rep1

StaronKacper

3/3/2020

qqplot

```
library(qqplotr)
```

```
## Loading required package: ggplot2
```

```
##
## Attaching package: 'qqplotr'
```

```
## The following objects are masked from 'package:ggplot2':
##
##     stat_qq_line, StatQqLine
```

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

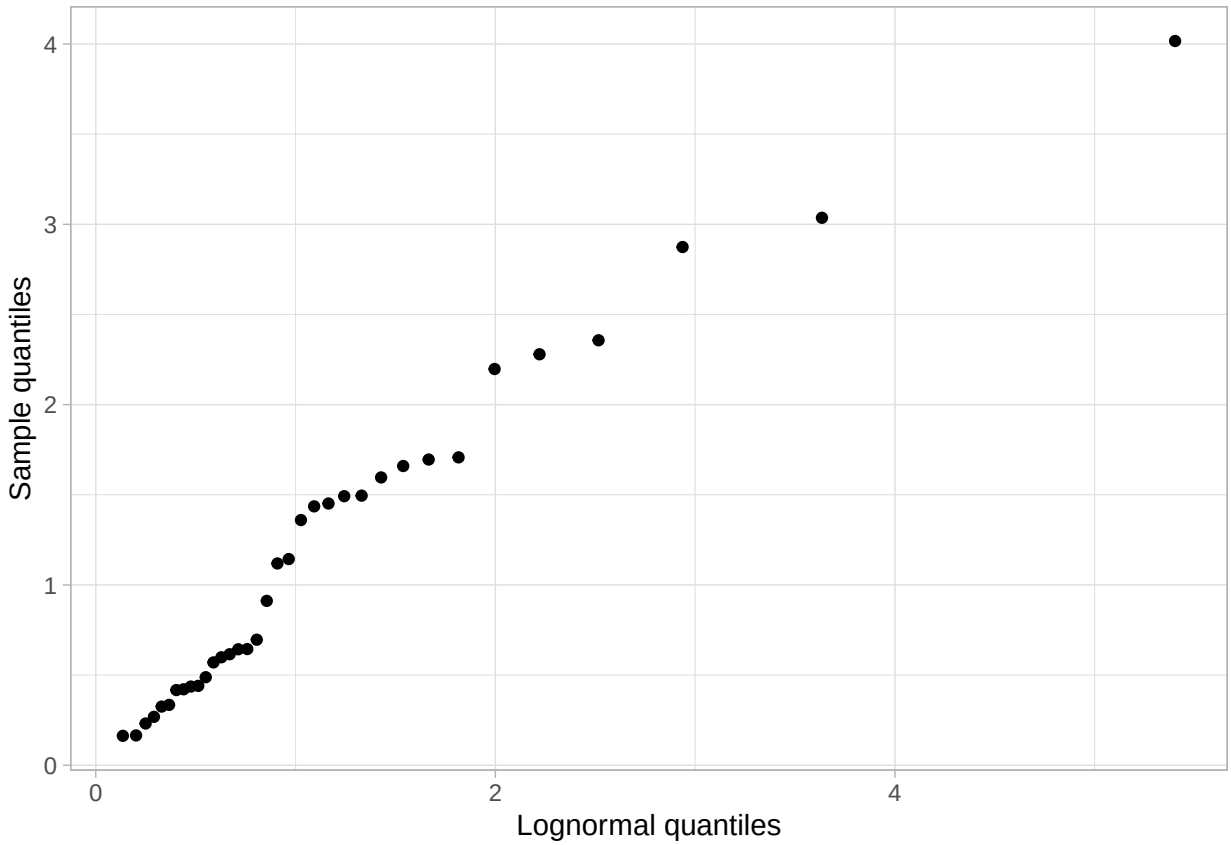
```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(gridExtra)
```

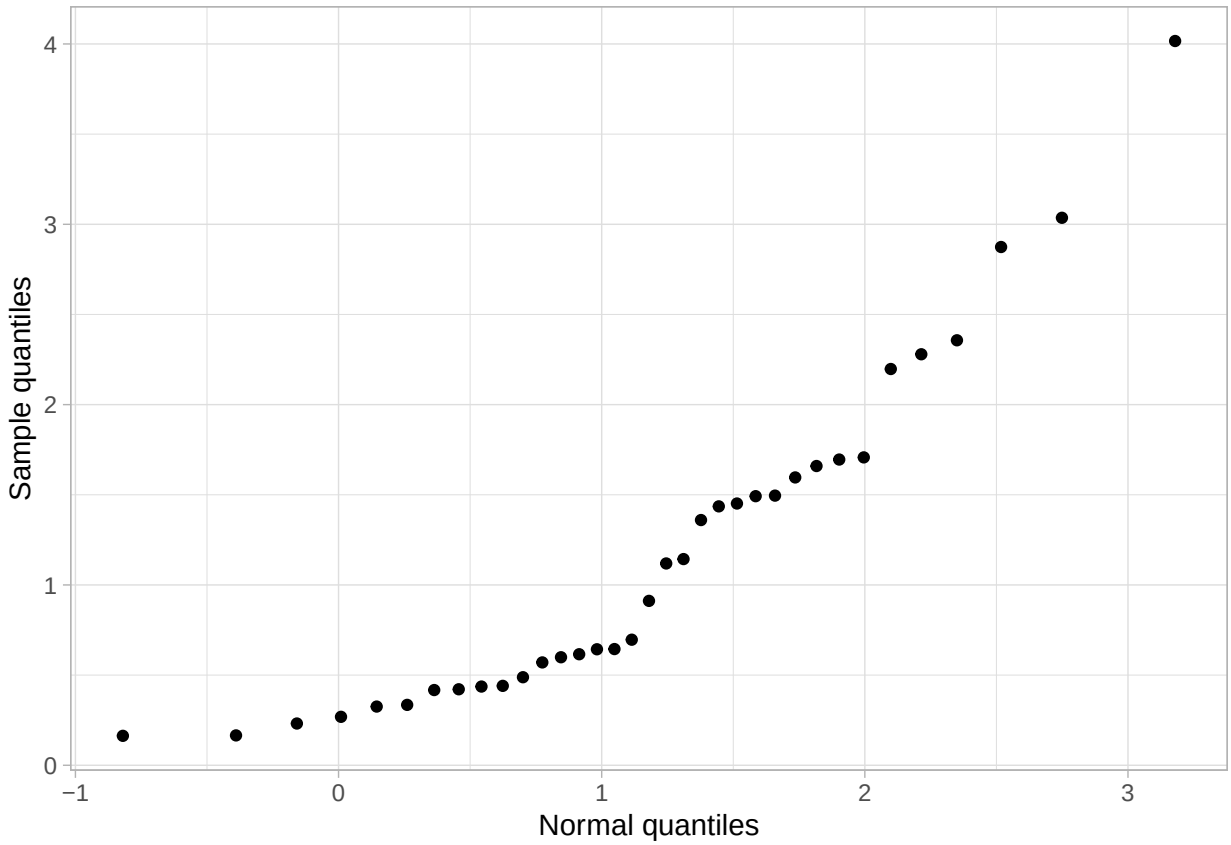
```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

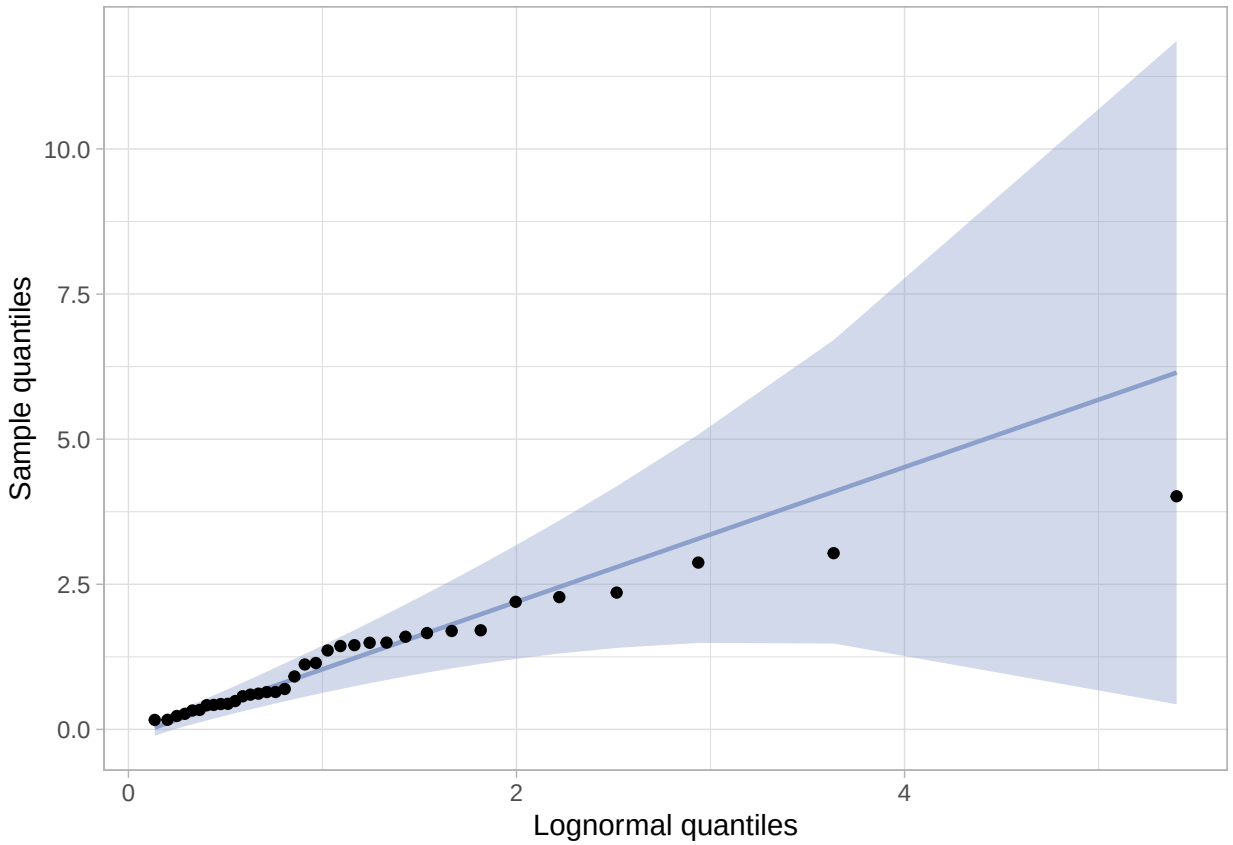
```
## ------------------------------------------------------------------------
## Background
## ------------------------------------------------------------------------

## ---- Code for Figure 1
set.seed(952017)
rand_sample <- data.frame(rs = rlnorm(35, meanlog = 0, sdlog = 1))

ggplot(data = rand_sample, mapping = aes(sample = rs)) +
  stat_qq_point(distribution = "lnorm") +
  ylab("Sample quantiles") +
  xlab("Lognormal quantiles") +
  theme_light()
```



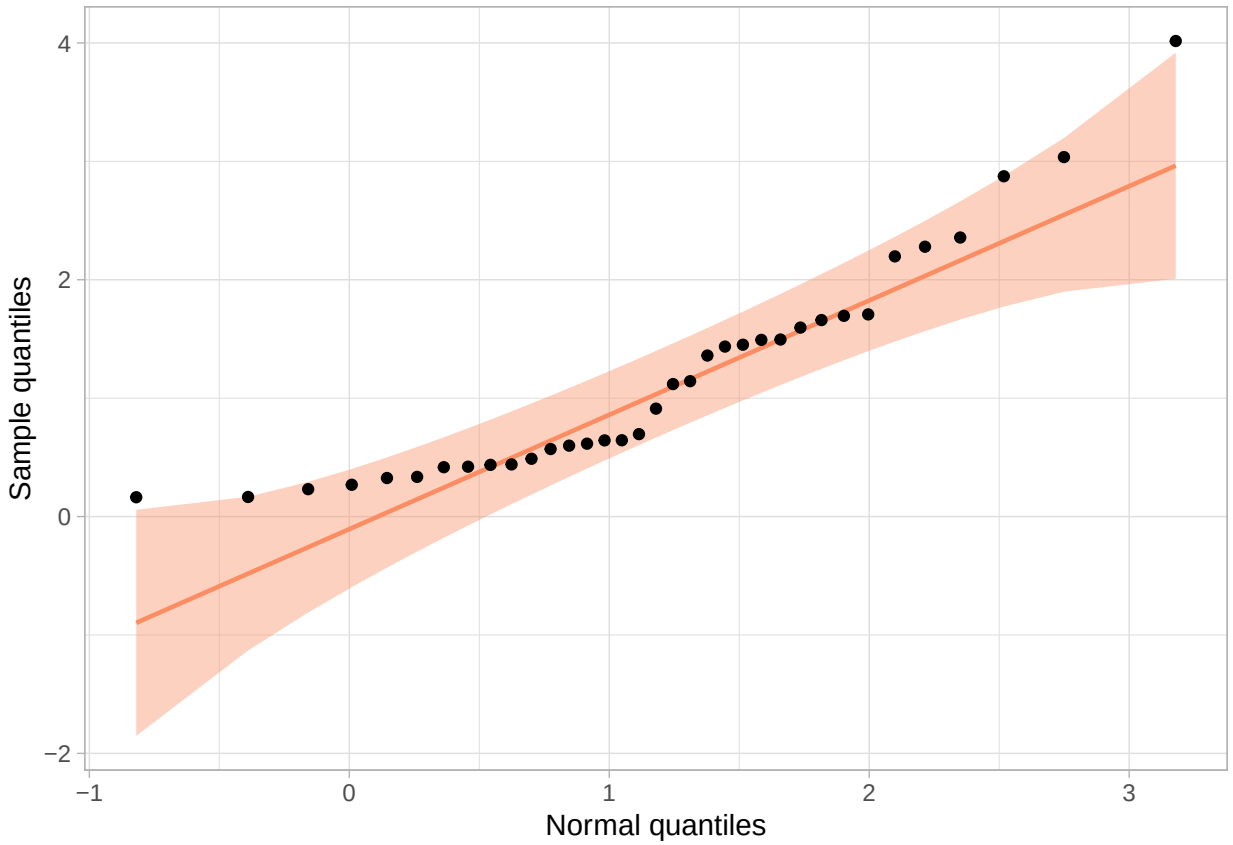

```
ggplot(data = rand_sample, mapping = aes(sample = rs)) +
  stat_qq_point() +
  ylab("Sample quantiles") +
  xlab("Normal quantiles") +
  theme_light()
```



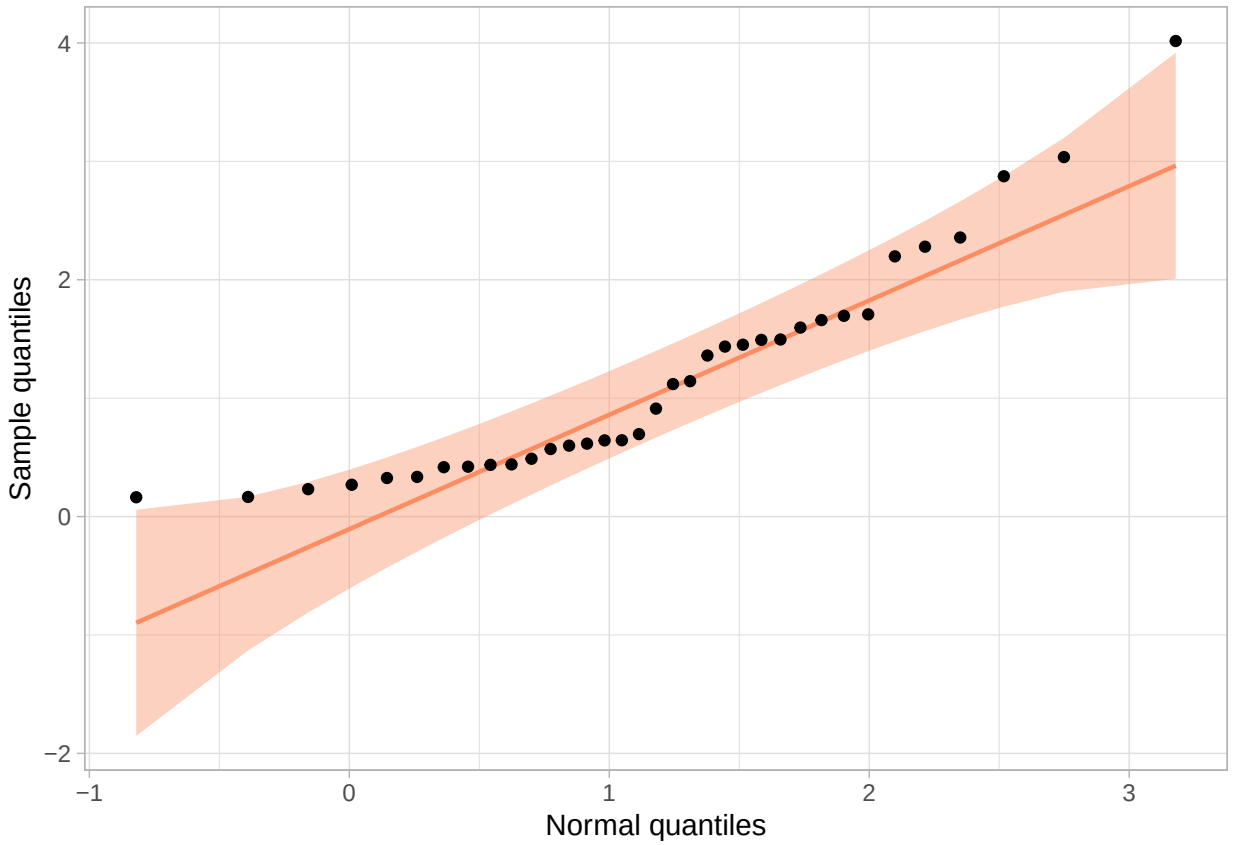

```
## ---- Code for Figure 2
ggplot(data = rand_sample, mapping = aes(sample = rs)) +
  stat_qq_band(distribution = "lnorm", fill = "#8DA0CB", alpha = 0.4) +
  stat_qq_line(distribution = "lnorm", colour = "#8DA0CB") +
  stat_qq_point(distribution = "lnorm") +
  ylab("Sample quantiles") +
  xlab("Lognormal quantiles") +
  theme_light()
```



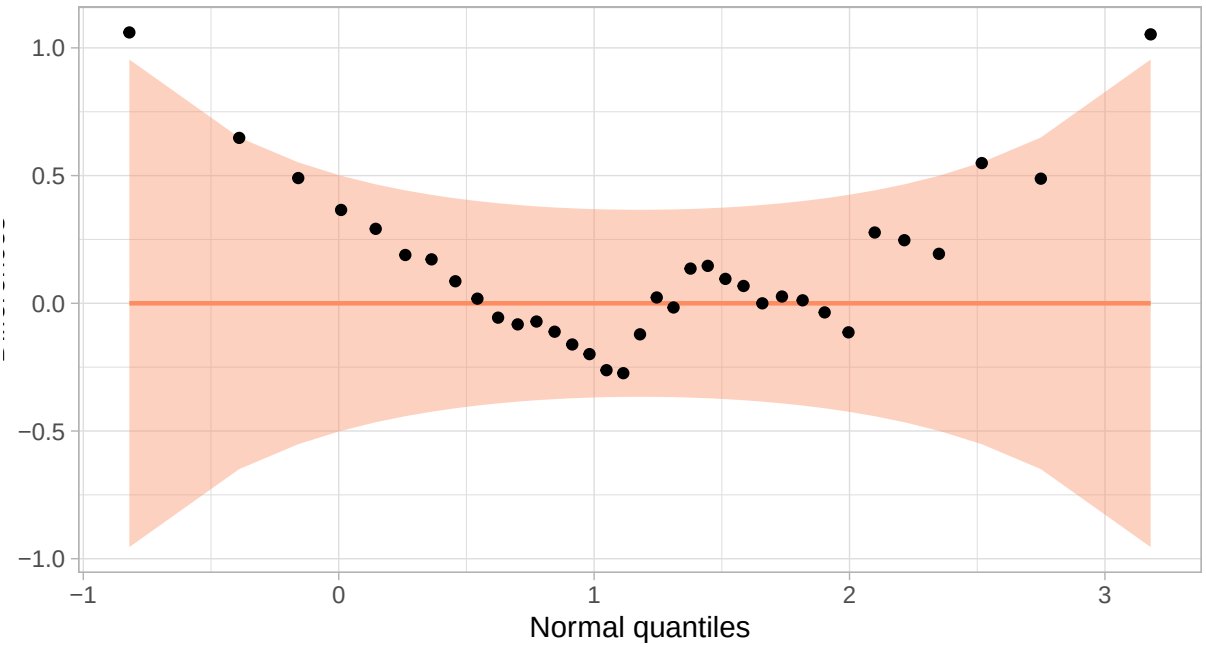

```
ggplot(data = rand_sample, mapping = aes(sample = rs)) +
  stat_qq_band(fill = "#FC8D62", alpha = 0.4) +
  stat_qq_line(colour = "#FC8D62") +
  stat_qq_point() +
  ylab("Sample quantiles") +
  xlab("Normal quantiles") +
  theme_light()
```



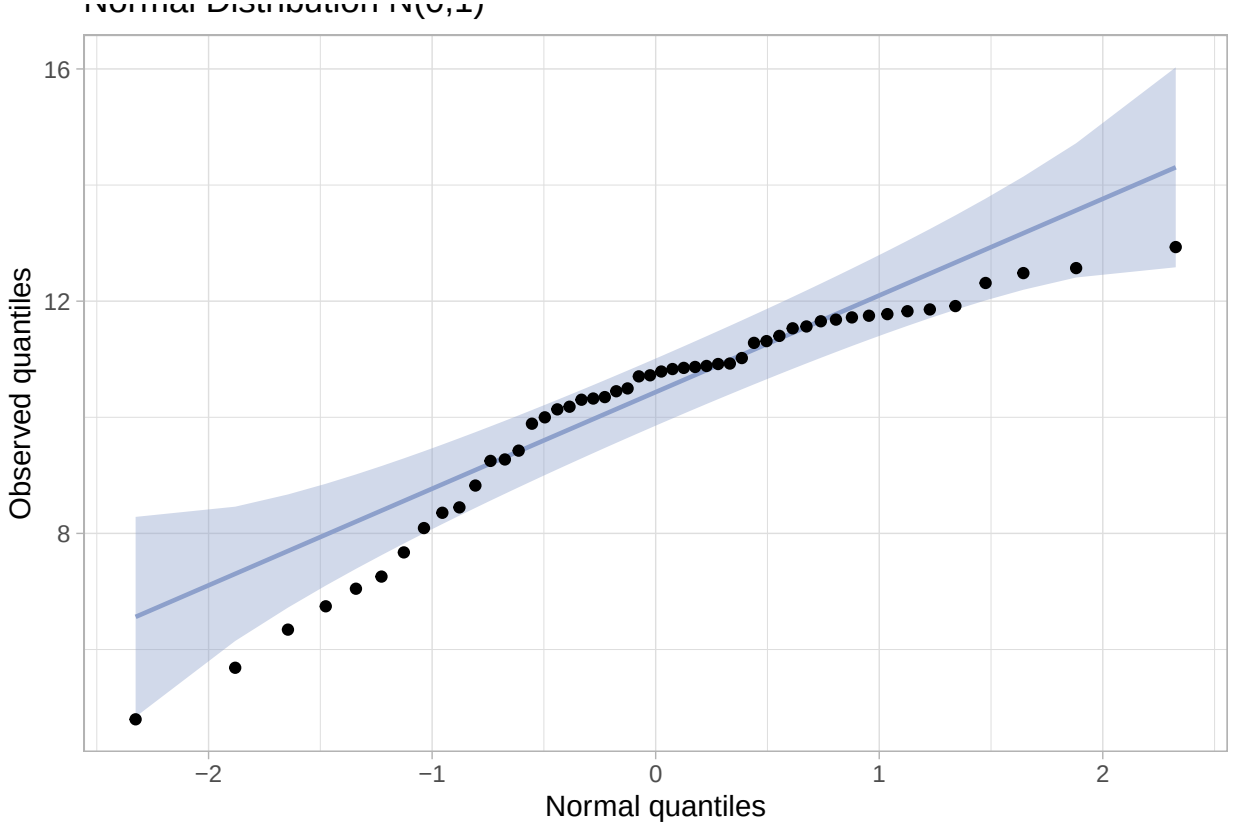

```
## ---- Code for Figure 3
set.seed(952017)
rand_sample <- data.frame(rs = rlnorm(35, meanlog = 0, sdlog = 1))

ggplot(data = rand_sample, mapping = aes(sample = rs)) +
  stat_qq_band(fill = "#FC8D62", alpha = 0.4) +
  stat_qq_line(colour = "#FC8D62") +
  stat_qq_point() +
  ylab("Sample quantiles") +
  xlab("Normal quantiles") +
  theme_light()
```



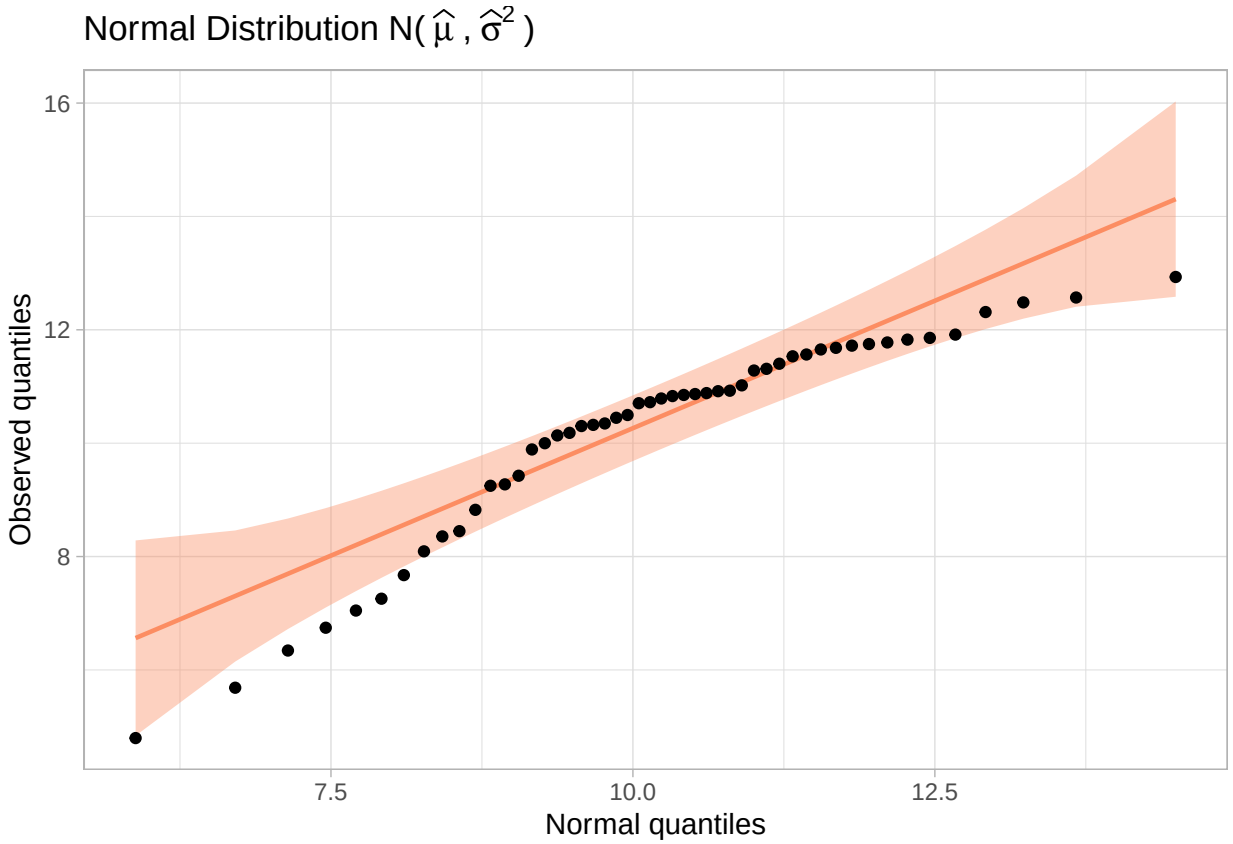

```
ggplot(data = rand_sample, mapping = aes(sample = rs)) +
  stat_qq_band(detrend = TRUE, fill = "#FC8D62", alpha = 0.4) +
  stat_qq_line(detrend = TRUE, colour = "#FC8D62") +
  stat_qq_point(detrend = TRUE) +
  ylab("Differences") +
  xlab("Normal quantiles") +
  theme_light() +
  coord_fixed(ratio = 1)
```



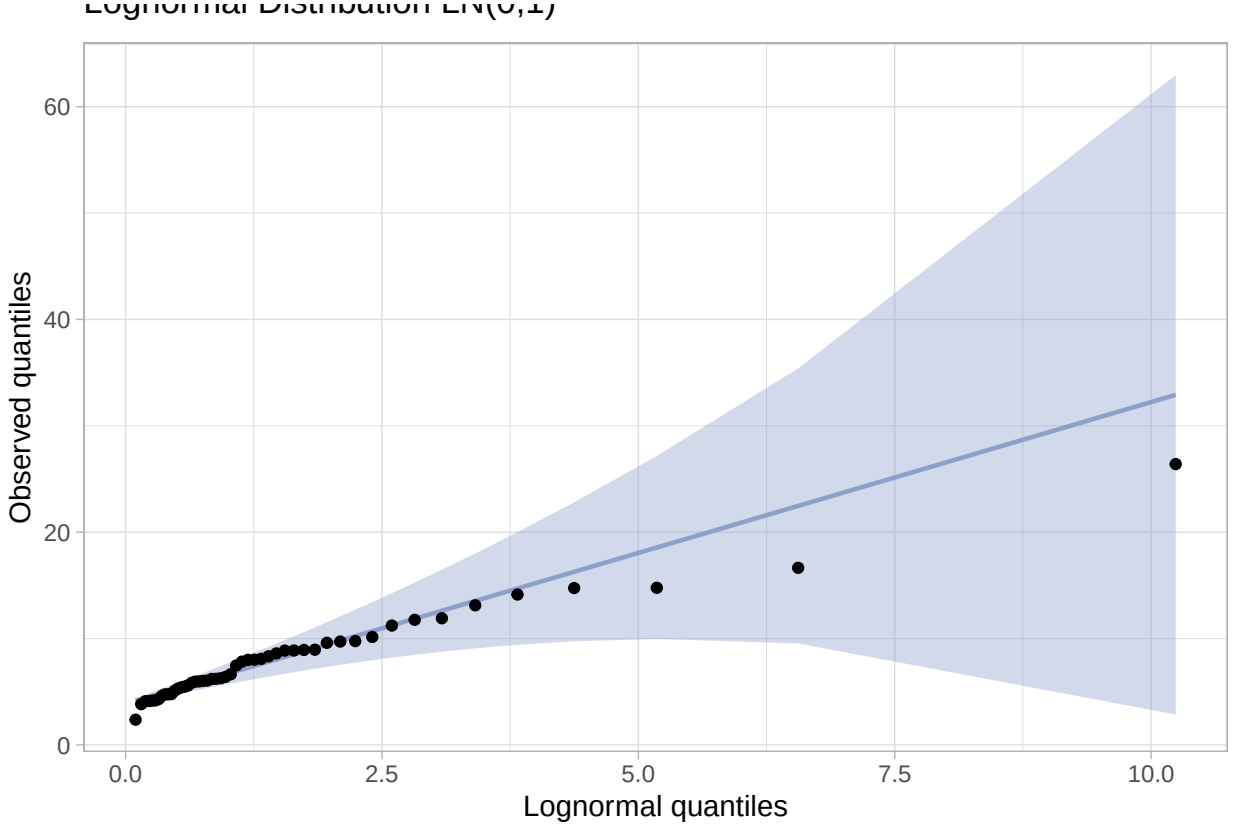

```
## ------------------------------------------------------------------------
## Implementing Q-Q plots in the ggplot2 framework
## ------------------------------------------------------------------------

## ---- Code for Figure 4
dframe <- data.frame(xnorm = rnorm(50, mean = 10, sd = 2),
                     xlnorm = rlnorm(50, mean = 2, sd = .5))

dframe %>% ggplot(aes(sample = xnorm)) +
  geom_qq_band(dparams = list(mean = 0, sd = 1), fill = "#8DA0CB", alpha = 0.4) +
  stat_qq_line(dparams = list(mean = 0, sd = 1), colour = "#8DA0CB") +
  stat_qq_point(dparams = list(mean = 0, sd = 1)) +
  theme_light() +
  xlab("Normal quantiles") +
  ylab("Observed quantiles") +
  ggtitle("Normal Distribution N(0,1)")
```



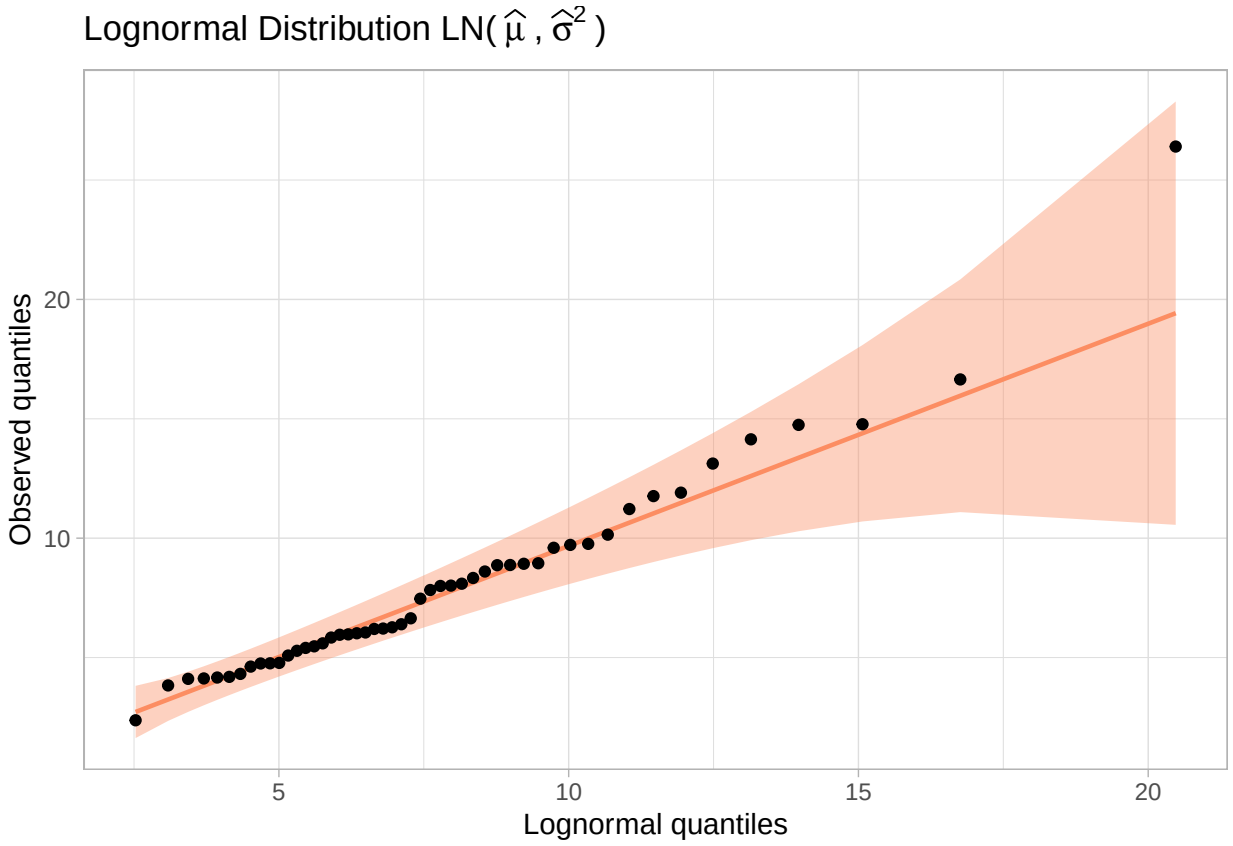

```
dframe %>% ggplot(aes(sample = xnorm)) +
  geom_qq_band(fill = "#FC8D62", alpha = 0.4) +
  stat_qq_line(colour = "#FC8D62") +
  stat_qq_point() +
  theme_light() +
  xlab("Normal quantiles") +
  ylab("Observed quantiles") +
  ggtitle(expression("Normal Distribution N("~widehat(mu)~","~widehat(sigma)^2~")"))
```




```
dframe %>% ggplot(aes(sample = xlnorm)) +
  geom_qq_band(distribution = "lnorm", dparams = list(mean = 0, sd = 1), fill = "#8DA0CB", alpha = 0.4) +
  stat_qq_line(distribution = "lnorm", dparams = list(mean = 0, sd = 1), colour = "#8DA0CB") +
  stat_qq_point(distribution = "lnorm", dparams = list(mean = 0, sd = 1)) +
  theme_light() +
  xlab("Lognormal quantiles") +
  ylab("Observed quantiles") +
  ggtitle(expression("Lognormal Distribution LN(0,1)"))
```



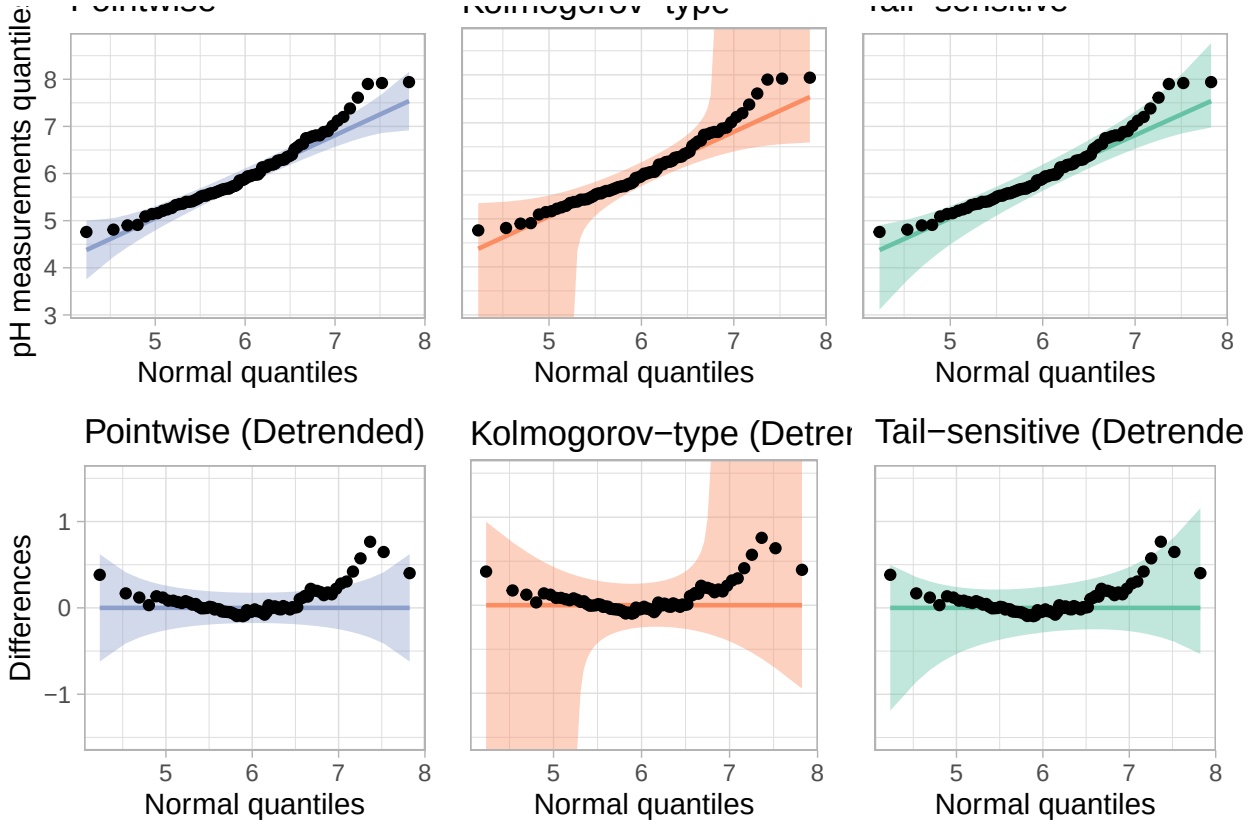

```
dframe %>% ggplot(aes(sample = xlnorm)) +
  geom_qq_band(distribution = "lnorm", fill = "#FC8D62", alpha = 0.4) +
  stat_qq_line(distribution = "lnorm", colour = "#FC8D62") +
  stat_qq_point(distribution = "lnorm") +
  theme_light() +
  xlab("Lognormal quantiles") +
  ylab("Observed quantiles") +
  ggtitle(expression("Lognormal Distribution LN("~widehat(mu)~","~widehat(sigma)^2~")"))
```



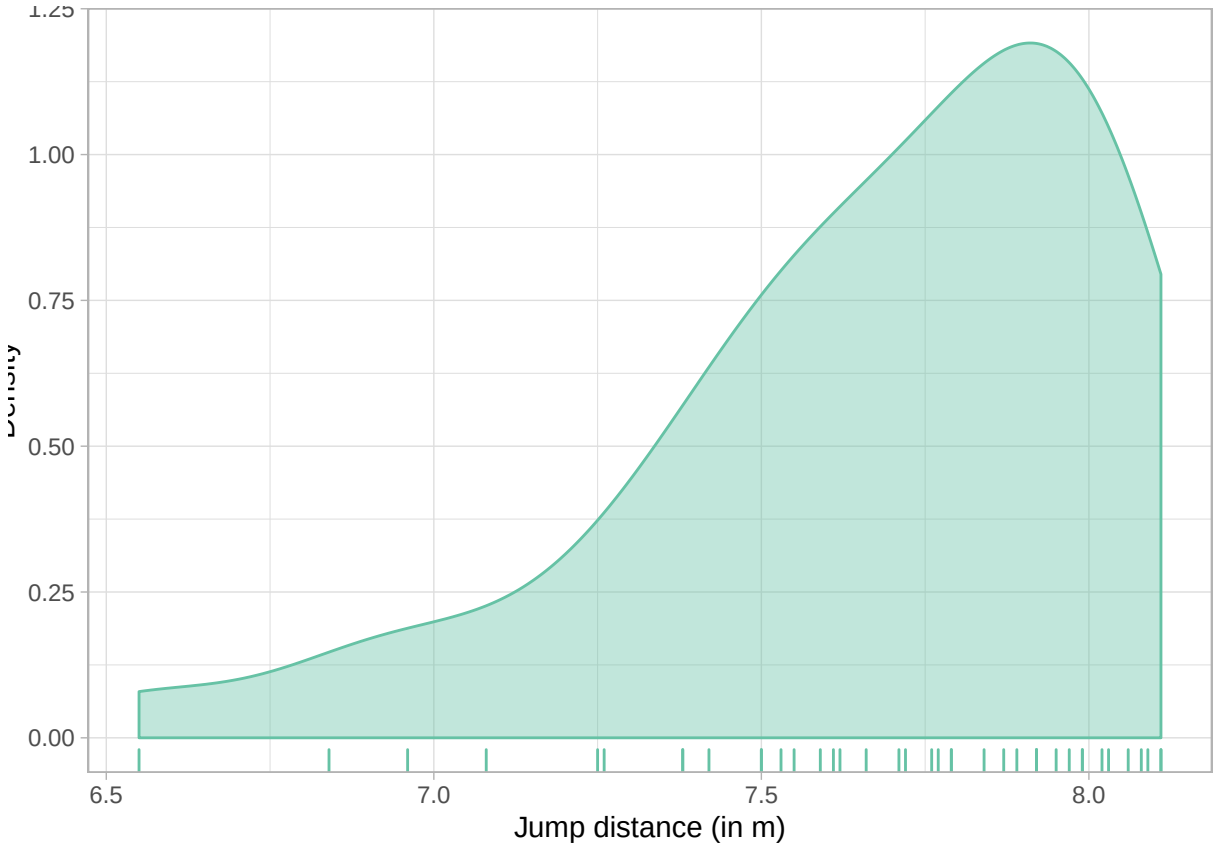

```
## ------------------------------------------------------------------------
## Constructing Q-Q plots with qqplotr
## ------------------------------------------------------------------------

## ---- Code for Figure 5
data(urine, package = "boot")
library(boot)
set.seed(9)

p11 <- urine %>% ggplot(aes(sample = ph)) +
  stat_qq_band(bandType = "pointwise", fill = "#8DA0CB", alpha = 0.4) +
  stat_qq_line(colour = "#8DA0CB") +
  stat_qq_point() +
  ggtitle("Pointwise") +
  xlab("Normal quantiles") +
  ylab("pH measurements quantiles") +
  theme_light() +
  coord_cartesian(ylim = c(3.2, 8.7))

p12 <- urine %>% ggplot(aes(sample = ph)) +
  stat_qq_band(bandType = "ks", fill = "#FC8D62", alpha = 0.4) +
  stat_qq_line(colour = "#FC8D62")+
  stat_qq_point() +
  ggtitle("Kolmogorov-type") +
  xlab("Normal quantiles") +
  ylab("pH measurements quantiles") +
  theme_light() +
```

```
  theme(axis.text.y = element_blank(),
          axis.ticks.y = element_blank(),
          axis.title.y = element_blank(),
          plot.title = element_text(vjust = 0),
          plot.margin = margin(t = 3.5, r = 5.5, b = 5.5, l = 5.5, unit = "pt")) +
  coord_cartesian(ylim = c(3.2, 8.7))

p13 <- urine %>% ggplot(aes(sample = ph)) +
  stat_qq_band(bandType = "ts", fill = "#66C2A5", alpha = 0.4) +
  stat_qq_line(colour = "#66C2A5") +
  stat_qq_point() +
  ggtitle("Tail-sensitive") +
  xlab("Normal quantiles") +
  ylab("Observed quantiles of pH measurements") +
  theme_light() +
  theme(axis.text.y = element_blank(),
        axis.ticks.y = element_blank(),
        axis.title.y = element_blank()) +
  coord_cartesian(ylim = c(3.2, 8.7))

p21 <- urine %>% ggplot(aes(sample = ph)) +
  stat_qq_band(bandType = "pointwise", fill = "#8DA0CB", alpha = 0.4, detrend = TRUE) +
  stat_qq_line(colour = "#8DA0CB", detrend = TRUE) +
  stat_qq_point(detrend = TRUE) +
  ggtitle("Pointwise (Detrended)") +
  xlab("Normal quantiles") +
  ylab("Differences") +
  theme_light() +
  coord_fixed(ratio=1, ylim = c(-1.5, 1.5))

p22 <- urine %>% ggplot(aes(sample = ph)) +
  stat_qq_band(bandType = "ks", fill = "#FC8D62", alpha = 0.4, detrend = TRUE) +
  stat_qq_line(colour = "#FC8D62", detrend = TRUE)+
  stat_qq_point(detrend = TRUE) +
  ggtitle("Kolmogorov-type (Detrended)") +
  xlab("Normal quantiles") +
  ylab("Differences") +
  theme_light() +
  theme(axis.text.y = element_blank(),
          axis.ticks.y = element_blank(),
          axis.title.y = element_blank(),
          plot.title = element_text(vjust = 0),
          plot.margin = margin(t = 3.5, r = 5.5, b = 5.5, l = 5.5, unit = "pt")) +
  coord_fixed(ratio=1, ylim = c(-1.5, 1.5))

p23 <- urine %>% ggplot(aes(sample = ph)) +
  stat_qq_band(bandType = "ts", fill = "#66C2A5", alpha = 0.4, detrend = TRUE) +
  stat_qq_line(colour = "#66C2A5", detrend = TRUE) +
  stat_qq_point(detrend = TRUE) +
  ggtitle("Tail-sensitive (Detrended)") +
  xlab("Normal quantiles") +
  ylab("Differences") +
  theme_light() +
```

```
  theme(axis.text.y = element_blank(),
        axis.ticks.y = element_blank(),
        axis.title.y = element_blank()) +
  coord_fixed(ratio=1, ylim = c(-1.5, 1.5))

grid.arrange(p11, p12, p13,
             p21, p22, p23,
             nrow = 2,
             ncol = 3,
             widths = c(1.11, 1, 1))
```

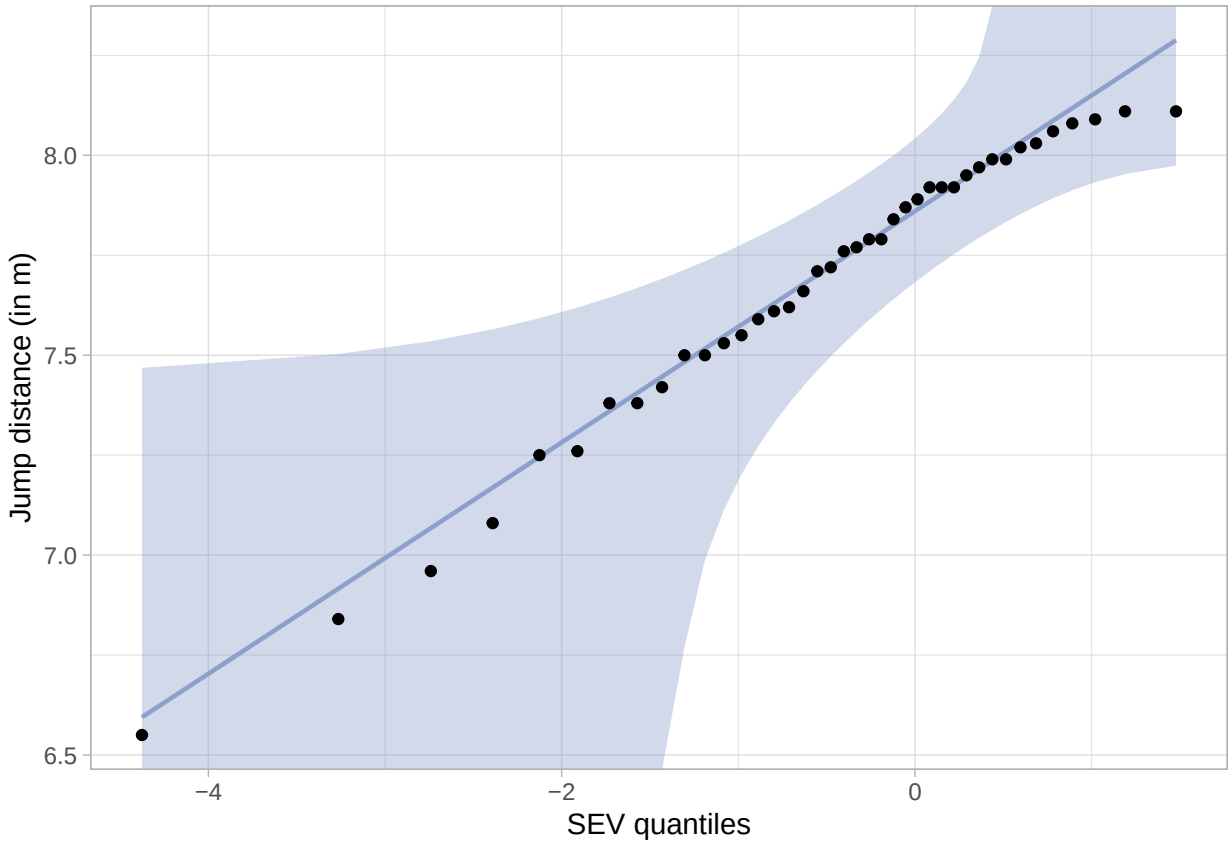



```
## ------------------------------------------------------------------------
## User-provided distributions
## ------------------------------------------------------------------------

## ---- Loading longjump data
data("longjump", package = "qqplotr")
longjump <- na.omit(longjump)

## ---- Code for Figure 6
ggplot(longjump, aes(x = distance)) +
  geom_density(fill = "#66C2A5", colour = "#66C2A5", alpha = 0.4) +
  geom_rug(colour = "#66C2A5") +
  xlab("Jump distance (in m)") +
  ylab("Density") +
  theme_light()
```



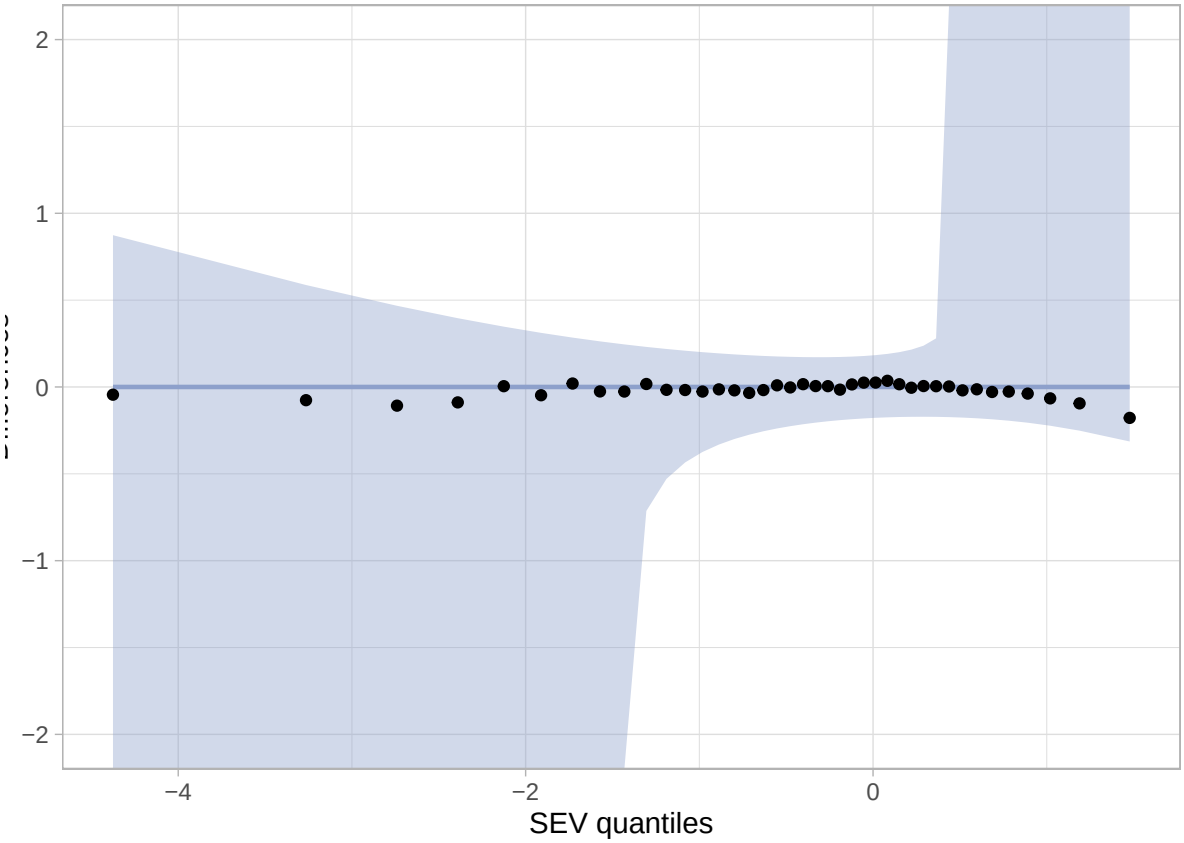

```r
## ---- Defining SEV distribution function
# CDF
psev <- function(q, mu = 0, sigma = 1) {
  z <- (q - mu) / sigma
  1 - exp(-exp(z))
}

# PDF
dsev <- function(x, mu = 0, sigma = 1) {
  z <- (x - mu) / sigma
  (1 / sigma) * exp(z - exp(z))
}

# Quantile function
qsev <- function(p, mu = 0, sigma = 1) {
  mu + log(-log(1 - p)) * sigma
}

# Simulation function
rsev <- function(n, mu = 0, sigma = 1) {
  qsev(runif(n), mu, sigma)
}

## ---- Code for Figure 7
ggplot(longjump, aes(sample = distance)) +
  stat_qq_band(distribution = "sev",
               bandType = "ks",
```

```
              dparams = list(mu = 0, sigma = 1),
              fill = "#8DA0CB",
              alpha = 0.4) +
  stat_qq_line(distribution = "sev",
              colour = "#8DA0CB",
              dparams = list(mu = 0, sigma = 1)) +
  stat_qq_point(distribution = "sev",
               dparams = list(mu = 0, sigma = 1)) +
  xlab("SEV quantiles") +
  ylab("Jump distance (in m)") +
  theme_light()
```



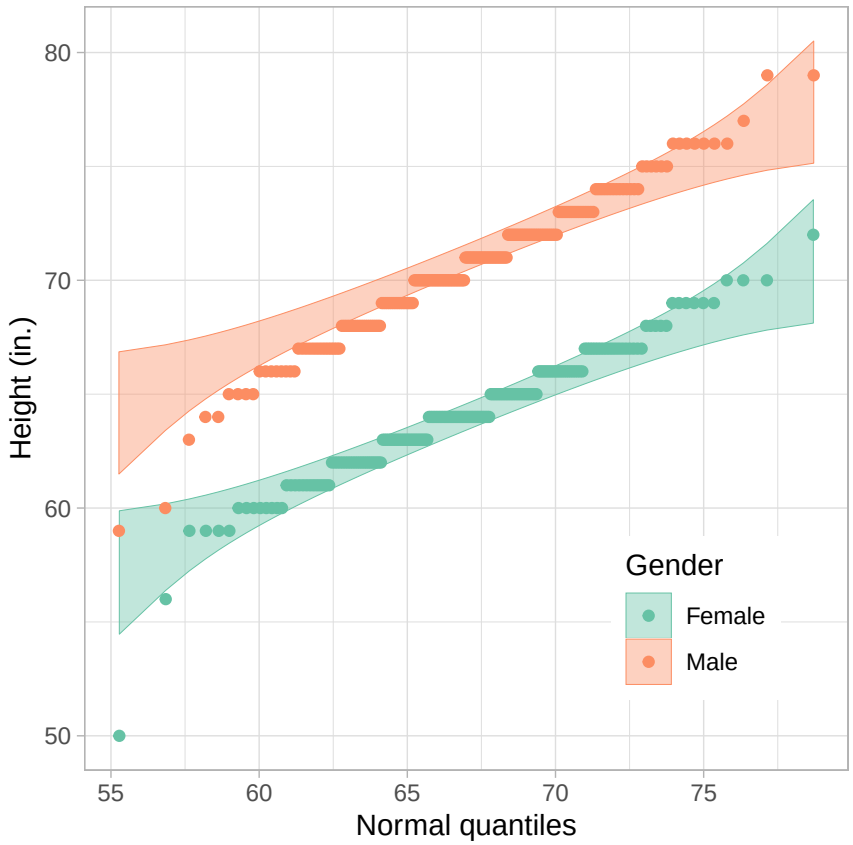

```
## ------------------------------------------------------------------------
## Detrending Q-Q plots
## ------------------------------------------------------------------------

## ---- Code for Figure 8
ggplot(longjump, aes(sample = distance)) +
  stat_qq_band(distribution = "sev",
              bandType = "ks",
              detrend = TRUE,
              dparams = list(mu = 0, sigma = 1),
              fill = "#8DA0CB",
              alpha = 0.4) +
  stat_qq_line(distribution = "sev",
              detrend = TRUE,
```

```
              dparams = list(mu = 0, sigma = 1),
              colour = "#8DA0CB") +
  stat_qq_point(distribution = "sev",
                detrend = TRUE,
                dparams = list(mu = 0, sigma = 1)) +
  xlab("SEV quantiles") +
  ylab("Differences") +
  theme_light() +
  coord_fixed(ratio = 1, ylim = c(-2, 2))
```



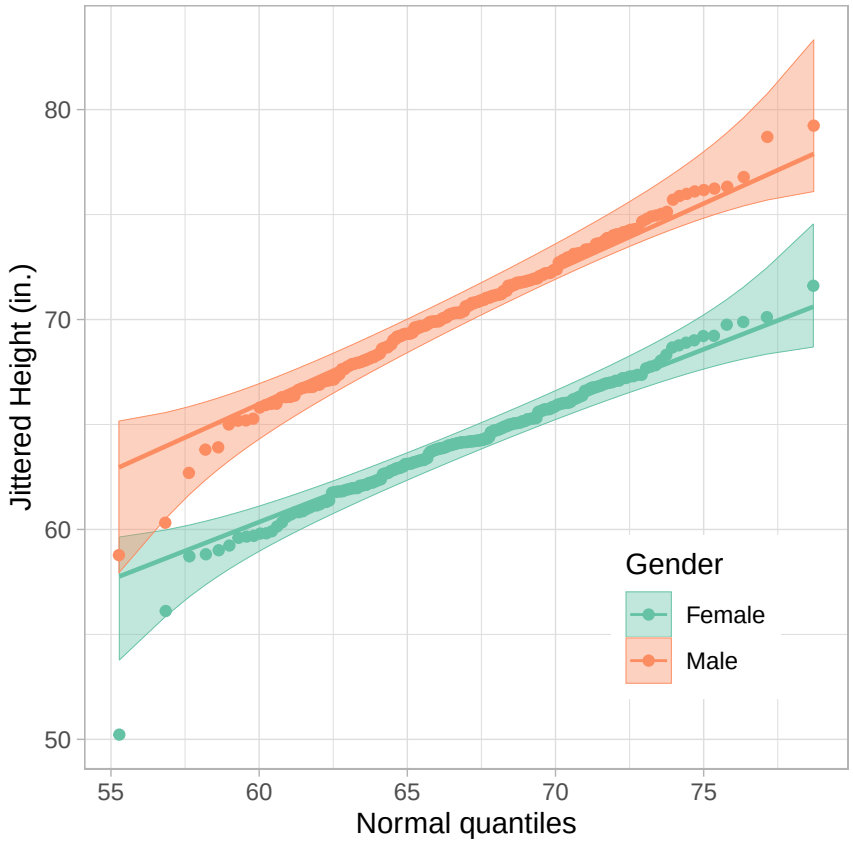

```
## ------------------------------------------------------------------------
## BRFSS Example
## ------------------------------------------------------------------------

data("iowa", package = "qqplotr")
set.seed(3145)

## ---- Data preprocessing
sample_ia <- iowa %>%
  tidyr::nest(-SEX) %>%
  mutate(
    data = data %>%
    purrr::map(.f = function(x) sample_n(x, size = 200))
    ) %>%
  tidyr::unnest(data) %>%
  dplyr::select(SEX, WTKG3, HTIN4) %>%
```

```
  mutate(Gender = c("Male", "Female")[SEX])


## ---- Code for figure 9
params <- iowa %>%
  filter(!is.na(HTIN4)) %>%
  summarize(m = mean(HTIN4), s = sd(HTIN4))

customization <- list(scale_fill_brewer(palette = "Set2"),
                      scale_colour_brewer(palette = "Set2"),
                      xlab("Normal quantiles"),
                      ylab("Height (in.)"),
                      coord_equal(),
                      theme_light(),
                      theme(legend.position = c(0.8, 0.2), aspect.ratio = 1))

sample_ia %>%
  ggplot(aes(sample = HTIN4, colour=Gender, fill=Gender)) +
  stat_qq_band(bandType = "ts",
               alpha = 0.4,
               dparams = list(mean = params$m, sd = params$s)) +
  stat_qq_point(dparams = list(mean = params$m, sd = params$s)) +
  customization
```




```
sample_ia %>%
  mutate(HTIN4.jitter = jitter(HTIN4, factor = 2)) %>%
  ggplot(aes(sample = HTIN4.jitter, colour = Gender, fill = Gender)) +
```

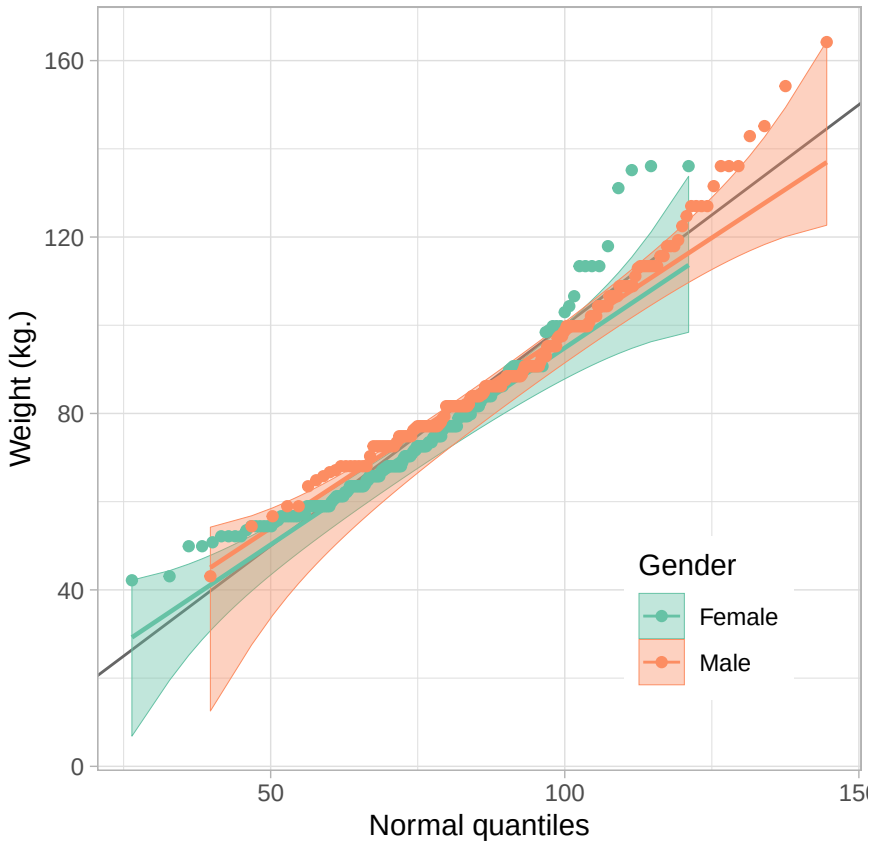
```
  stat_qq_band(bandType = "ts",
               alpha = 0.4,
               dparams = list(mean = params$m, sd = params$s)) +
  stat_qq_line(dparams = list(mean = params$m, sd = params$s)) +
  stat_qq_point(dparams = list(mean = params$m, sd = params$s)) +
  customization +
  ylab("Jittered Height (in.)")
```



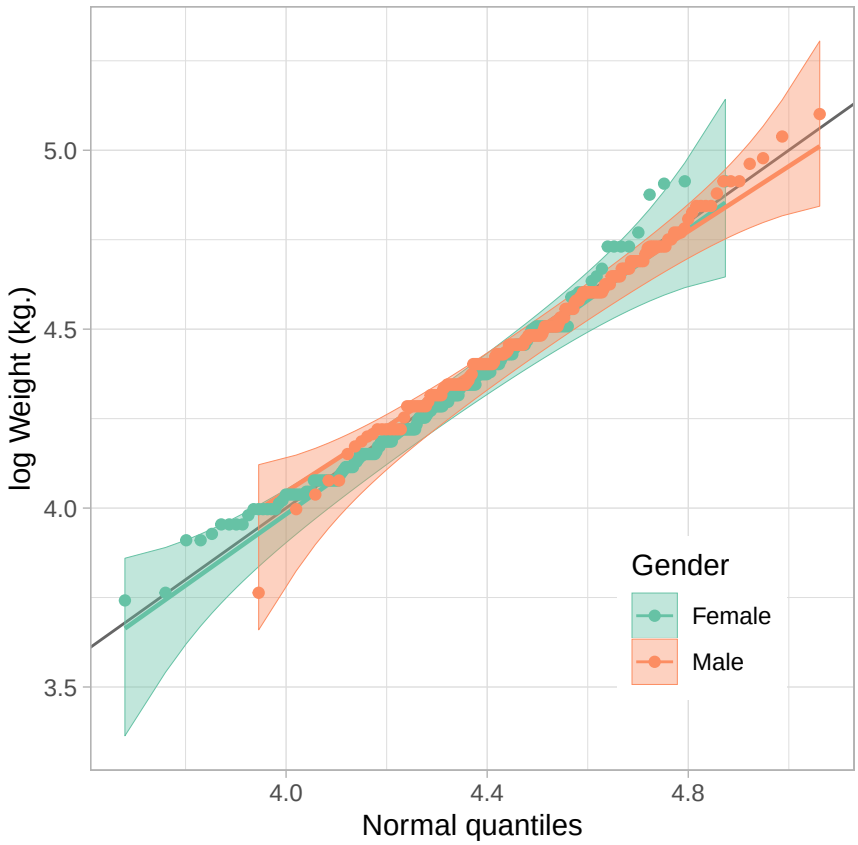

```
## ---- Code for table 1
bygender <- iowa %>%
  group_by(SEX) %>%
  summarize(mean.height = mean(HTIN4, na.rm = TRUE),
            sd.height = sd(HTIN4, na.rm = TRUE),
            mean.weight = mean(log(WTKG3), na.rm = TRUE),
            sd.weight = sd(log(WTKG3), na.rm = TRUE))
bygender$SEX <- c("Male", "Female")

total <- iowa %>%
  summarize(mean.height = mean(HTIN4, na.rm = TRUE),
            sd.height = sd(HTIN4, na.rm = TRUE),
            mean.weight = mean(log(WTKG3), na.rm = TRUE),
            sd.weight = sd(log(WTKG3), na.rm = TRUE))

bygender[3,] <- c("Total", total)
names(bygender) <- c("SEX", "mean height (in)", "sd (in)", "mean log weight (kg)", "sd (kg)")
knitr::kable(bygender)
```

SEX	mean height (in)	sd (in)	mean log weight (kg)	sd (kg)
Male	70.55043	2.966460	9.100936	0.2011216
Female	64.50972	2.911454	8.887950	0.2254663
Total	66.99109	4.176295	8.978768	0.2397852

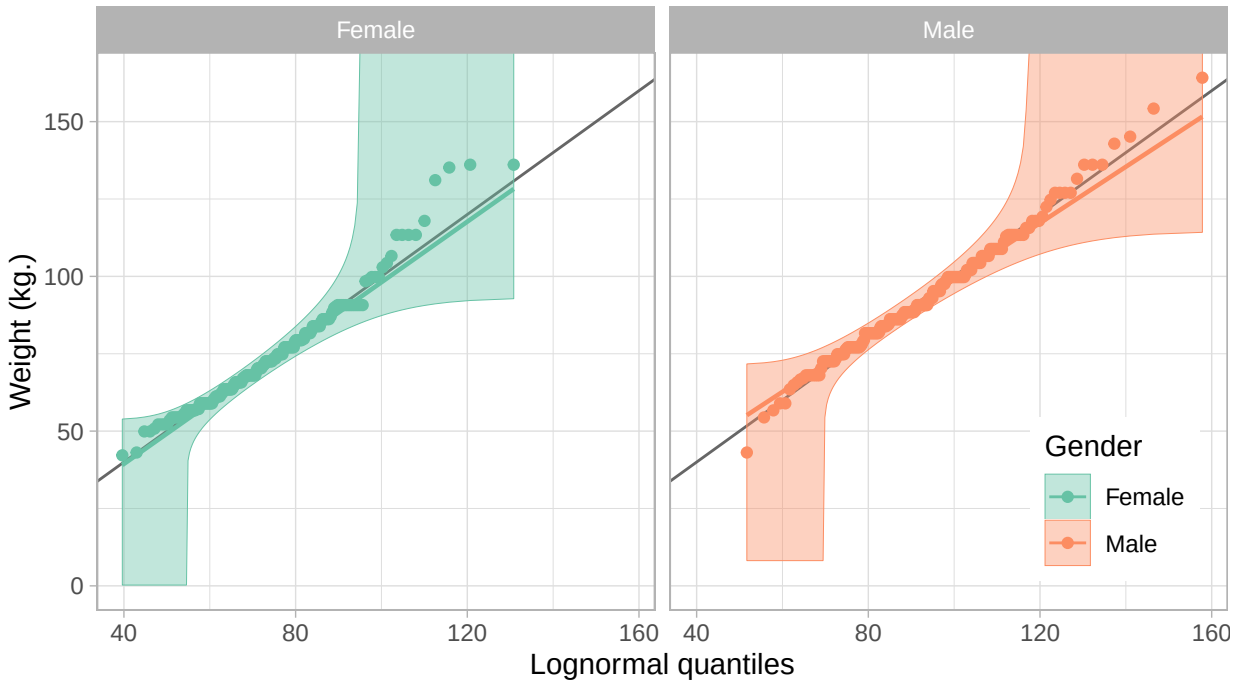
```
## ---- Code for figure 10
sample_ia %>%
  ggplot(aes(sample = WTKG3 / 100, colour = Gender, fill = Gender)) +
  geom_abline(colour = "grey40") +
  stat_qq_band(bandType = "ts", alpha = 0.4) +
  stat_qq_line() +
  stat_qq_point() +
  customization +
  ylab("Weight (kg.)")
```




```
sample_ia %>%
  ggplot(aes(sample = log(WTKG3/100), colour=Gender, fill=Gender)) +
  geom_abline(colour = "grey40") +
  stat_qq_band(bandType = "ts", alpha = 0.4) +
  stat_qq_line() +
  stat_qq_point() +
  customization +
  ylab("log Weight (kg.)")
```




```
## ---- Code for figure 11
sample_ia %>%
  ggplot(aes(sample = WTKG3 / 100, colour = Gender, fill = Gender)) +
  geom_abline(colour = "grey40") +
  stat_qq_band(bandType = "ks", distribution = "lnorm", alpha = 0.4) +
  stat_qq_line(distribution = "lnorm") +
  stat_qq_point(distribution = "lnorm") +
  customization +
  facet_grid(. ~ Gender) +
  xlab("Lognormal quantiles") +
  ylab("Weight (kg.)") +
  theme(legend.position = c(0.9, 0.2))
```



##DotPipes

```
library("wrapr")
```

```
##
## Attaching package: 'wrapr'
```

```
## The following object is masked from 'package:dplyr':
##
##     coalesce
```

```
5 %.>% sin(.)
```

```
## [1] -0.9589243
```

```
print(.)
```

```
## [1] 5
```

```
5 %.>% {1 + .}
```

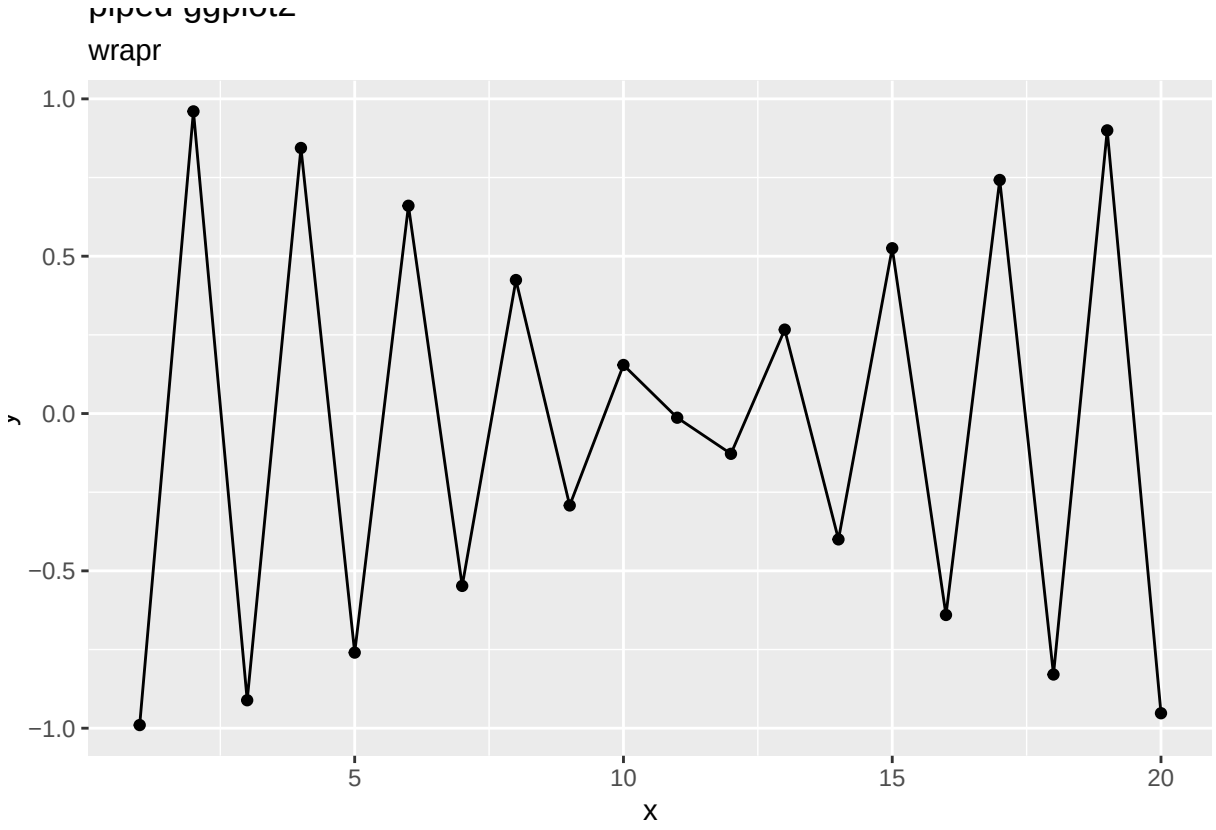
```
## [1] 6
```

```
library("dplyr")
disp <- 4
mtcars %.>%
  filter(., .data$cyl == .env$disp) %.>%
  nrow(.)
```

```
## [1] 11
```

```
library("ggplot2")
apply_left.gg <- function(pipe_left_arg,
                          pipe_right_arg,
                          pipe_environment,
                          left_arg_name,
                          pipe_string,
```

```
                           right_arg_name) {
  pipe_right_arg <- eval(pipe_right_arg,
                         envir = pipe_environment,
                         enclos = pipe_environment)
  pipe_left_arg + pipe_right_arg
}


data.frame(x = 1:20) %.>%
  mutate(., y = cos(3*x)) %.>%
  ggplot(., aes(x = x,  y = y)) %.>%
  geom_point() %.>%
  geom_line() %.>%
  ggtitle("piped ggplot2",
          subtitle = "wrapr")
```




```
library("rquery")
```

```
##
## Attaching package: 'rquery'
```

```
## The following object is masked from 'package:ggplot2':
##
##     arrow
```

```
optree <- mk_td(table_name = "d", columns = "x") %.>%
  extend_nse(., y = cos(2*x))
```

```
class(optree)
```

```
## [1] "relop_extend" "relop"
```

```
print(optree)
```

```
## [1] "mk_td(\"d\", c( \"x\")) %.>% extend(., y := cos(2 * x))"
```

```
column_names(optree)
```

```
## [1] "x" "y"
```

```
columns_used(optree)
```

```
## $d
## [1] "x"
```

```
db = DBI::dbConnect(RSQLite::SQLite(),
                    ":memory:")

options(list("rquery.rquery_db_executor" = list(db = db)))
data.frame(x = 1:3) %.>% optree
```

```
##   x          y
## 1 1 -0.4161468
## 2 2 -0.6536436
## 3 3  0.9601703
```

```
d1 <- data.frame(x = 1)
d2 <- data.frame(x = 2)
tryCatch(
  d1 %.>% d2,
  error = function(e) { invisible(cat(format(e))) })
```

```
## wrapr::apply_right_S4 default called with classes:
##  d1 data.frame
##  d2 data.frame
##   must have a more specific S4 method defined to dispatch
##  NULL
```

```
setMethod(
  "apply_right_S4",
  signature = c("data.frame", "data.frame"),
  definition = function(pipe_left_arg,
                        pipe_right_arg,
                        pipe_environment,
                        left_arg_name,
                        pipe_string,
                        right_arg_name) {
    rbind(pipe_left_arg, pipe_right_arg)
    })
d1 %.>% d2
```

```
##   x
## 1 1
## 2 2
```

```
d1 %.>% data.frame(x = 2)
```

```
##   x
## 1 2
```

```
library("magrittr")
5 %>% sin
```

```
## [1] -0.9589243
```

```
`%userpipe%`<- magrittr::`%>%`
tryCatch(
  5 %userpipe% sin,
  error = function(e) {e})
```

```
## <simpleError in pipes[[i]]: subscript out of bounds>
```

```
`%userpipe%`<- wrapr::`%.>%`
5 %userpipe% sin
```

```
## [1] -0.9589243
```

```
library("magrittr")
5 %>% substitute
```

```
## value
```

```
tryCatch(
  5 %>% base::sin,
        error = function(e) {e})
```

```
## <simpleError in .::base: unused argument (sin)>
```

```
5 %.>% substitute
```

```
## [1] 5
```

```
5 %.>% base::sin
```

```
## [1] -0.9589243
```

```
d <- data.frame(x = 1:5, y = c(1, 1, 0, 1, 0))
model <- glm(y~x, family = binomial, data = d)
apply_right.glm <-
  function(pipe_left_arg,
           pipe_right_arg,
           pipe_environment,
           left_arg_name,
           pipe_string,
           right_arg_name) {
    predict(pipe_right_arg,
            newdata = pipe_left_arg,
            type ='response')
    }
data.frame(x = c(1, 3)) %.>% model
```

```
##         1         2
## 0.9428669 0.6508301
```

```
db = DBI::dbConnect(RSQLite::SQLite(),
                    ":memory:")
apply_right.SQLiteConnection <-
  function(pipe_left_arg,
```

```
           pipe_right_arg,
           pipe_environment,
           left_arg_name,
           pipe_string,
           right_arg_name) {
    DBI::dbGetQuery(pipe_right_arg, pipe_left_arg)
    }
"SELECT * FROM sqlite_temp_master" %.>% db
```

```
## [1] type     name     tbl_name rootpage sql
## <0 rows> (or 0-length row.names)
```

```
apply_left.character <- function(pipe_left_arg,
                                 pipe_right_arg,
                                 pipe_environment,
                                 left_arg_name,
                                 pipe_string,
                                 right_arg_name) {
  pipe_right_arg <- eval(pipe_right_arg,
                         envir = pipe_environment,
                         enclos = pipe_environment)
  paste0(pipe_left_arg, pipe_right_arg)
  }
"a" %.>% "b" %.>% "c"
```

```
## [1] "abc"
```

```
apply_left.formula <- function(pipe_left_arg,
                               pipe_right_arg,
                               pipe_environment,
                               left_arg_name,
                               pipe_string,
                               right_arg_name) {
  pipe_right_arg <- eval(pipe_right_arg,
                         envir = pipe_environment,
                         enclos = pipe_environment)
  pipe_right_arg <- paste(pipe_right_arg, collapse = " + ")
  update(pipe_left_arg, paste(" ~ . +", pipe_right_arg))
  }
(y~a) %.>% c("b", "c", "d") %.>% "e"
```

```
## y ~ a + b + c + d + e
```