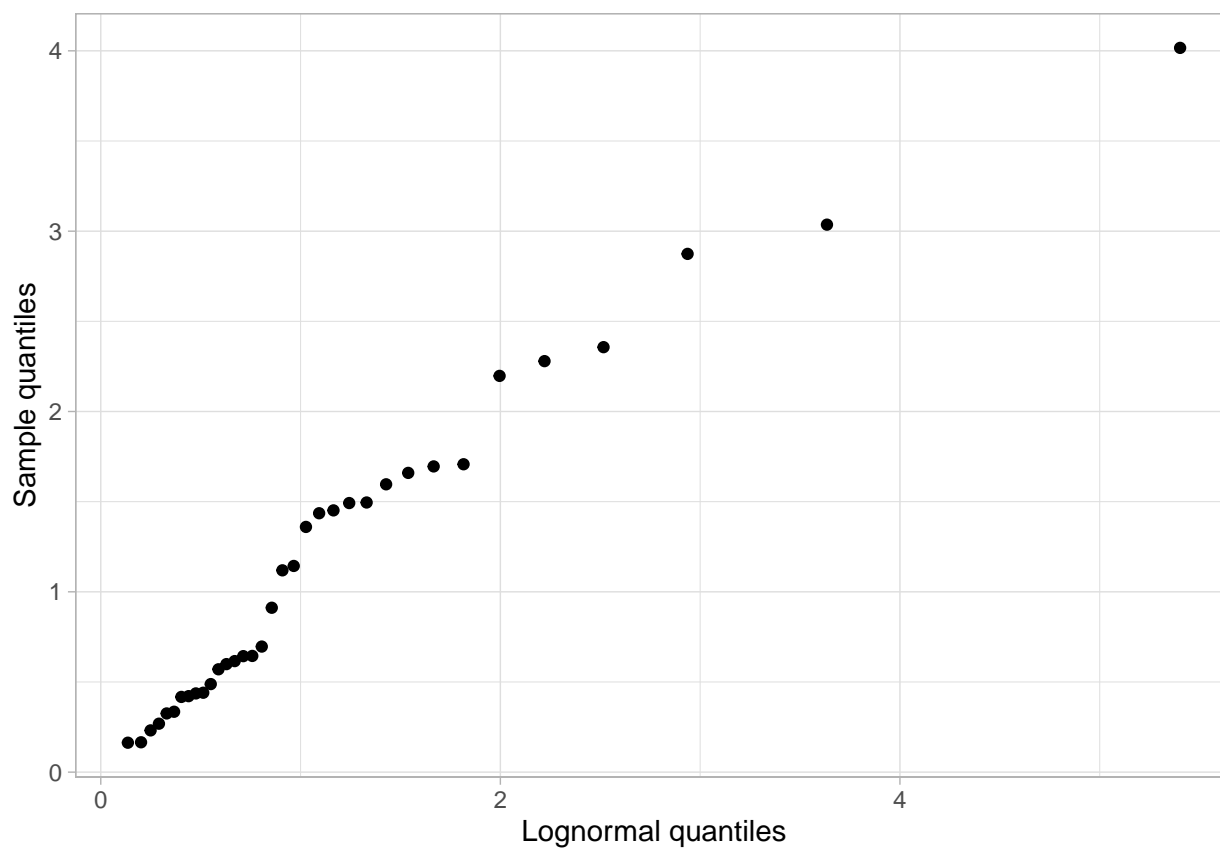


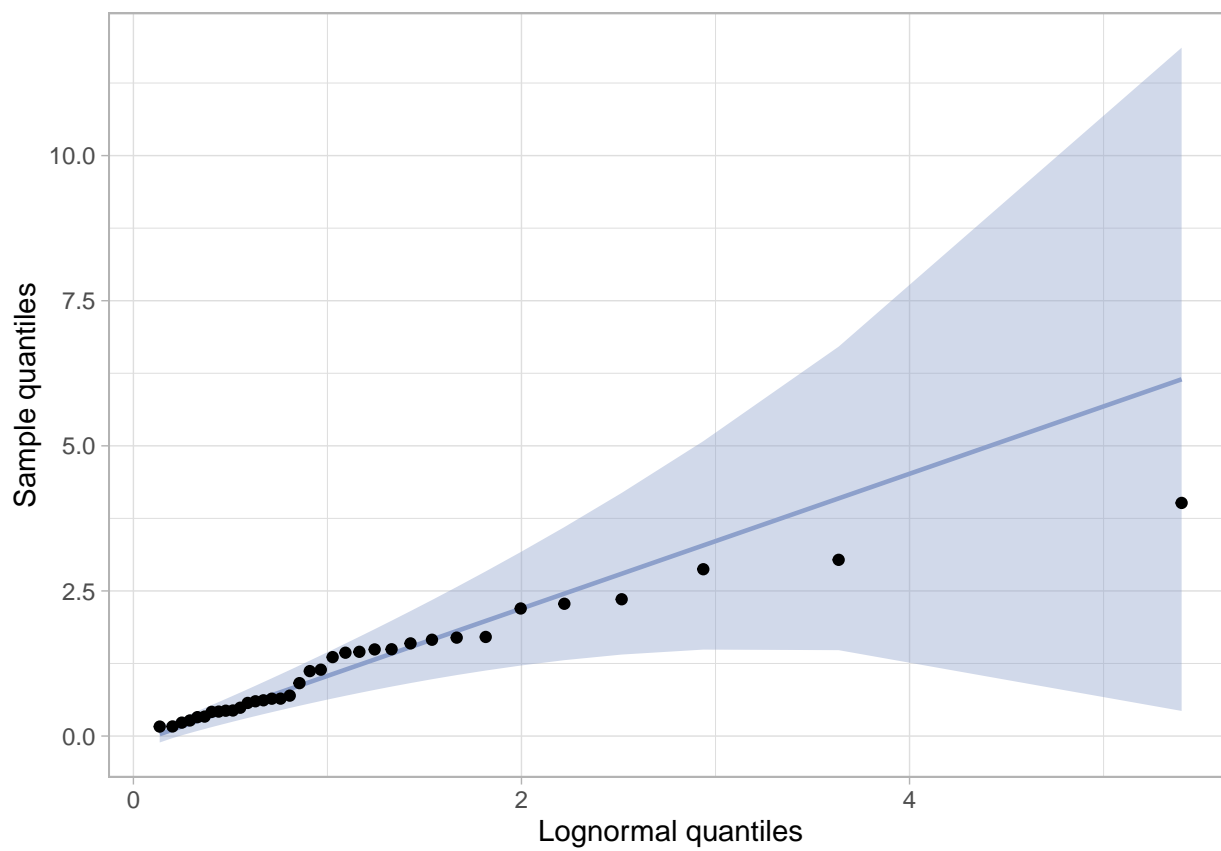
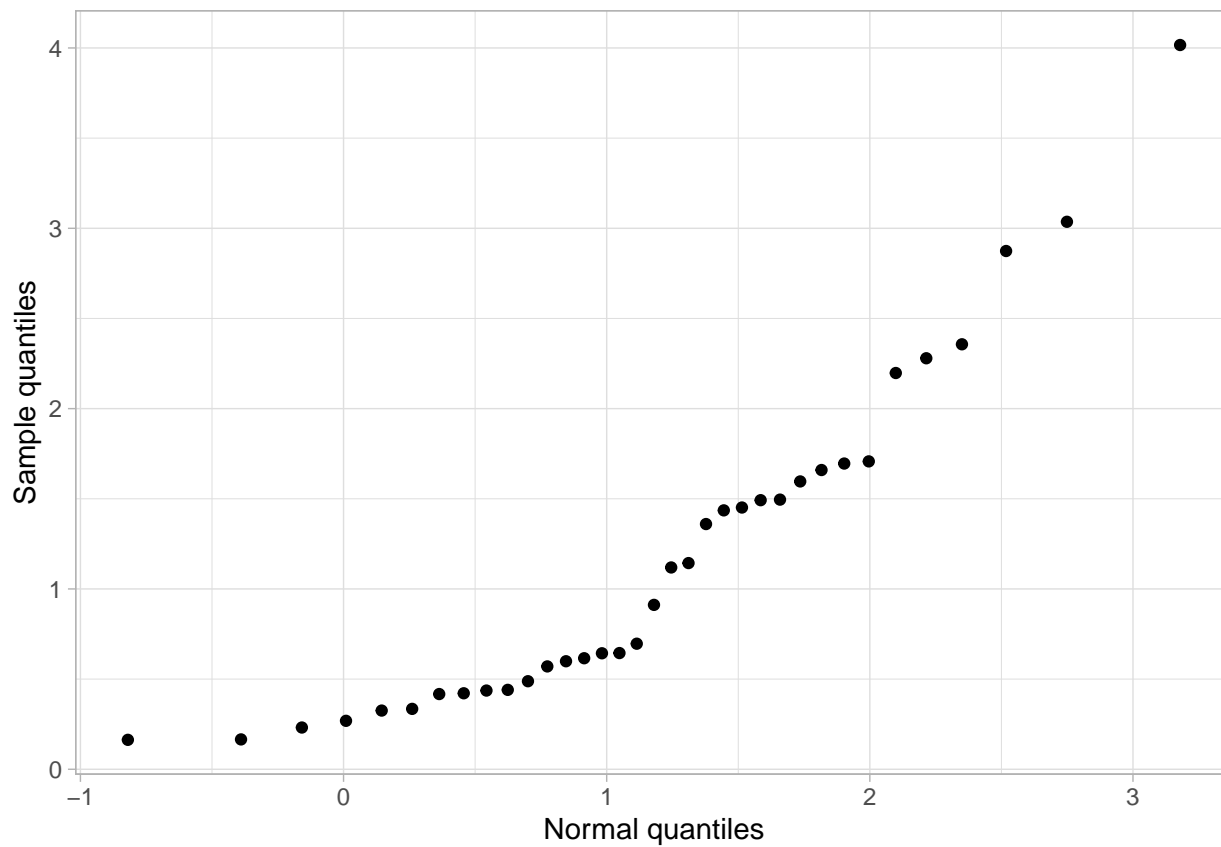
laby2

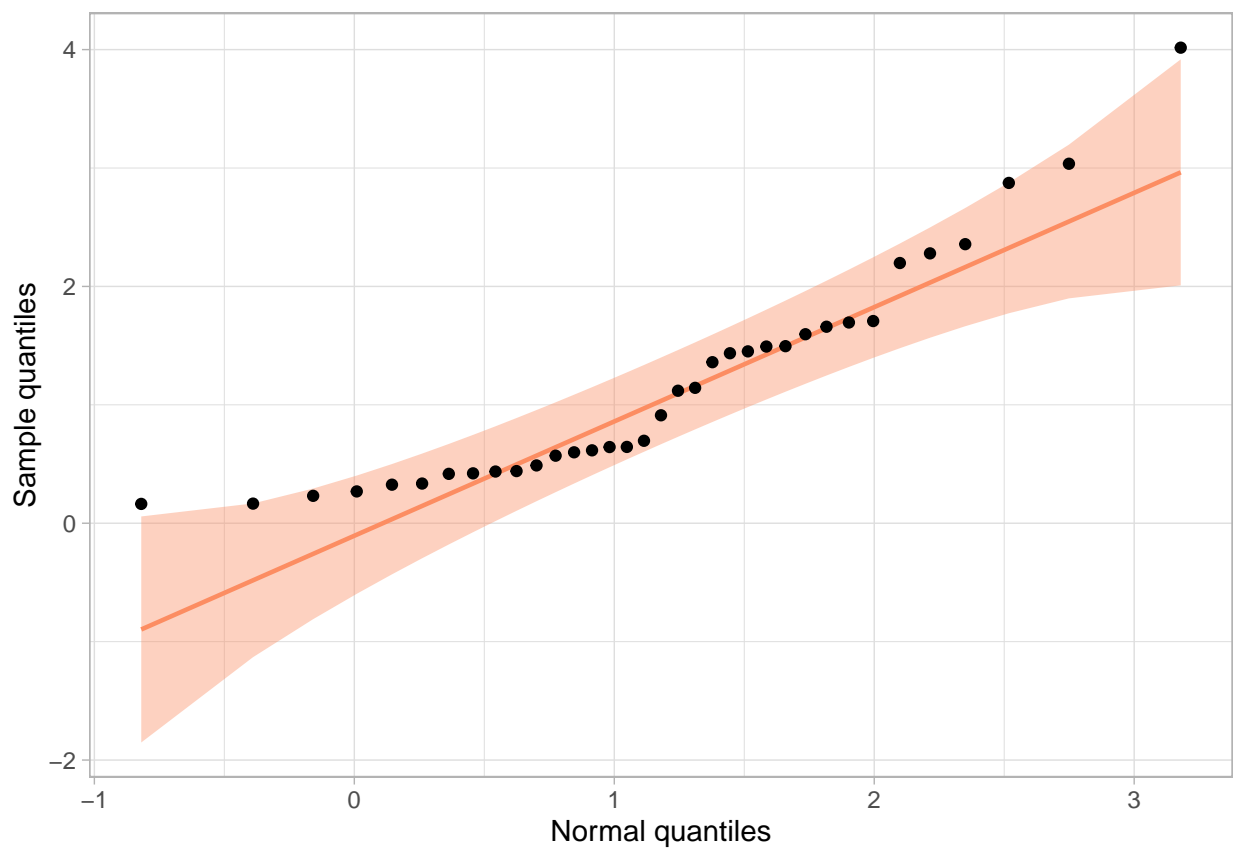
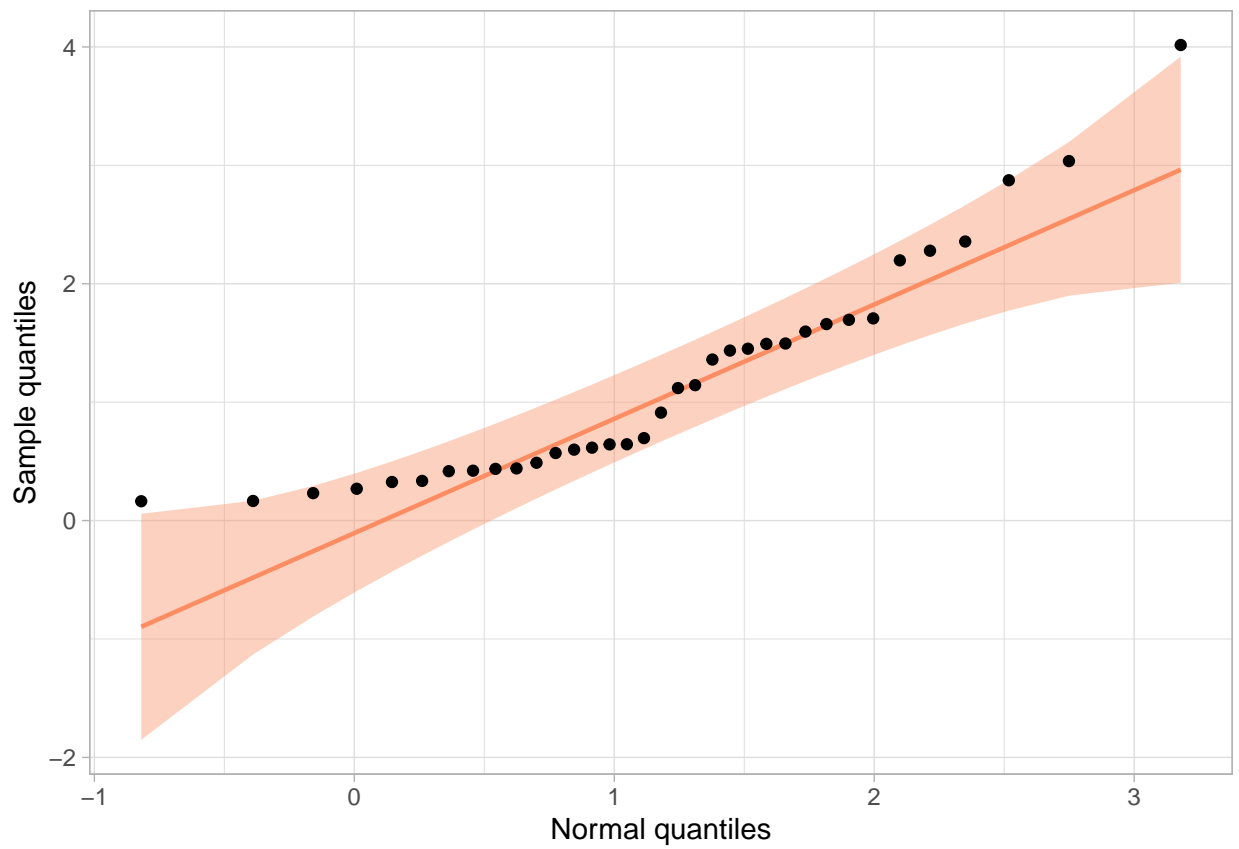
Jakub Szypuła
3/3/2020

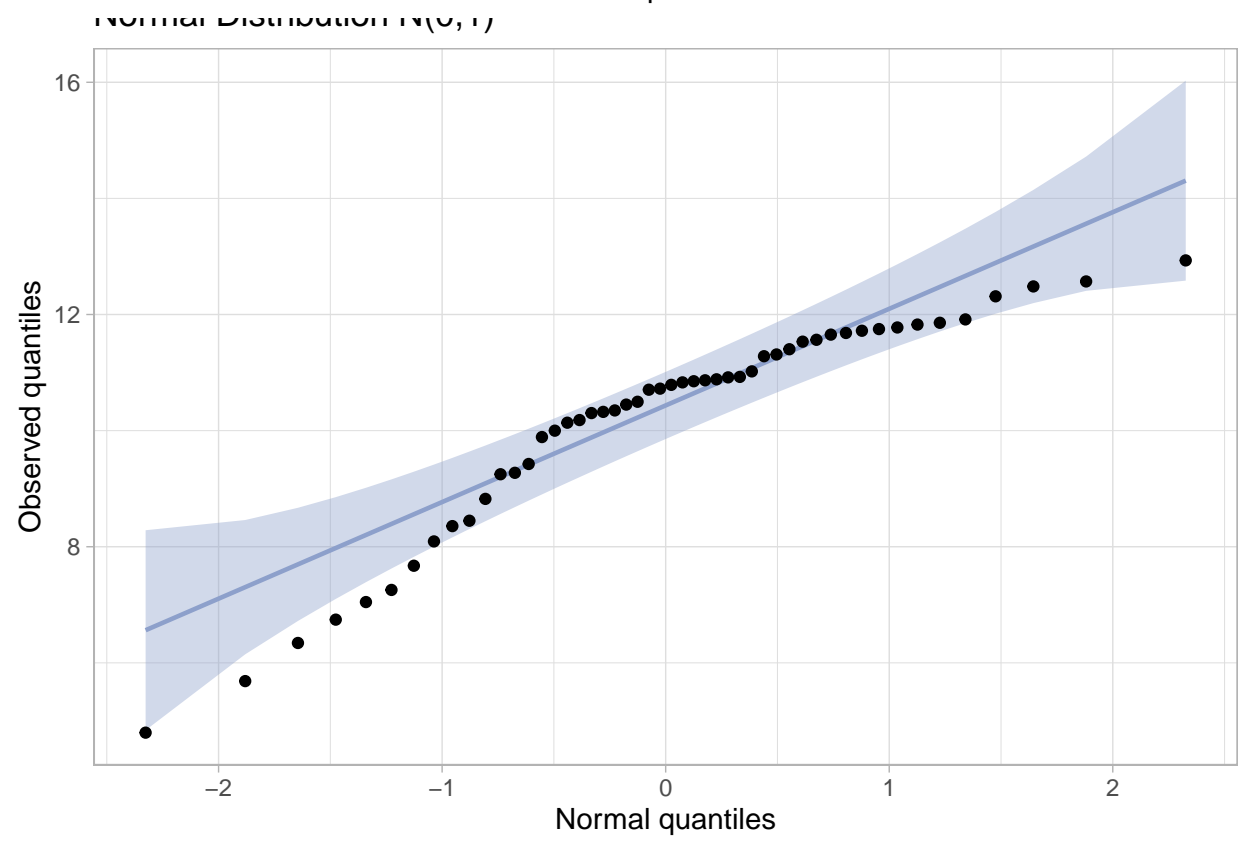
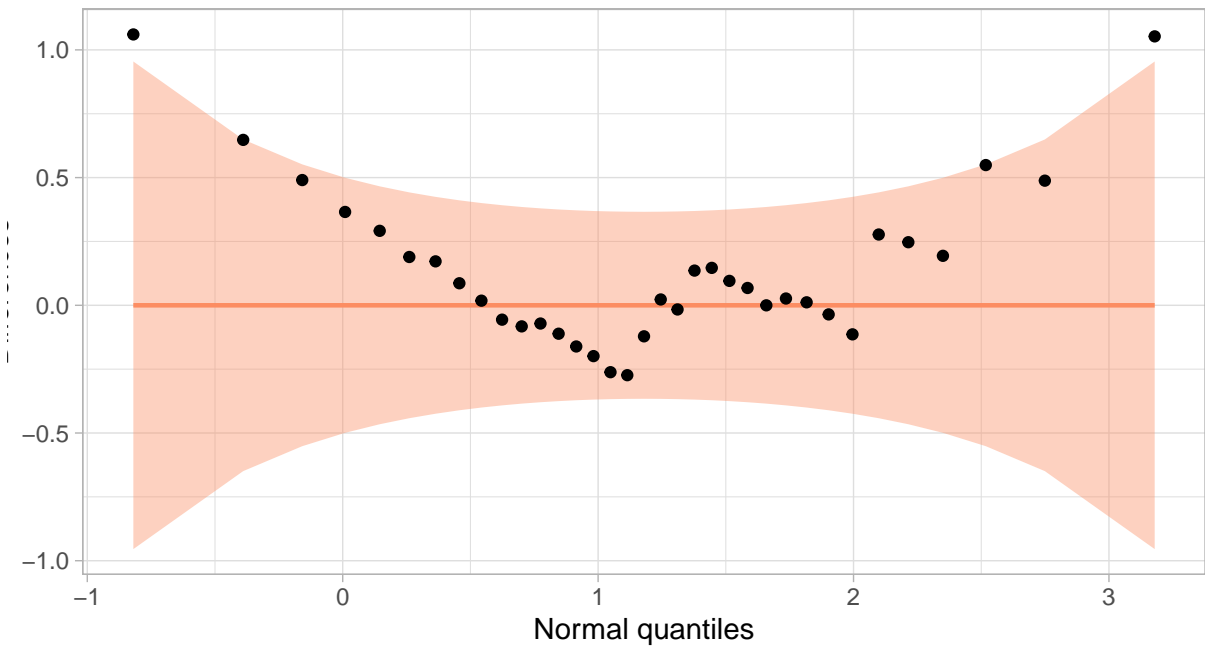
Wykresy

Zreprodukowane wykresy poniżej:

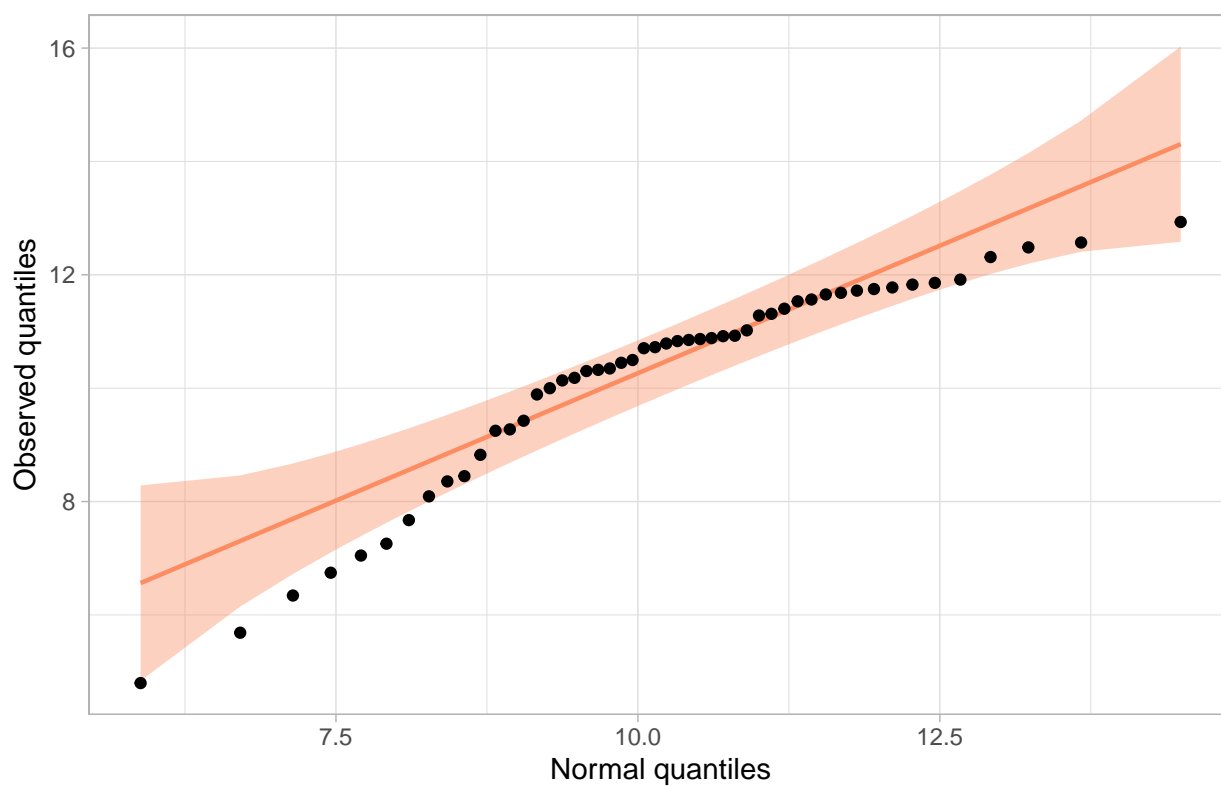




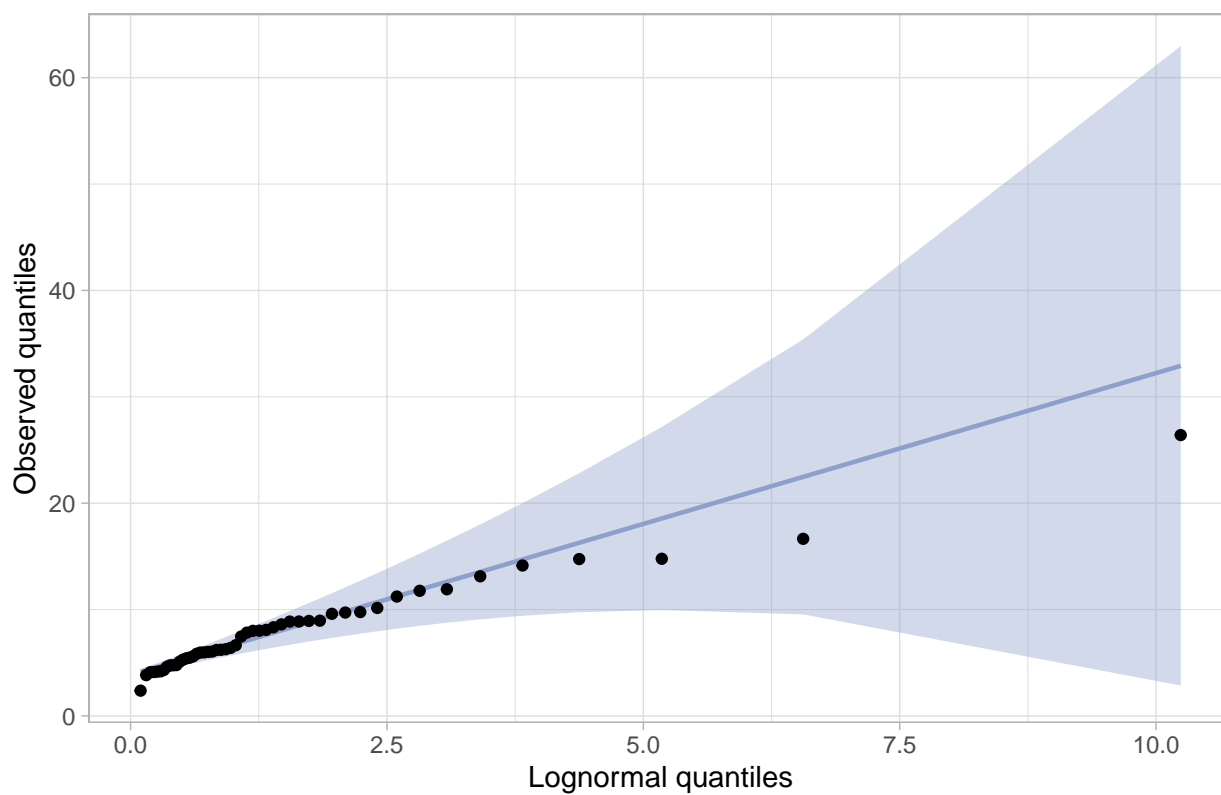




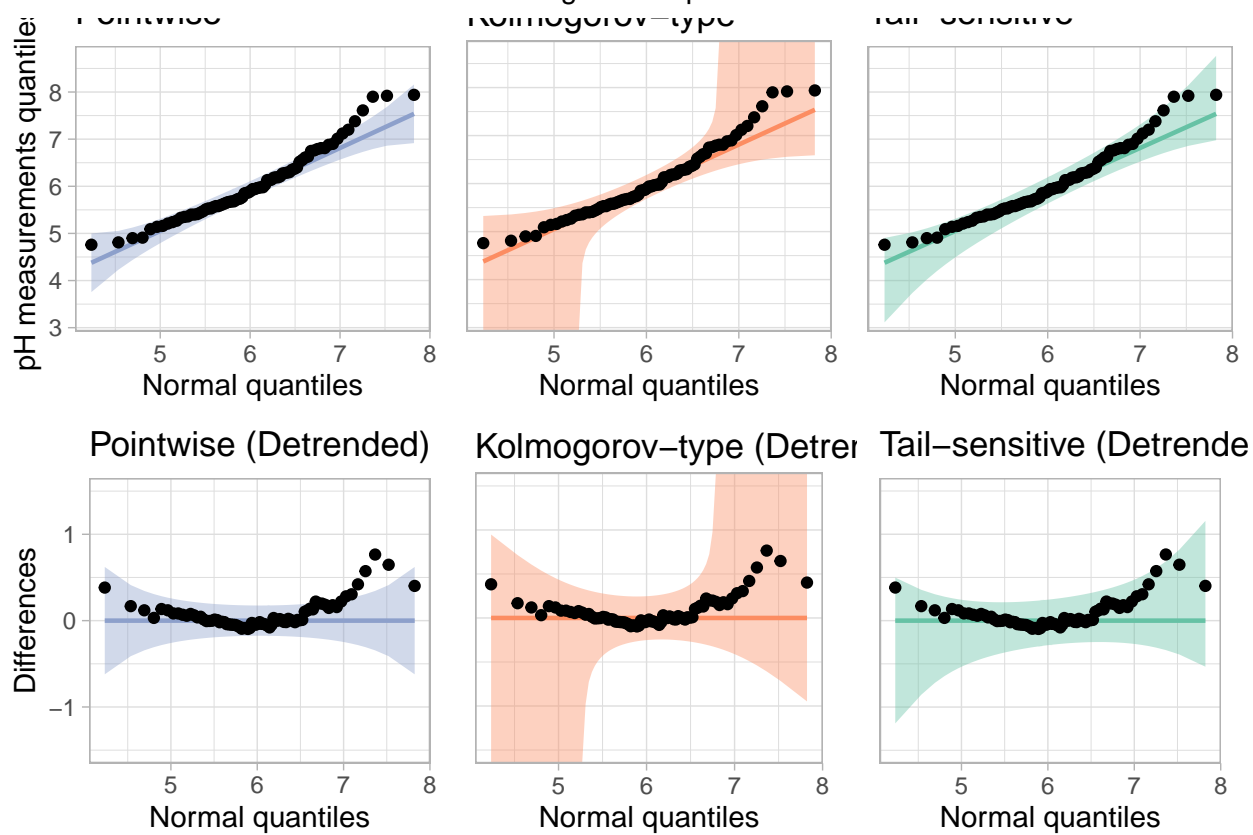
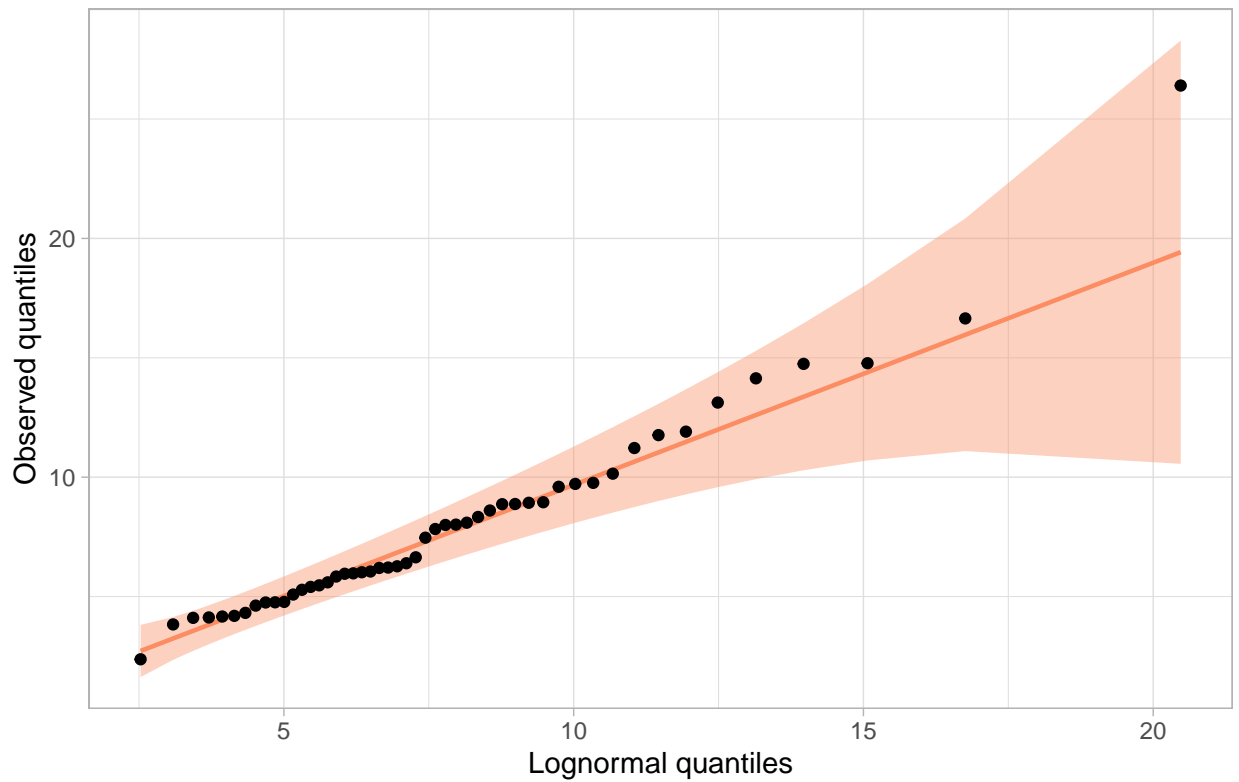
Normal Distribution $N(\hat{\mu}, \hat{\sigma}^2)$

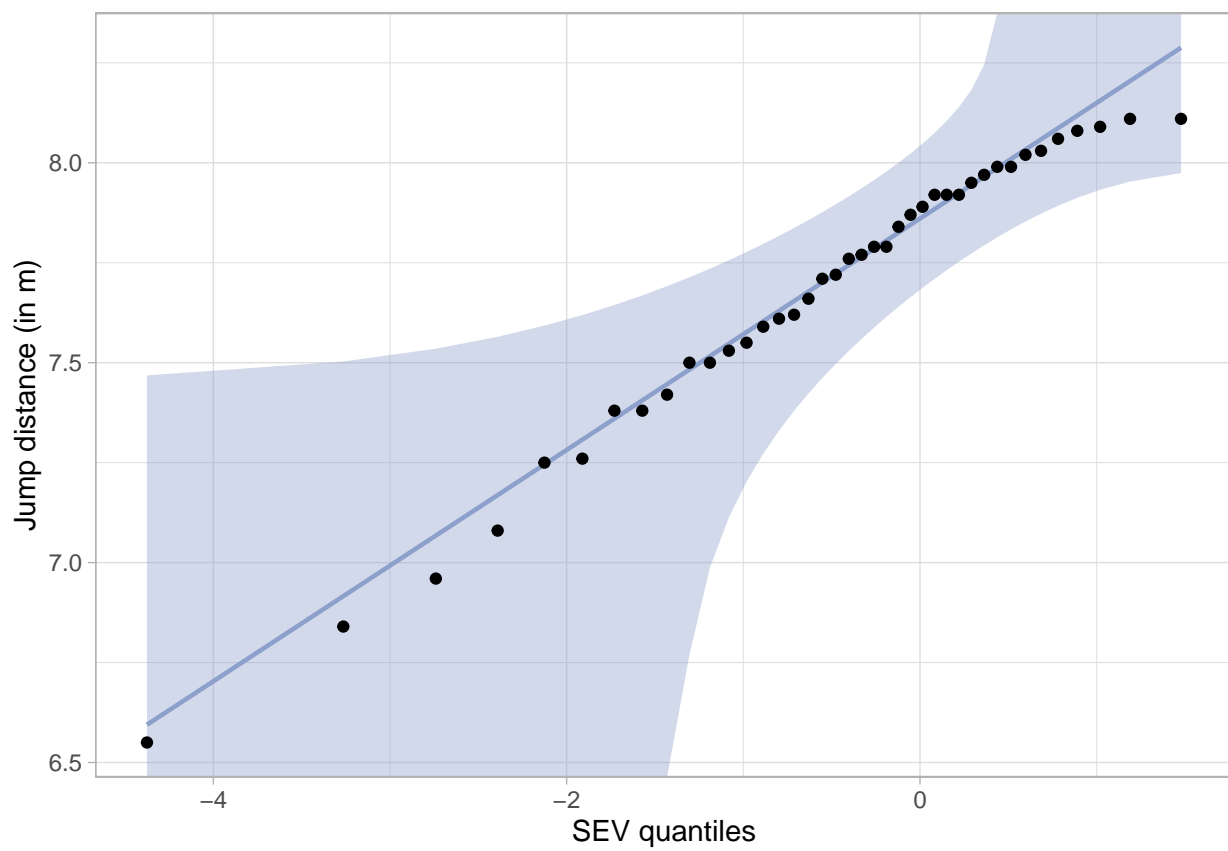
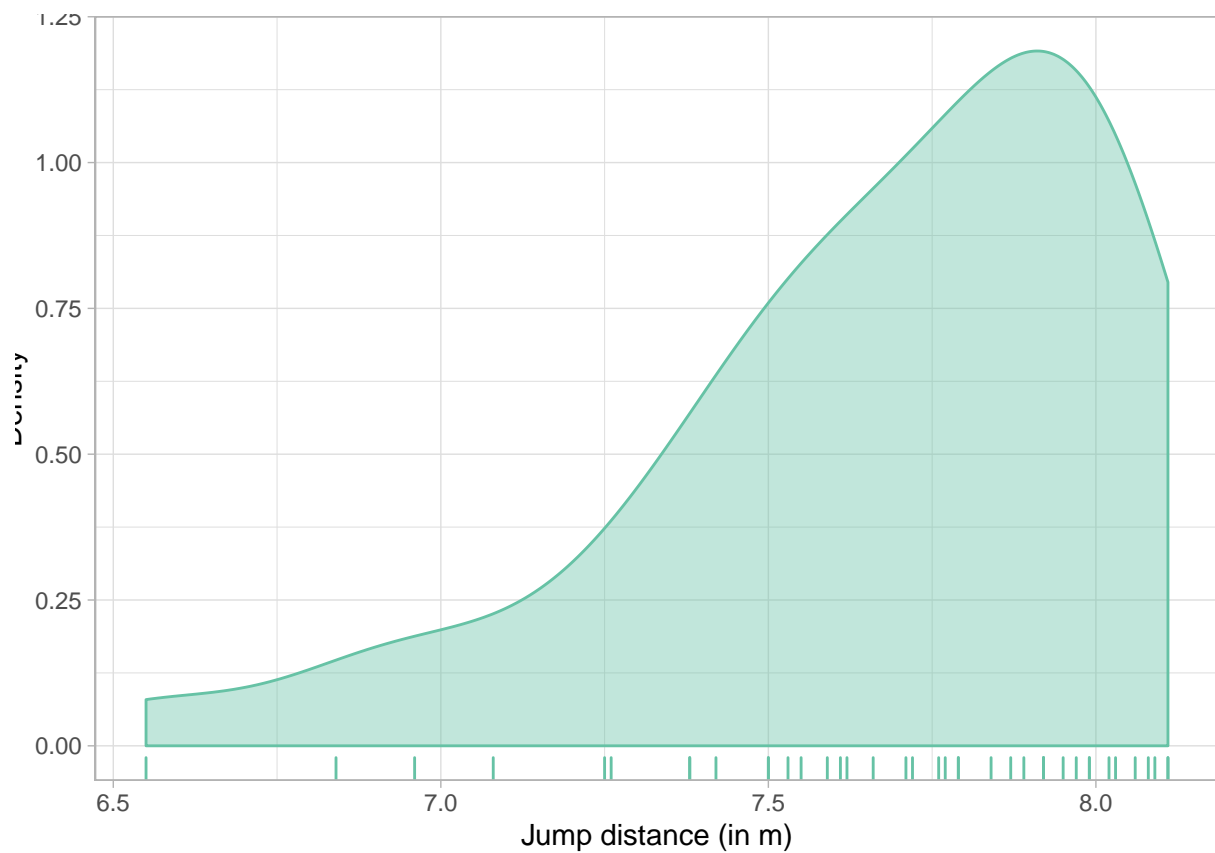


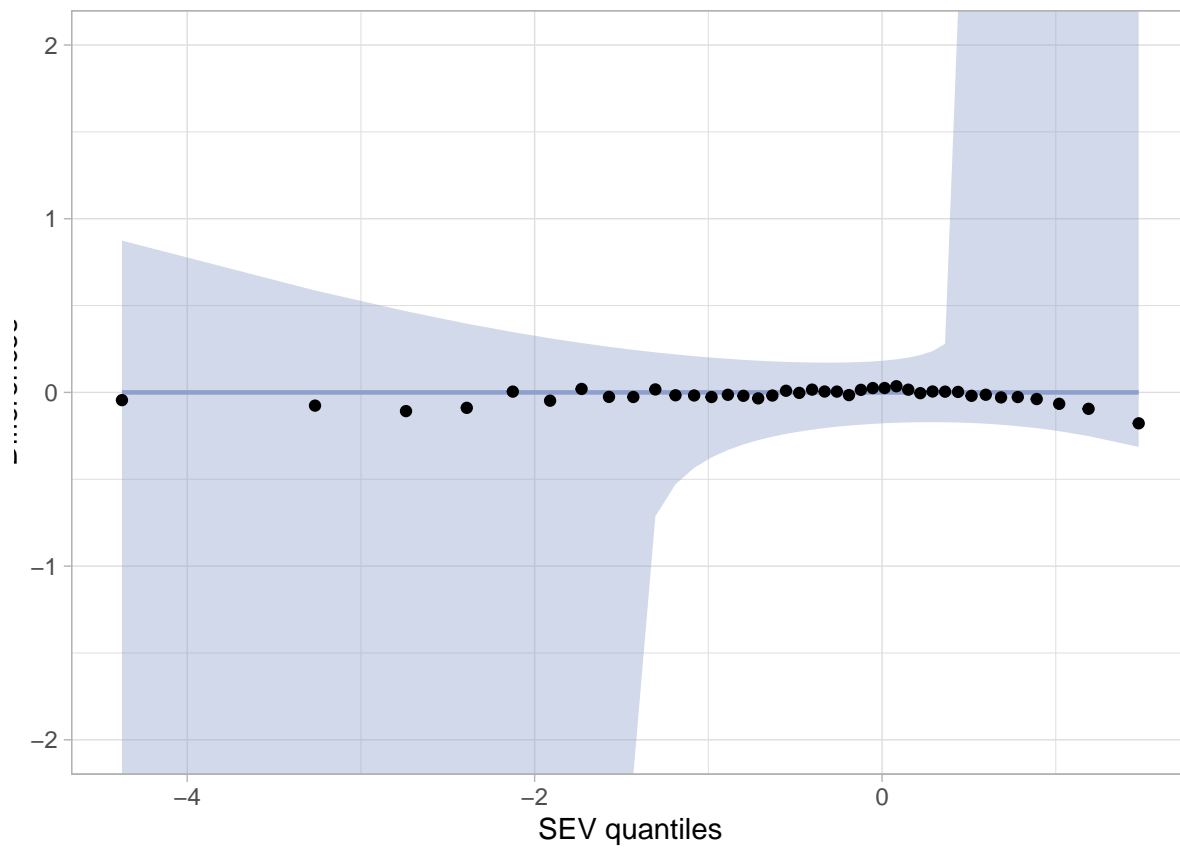
Lognormal Distribution $LN(0, 1)$



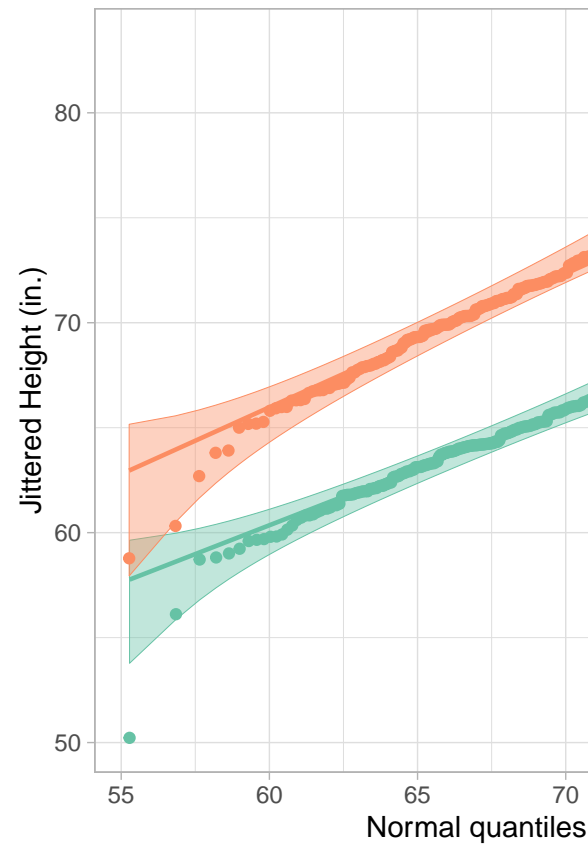
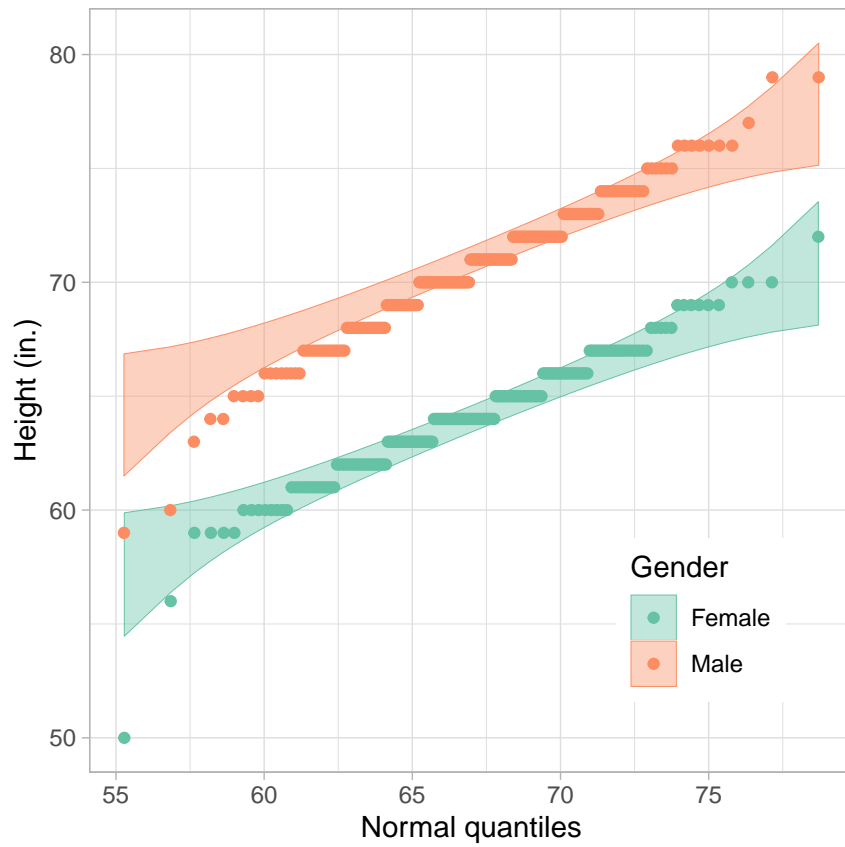
Lognormal Distribution $LN(\hat{\mu}, \hat{\sigma}^2)$



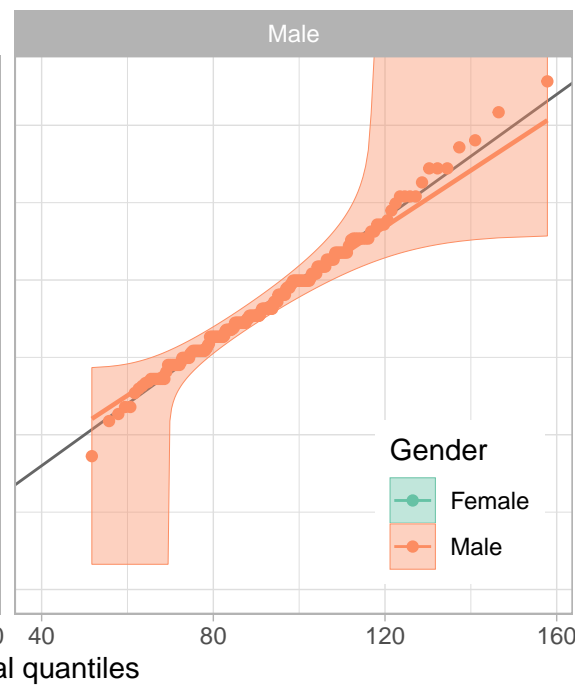
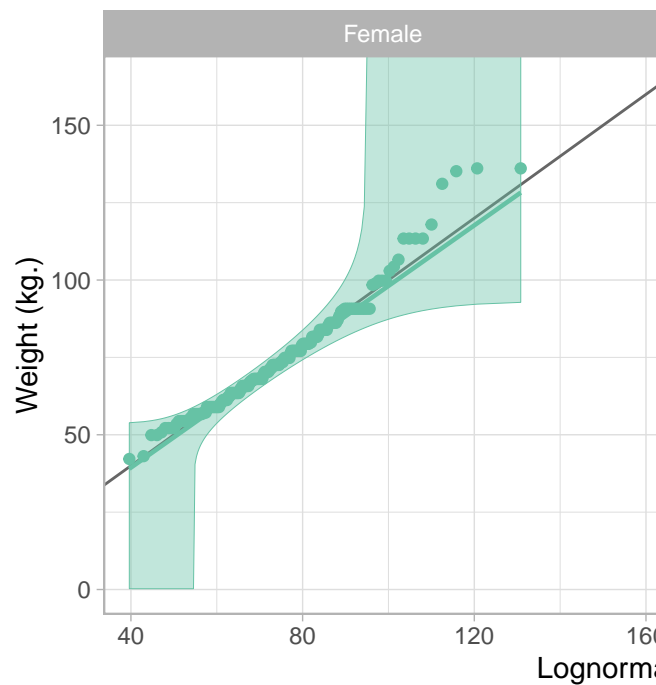
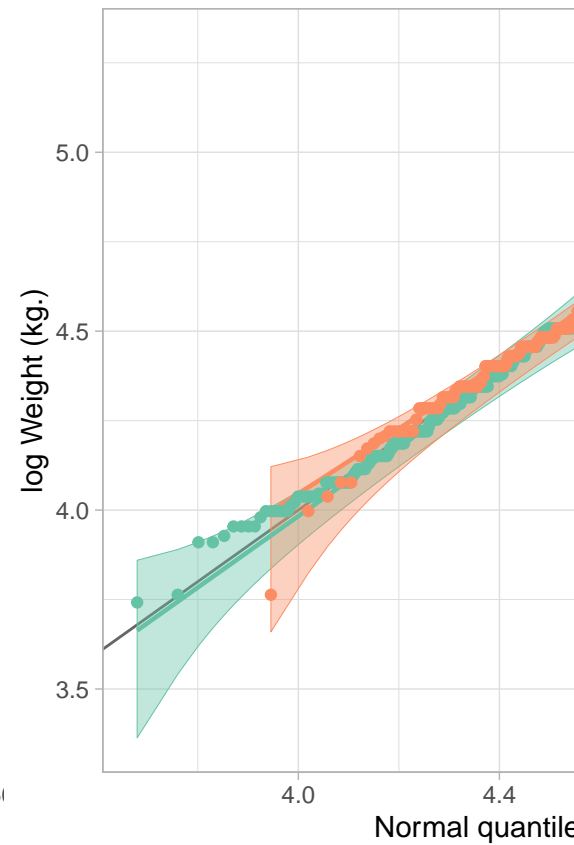
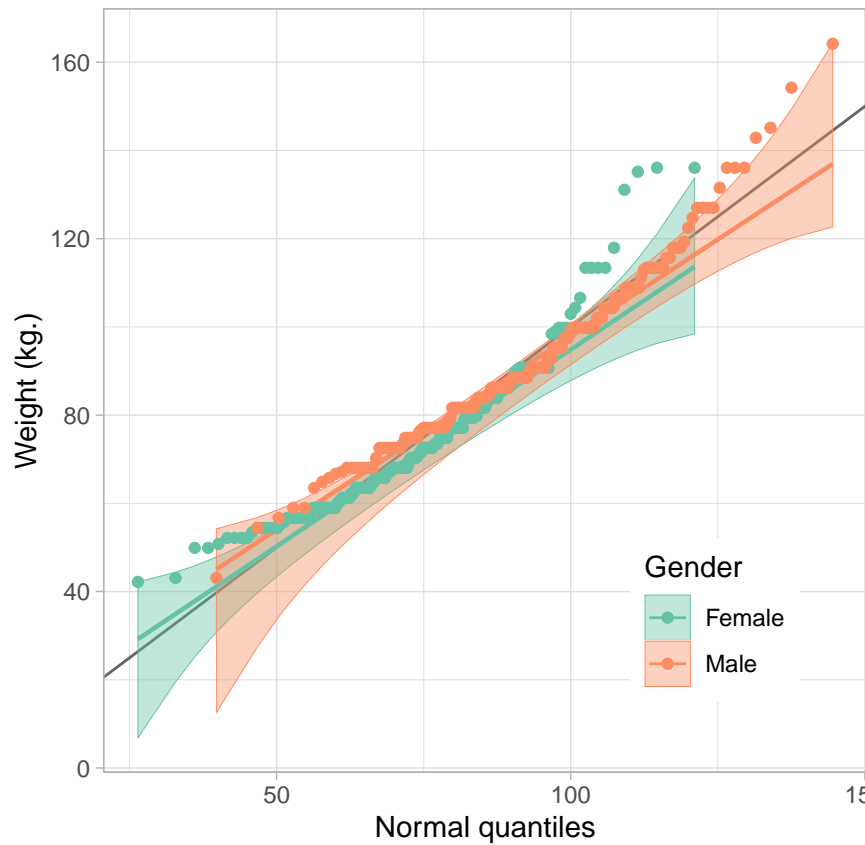




```
## Warning: All elements of `...` must be named.
## Did you want `data = c(WTKG3, HTIN4)`?
```

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



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

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

## [1] -0.9589243
print(.)

## [1] 5
5 %>% {1 + .}

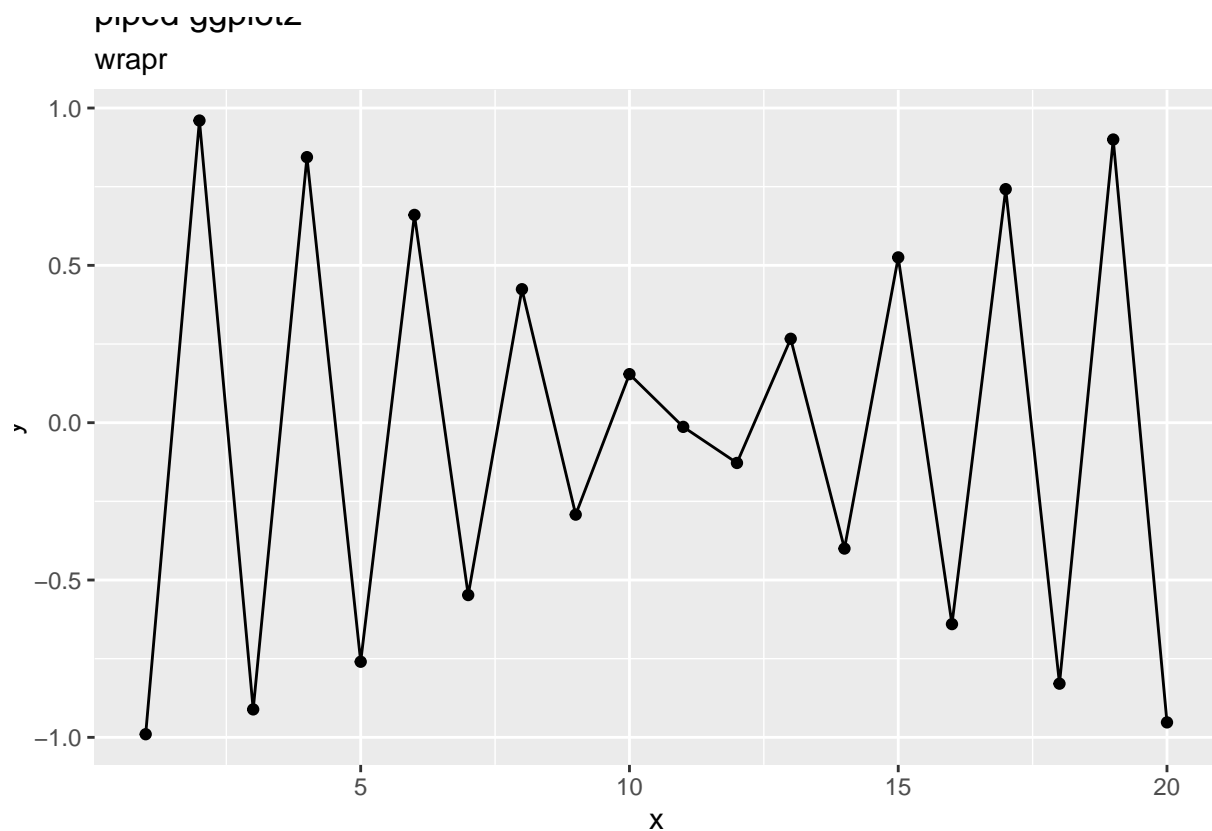
## [1] 6
5 %>% (1 + .)

## [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")
```



```
##
## Attaching package: 'rquery'
## The following object is masked from 'package:ggplot2':
##
##   arrow
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"
# get a database connection
db = DBI::dbConnect(RSQLite::SQLite(),
  ":memory:")
# make our db connection available to rquery package
options(list("rquery.rquery_db_executor" = list(db = db)))
data.frame(x = 1:3) %>% optree # apply optree to d

##   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

```

```

library("magrittr")
5 %>% substitute

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

## <simpleError in .::base: unused argument (sin)>
library("wrapr")
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

# get a database connection
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
```