Lab Report 2: Spotify

Gunchica Bhalla

1. Distribution Characteristics

```
knitr::opts_chunk$set(message=FALSE, warning=FALSE)
## Read data here
library('tidyverse')
```

```
## ── Attaching packages ─────────────────────────────────────────────────
──────────────────────────────────── tidyverse 1.3.0 ──
```

```
## ✓ ggplot2 3.3.1     ✓ purrr   0.3.4
## ✓ tibble  3.0.1     ✓ dplyr   1.0.0
## ✓ tidyr   1.1.0     ✓ stringr 1.4.0
## ✓ readr   1.3.1     ✓ forcats 0.5.0
```

```
## ── Conflicts ────────────────────────────────────────────────────────
──────────────────────────── tidyverse_conflicts() ──
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
#library('RSQLite')
library('qqtest')
library('lubridate')
```

```
##
## Attaching package: 'lubridate'
```

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

```
library('gganimate')
```

```
## No renderer backend detected. gganimate will default to writing frames to separ
ate files
## Consider installing:
## - the `gifski` package for gif output
## - the `av` package for video output
## and restarting the R session
```

```
library('plotly')
```

```
##
## Attaching package: 'plotly'
```

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

```
## The following object is masked from 'package:stats':
##
##     filter
```

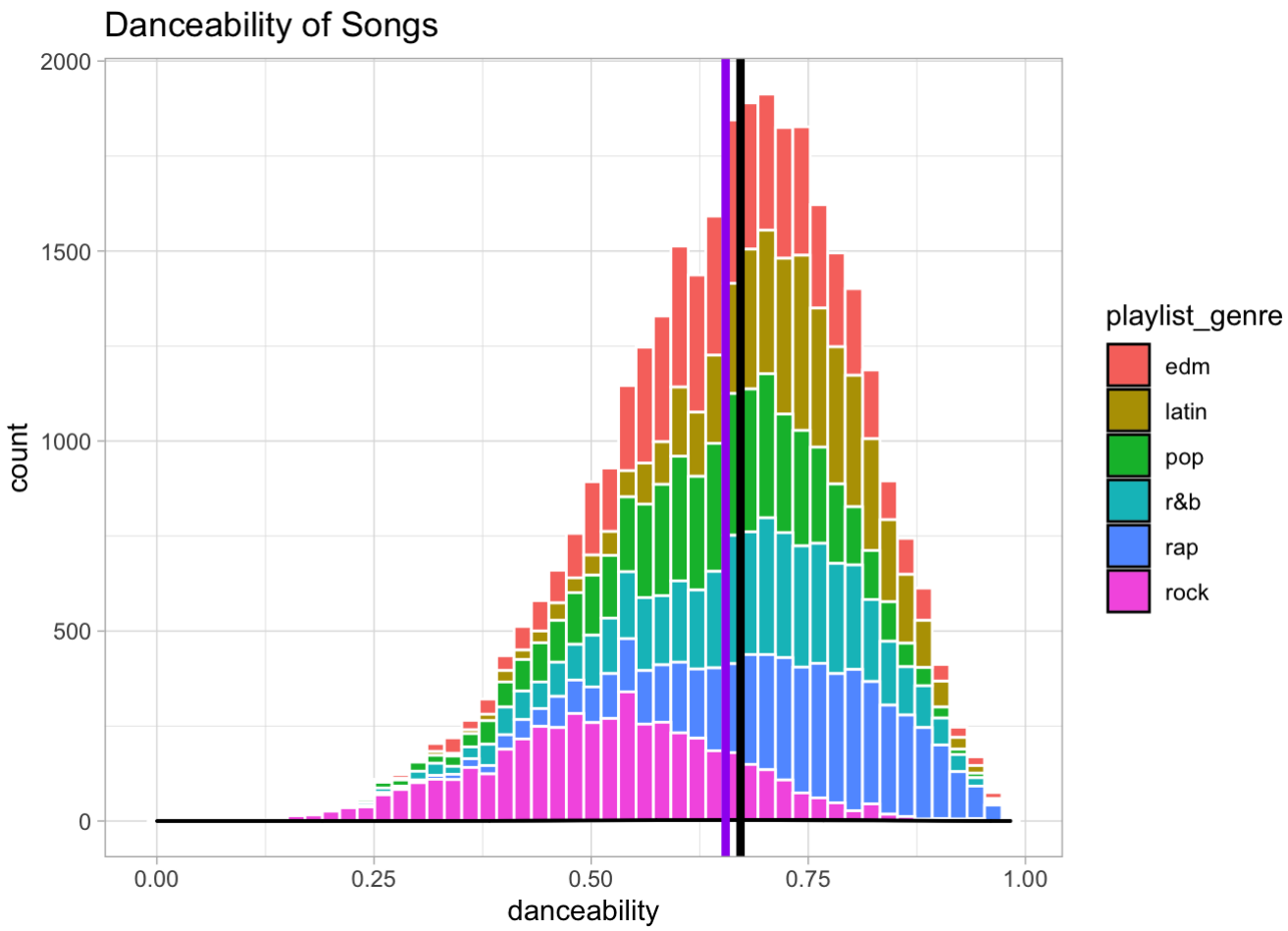
```
## The following object is masked from 'package:graphics':
##
##     layout
```

```
songs <-read.csv('Data/Data-1/spotify_songs.csv')
# ...

## call tidyverse and any other library you'll need ( you can call later if you wa
nt )

str(songs)
```

```
## 'data.frame':    32833 obs. of  23 variables:
##  $ track_id                : chr  "6f807x0ima9a1j3VPbc7VN" "0r7CVbZTWZgbTCYdfa2
P31" "1z1Hg7Vb0AhHDiEmnDE79l" "75FpbthrwQmzHlBJLuGdC7" ...
##  $ track_name              : chr  "I Don't Care (with Justin Bieber) - Loud Lux
ury Remix" "Memories - Dillon Francis Remix" "All the Time - Don Diablo Remix" "Ca
ll You Mine - Keanu Silva Remix" ...
##  $ track_artist            : chr  "Ed Sheeran" "Maroon 5" "Zara Larsson" "The C
hainsmokers" ...
##  $ track_popularity        : int  66 67 70 60 69 67 62 69 68 67 ...
##  $ track_album_id          : chr  "2oCs0DGTsRO98Gh5ZSl2Cx" "63rPSO264uRjW1X5E6c
Wv6" "1HoSmj2eLcsrR0vE9gThr4" "1nqYsOef1yKKuGOVchbsk6" ...
##  $ track_album_name        : chr  "I Don't Care (with Justin Bieber) [Loud Luxu
ry Remix]" "Memories (Dillon Francis Remix)" "All the Time (Don Diablo Remix)" "Ca
ll You Mine - The Remixes" ...
##  $ track_album_release_date: chr  "2019-06-14" "2019-12-13" "2019-07-05" "2019-
07-19" ...
##  $ playlist_name           : chr  "Pop Remix" "Pop Remix" "Pop Remix" "Pop Remi
x" ...
##  $ playlist_id             : chr  "37i9dQZF1DXcZDD7cfEKhW" "37i9dQZF1DXcZDD7cfE
KhW" "37i9dQZF1DXcZDD7cfEKhW" "37i9dQZF1DXcZDD7cfEKhW" ...
##  $ playlist_genre          : chr  "pop" "pop" "pop" "pop" ...
##  $ playlist_subgenre       : chr  "dance pop" "dance pop" "dance pop" "dance po
p" ...
##  $ danceability            : num  0.748 0.726 0.675 0.718 0.65 0.675 0.449 0.54
2 0.594 0.642 ...
##  $ energy                  : num  0.916 0.815 0.931 0.93 0.833 0.919 0.856 0.90
3 0.935 0.818 ...
##  $ key                     : int  6 11 1 7 1 8 5 4 8 2 ...
##  $ loudness                : num  -2.63 -4.97 -3.43 -3.78 -4.67 ...
##  $ mode                    : int  1 1 0 1 1 1 0 0 1 1 ...
##  $ speechiness             : num  0.0583 0.0373 0.0742 0.102 0.0359 0.127 0.062
3 0.0434 0.0565 0.032 ...
##  $ acousticness            : num  0.102 0.0724 0.0794 0.0287 0.0803 0.0799 0.18
7 0.0335 0.0249 0.0567 ...
##  $ instrumentalness        : num  0.00 4.21e-03 2.33e-05 9.43e-06 0.00 0.00 0.0
0 4.83e-06 3.97e-06 0.00 ...
##  $ liveness                : num  0.0653 0.357 0.11 0.204 0.0833 0.143 0.176 0.
111 0.637 0.0919 ...
##  $ valence                 : num  0.518 0.693 0.613 0.277 0.725 0.585 0.152 0.3
67 0.366 0.59 ...
##  $ tempo                   : num  122 100 124 122 124 ...
##  $ duration_ms             : int  194754 162600 176616 169093 189052 163049 187
675 207619 193187 253040 ...
```

```
## Set theme to something else (optional)
theme_set(theme_minimal())
```

```
genre <- unique(songs$playlist_genre)
ggplot(data=songs,aes(x=danceability,fill=playlist_genre))+
  geom_histogram(colour="white",bins=50) +
  geom_density(alpha=0.5) + ggtitle("Danceability of Songs") +
  geom_vline(aes(xintercept=median(danceability,na.rm = T)),colour="black",
             size=1.5) +
  geom_vline(aes(xintercept=mean(danceability,na.rm = T)),colour="purple",size=1.5
)+
  theme_light()
```



The graph represents the danceability and the count of songs with that danceability.The graph looks symetric on glance but is negative (left) skewed as the mean (0.6548495) is smaller than the median(0.672).The mean is represented by the purple line and the median by the black line.The tails are uniformaly thick through out the ranges.

```
library("moments")
fourmoments <- function(rv){
  c('Mean' = mean(rv),
    'Variance' = var(rv),
    'Skewness' = skewness(rv),
    'Kurtosis' = kurtosis(rv))
}

fourmoments(songs$danceability)
```

```
##        Mean    Variance    Skewness    Kurtosis
##  0.65484952  0.02104975 -0.50446539  3.01001783
```

Assuming that the the four moments were to be calculated for the choosen continous varianble (danceability). We see that the skewness is negative thus we know that this is a left skwed graph. As for the Kurtosis the value is very close to 3 (normal distribution) but as an unlikey even it makes this data lepokutic(thick tailed)

```
library("moments")
song <- data.frame('Danceability'= songs$danceability,
                   'Popularity'=songs$track_popularity,
                   'Loudness'= songs$loudness,
                   'Energy'=songs$energy,
                   'Speechiness'=songs$speechiness,
                   'Acoustics'=songs$acousticness,
                   'Instrumentalness'= songs$instrumentalness,
                   'Liveliness'=songs$liveness,
                   'Duration'=songs$duration_ms)
round(apply(song,2,fourmoments),4)
```

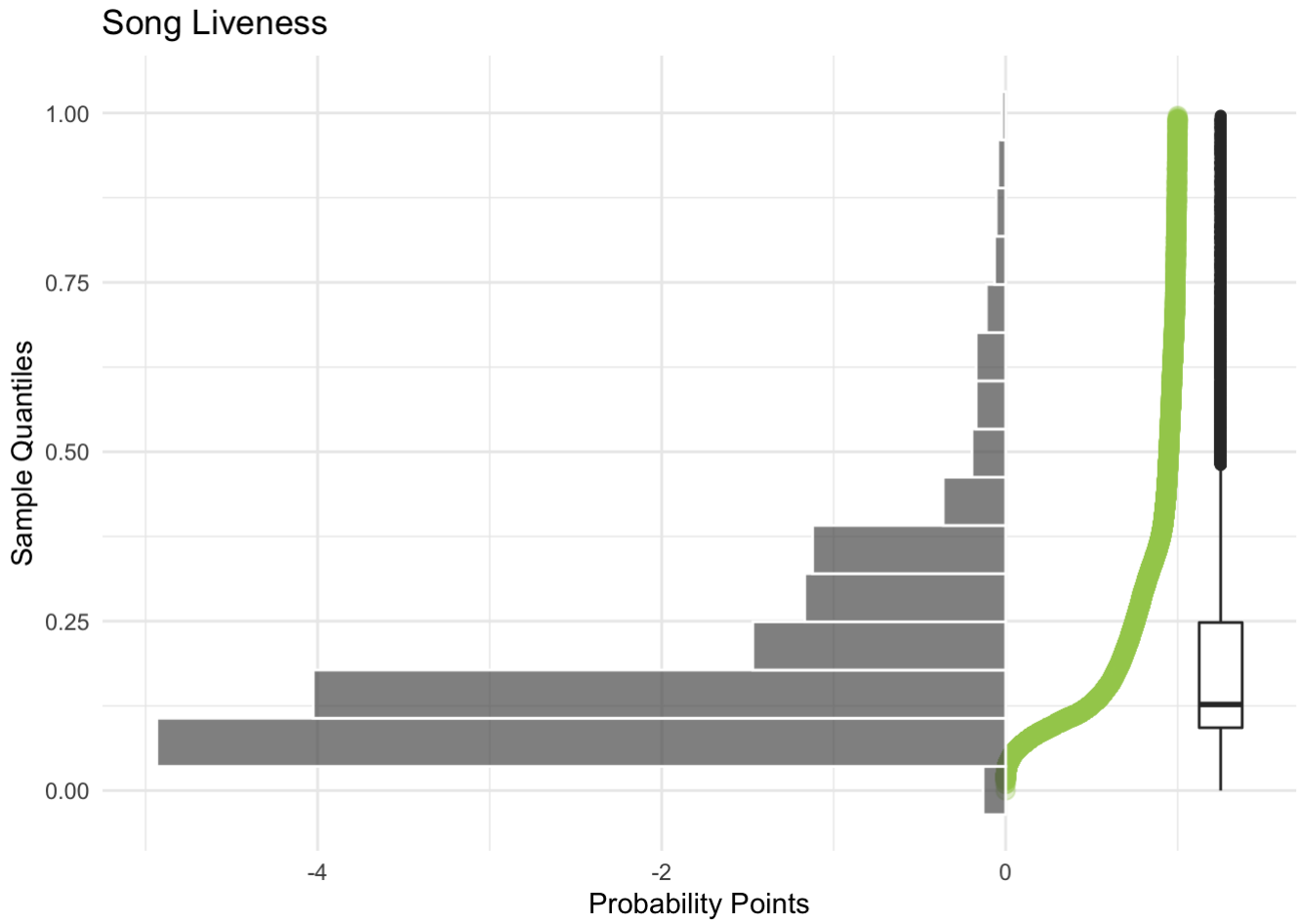
```
##          Danceability Popularity Loudness  Energy Speechiness Acoustics
## Mean           0.6548    42.4771  -6.7195  0.6986      0.1071    0.1753
## Variance       0.0210   624.2040   8.9308  0.0327      0.0103    0.0482
## Skewness      -0.5045    -0.2333  -1.3640 -0.6363      1.9669    1.5947
## Kurtosis       3.0100     2.0672   7.4901  3.0003      7.2600    4.8779
##          Instrumentalness Liveliness     Duration
## Mean               0.0847     0.1902 2.257998e+05
## Variance           0.0503     0.0238 3.580108e+09
## Skewness           2.7593     2.0766 1.149800e+00
## Kurtosis           9.2729     8.0650 5.698600e+00
```

Here we have the four moments for most of the numeric attributes of the sond database. Looking at the attributes individually we can tell that: Danceability is left skewed as the Skewness(-0.5045) is negative. Popularity is aslo left skewed as the Skewness(-0.2333) is negative. Loudness is aslo left skewed as the Skewness(-1.3640) is negative. Energy is aslo left skewed as the Skewness(-0.6363) is negative. Speechiness is right skewed as the Skewness(1.9669) is positive. Acoustics is right skewed as the Skewness(1.5947) is positive. Instrumentalness is right skewed as the Skewness(2.7593) is positive. Liveliness is right skewed as the Skewness(2.0766) is positive. Duration is right skewed as the Skewness(1.149800e+00) is positive. In conclusion danceability, track_popularity, loudness and energy are left skewed and speechinees, acoustics,instrumentalness liveliness and duration are right skewed.

Moving onto the Kurtosis we can see that Instrumentality has the highest value of Kurtosis (9.2729) signifying that the this is data will have the thickest tail in compariosn to the others making the Instrumentality lepokurtic in compariosn to the others. On the other hand Popularity has the lowest Kurtosis value(2.0672) thus this will have the thinnest tail making it platykurtic.

2. Quantile Plot

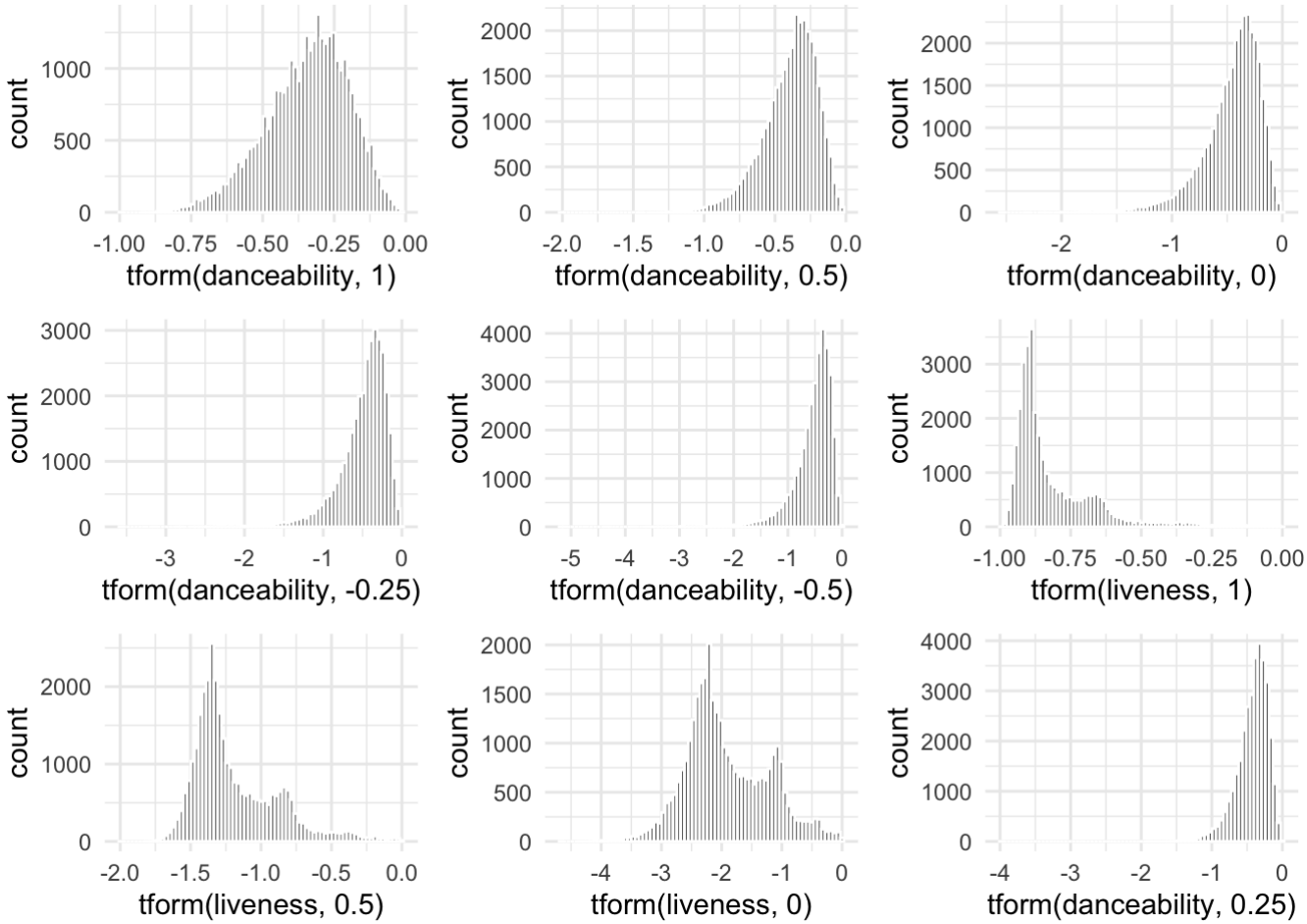
```
ggplot(songs, aes(y=liveness)) +
  geom_point(aes(y=sort(liveness), x=seq(0,1, length.out = length(liveness))), col
our='darkolivegreen3',
             alpha = .3, size =3 ) +
  geom_histogram(aes(x = -..density..), alpha = .7, colour = 'white', bins=15) +
  geom_boxplot(aes(x = 1.25), width=.25) +
  labs(x = 'Probability Points', y= 'Sample Quantiles', title='Song Liveness')
```



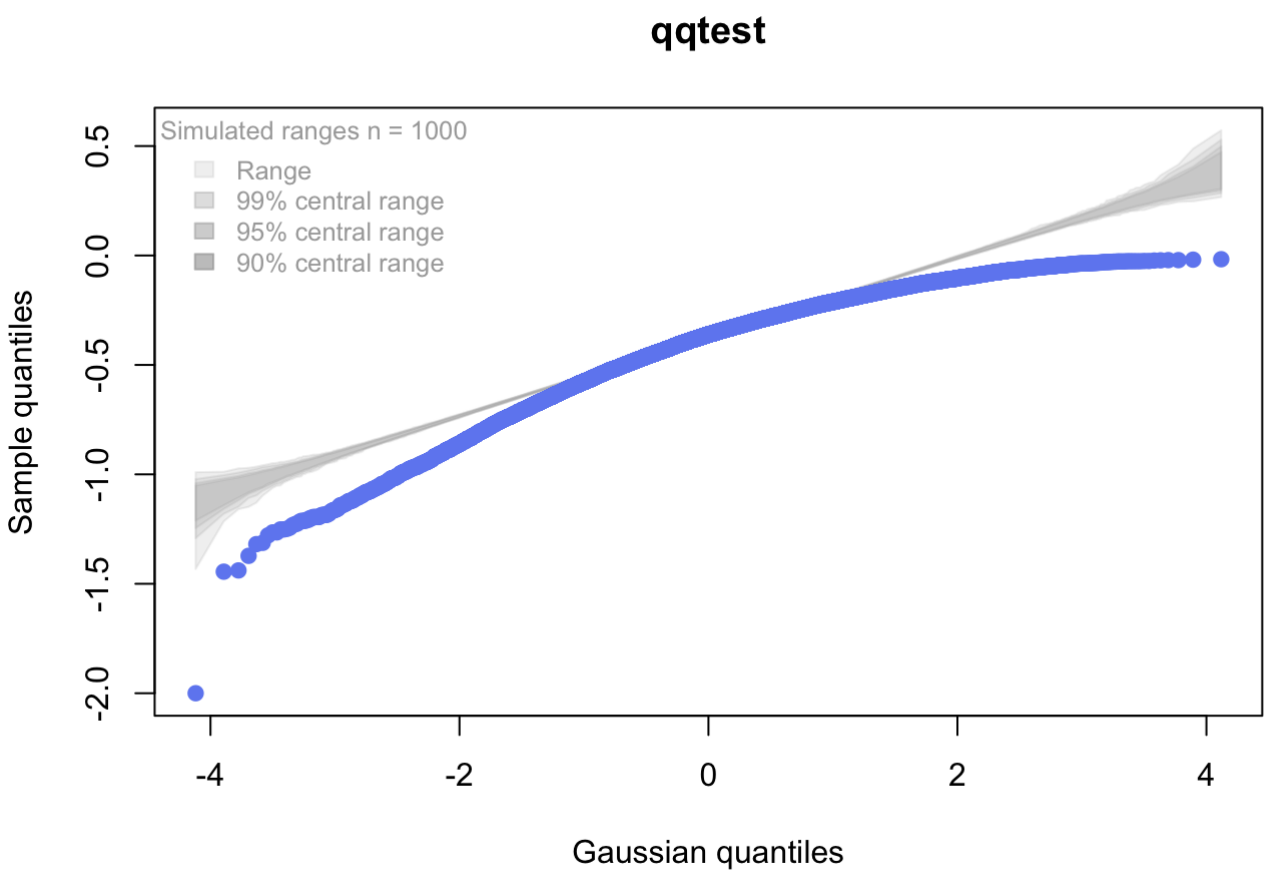
The liveliness attribute of the songs table is unimodal with 0.111 as the mode for the data.The median of the plot is 0.1270 it is the bold black line in the middle of the box The horizontal black line(base of the box) closer to the x-axis is the first quartile whose value is 0.0927 and the thrid quartile(top of the box) is the horizontal black line above the bold line its value os 0.2480. The minimum value is 0 and maxium value is 0.996.

3. Fitting Distributions

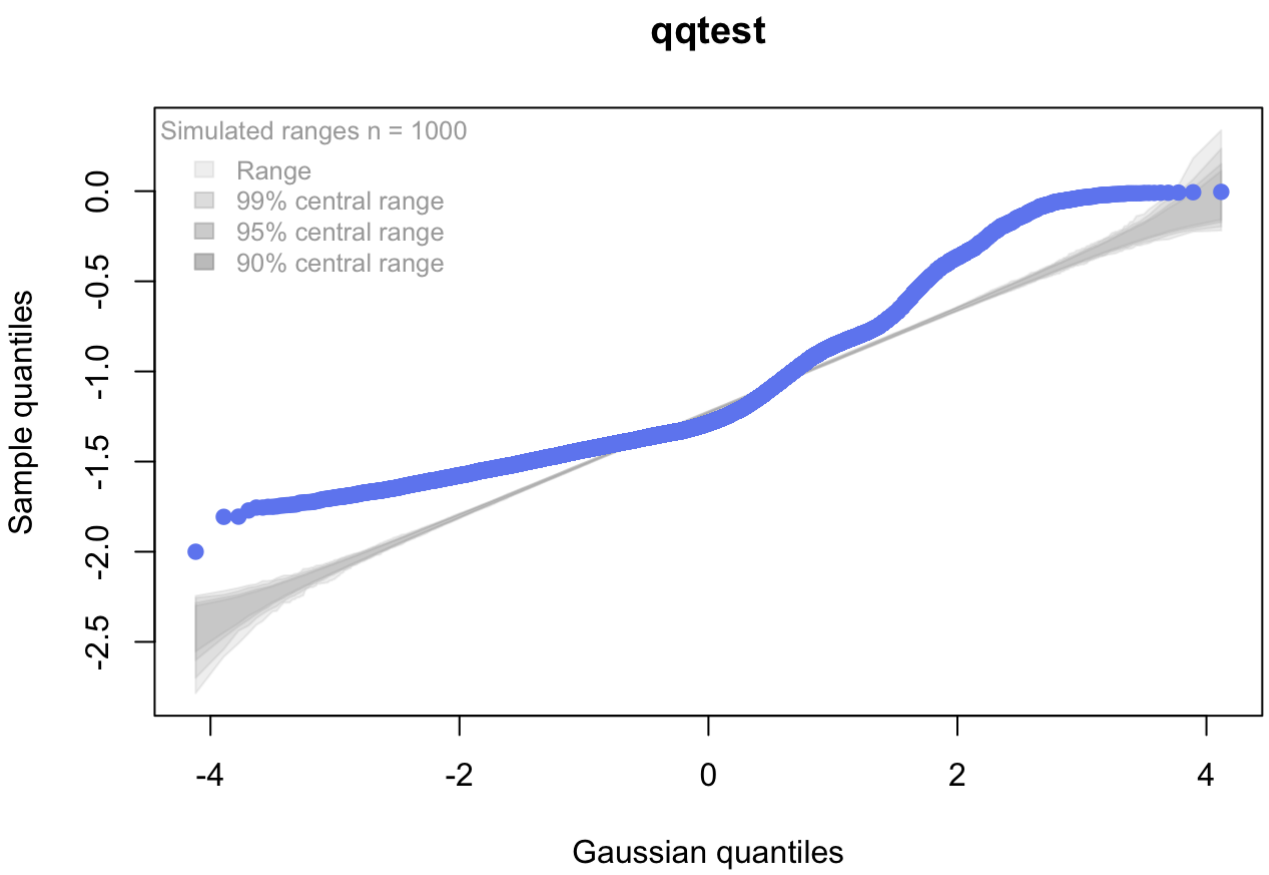
```
tform <- function(x, alpha){
  if(alpha == 0) {log(x)} else {(x^alpha-1)/alpha}
}



library(gridExtra)
g1 <- ggplot(songs, aes(x=tform(danceability,1))) +
  geom_histogram(colour='white',bins = 75)
g2 <- ggplot(songs, aes(x=tform(danceability,0.5))) +
  geom_histogram(colour='white',bins = 75)
g3 <- ggplot(songs, aes(x=tform(danceability,0))) +
  geom_histogram(colour='white',bins = 75)
g7 <- ggplot(songs, aes(x=tform(danceability,-0.25))) +
  geom_histogram(colour='white',bins = 75)
g8 <- ggplot(songs, aes(x=tform(danceability,-0.5))) +
  geom_histogram(colour='white',bins = 75)

g4 <- ggplot(songs, aes(x=tform(liveness,1))) +
  geom_histogram(colour='white',bins = 75)
g5 <- ggplot(songs, aes(x=tform(liveness,0.5))) +
  geom_histogram(colour='white',bins = 75)
g6 <- ggplot(songs, aes(x=tform(liveness,0))) +
  geom_histogram(colour='white',bins = 75)
g9 <- ggplot(songs, aes(x=tform(danceability,0.25))) +
  geom_histogram(colour='white',bins = 75)


grid.arrange(g1,g2,g3,g7,g8,g4,g5,g6,g9)
```




```
qqtest(tform(songs$danceability,0.5))
```

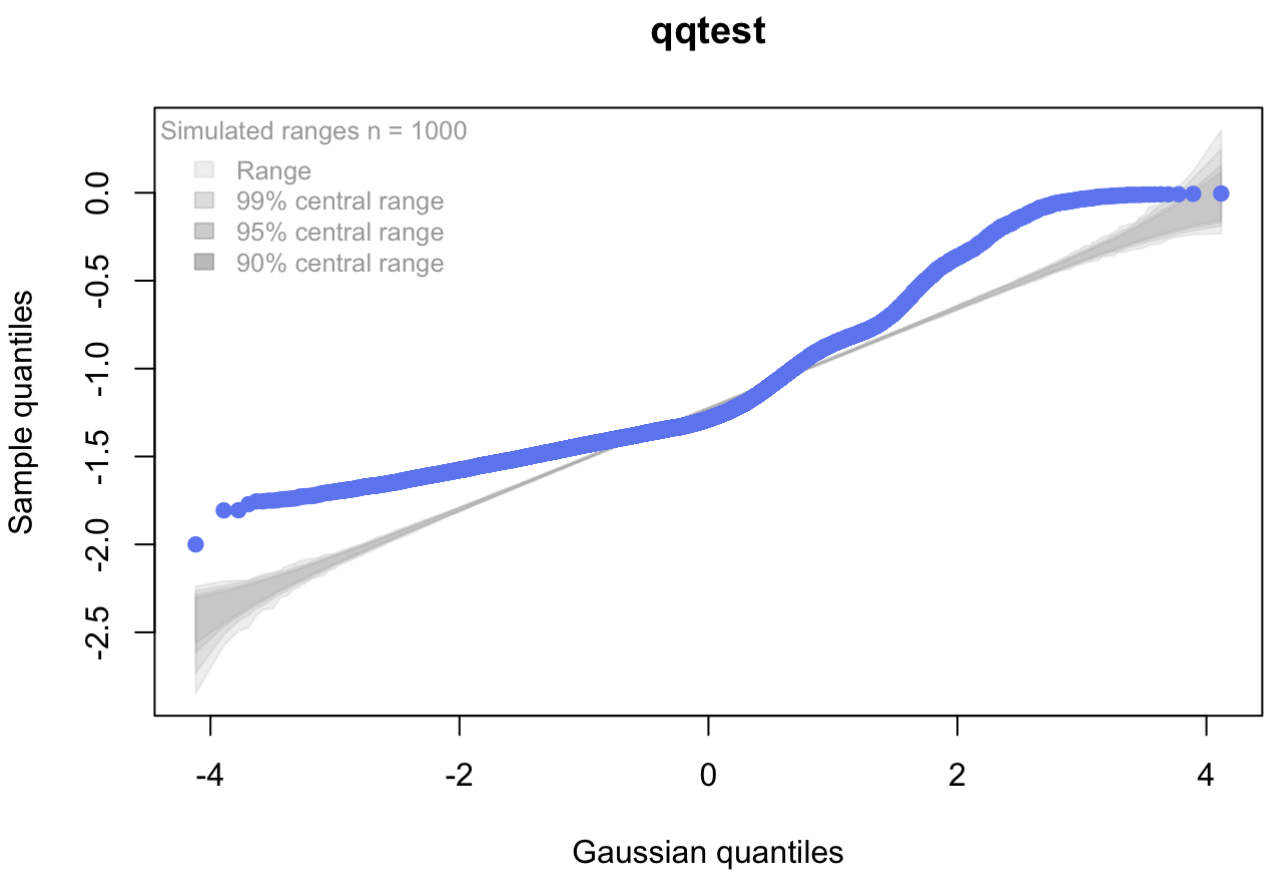



```
qqtest(tform(songs$liveness,0.5))
```



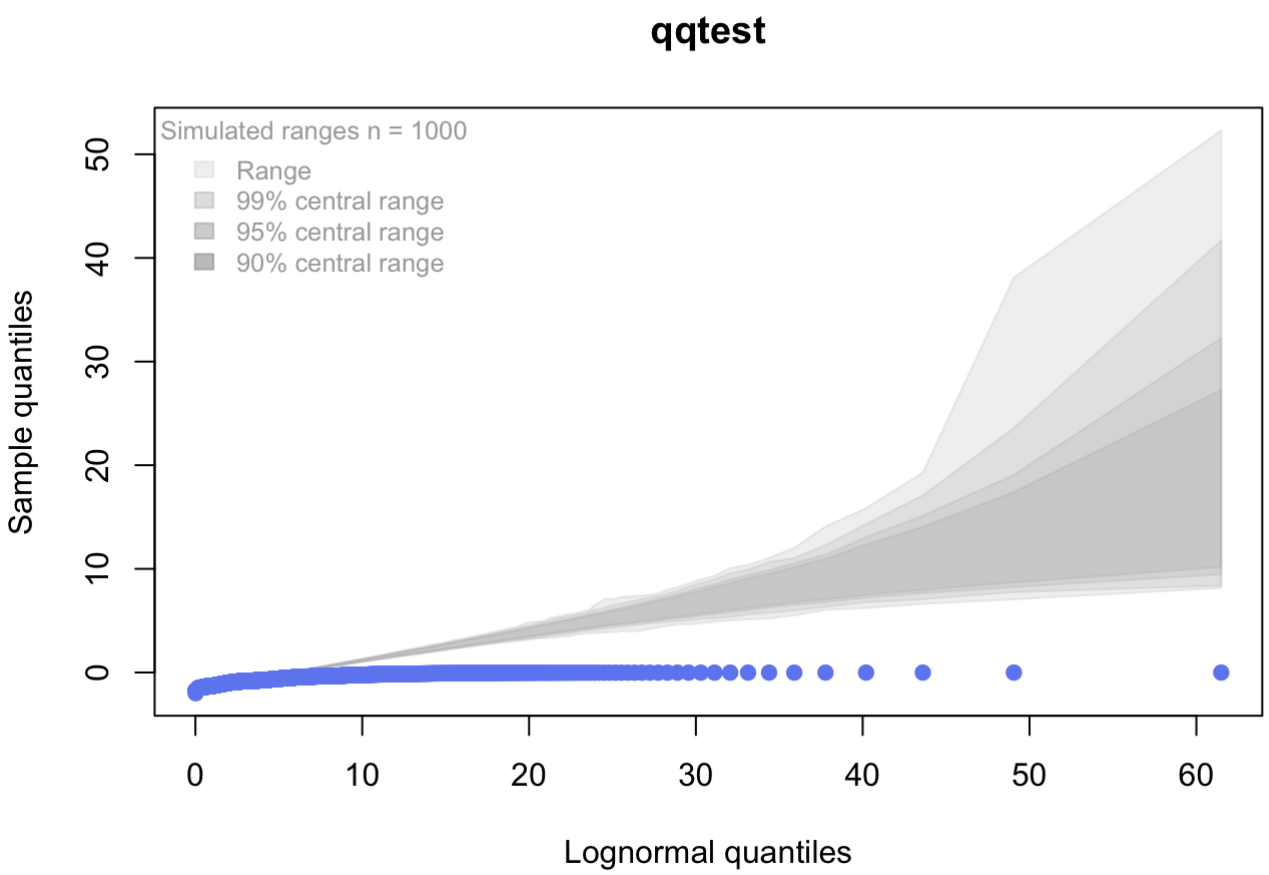
Songs liveness is right skewed but at alpha 0.5 it resembles a normal distribution.

```
qqtest(tform(songs$liveness,0.5),dist = 'normal')
```



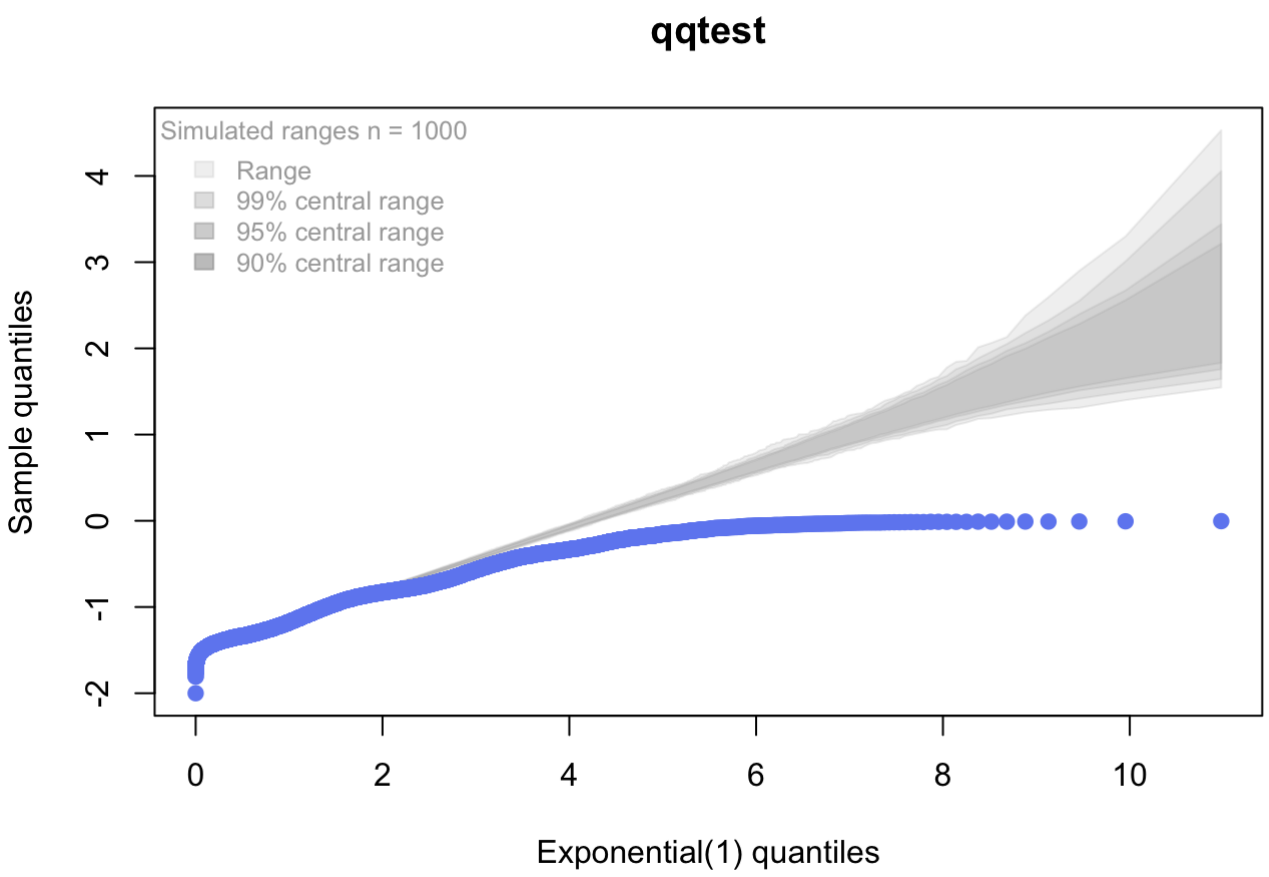
It is not a normal distribution

```
qqtest(tform(songs$liveness,0.5),dist = 'log-normal')
```



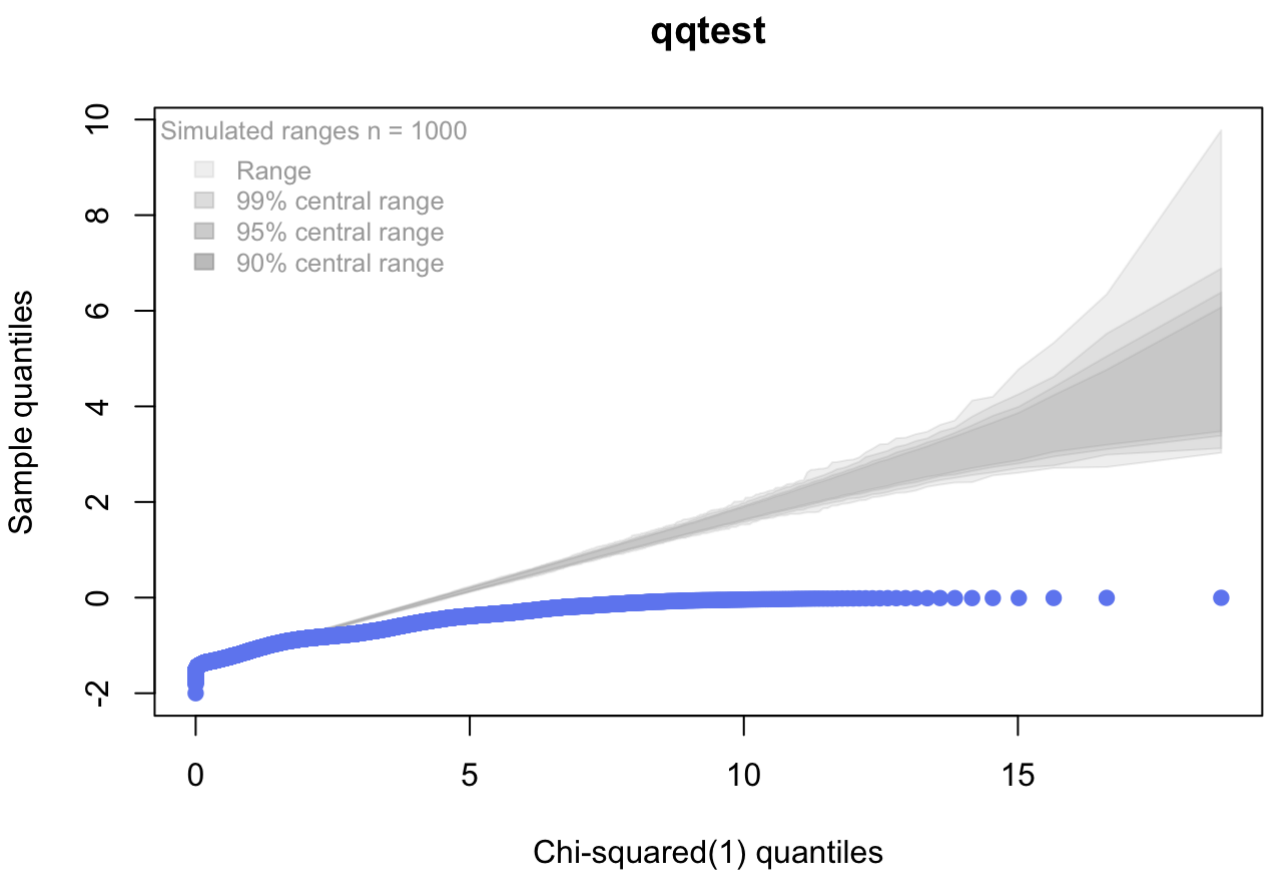
The distribution is not a lognormal distribution

```
qqtest(tform(songs$liveness,0.5),dist = 'exponential')
```



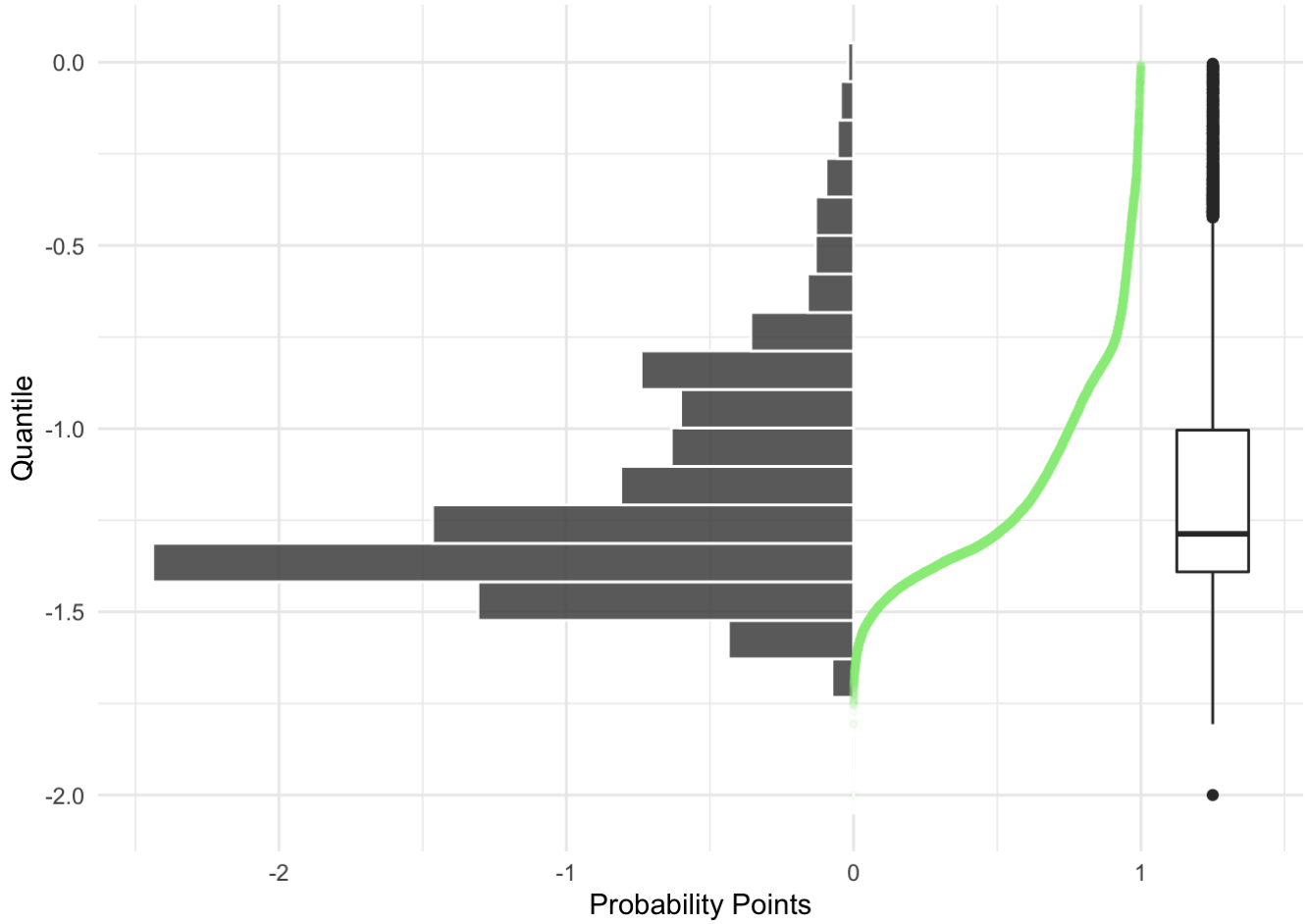
The distribtion is not exponential

```
qqtest(tform(songs$liveness,0.5),dist = 'chi-squared')
```

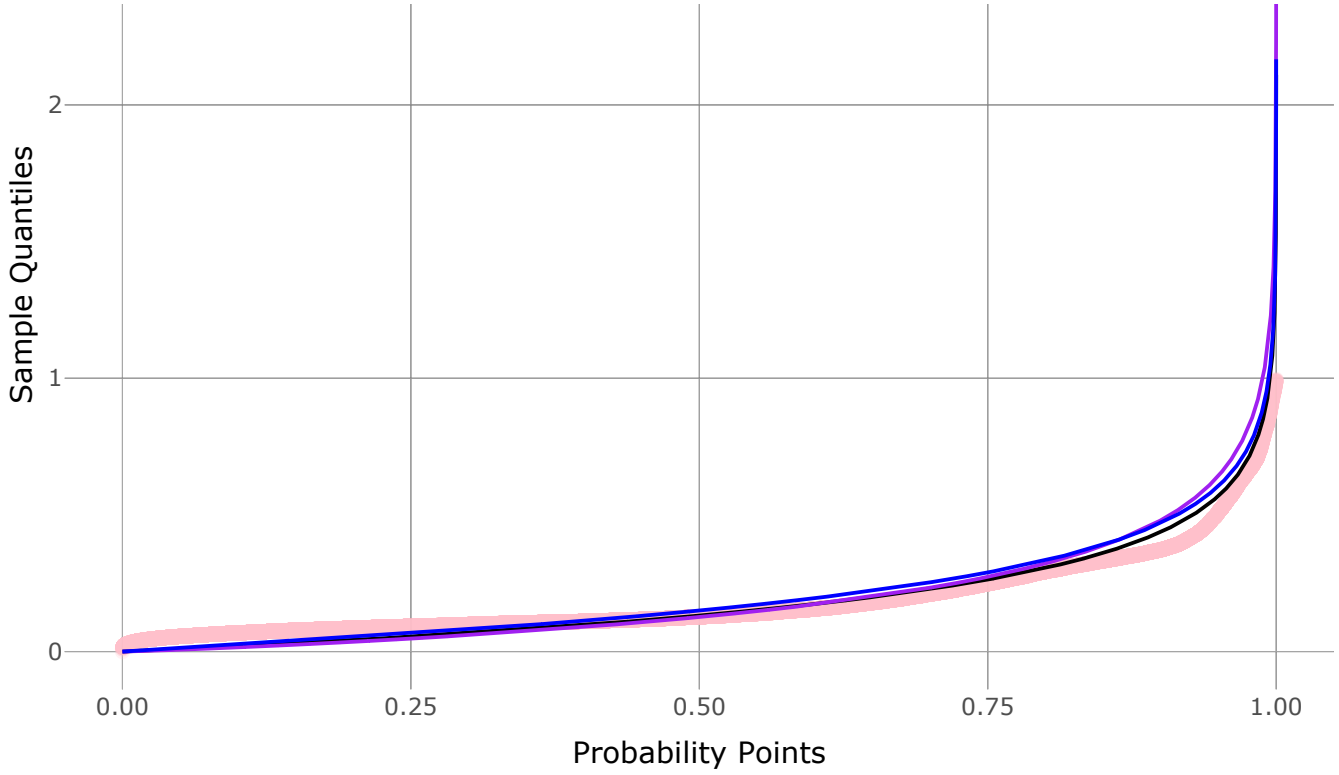


The distribution is not chi-squared

```
sample  <- data.frame(X = tform(songs$liveness,0.5))

ggplot(sample, aes(y=X)) +
  geom_histogram(aes(x = -..density..), alpha = .9, colour = 'white', bins=20) +
  geom_point(aes(y=sort(X), x=ppoints(X)), colour='light green', alpha = .05, size
=1 ) +
  geom_boxplot(aes(x = 1.25), width=.25)  +
  labs(x='Probability Points', y='Quantile')
```




```
library('plotly')
rate <- 1/mean(songs$liveness)

probs    <- ppoints(songs$liveness)
q.exp    <- qexp(probs, rate)
q.wbl    <- qweibull(probs, .9, 1/rate)
q.gmm    <- qgamma(probs, 1.1, rate)
q.sample <- sort(songs$liveness)

g <- ggplot() +
  geom_point(aes(y = q.sample, x=probs), colour = 'pink', alpha = .2) +
  geom_line(aes(y = q.exp,   x=probs)) +
  geom_line(aes(y = q.wbl,   x=probs), colour = 'purple') +
  geom_line(aes(y = q.gmm,   x=probs), colour = 'blue') +
  labs(x = 'Probability Points', y= 'Sample Quantiles')


ggplotly(g)
```


Sample Quantiles
2
1
0
0.00
0.25
0.50
0.75
1.00
Probability Points

4. Summarizing Data

```
d1 <- songs %>%
  mutate(Date=as.Date(track_album_release_date,format='%d-%m-%y')) %>%
  select(c('liveness','playlist_genre','track_album_release_date')) %>%
  mutate(Year = as.Date(track_album_release_date,format= '%Y')) %>%
  group_by(playlist_genre,Year) %>%
  summarize_at(vars(liveness),mean, na.rm=T)
  h <- head(d1)
  t <-tail(d1)
  h
```

playlist_genre <chr>	Year <date>	liveness <dbl>
edm	1977-07-06	0.11000
edm	1978-07-06	0.25700
edm	1979-07-06	0.06605
edm	1980-07-06	0.23100
edm	1981-07-06	0.03670
edm	1982-07-06	0.04235

6 rows

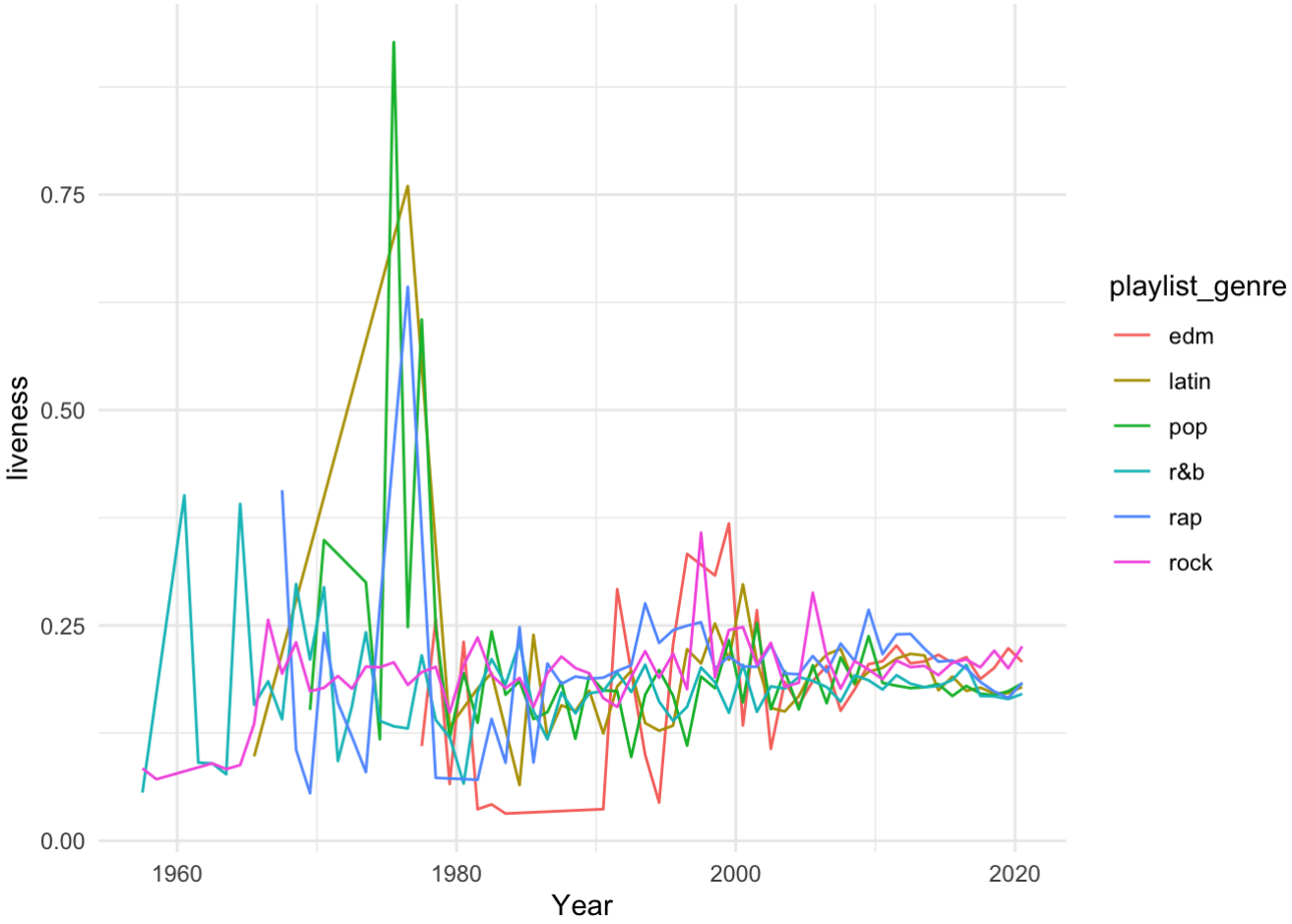
```
  t
```

playlist_genre <chr>	Year <date>	liveness <dbl>
rock	2015-07-06	0.2067241
rock	2016-07-06	0.2105325
rock	2017-07-06	0.2013155
rock	2018-07-06	0.2207062
rock	2019-07-06	0.2002295
rock	2020-07-06	0.2257625

6 rows

5. Plot the Time Series

```
library('gganimate')
ggplot(d1, aes(x=Year, y=liveness, colour =playlist_genre)) +
  geom_line() +
  #transition_reveal(Year)+
  #geom_text(aes(x=Year, label=playlist_genre),hjust=0)+
  theme(legend.position = 'right')
```



6. CLT

```
groups <- sample(1:300, size = nrow(songs), replace = T)
head(groups)
```

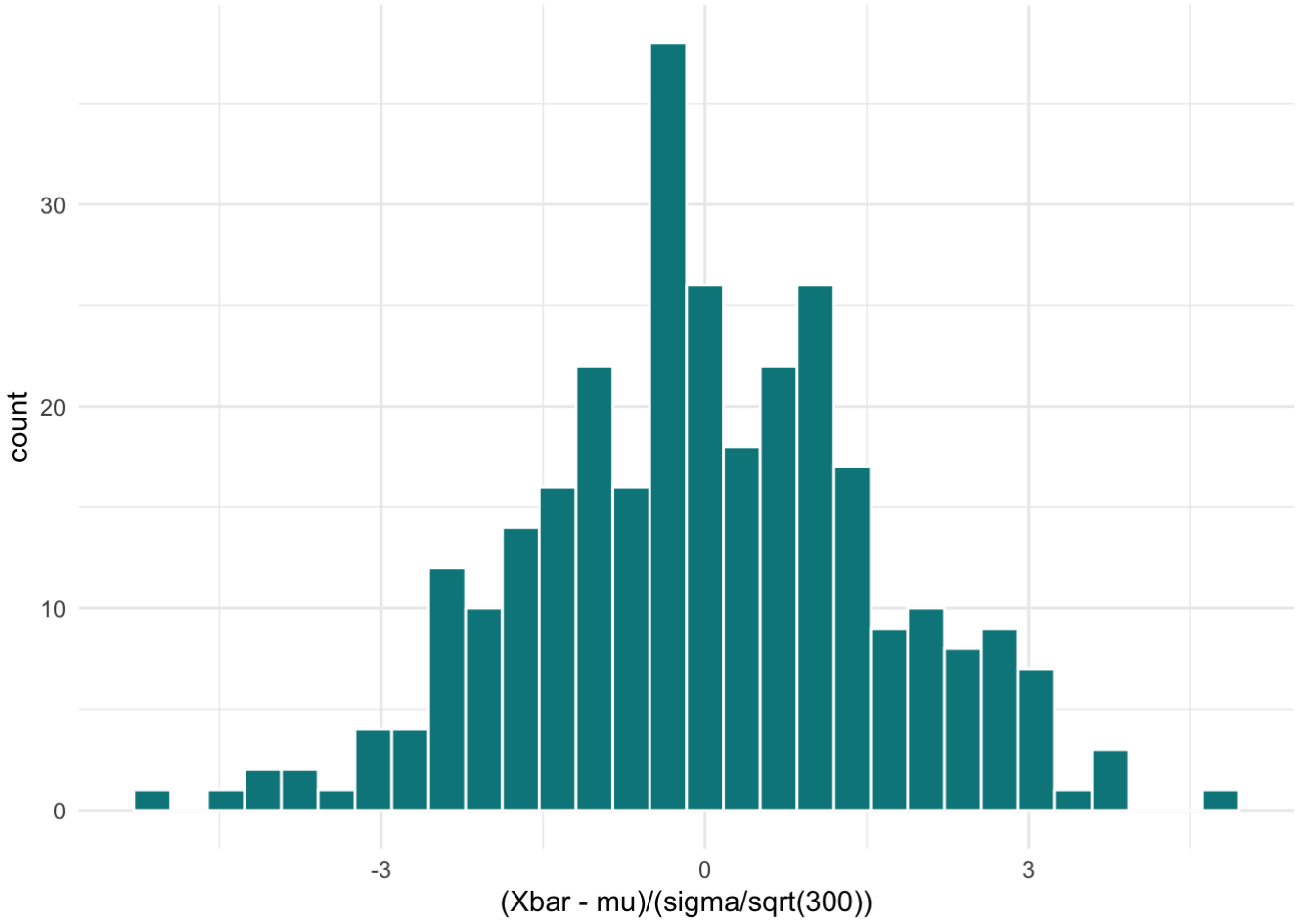
```
## [1] 165 273 264 209 281  54
```

```
songs$group <- groups
songs %>% select(energy:valence, group) %>% head
```

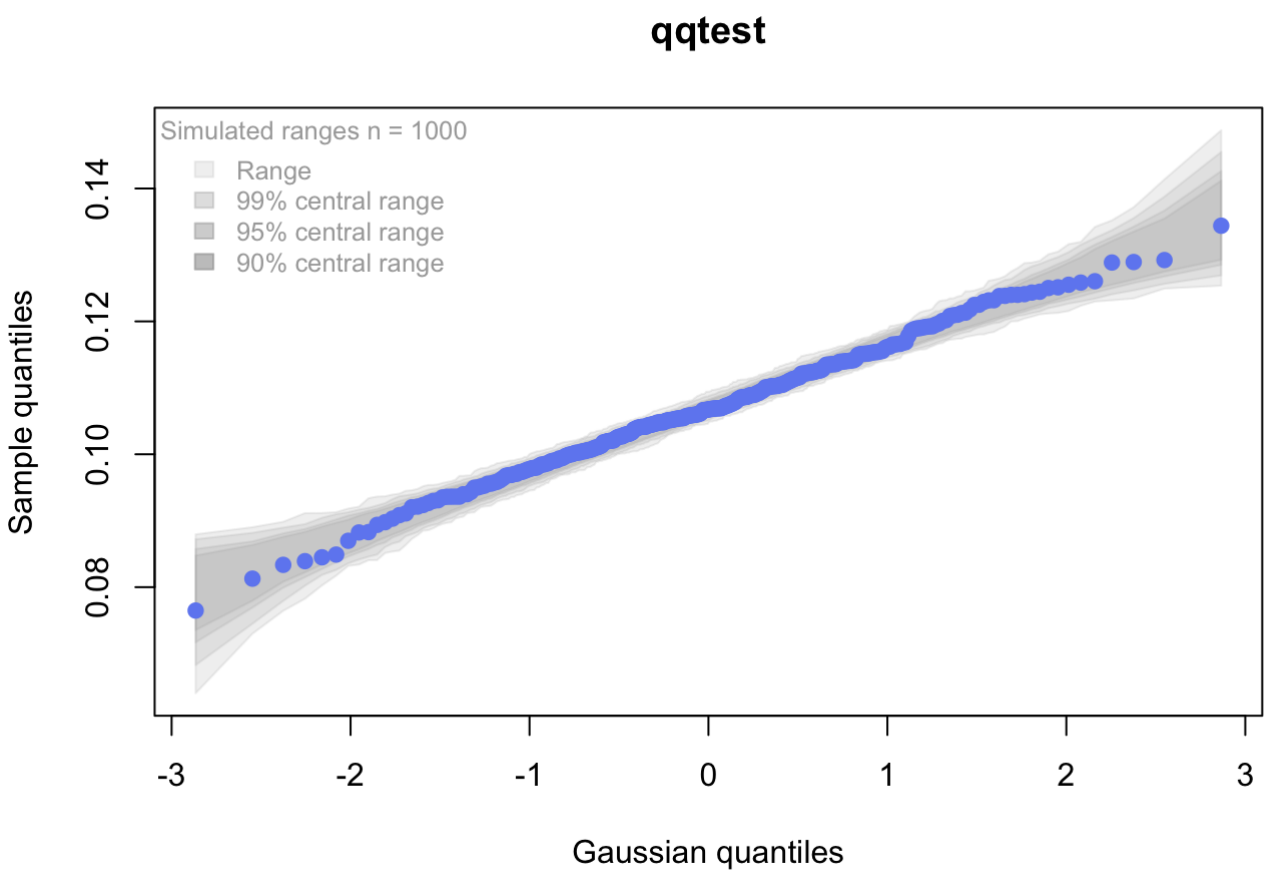
	ener...	...	loudness	...	speechiness	acousticness	instrumentalness	liveness	valence
	<dbl>	<int>	<dbl>	<int>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	0.916	6	-2.634	1	0.0583	0.1020	0.00e+00	0.0653	0.518
2	0.815	11	-4.969	1	0.0373	0.0724	4.21e-03	0.3570	0.693
3	0.931	1	-3.432	0	0.0742	0.0794	2.33e-05	0.1100	0.613
4	0.930	7	-3.778	1	0.1020	0.0287	9.43e-06	0.2040	0.277
5	0.833	1	-4.672	1	0.0359	0.0803	0.00e+00	0.0833	0.725
6	0.919	8	-5.385	1	0.1270	0.0799	0.00e+00	0.1430	0.585

6 rows | 1-10 of 11 columns

```
Xbars <- songs %>% group_by(group)%>% summarize(Xbar=mean(speechiness))

mu    <- mean(songs$speechiness)
sigma <- sd(songs$speechiness)

ggplot(Xbars, aes(x=(Xbar-mu)/(sigma/sqrt(300)))) +
  geom_histogram(colour='white',fill='turquoise4')
```




```
qqtest(Xbars$Xbar)
```



The magic does work.It is transformed to a normal distribution