R package qshift: A functional programming approach to shifted probability distributions

Introduction

Shifted distributions are used daily in many statistical fields.
Given a random variable X with distribution function $F(x)$, a shifted random variable Y from X is a random variable with a constant, k, added, defined as $Y = X + k$, with distribution function $F(x - k)$. Y support changes according to shift value.

The aim of this paper is to describe an R package, `qshift` which generates shifted distribution functions for any distribution chosen. The package was realized using a functional programming approach and inspires its computational method to Andrea Spanò article "R package `qdist`: A functional programming approach to truncated probability distributions", which provides an alternative method for computing from truncated probability distributions using a functional programming approach.

Shifted probability functions in R

R does not provide a specific package for shifted distribution functions, but it provides package `stats` which contains a wide set of functions for standard distributions: probability, density, quantile and random number generation functions. All them share a stable naming convention both for the names of the functions and in the first argument of these functions. In particular:

- density functions: `d<dist>(x)` with `x` vector of quantiles;
- probability distribution functions: `p<dist>(q)` with `q` vector of quantiles;
- quantile functions: `q<dist>(p)` with `p` vector of probabilities;
- random number generation: `r<dist>(n)` with `n` number of observations.

where `<dist>` indicates the conventional name of distribution family.

Let us see an example.
Considering the Normal distribution (`norm`), we have: `dnorm(x)`, `pnorm(q)`, `qnorm(p)` and `rnorm(n)`.

So, we use to write:

```
dnorm(x = 6:10, mean = 8, sd = 1)
```

```
## [1] 0.05399097 0.24197072 0.39894228 0.24197072 0.05399097
```

to get density values at 6:10 from a Normal distribution with parameters `mean = 8` and `sd = 1`.

Going back to shifted distribution, we could easily write a shifted version for `dnorm()`, as:

```
shift_dnorm <- function(x, mean = 0, sd = 1, shift = 0, ...){
            x_shifted <- x - shift
            density <- dnorm(x = x_shifted, mean = mean, sd = sd, ...)
            return(density)
}
```

So we replicate the previous example but shifting the Normal distribution of 1 unit:

```r
shift_dnorm(x = 6:10, mean = 8, sd = 1, shift = 1)
```

```
## [1] 0.004431848 0.053990967 0.241970725 0.398942280 0.241970725
```

At first view, this approach seems easy and cool but thinking about it, we realize that we would need to write a different function for each probability distribution we would like to work with. Functional programming approach could be a solution, as Andrea Spanò demonstrates in his article (see "R package `qdist`: A functional programming approach to truncated probability distributions" for more details).

`qshift` package includes four general functions which compute shifted distribution functions for any distribution chosen:

- `dshift()` for shifted density functions;
- `pshift()` for shifted probability distributions;
- `qshift()` for shifted quantile functions;
- `rshift()` for shifted random numbers generation functions.

`qshift` package is available at https://github.com/veronicagiro/qshift.

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

The four functions share the same logic: they take as input a probability distribution name, as a character string, and return the equivalent shifted distribution as a function object.

```r
require(qshift)
dsnorm <- dshift("norm") # density function
psnorm <- pshift("norm") # probability function
qsnorm <- qshift("norm") # quantile function
rsnorm <- rshift("norm") # random generation function
```

Talking about the computational process, package `qshift` is based on two key concepts being part of the functional nature of the R programming language:

- the environment of a function can be used as a placeholder for other objects;
- function formals can be manipulated.

For example, let us consider `dshift()`. As we seen, it takes as input the distribution name (`norm`) and proceeds as follows:

- gets the corresponding function `dnorm()`;
- uses `dnorm()` to create a function, say `density()` that computes shifted density for normal distributions;
- modify the formals of `density()` so that it has the same formals as `dnorm()` plus `shift`, corresponding to the shift value;
- returns `density()`.

So, the function `dsnorm()` has the same formals of the function `dnorm()`, plus the parameter `shift` which indicates the shift value to be applied to distribution; by default it is set as 0.

```r
args(dsnorm)
```

```
## function (x, mean = 0, sd = 1, log = FALSE, shift = 0)
## NULL
```

```r
args(dnorm)
```

```
## function (x, mean = 0, sd = 1, log = FALSE)
## NULL
```

Other references are in chapters Environments and Functions of Andrea Spanò weeb Book Ramarro, available on Quantide web site.

Examples

Let us consider the Normal distribution and the previously defined functions `dsnorm()`, `psnorm()`, `qsnorm()` and `rsnorm()`.

- Density function, `dsnorm()`.
  This function has a double use:
  - generate probability values from a non-shifted normal distribution by leaving parameter `shift` set to its default (`shift = 0`)
  - generate probability values from any shifted normal distribution by setting `shift` value

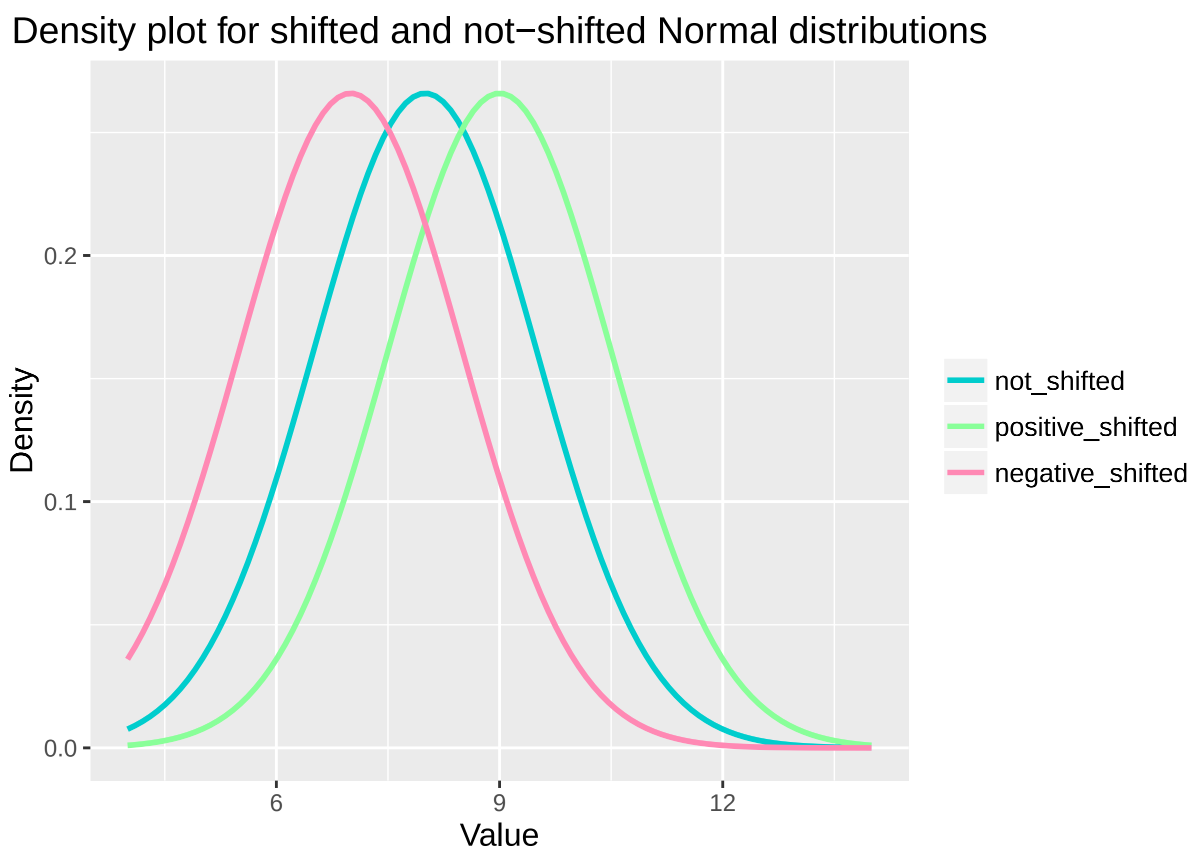
  ```r
  x <- seq(4, 14, len = 100)
  not_shifted <- dsnorm(x = x, mean = 8, sd = 1.5)
  positive_shifted <- dsnorm(x = x, mean = 8, sd = 1.5, shift = 1)
  negative_shifted <- dsnorm(x = x, mean = 8, sd = 1.5, shift = -1)
  ```

  We can visualize these results by plotting them:

  ```r
  require(ggplot2)
  df <- data.frame(x=x,
               dsnorm_type = factor(c(rep("not_shifted", times=length(x)),
                  rep("positive_shifted", times=length(x)),
                  rep("negative_shifted", times=length(x))),
                  levels = c("not_shifted", "positive_shifted", "negative_shifted")),
                value = c(not_shifted, positive_shifted, negative_shifted))

  pl <- ggplot(data = df, mapping = aes(x=x, y=value, col=dsnorm_type)) +
    geom_line(size=1) +
    scale_color_manual(values = c("not_shifted" = "cyan3",
                                  "positive_shifted" = "#89FF99",
                                  "negative_shifted" = "#FF89B4")) +
    xlab("Value") + ylab("Density") +
    ggtitle("Density plot for shifted and not-shifted Normal distributions") +
    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)
  ```



- Probability function, `spnorm()`:

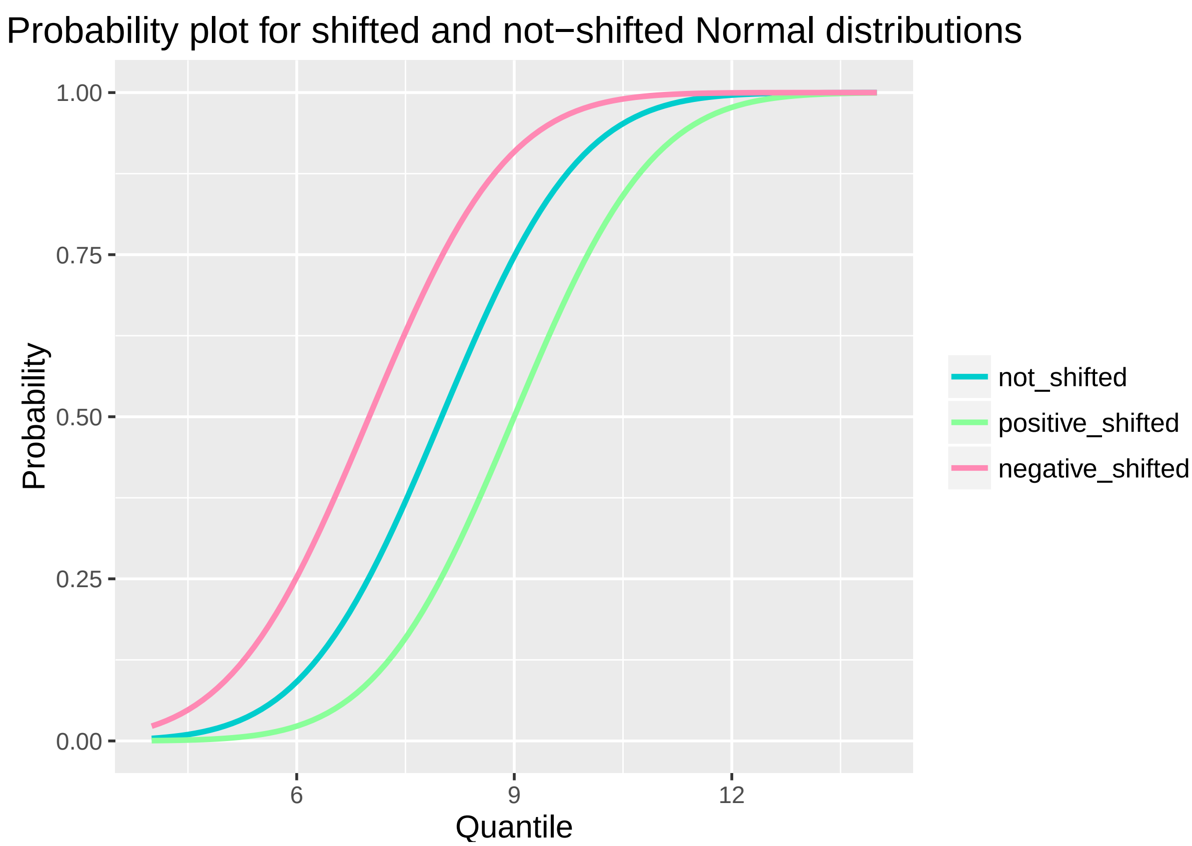
```
not_shifted <- psnorm(q = x, mean = 8, sd = 1.5)
positive_shifted <- psnorm(q = x, mean = 8, sd = 1.5, shift = 1)
negative_shifted <- psnorm(q = x, mean = 8, sd = 1.5, shift = -1)
```

We can visualize these results by plotting them:

```
df <- data.frame(x=x,
             psnorm_type = factor(c(rep("not_shifted", times=length(x)),
                rep("positive_shifted", times=length(x)),
                rep("negative_shifted", times=length(x))),
                levels = c("not_shifted", "positive_shifted", "negative_shifted")),
              value = c(not_shifted, positive_shifted, negative_shifted))

pl <- ggplot(data = df, mapping = aes(x=x, y=value, col=psnorm_type)) +
  geom_line(size=1) +
  scale_color_manual(values = c("not_shifted" = "cyan3",
                            "positive_shifted" = "#89FF99",
                            "negative_shifted" = "#FF89B4")) +
  xlab("Quantile") + ylab("Probability") +
  ggtitle("Probability plot for shifted and not-shifted Normal distributions") +
  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, `qsnorm()`:

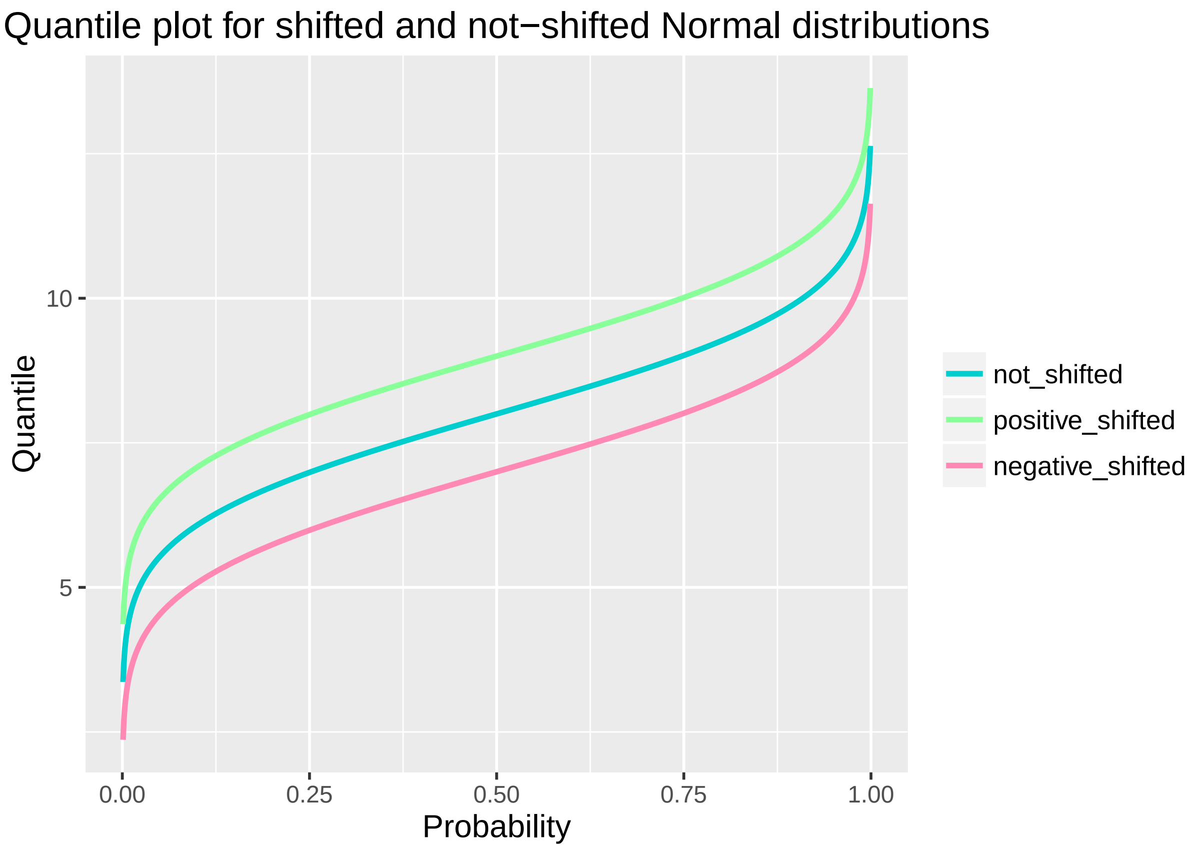
```
x <- (1:999)/1000
not_shifted <- qsnorm(p = x, mean = 8, sd = 1.5)
positive_shifted <- qsnorm(p = x, mean = 8, sd = 1.5, shift = 1)
negative_shifted <- qsnorm(p = x, mean = 8, sd = 1.5, shift = -1)
```

We can visualize these results by plotting them:

```
df <- data.frame(x=x,
             qsnorm_type = factor(c(rep("not_shifted", times=length(x)),
                rep("positive_shifted", times=length(x)),
                rep("negative_shifted", times=length(x))),
                levels = c("not_shifted", "positive_shifted", "negative_shifted")),
              value = c(not_shifted, positive_shifted, negative_shifted))

pl <- ggplot(data = df, mapping = aes(x=x, y=value, col=qsnorm_type)) +
  geom_line(size=1) +
  scale_color_manual(values = c("not_shifted" = "cyan3",
                                "positive_shifted" = "#89FF99",
                                "negative_shifted" = "#FF89B4")) +
  ylab("Quantile") + xlab("Probability") +
  ggtitle("Quantile plot for shifted and not-shifted Normal distributions") +
  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, `rsnorm()`:

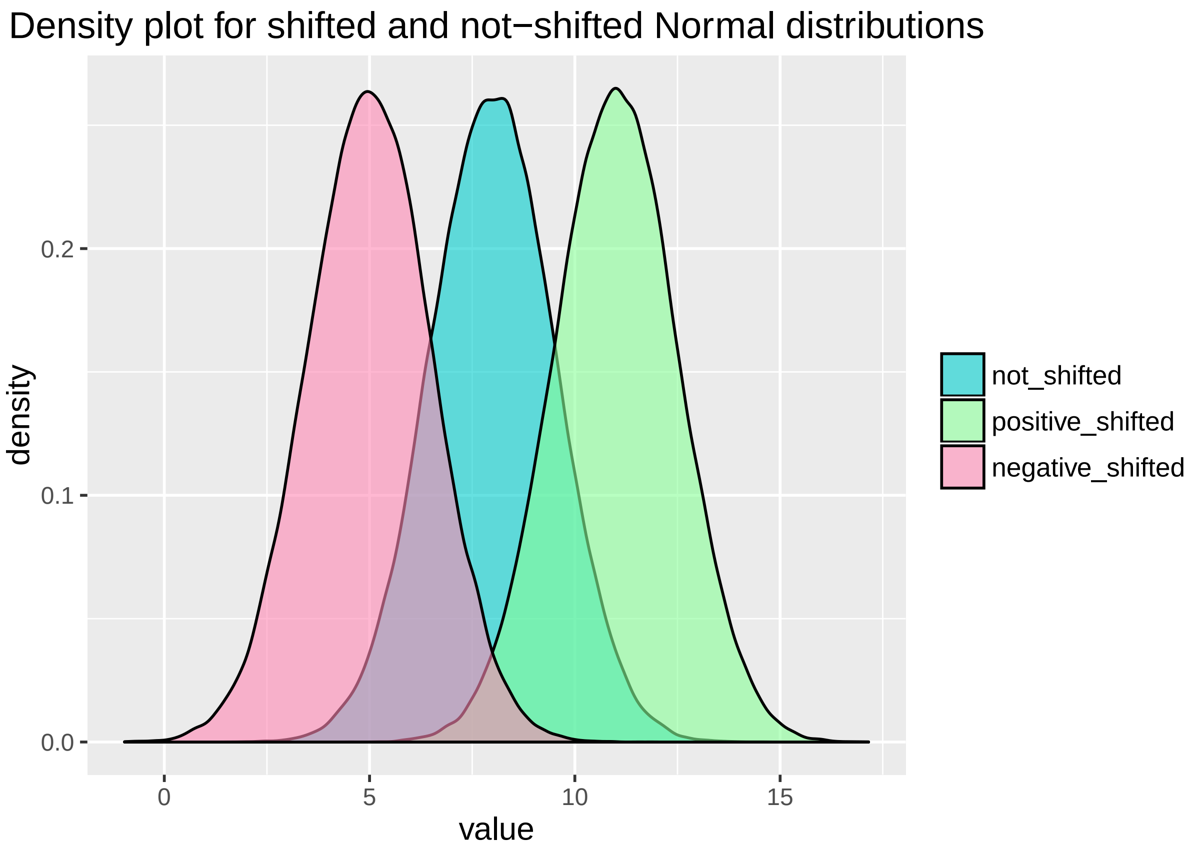
```
x <- 100000
not_shifted <- rsnorm(n = x, mean = 8, sd = 1.5)
positive_shifted <- rsnorm(n = x, mean = 8, sd = 1.5, shift = 3)
negative_shifted <- rsnorm(n = x, mean = 8, sd = 1.5, shift = -3)
```

We can visualize these results by plotting them:

```
df <- data.frame(rsnorm_type = factor(c(rep("not_shifted", times=x),
                rep("positive_shifted", times=x),
                rep("negative_shifted", times=x)),
                levels = c("not_shifted", "positive_shifted", "negative_shifted")),
              value = c(not_shifted, positive_shifted, negative_shifted))

pl <- pl <- ggplot(data = df, mapping = aes(x=value, group=rsnorm_type, fill=rsnorm_type)) +
  geom_density(alpha=0.6) +
scale_fill_manual(values = c("not_shifted" = "cyan3",
                            "positive_shifted" = "#89FF99",
                            "negative_shifted" = "#FF89B4")) +
  ggtitle("Density plot for shifted and not-shifted Normal distributions") +
  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)
```



As expected, the Normal shifted distribution is a Normal distribution with changed mean. The change in the mean is equal to the shift.

As we know, Normal distribution support goes from $-\infty$ to $+\infty$. What happens considerig a distribution with a limited support?
Let us consider the Uniform distribution.

First of all we have to define its probability functions:

```
dsunif <- dshift("unif")
psunif <- pshift("unif")
qsunif <- qshift("unif")
rsunif <- rshift("unif")
```

We limited Uniform support between 6 and 12.

- Density function, `dsunif()`:

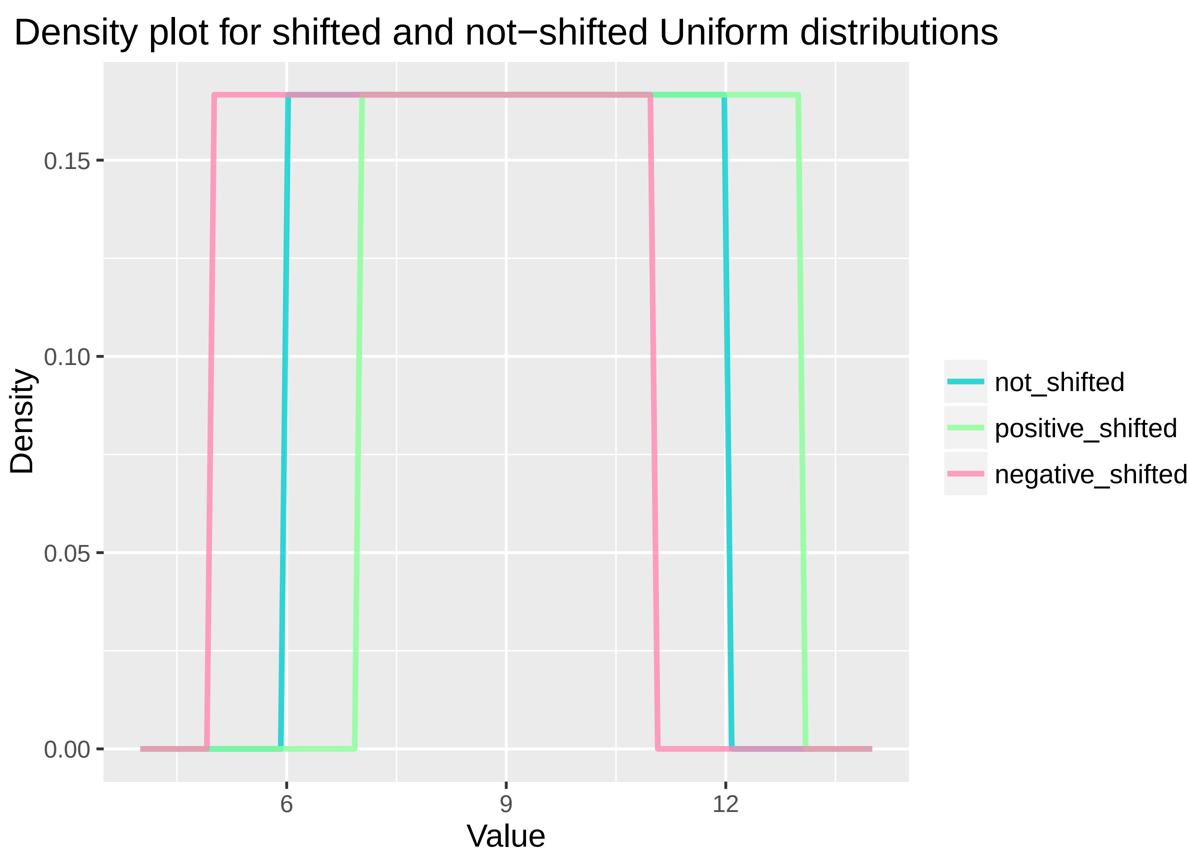
  ```
  x <- seq(4, 14, len = 100)
  not_shifted <- dsunif(x = x, min = 6, max = 12)
  positive_shifted <- dsunif(x = x, min = 6, max = 12, shift = 1)
  negative_shifted <- dsunif(x = x, min = 6, max = 12, shift = -1)
  ```

  We can visualize these results by plotting them:

```r
df <- data.frame(x=x,
             dsunif_type = factor(c(rep("not_shifted", times=length(x)),
                rep("positive_shifted", times=length(x)),
                rep("negative_shifted", times=length(x))),
                levels = c("not_shifted", "positive_shifted", "negative_shifted")),
              value = c(not_shifted, positive_shifted, negative_shifted))

pl <- ggplot(data = df, mapping = aes(x=x, y=value, col=dsunif_type)) +
  geom_line(size=1, alpha =0.8) +
  scale_color_manual(values = c("not_shifted" = "cyan3",
                                "positive_shifted" = "#89FF99",
                                "negative_shifted" = "#FF89B4")) +
  xlab("Value") + ylab("Density") +
  ggtitle("Density plot for shifted and not-shifted Uniform distributions") +
  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)
```



- Probability function, `psunif()`:

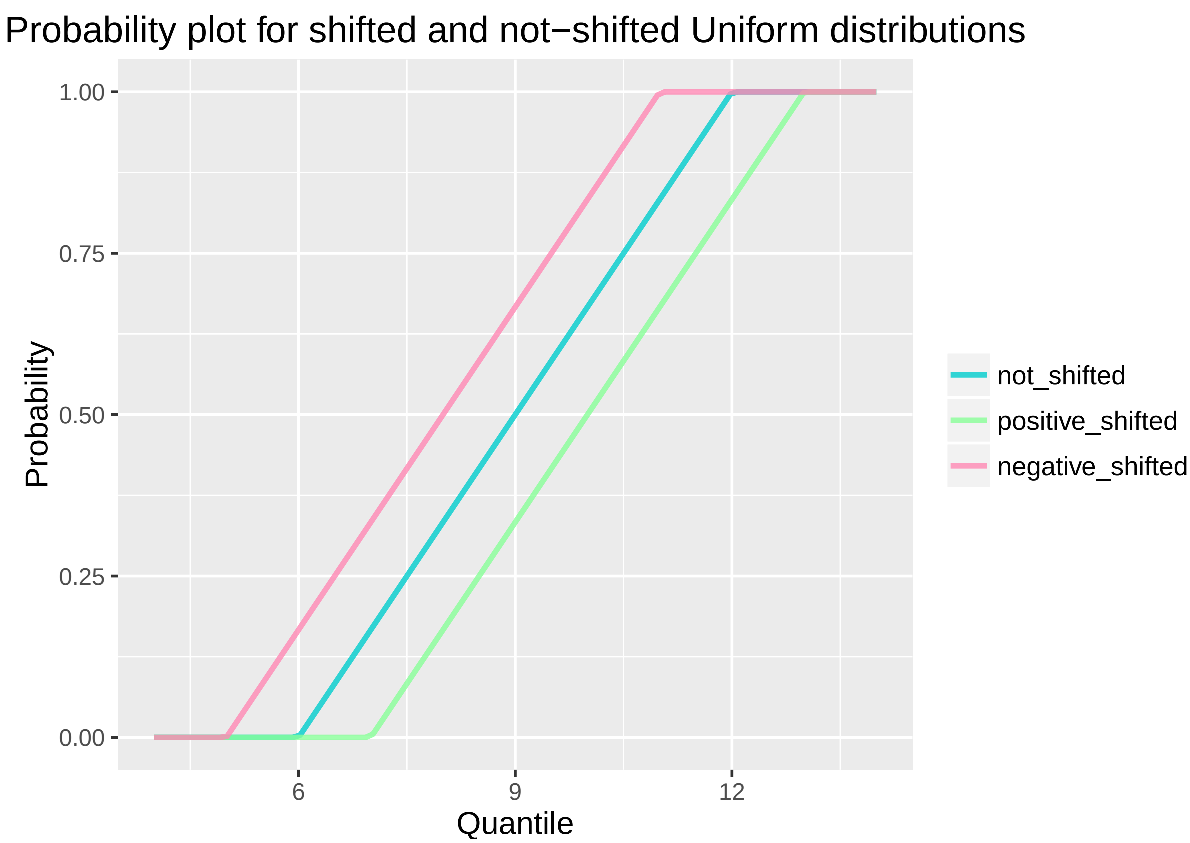
```r
not_shifted <- psunif(q = x, min = 6, max = 12)
positive_shifted <- psunif(q = x, min = 6, max = 12, shift = 1)
negative_shifted <- psunif(q = x, min = 6, max = 12, shift = -1)
```

We can visualize these results by plotting them:

```r
df <- data.frame(x=x,
             psunif_type = factor(c(rep("not_shifted", times=length(x)),
                rep("positive_shifted", times=length(x)),
                rep("negative_shifted", times=length(x))),
                levels = c("not_shifted", "positive_shifted", "negative_shifted")),
              value = c(not_shifted, positive_shifted, negative_shifted))

pl <- ggplot(data = df, mapping = aes(x=x, y=value, col=psunif_type)) +
  geom_line(size=1, alpha =0.8) +
  scale_color_manual(values = c("not_shifted" = "cyan3",
                          "positive_shifted" = "#89FF99",
                          "negative_shifted" = "#FF89B4")) +
  xlab("Quantile") + ylab("Probability") +
  ggtitle("Probability plot for shifted and not-shifted Uniform distributions") +
  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, `qsunif()`:

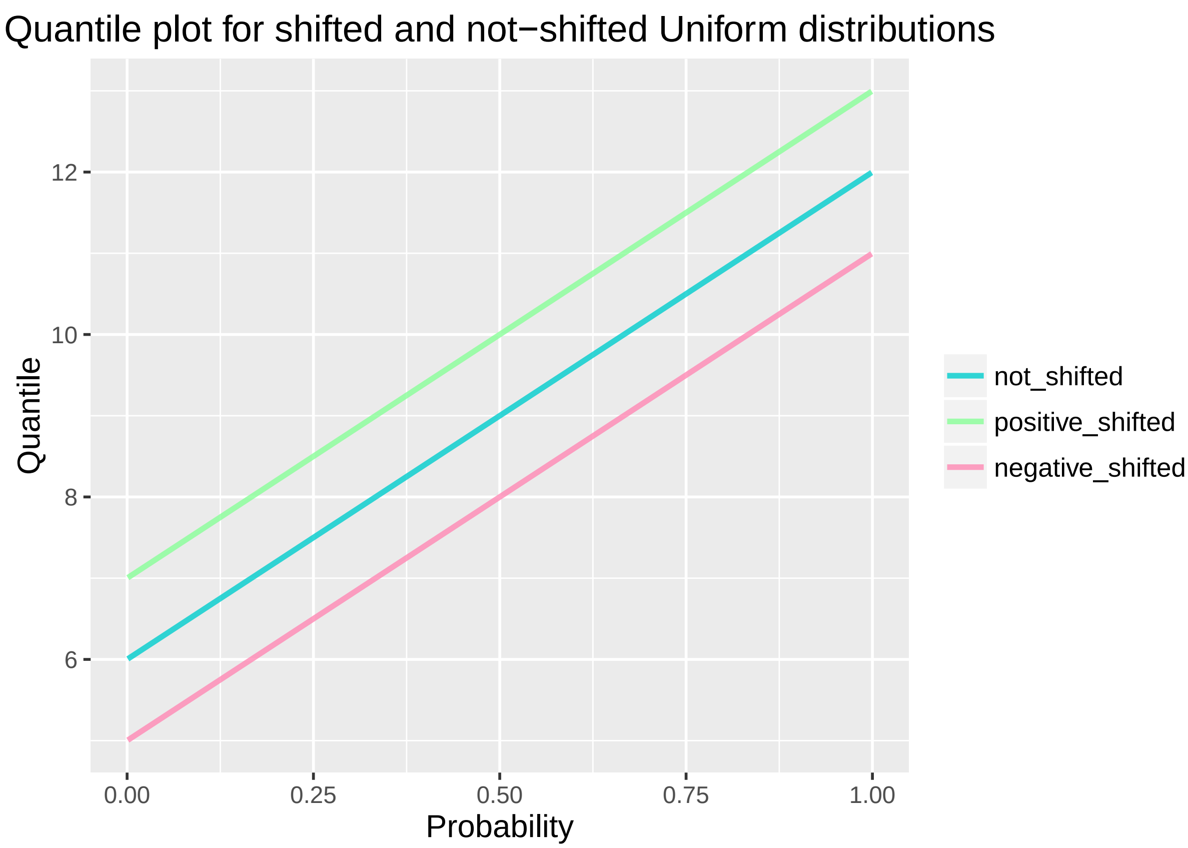
```r
x <- (1:999)/1000
not_shifted <- qsunif(p = x, min = 6, max = 12)
positive_shifted <- qsunif(p = x, min = 6, max = 12, shift = 1)
negative_shifted <- qsunif(p = x, min = 6, max = 12, shift = -1)
```

We can visualize these results by plotting them:

```
df <- data.frame(x=x,
             qsunif_type = factor(c(rep("not_shifted", times=length(x)),
                rep("positive_shifted", times=length(x)),
                rep("negative_shifted", times=length(x))),
                levels = c("not_shifted", "positive_shifted", "negative_shifted")),
              value = c(not_shifted, positive_shifted, negative_shifted))

pl <- ggplot(data = df, mapping = aes(x=x, y=value, col=qsunif_type)) +
  geom_line(size=1, alpha =0.8) +
  scale_color_manual(values = c("not_shifted" = "cyan3",
                                "positive_shifted" = "#89FF99",
                                "negative_shifted" = "#FF89B4")) +
  ylab("Quantile") + xlab("Probability") +
  ggtitle("Quantile plot for shifted and not-shifted Uniform distributions") +
  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, `rsunif()`:

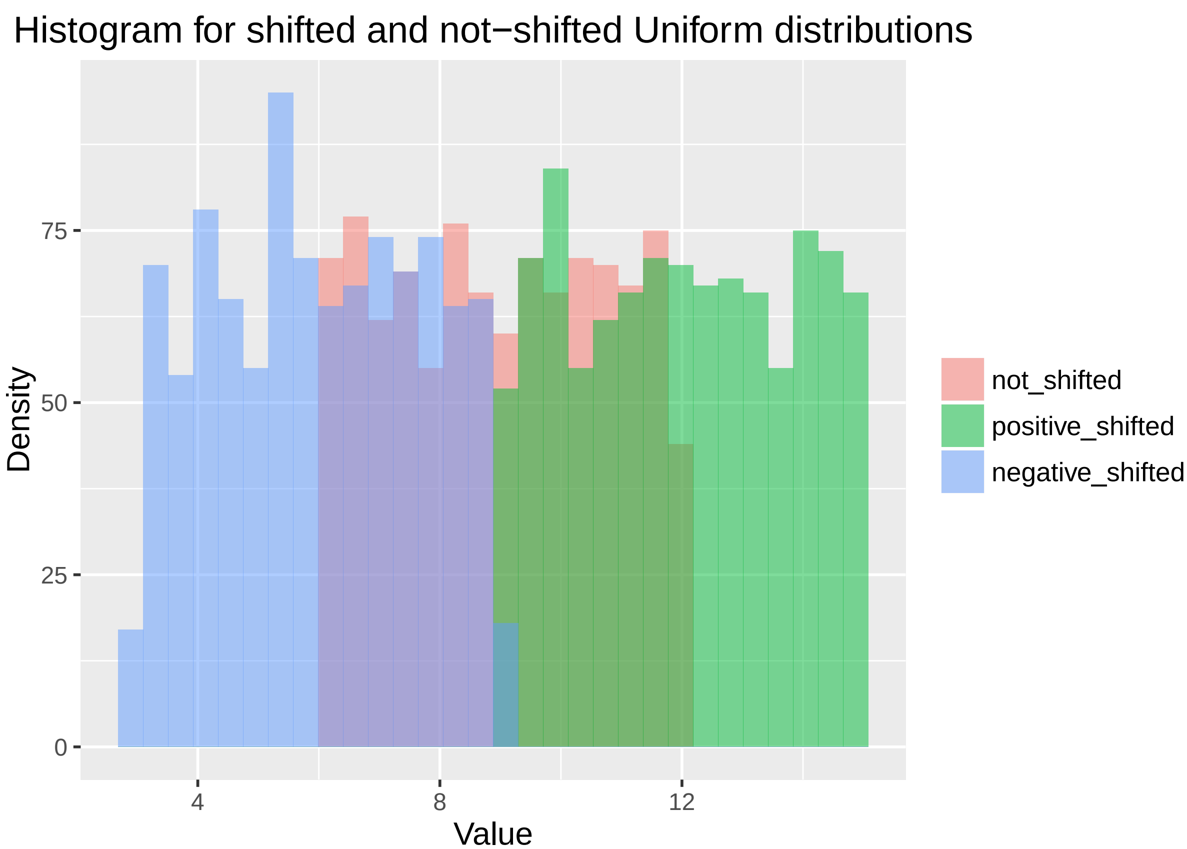
```
x <- 1000
not_shifted <- rsunif(n = x, min = 6, max = 12)
positive_shifted <- rsunif(n = x, min = 6, max = 12, shift = 3)
negative_shifted <- rsunif(n = x, min = 6, max = 12, shift = -3)
```

We can visualize these results by plotting them:

```r
df <- data.frame(rsunif_type = factor(c(rep("not_shifted", times=x),
                rep("positive_shifted", times=x),
                rep("negative_shifted", times=x)),
                levels = c("not_shifted", "positive_shifted", "negative_shifted")),
              value = c(not_shifted, positive_shifted, negative_shifted))


pl <- ggplot(data = df, mapping = aes(x=value, fill=rsunif_type, group=rsunif_type)) +
  geom_histogram(alpha=0.5, bins = 30, position="identity") +
scale_color_manual(values = c("not_shifted" = "cyan3",
                              "positive_shifted" = "#89FF99",
                              "negative_shifted" = "#FF89B4")) +
  xlab("Value") + ylab("Density") +
  ggtitle("Histogram for shifted and not-shifted Uniform distributions") +
  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)
```



As we see, Uniform support changes according to the transformation (the shift) applied to the distribution.

Extending the computation

Further implementations are possible starting from the definition of a shifted distribution.
Suppose we want to define a function for maximum likelihood estimate for the shifted Weibull distribution.

Firstly let us define the shifted density function for the Weibull distribution:

```
dsweibull <- dshift("weibull")
```

and, subsequently, by using `dsweibull()` within an estimator function:

```
lsweibull <- function(x){
  # starting values for parameters (scale, shape) and shift
  shift <- min(x) - 0.01
  x1 <- x - shift
  shape <- (sd(x1)/mean(x1))^(-1.086)
  scale <- mean(x1)/gamma(1+1/shape)
  # parameters vector definition
  theta <- c(shape, scale, shift)
  # likelihood function
  ll <- function(theta, x){
    shape <- theta[1]
    scale <- theta[2]
    shift <- theta[3]
    ld <- dsweibull(x = x, shape = shape, scale = scale, shift = shift, log = TRUE)
    -sum(ld)
  }
  # maximum likelihood estimation
  optim(par = theta, fn = ll, x = x)[["par"]]
}
```

Let us see how it works on a random generated sample from a shifted Weibull distribution:

```
# generate a random sample from a shifted weibull distribution
rsweibull <- rshift("weibull")
x <- rsweibull(n = 100000, shape = 1, scale = 5, shift = 1)

# maximum likelihood estimate for the shifted weibull distribution
lsweibull(x = x)
```

```
## [1] 1.000621 5.026303 1.000032
```

Bibliography

Pace L., Salvan A. . *Introduzione alla Statistica.* CEDAM. 2010

Spanò A. . *R package qdist: A functional programming approach to truncated probability ditributions.* url

Wickham H. . *Advanced r.* http://adv-r.had.co.nz/. Accessed: 2016-08-24.