Using the Fast Fourier Transform. *Journal of Computational Finance*, *2*, 61–73.

Chourdakis, K. (2005). Option pricing using the fractional FFT. *Journal of Computational Finance*, *8*(2), 1–18.

Cohen, H., Villegas, F. R., & Zagier, D. (2000). Convergence Acceleration of Alternating Series. *Experimental Mathematics*, *9*(1), 3–12.

Gil-Pelaez, J. (1951). Note on the inversion theorem. *Biometrika*, *38*(3-4), 481–482.

Held, M. (2014, August). CFH Toolbox (Characteristic Function Option Pricing) - File Exchange - MATLAB Central.

Hürlimann, W. (2013). Improved FFT approximations of probability functions based on modified quadrature rules. In *International Mathematical Forum* (Vol. 8, pp. 829–840).

Kim, Y. S., Rachev, S., Bianchi, M. L., & Fabozzi, F. J. (2009). Computing VaR and AVaR in infinitely divisible distributions.

Sidi, A. (1982). The Numerical Evaluation of Very Oscillatory Infinite Integrals by Extrapolation. *Mathematics of Computation*, *38*(158), 517–529.

Sidi, A. (1988). A User-Friendly Extrapolation Method for Oscillatory Infinite Integrals. *Mathematics of Computation*, *51*(183), 249–266.

Waller, L. A., Turnbull, B. W., & Hardin, J. M. (1995). Obtaining Distribution Functions by Numerical Inversion of Characteristic Functions with Applications. *The American Statistician*, *49*(4), 346–350.

Witkovsky, V. (2016). Numerical inversion of a characteristic function: An alternative tool to form the probability distribution of output quantity in linear measurement models. *ACTA IMEKO*, *5*(3), 32–44.

Witkovsky, V., Wimmer, G., & Duby, T. (2015). Logarithmic Lambert WxF random variables for the family of chi-squared distributions and their applications.