R functional programming approach for truncated and shifted and for shifted and truncated probability distribution functions

Introduction

R programming functional nature allows programmer to implement and manipulate functions as needed. Functional programming is a style of programming that emphasizes the evaluation of expressions rather than the execution of commands, so these functions produce results that depend only on their inputs and not on the program state. This style has a great power and includes lots of tools that combined together can allows us to produce results with the most efficient computational method. One of them is the functional factory, which is a factory for making new functions. It allows us to use closures, which are functions together with a referencing environment, to write specific functions that, in turn, can be used to generate new functions. So we can have a double layer of development: a first layer that is used to do all the complex work in common to all functions and a second layer that defines the details of each function. This concept was used to develop the computational method proposed by Andrea Spanò in "R package qdist: A functional programming approach to truncated probability distributions", and that I used in "R package qshift: A functional programming approach to shifted probability distributions", that produces functions for truncated and shifted distributions, respectively. `qdist` and `qshift` packages are available on Github:

```
devtools::install_github("veronicagiro/qtruncate")
devtools::install_github("veronicagiro/qshift")
```

In the previously cited articles, we dealt separately with shifted and truncated distributions. But in real life, we sometimes work with random variables arising from more than one transformation. So in this article, we will demonstrate that we can produce functions for truncated and shifted or for shifted and truncated distributions only combining the functions of the `qdist` and `qshift` packages. This is possible because of functional programming approach used to build them.

Distribution functions for truncated and shifted or for shifted and truncated distributions

R provides density, probability, quantile and random number generator functions in `stats` package. These functions share a stable naming convention both for the names of the functions and in the first argument of these functions. In particular:

- `d<dist>(x)` with `x` vector of quantiles, identifies density functions
- `p<dist>(q)` with `q` vector of quantiles, identifies probability distribution functions
- `q<dist>(p)` with `p` vector of probabilities, identifies quantile functions
- `r<dist>(n)` with `n` number of observations, identifies random number generator functions

where `<dist>` indicates the distribution family.

As a result we have: `dweibull(x)`, `pweibull(q)`, `qweibull(p)` and `rweibull(n)` for Weibull distribution and similarly for all other distributions.

Andrea Spanò proposed in `qdist` a functional programming method that computes truncated distribution functions for each distribution chosen without changing the well structured programming style and syntax of the previous functions.

The computational method proposed in `qdist` is based on two R key concepts:

- the environment of a function can be used as a placeholder for other objects (for details see Andrea Spanò article "R package qdist: A functional programming approach to truncated probability distributions")
- function formals can be manipulated

and it is made up of two steps:

1. defines the required functions suitable for computing from a specific truncated probability distribution by using the functions available within its package. These functions take as input a probability distribution as a character string, in this case the Weibull distribution, and return the equivalent truncated distribution function as object:

```r
require(qdist)
dtweibull <- dtruncate("weibull") # density function
ptweibull <- ptruncate("weibull") # probability function
qtweibull <- qtruncate("weibull") # quantile function
rtweibull <- rtruncate("weibull") # random generator function
```

2. uses the newly created functions (`dtweibull()`, `ptweibull()`, `qtweibull()`, `rtweibull()`) for computing from truncated probability distributions.

Let us consider `dtweibull()`. It has the same formals as the original `dweibull()`, plus two extra parameters: `L` and `U`, respectively for lower and upper truncation thresholds set by default to `-Inf` and `+Inf`:

```r
args(dweibull)
```

```
## function (x, shape, scale = 1, log = FALSE)
## NULL
```

```r
args(dtweibull)
```

```
## function (x, shape, scale = 1, log = FALSE, L = -Inf, U = Inf)
## NULL
```

As `dweibull()` is the function that computes the density function for a Weibull distribution, conventionally named `weibull`, `dtweibull()` is the function that computes the density function for a truncated Weibull distribution, named `tweibull`.

Suppose we want to generate the density function for a **truncated and shifted** Weibull distribution. We need `qshift` package. `qshift` package shares the same logic and the same methodology of `qdist`. The difference is that its functions are designed to generate functions for shifted distributions. In particular, `qshift` package includes the following four functions:

- `dshift()`, for density function
- `pshift()`, for distribution function
- `qshift()`, for quantile function

- `rshift()`, for random generator function

So for computing the density function of a trucated and shifted Weibull distribution we have to set `tweibull` distribution as `dshift()` argument:

```r
require(qshift)
dstweibull <- dshift("tweibull")
```

So `dstweibull()` is the function that computes the density function of `stweibull`, which is the truncated and shifted Weibull distribution.

```r
args(dstweibull)
```

```
## function (x, shape, scale = 1, log = FALSE, L = -Inf, U = Inf,
##     shift = 0)
## NULL
```

Looking at the function arguments, we see that the syntax style is that of `dweibull()` function plus `L` and `U` arguments for truncated component of the distribution and `shift` argument for shifted component of the distribution.

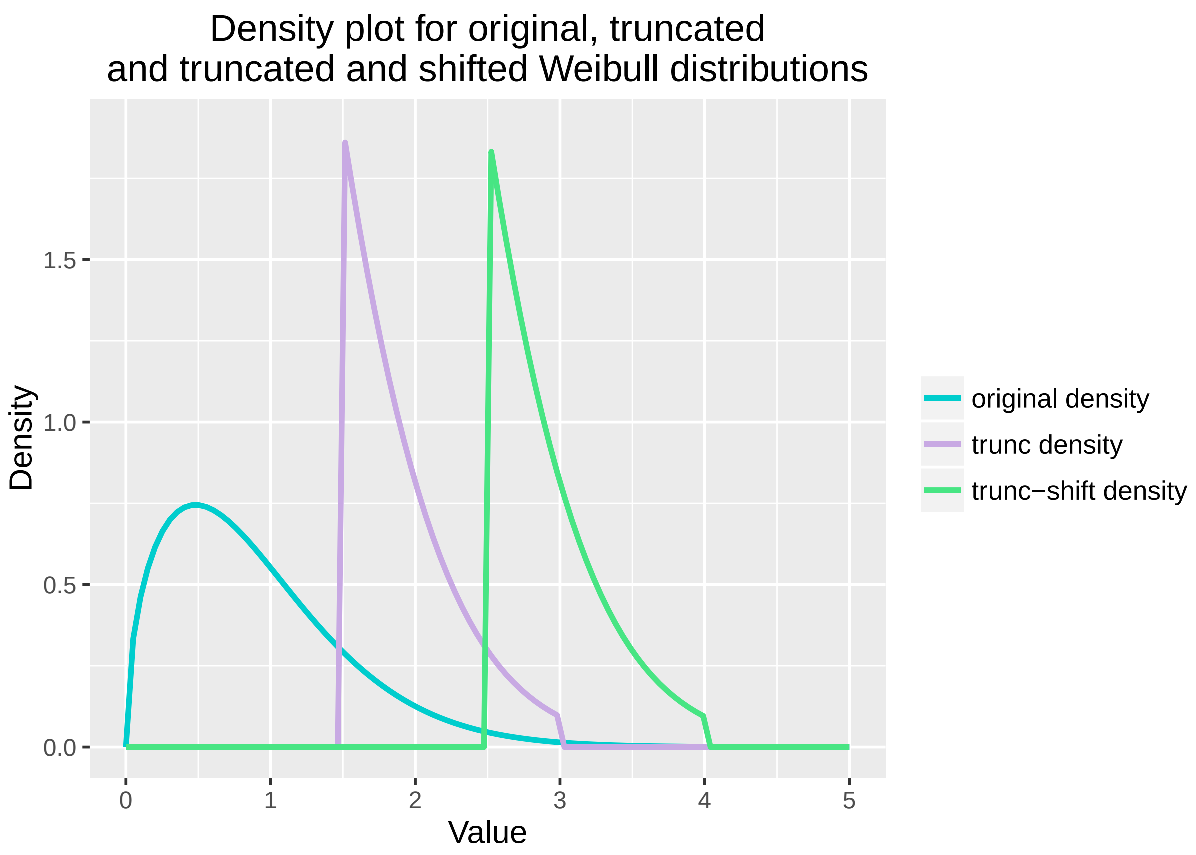
Let us graphically compare the results of `dweibull()`, `dtweibull()` and `dstweibull()` functions:

```r
x <- seq(0, 5, len = 100)
orig_density <- dweibull(x = x, shape = 1.5, scale = 1)
trunc_density <- dtweibull(x = x, shape = 1.5, scale = 1, L = 1.5, U = 3)
trunc_shift_density <- dstweibull(x = x, shape = 1.5, scale = 1, L = 1.5, U = 3, shift = 1)
```

```r
df <- data.frame(x=x,
                 type = factor(c(rep("original density", times=length(x)),
                    rep("trunc density", times=length(x)),
                    rep("trunc-shift density", times=length(x))),
                    levels = c("original density", "trunc density", "trunc-shift density")),
                  value = c(orig_density, trunc_density, trunc_shift_density))

require(ggplot2)
pl <- ggplot(data = df, mapping = aes(x=x, y=value, col=type)) +
  geom_line(size=1) +
  scale_color_manual(values = c("original density" = "cyan3",
                                "trunc density" = "#c8a9e3",
                                "trunc-shift density" = "#47e583")) +
  xlab("Value") + ylab("Density") + ylim(c(0,1.9)) +
  ggtitle(paste("Density plot for original, truncated",
                "and truncated and shifted Weibull distributions", sep = "\n")) +
  theme(legend.title=element_blank(), legend.text= element_text(size=10),
  plot.title = element_text(size=14), axis.title=element_text(size=12))

print(pl)
```



The inverse procedure generates the density function for a **shifted and truncated** Weibull distribution.

Starting from `dshift()` function of `qshift` package, we define `dsweibull()`, the density function of `sweibull`, which is the shifted Weibull distribution:

```
dsweibull <- dshift("weibull") # density function
```

We have to define also the others functions for shifted distribution, as they are implicitly necessary for further step:

```
psweibull <- pshift("weibull") # probability function
qsweibull <- qshift("weibull") # quantile function
rsweibull <- rshift("weibull") # random generator function
```

The density function of a shifted and truncated Weibull distribution is generated setting `sweibull` distribution as `ptruncate()` argument:

```
dtsweibull <- dtruncate("sweibull")
```

So `dtsweibull()` is the function that computes the density function of `tsweibull`, which is the shifted and truncated Weibull distribution.

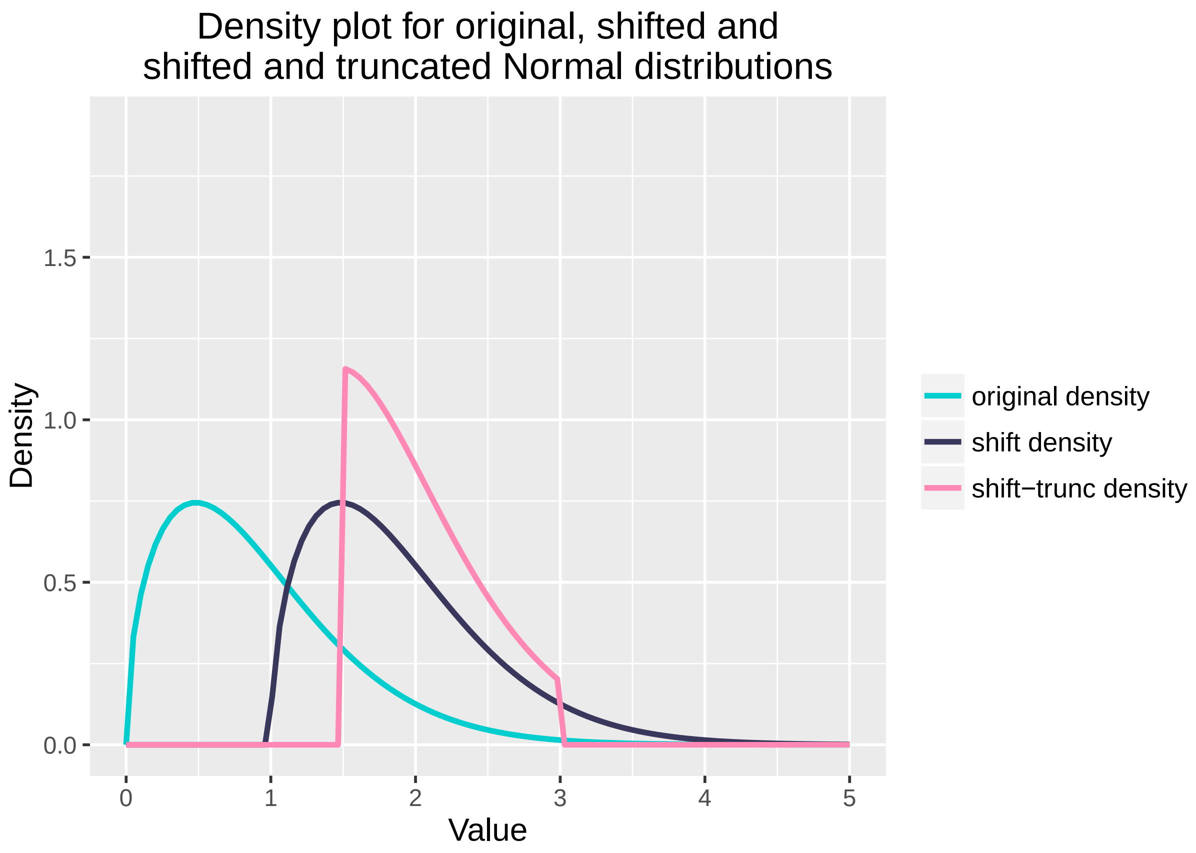
Let us graphically represent the output of the three functions:

```
orig_density <- dweibull(x = x, shape = 1.5, scale = 1)
shift_density <- dsweibull(x = x, shape = 1.5, scale = 1, shift = 1)
shift_trunc_density <- dtsweibull(x = x, shape = 1.5, scale = 1, L = 1.5, U = 3, shift = 1)
```

```r
df <- data.frame(x=x,
                 type = factor(c(rep("original density", times=length(x)),
                    rep("shift density", times=length(x)),
                    rep("shift-trunc density", times=length(x))),
                    levels = c("original density", "shift density", "shift-trunc density")),
                  value = c(orig_density, shift_density, shift_trunc_density))

pl <- ggplot(data = df, mapping = aes(x=x, y=value, col=type)) +
  geom_line(size=1) +
  scale_color_manual(values = c("original density" = "cyan3",
                                "shift density" = "#39375b",
                                "shift-trunc density" = "#FF89B4")) +
  xlab("Value") + ylab("Density") + ylim(c(0,1.9)) +
  ggtitle(paste("Density plot for original, shifted and",
                "shifted and truncated Normal distributions", sep = "\n")) +
  theme(legend.title=element_blank(), legend.text= element_text(size=10),
  plot.title = element_text(size=14), axis.title=element_text(size=12))

print(pl)
```

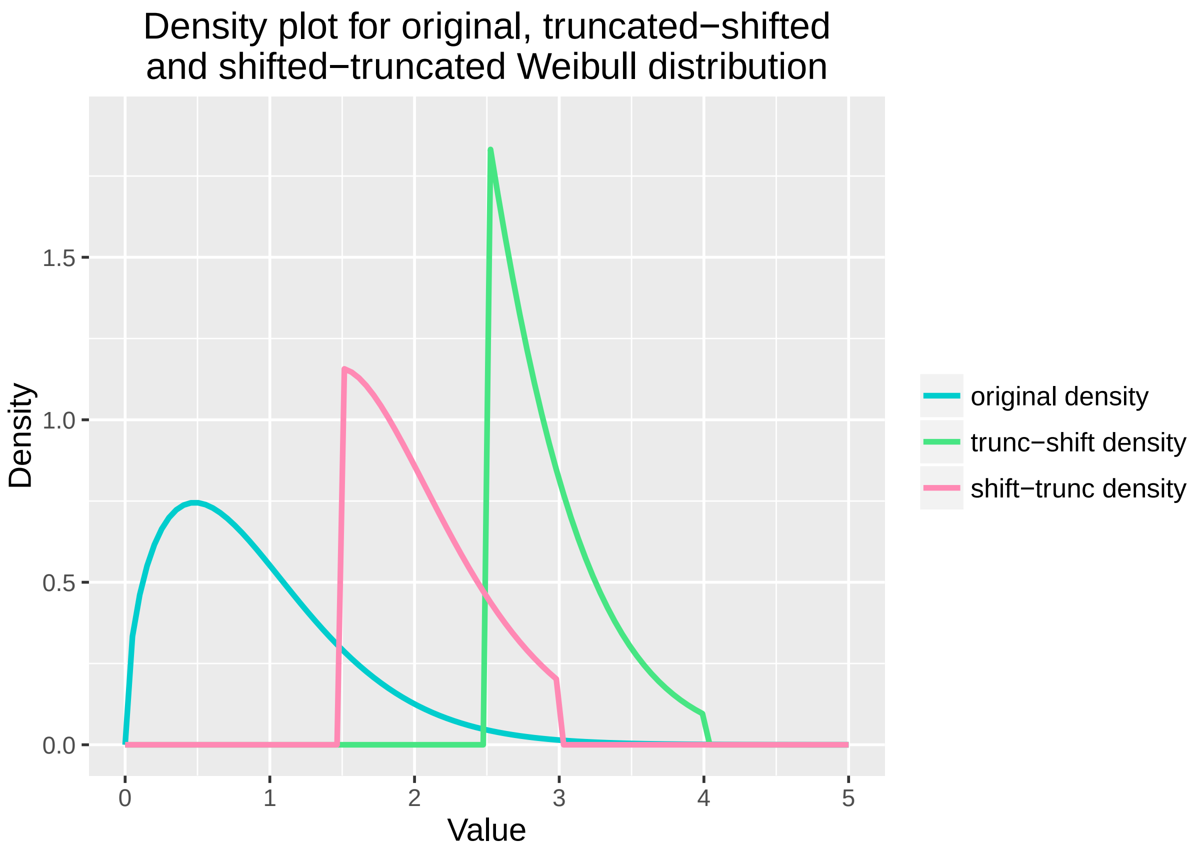


Let us see graphically the differences between the density function of original Weibull distribution, of truncated and shifted Weibull distribution and of shifted and truncated Weibull distribution. We refers to values computed in the previous examples.

```r
df <- data.frame(x=x,
                 type = factor(c(rep("original density", times=length(x)),
                    rep("trunc-shift density", times=length(x)),
                     rep("shift-trunc density", times=length(x))),
                     levels = c("original density", "trunc-shift density", "shift-trunc density")),
                   value = c(orig_density, trunc_shift_density, shift_trunc_density))

pl <- ggplot(data = df, mapping = aes(x=x, y=value, col=type)) +
  geom_line(size=1) +
  scale_color_manual(values = c("original density" = "cyan3",
                                "trunc-shift density" = "#47e583",
                                "shift-trunc density" = "#FF89B4")) +
  xlab("Value") + ylab("Density") + ylim(c(0,1.9)) +
  ggtitle(paste("Density plot for original, truncated-shifted",
                "and shifted-truncated Weibull distribution", sep="\n")) +
  theme(legend.title=element_blank(), legend.text= element_text(size=10),
  plot.title = element_text(size=14), axis.title=element_text(size=12))

print(pl)
```



Looking at the figure we can see that a truncated and shifted distribution is different from a shifted and truncated one.

Let us analyze the other distribution functions:

- probability distribution function of a truncated and shifted Weibull distribution and of a shifted and truncated Weibull distribution:

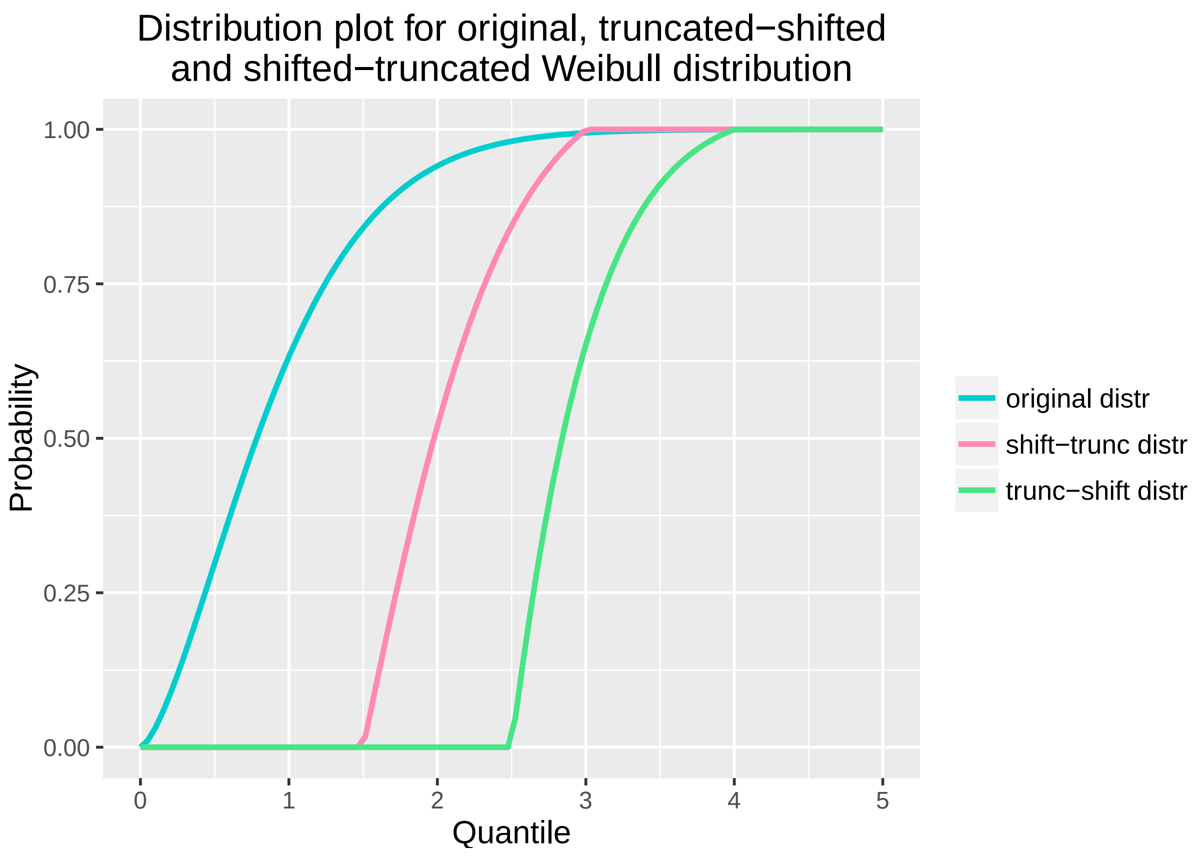
```r
# probability distribution function of a truncated and shifted Weibull distribution
pstweibull <- pshift("tweibull")
# probability distribution function of a shifted and truncated Weibull distribution
ptsweibull <- ptruncate("sweibull")
```

```r
orig_distr <- pweibull(q = x, shape = 1.5, scale = 1)
trunc_shift_distr <- pstweibull(q = x, shape = 1.5, scale = 1, L = 1.5, U = 3, shift = 1)
shift_trunc_distr <- ptsweibull(q = x, shape = 1.5, scale = 1, L = 1.5, U = 3, shift = 1)
```

```r
df <- data.frame(x=x,
                 type = factor(c(rep("original distr", times=length(x)),
                    rep("shift-trunc distr", times=length(x)),
                    rep("trunc-shift distr", times=length(x))),
                    levels = c("original distr", "shift-trunc distr", "trunc-shift distr")),
                  value = c(orig_distr, shift_trunc_distr, trunc_shift_distr))

pl <- ggplot(data = df, mapping = aes(x=x, y=value, col=type)) +
  geom_line(size=1) +
  scale_color_manual(values = c("original distr" = "cyan3",
                                "shift-trunc distr" = "#FF89B4",
                                "trunc-shift distr" = "#47e583")) +
  xlab("Quantile") + ylab("Probability") +
  ggtitle(paste("Distribution plot for original, truncated-shifted",
                "and shifted-truncated Weibull distribution", sep="\n")) +
  theme(legend.title=element_blank(), legend.text= element_text(size=10),
  plot.title = element_text(size=14), axis.title=element_text(size=12))

print(pl)
```



- quantile function of a truncated and shifted Weibull distribution and of a shifted and truncated Weibull distribution:

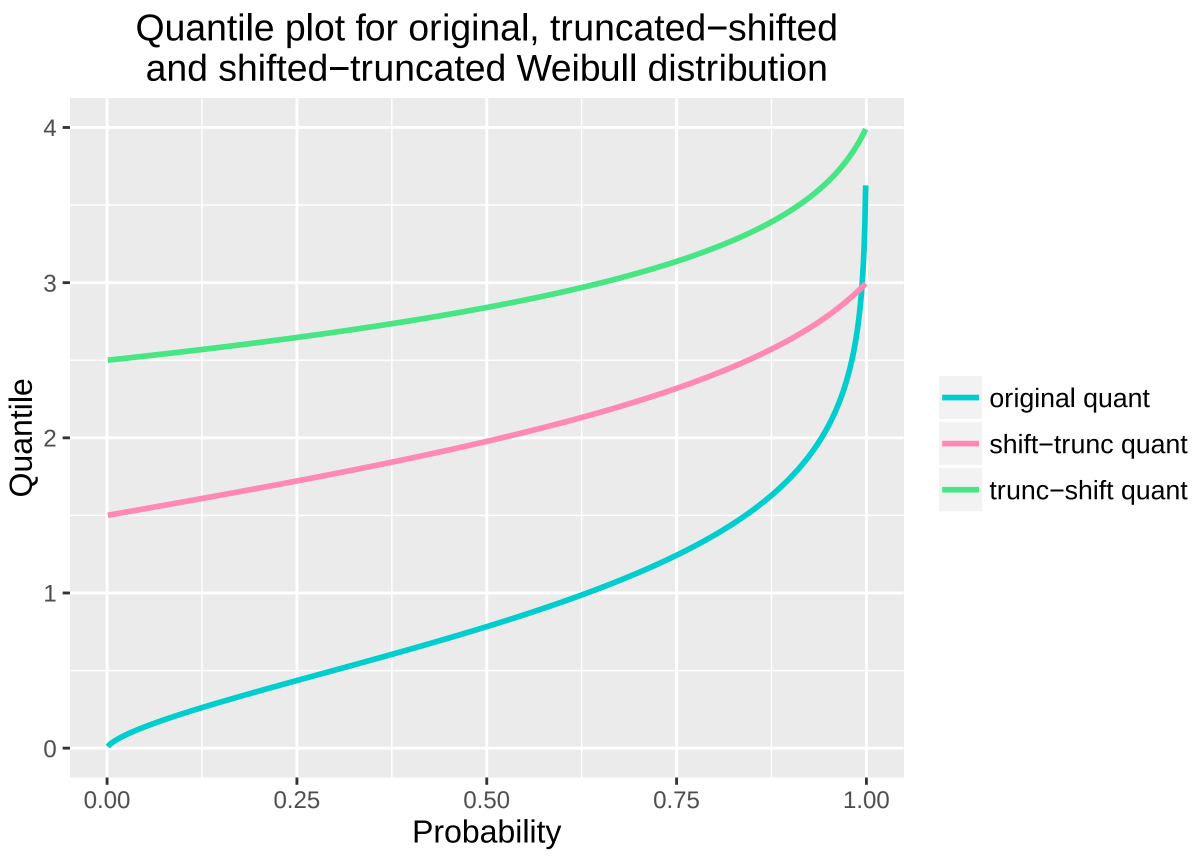
```
# quantile function of a truncated and shifted Weibull distribution
qstweibull <- qshift("tweibull")
# quantile function of a shifted and truncated Weibull distribution
qtsweibull <- qtruncate("sweibull")
```

```
x <- 1:999/1000
orig_quant <- qweibull(p = x, shape = 1.5, scale = 1)
trunc_shift_quant <- qstweibull(p = x, shape = 1.5, scale = 1, L = 1.5, U = 3, shift = 1)
shift_trunc_quant <- qtsweibull(p = x, shape = 1.5, scale = 1, L = 1.5, U = 3, shift = 1)
```

```
df <- data.frame(x=x,
                 type = factor(c(rep("original quant", times=length(x)),
                    rep("shift-trunc quant", times=length(x)),
                    rep("trunc-shift quant", times=length(x))),
                    levels = c("original quant", "shift-trunc quant", "trunc-shift quant")),
                  value = c(orig_quant, shift_trunc_quant, trunc_shift_quant))

pl <- ggplot(data = df, mapping = aes(x=x, y=value, col=type)) +
  geom_line(size=1) +
  scale_color_manual(values = c("original quant" = "cyan3",
                                "shift-trunc quant" = "#FF89B4",
                                "trunc-shift quant" = "#47e583")) +
  xlab("Probability") + ylab("Quantile") +
  ggtitle(paste("Quantile plot for original, truncated-shifted",
                "and shifted-truncated Weibull distribution", sep="\n")) +
  theme(legend.title=element_blank(), legend.text= element_text(size=10),
  plot.title = element_text(size=14), axis.title=element_text(size=12))

print(pl)
```



- random generation function of a truncated and shifted Weibull distribution and of a shifted and truncated Weibull distribution:

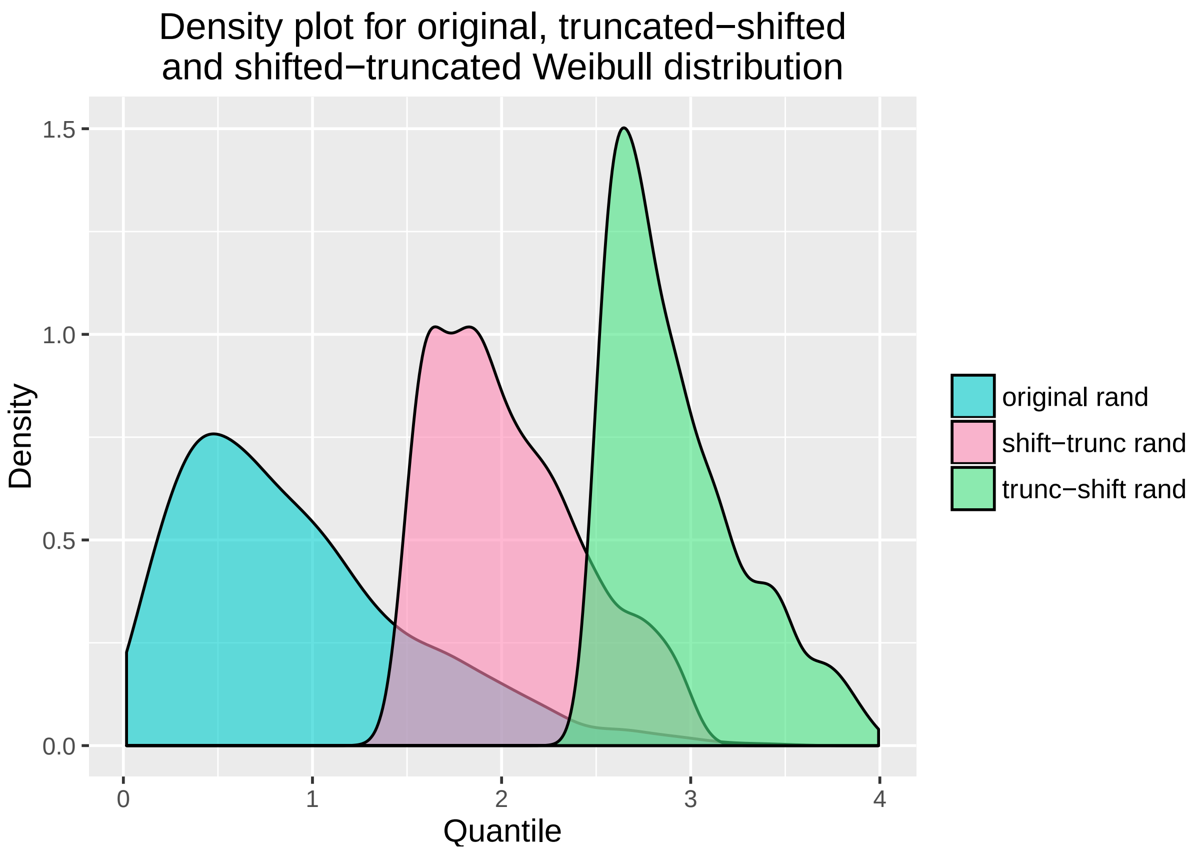
```
# random generation function of a truncated and shifted Weibull distribution
rstweibull <- rshift("tweibull")
# random generation function of a shifted and truncated Weibull distribution
rtsweibull <- rtruncate("sweibull")
```

```
orig_rand <- rweibull(n=1000, shape = 1.5, scale = 1)
trunc_shift_rand <- rstweibull(n=1000, shape = 1.5, scale = 1, L = 1.5, U = 3, shift = 1)
shift_trunc_rand <- rtsweibull(n=1000, shape = 1.5, scale = 1, L = 1.5, U = 3, shift = 1)
```

```
df <- data.frame(x=1:1000,
                 type = factor(c(rep("original rand", times=1000),
                    rep("shift-trunc rand", times=1000),
                    rep("trunc-shift rand", times=1000)),
                    levels = c("original rand", "shift-trunc rand", "trunc-shift rand")),
                  value = c(orig_rand, shift_trunc_rand, trunc_shift_rand))
```

```r
pl <- ggplot(data = df, mapping = aes(x=value, group=type, fill=type)) +
  geom_density(alpha=0.6) +
  scale_fill_manual(values = c("original rand" = "cyan3",
                               "shift-trunc rand" = "#FF89B4",
                               "trunc-shift rand" = "#47e583")) +
  ggtitle(paste("Density plot for original, truncated-shifted",
                "and shifted-truncated Weibull distribution", sep="\n")) +
  xlab("Quantile") + ylab("Density") +
  theme(legend.title=element_blank(), legend.text= element_text(size=10),
  plot.title = element_text(size=14), axis.title=element_text(size=12))

print(pl)
```



Extending the computation

Once we demonstrated that we can generate distribution functions for truncated and shifted and for shifted and truncated distributions, we can use them for further implementations. Suppose we want to define a function for maximum likelihood estimate for truncated and shifted distributions and one for shifted and truncated distributions. In the following examples, we will continue to consider the Weibull distribution.

Let us start from the function for maximum likelihood estimate for truncated and shifted Weibull distribution. We consider the previously defined density function, `dstweibull()`, for a truncated and shifted Weibull distribution and we use it inside the estimator function:

```r
lstweibull <- function(x, shift = 0 ,L = -Inf, U = Inf){
  x1 <- x - shift
  # starting values for parameters (scale, shape)
  shape <- (sd(x1)/mean(x1))^(-1.086)
  scale <- mean(x1)/gamma(1+1/shape)
  # parameters vector definition
  theta <- c(shape, scale)
  # likelihood function
  ll <- function(theta, x, shift = 0, L = -Inf, U = Inf){
    shape <- theta[1]
    scale <- theta[2]
    ld <- dstweibull(x = x, shape = shape, scale = scale, shift = shift, L = L, U = U, log=TRUE)
    -sum(ld)
  }
  # maximum likelihood estimation
  optim(par = theta, fn = ll, x = x, shift=shift, L = L, U = U, method = "Nelder-Mead")[["par"]]
}
```

Let's test it to see how it works.
We consider the previously defined random generator function `rstweibull()` for a truncated and shifted Weibull distribution to create the sample for testing the maximum likelihood estimate function:

```r
x <- rstweibull(n = 100000, shape = 1.6, scale = 1, shift = 1, L = 1, U = 2.5)
# maximum likelihood estimate for the shifted and truncated weibull distribution
lstweibull(x = x, shift=1, L = 1, U = 2.5)
```

```
## [1] 1.615796 1.006976
```

The maximum likelihood estimate function produces a good estimate of the scale and shape parameters of truncated and shifted Weibull distribution.

Let us continue with the function for maximum likelihood estimate for shifted and truncated Weibull distribution.

We will consider the previously defined density function, `dtsweibull()`, for a shifted and truncated Weibull distribution and we use it inside the estimator function:

```r
ltsweibull <- function(x, shift = 0, L = -Inf, U = Inf){
  x1 <- x - shift
  # starting values for parameters (scale, shape)
  shape <- (sd(x1)/mean(x1))^(-1.086)
  scale <- mean(x1)/gamma(1+1/shape)
  # parameters vector definition
  theta <- c(shape, scale)
  # likelihood function
  ll <- function(theta, x, L = -Inf, U = Inf){
    shape <- theta[1]
    scale <- theta[2]
    ld <- dtsweibull(x = x, shape = shape, scale = scale, shift = shift, L = L, U = U, log=T)
    -sum(ld)
  }
  # maximum likelihood estimation
  optim(par = theta, fn = ll, x = x, L = L, U = U, method = "Nelder-Mead")[["par"]]
}
```

Let's test it to see how it works.
We consider the previously defined random generator function `rtsweibull()` for a shifted and truncated Weibull distribution to create the sample for testing the maximum likelihood estimate function:

```
x <- rtsweibull(n = 100000, shape = 1.6, scale = 1, shift = 1, L = 0.5, U = 3)
# maximum likelihood estimate for the shifted weibull distribution
ltsweibull(x = x, shift = 1, L = 0.5, U = 3)
```

```
## [1] 1.6106247 0.9964173
```

Also in this case, the maximum likelihood estimate function produces a good estimate of the scale and shape parameters of shifted and truncated Weibull distribution.

Bibliography

Giro V. . R package qshift: A functional programming approach to shifted probability ditributions. url
Spanò A. . R package qdist: A functional programming approach to truncated probability ditributions. url
Spanò A. . Ramarro. R for Developers. http://www.quantide.com/ramarro-r-for-developers/. Accessed: 2016-09-20
Wickham H. . Advanced r. http://adv-r.had.co.nz/. Accessed: 2016-09-20.