DSS Normality Testing

Alvin Murphy

Table of contents


```
install.packages("stringr")                # Install packages and libraries in R
library(stringr, quietly = TRUE)
# suppressPackageStartupMessages(library("stringr"))
```

```
install.packages("dplyr")
library(dplyr, quietly = TRUE)

install.packages("ggplot2")
install.packages("GGally")
library(ggplot2, quietly = TRUE)
library(GGally, quietly = TRUE)

# Note that loading MASS will cause issues with dplyr select
library("MASS")
```

```
options(warn=-1)
```

```
setwd('/home/jovyan/work/data')
```

1 Load Data Files

```
# macData <- read.csv('DSS_SpanData-mac-2022-05-02 18_38_26_s10-5-1.csv', header = TRUE)
# linpcData <- read.csv('DSS_SpanData-linuxpc-2022-06-06 17_38_29_s10-5-1.csv', header = T
# rpi4Data <- read.csv('DSS_SpanData-rpi4-2022-06-06 17_52_59_s10-5-1.csv', header = TRUE)
# awsEC2Data <- read.csv('DSS_SpanData-aws_ec2-2022-06-07 17_44_08_s10-5-1.csv', header =
cci_Data <- read.csv('DSS_SpanData-odu_cci-2022-06-28 17_47_20_s10-5-1.csv', header = TRUE
```

1.1 Review and Tag Data

```
dssData <- cci_Data
summary(dssData)
```

```
   Trace.ID          Trace.name         Start.time          Duration
 Length:100         Length:100         Length:100         Length:100
 Class :character   Class :character   Class :character   Class :character
 Mode  :character   Mode  :character   Mode  :character   Mode  :character
```

```
head(dssData[, c(1,2)])
head(dssData[, c(3,4)])
```

A data.frame: 6 × 2

	Trace.ID <chr>	Trace.name <chr>
1	5a8ec9277c323283b32729fe0994a647	dss-prototype: /TE
2	790bb217abfb736f03db40836260ee43	dss-prototype: /tracks
3	e80d6b686e2f994d2d1c8c55f9f0bf5b	dss-prototype: /IAD
4	90707755d04cb477ff4a5d875754c781	dss-prototype: /RIC
5	364af60c5cb43364ac74731f34f2647a	dss-prototype: /WA
6	5a61b04ef48413f4e0a232e8271f20cd	dss-prototype: /TE

A data.frame: 6 × 2

	Start.time <chr>	Duration <chr>
1	2022-06-28 21:18:36.903	7.85 ms
2	2022-06-28 21:18:35.892	5.42 ms
3	2022-06-28 21:18:34.236	652 ms
4	2022-06-28 21:18:32.842	390 ms

	Start.time <chr>	Duration <chr>
5	2022-06-28 21:18:31.826	11.4 ms
6	2022-06-28 21:18:30.814	7.18 ms

2 Convert Data into Useable Metrics

To make the data more usable and easier to understand we apply conversions from text to numeric and add additional columns with supporting information. A **useCase** column is added to identify specific DSS request use cases; e.g. Get Dulles Airport Data. The data also indicates whether the request is managed internally or a connection to an external service is required to provided a response (i.e., https://opensky-network.org). A **numContainers** column is added to indicate the number of containers involved in providing a use case response (e.g. independent variable). An **ext** column is added to indicate whether an API external to the Docker environment is used; e.g., ext = TRUE for OpenSky API calls.

```
## Dictionary for converting data

DSSoperations <- c(
    "dss-prototype: /IAD" = "Get Dulles Airport Data (External)",
    "dss-prototype: /RIC" = "Get Richmond Airport Data (External)",
    "dss-prototype: /tracks" = "Get Stored Local DSS Tracks (Internal)",
    "dss-prototype: /TE" = "Trial Engage (Internal)",
    "dss-prototype: /WA" = "Assess Weapons (Internal)"
)

DSSuseCaseNum <- c(
    "dss-prototype: /IAD" = 4,
    "dss-prototype: /RIC" = 5,
    "dss-prototype: /tracks" = 1,
    "dss-prototype: /TE" = 2,
    "dss-prototype: /WA" = 3
)

DSSexternal <- c(
    "dss-prototype: /IAD" = TRUE,
    "dss-prototype: /RIC" = TRUE,
    "dss-prototype: /tracks" = FALSE,
    "dss-prototype: /TE" = FALSE,
    "dss-prototype: /WA" = FALSE
)
```

```
DSStraceShortName <- c(
    "dss-prototype: /IAD" = "/IAD",
    "dss-prototype: /RIC" = "/RIC",
    "dss-prototype: /tracks" = "/tracks",
    "dss-prototype: /TE" = "/TE",
    "dss-prototype: /WA" = "/WA"
)
```

2.1 Add Additional Column Descriptors

```
spanData <- dssData
spanMetrics <- spanData
```

```
spanMetrics$useCase <- DSSoperations[spanMetrics$Trace.name]
spanMetrics$useCaseNum <- DSSuseCaseNum[spanMetrics$Trace.name]

spanMetrics$ext = DSSexternal[spanMetrics$Trace.name]
spanMetrics$Trace.name = DSStraceShortName[spanMetrics$Trace.name]

# truncate span ID
# spanMetrics$Trace.ID <- str_sub(spanMetrics$Trace.ID,1,4)


# summary(spanMetrics)
# head(spanMetrics)
# tail(spanMetrics)

# spanMetrics
```

```
# Convert character data into numeric metrics

for(index in 1:nrow(spanMetrics)) {       # for-loop over rows

    # Convert span duration

    char = spanMetrics[index,4]
    len = str_length(char)
    duration = str_sub(char,1,(len-3))
    units = str_sub(char,(len-1),len)
```

```
    duration = as.numeric(duration)

    # print(duration)
    # print(units)

    if(units == 'ms') {
        duration = duration               # Keep ms
    } else if (units == 'µs') {
        duration = duration * 0.001       # Convert µs to ms
    } else if (units == ' s') {
        duration = duration * 1000        # Convert s to ms
    } else {
        print ('Unable to find specified units')
        print (units)
    }

    # if(units == 'ms') {
    #     duration = duration / 1000        # Convert ms to s
    # } else if (units == 'µs') {
    #     duration = duration / 1000000     # Convert µs to s
    # } else if (units == ' s') {
    #     duration = duration               # Keep s
    # } else {
    #     print ('Unable to find specified units')
    #     print (units)
    # }


    spanMetrics[index,4] = duration

    # Convert time

    # time = spanMetrics[index,3]
    # epoch <- as.POSIXct(time)
    # epoch_int <- as.integer(epoch)
    # spanMetrics[index,3] = epoch_int
}

# spanMetrics
```

```r
# Convert columns from char to numeric

spanMetrics$Duration = as.numeric(spanMetrics$Duration)
# spanMetrics$Start.time = as.numeric(spanMetrics$Start.time)

# spanMetrics
```

```r
spanMetrics$Trace.name <- as.factor(spanMetrics$Trace.name)
spanMetrics$useCase <- as.factor(spanMetrics$useCase)
spanMetrics$Trace.useCaseNum <- as.factor(spanMetrics$useCaseNum)
```

```r
summary(spanMetrics)

# # sort span metrics by use case number
# spanMetricsA <- arrange(spanMetrics, useCaseNum)

head(spanMetrics[, c(2,3,4,5)])
head(spanMetrics[, c(6,7)])

# spanMetricsA
```

```
  Trace.ID           Trace.name  Start.time           Duration
Length:100         /IAD   :20   Length:100         Min.   :   4.77
Class :character   /RIC   :20   Class :character   1st Qu.:   7.21
Mode  :character   /TE    :20   Mode  :character   Median :  11.55
                   /tracks:20                      Mean   : 184.71
                   /WA    :20                      3rd Qu.: 340.75
                                                   Max.   :1500.00
                                    useCase     useCaseNum    ext
Assess Weapons (Internal)              :20   Min.   :1    Mode :logical
Get Dulles Airport Data (External)     :20   1st Qu.:2    FALSE:60
Get Richmond Airport Data (External)   :20   Median :3    TRUE :40
Get Stored Local DSS Tracks (Internal):20   Mean   :3
Trial Engage (Internal)                :20   3rd Qu.:4
                                             Max.   :5

Trace.useCaseNum
1:20
2:20
3:20
4:20
5:20
```

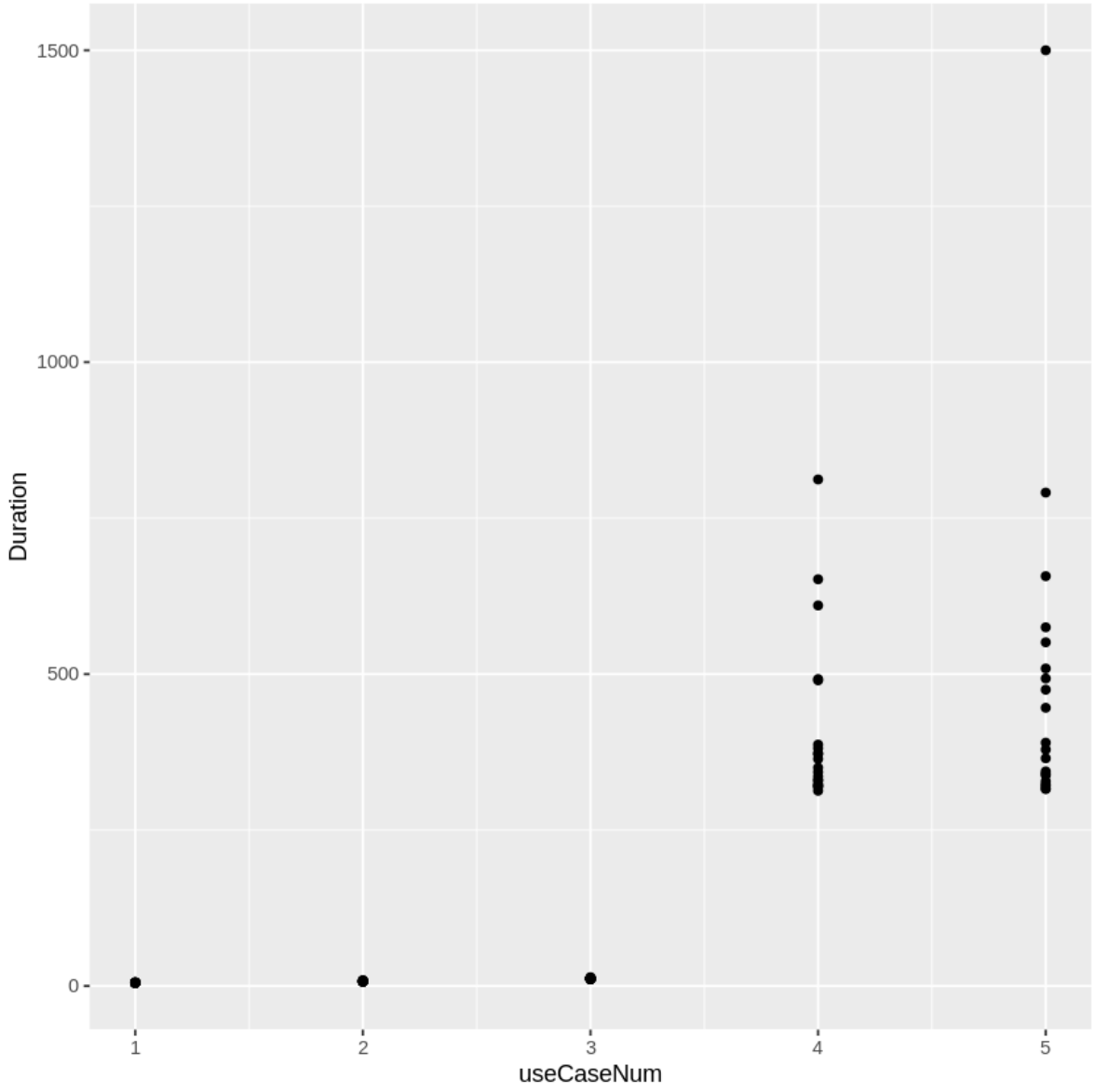
A data.frame: 6 × 4

	Trace.name <fct>	Start.time <chr>	Duration <dbl>	useCase <fct>
1	/TE	2022-06-28 21:18:36.903	7.85	Trial Engage (Internal)
2	/tracks	2022-06-28 21:18:35.892	5.42	Get Stored Local DSS Tracks (Internal)
3	/IAD	2022-06-28 21:18:34.236	652.00	Get Dulles Airport Data (External)
4	/RIC	2022-06-28 21:18:32.842	390.00	Get Richmond Airport Data (External)
5	/WA	2022-06-28 21:18:31.826	11.40	Assess Weapons (Internal)
6	/TE	2022-06-28 21:18:30.814	7.18	Trial Engage (Internal)

A data.frame: 6 × 2

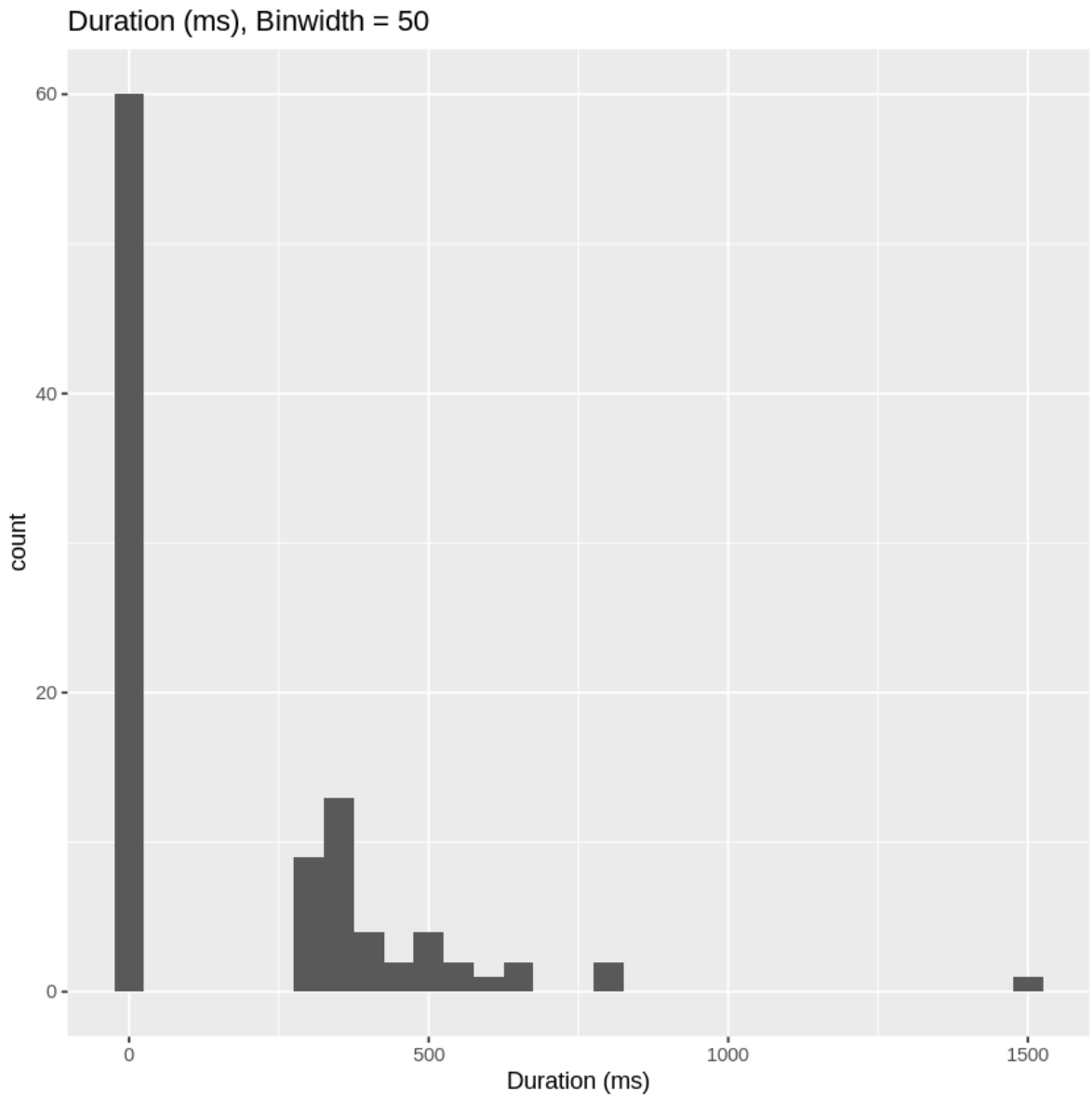
	useCaseNum <dbl>	ext <lgl>
1	2	FALSE
2	1	FALSE
3	4	TRUE
4	5	TRUE
5	3	FALSE
6	2	FALSE

```
qplot(useCaseNum, Duration, data = spanMetrics)
```

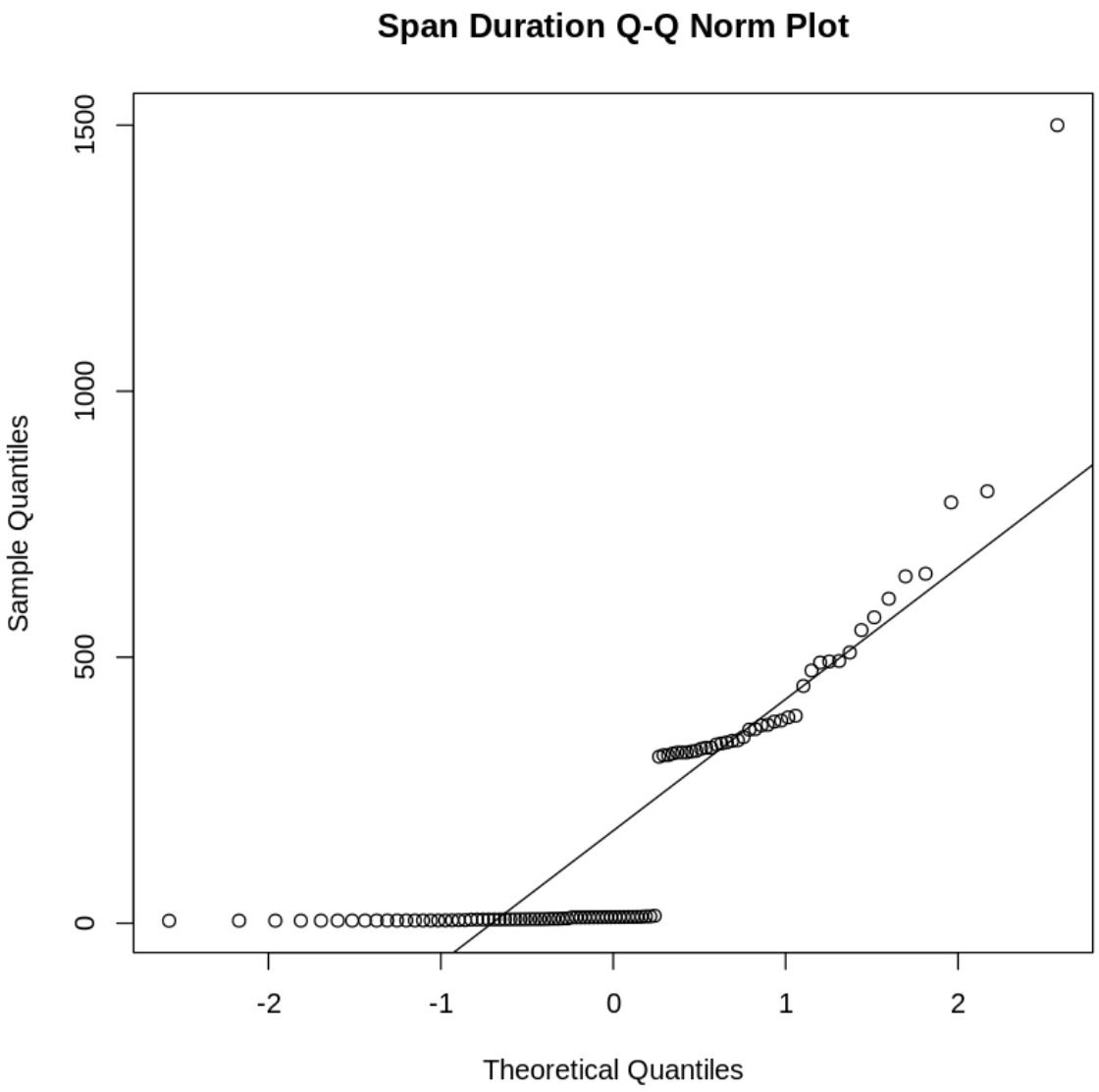



```
# Remove outliers
aSpan <- spanMetrics
# outliers <- boxplot(aSpan$Duration, plot = FALSE)$out
# outliers

# aSpan <- aSpan[-which(aSpan$Duration %in% outliers),]
```

```
spanMetrics %>%
# aSpan %>%
    ggplot(aes(Duration)) + geom_histogram(binwidth = 50) +
    ggtitle("Duration (ms), Binwidth = 50") +
    xlab("Duration (ms)")
```



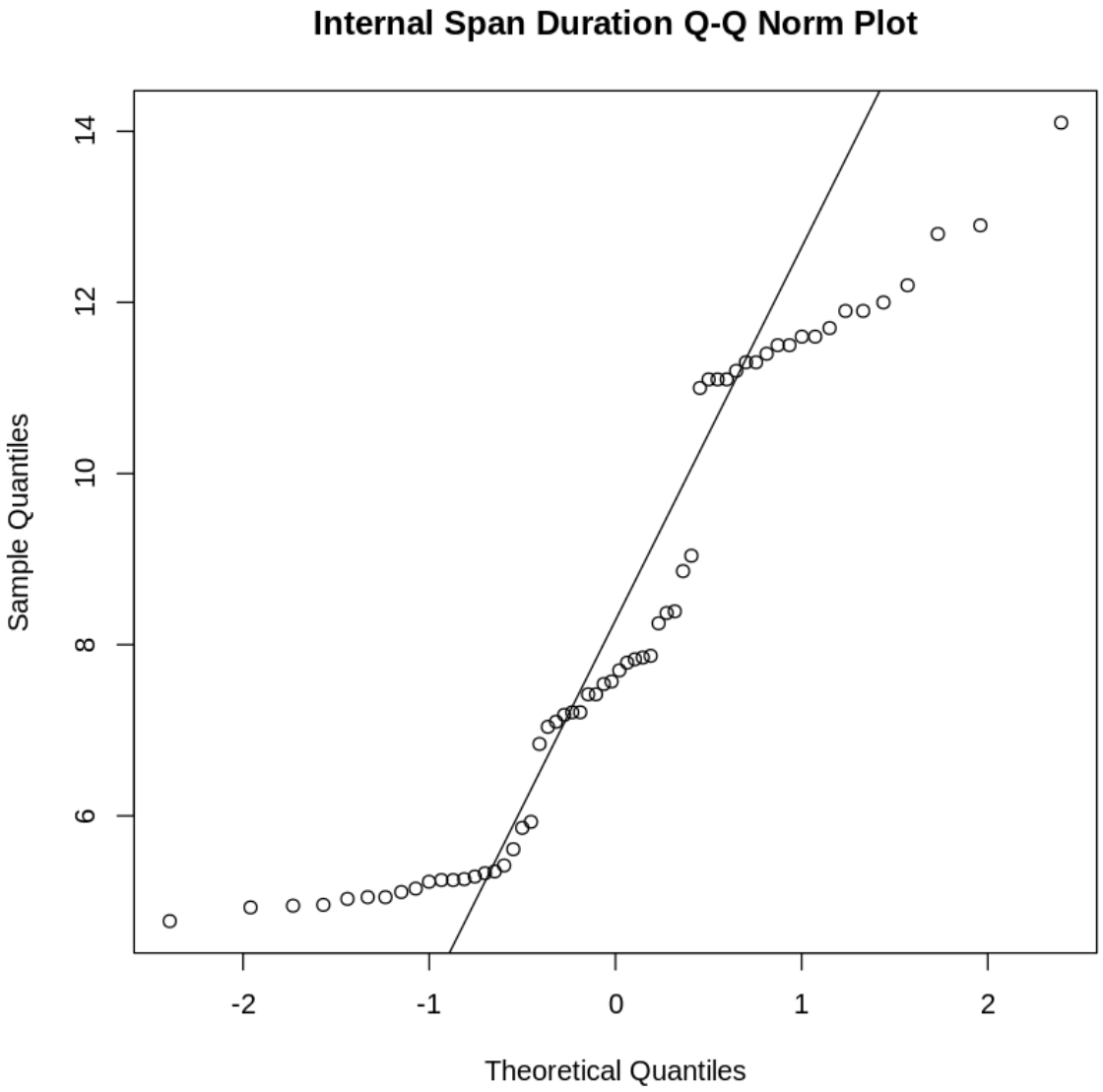
```
qqnorm(aSpan$Duration, main="Span Duration Q-Q Norm Plot")
qqline(aSpan$Duration)
```




```
# Separate Internal Data
# Could use ext == FALSE
```

```
tracksSpanData = subset(aSpan, useCaseNum == 1)
TE_SpanData = subset(aSpan, useCaseNum == 2)
WA_SpanData = subset(aSpan, useCaseNum == 3)

internalSpanData <- rbind(tracksSpanData, TE_SpanData, WA_SpanData)
# internalSpanData <- rbind(WA_SpanData)
dssSpanData <- rbind(TE_SpanData, WA_SpanData)
```

```
qqnorm(internalSpanData$Duration, main="Internal Span Duration Q-Q Norm Plot")
qqline(internalSpanData$Duration)

# qqnorm(dssSpanData$Duration, main="DSS Span Duration Q-Q Norm Plot")
# qqline(dssSpanData$Duration)
```




```
outliers <- which(internalSpanData$Duration > 50) #outlier rows
outliers
# iSpan <- internalSpanData[!outliers,]
# iSpan <- dssSpanData[!dssSpanData$Duration > 50,]
iSpan <- internalSpanData[!internalSpanData$Duration > 50,]
    # Remove if duration is greater than a value
```

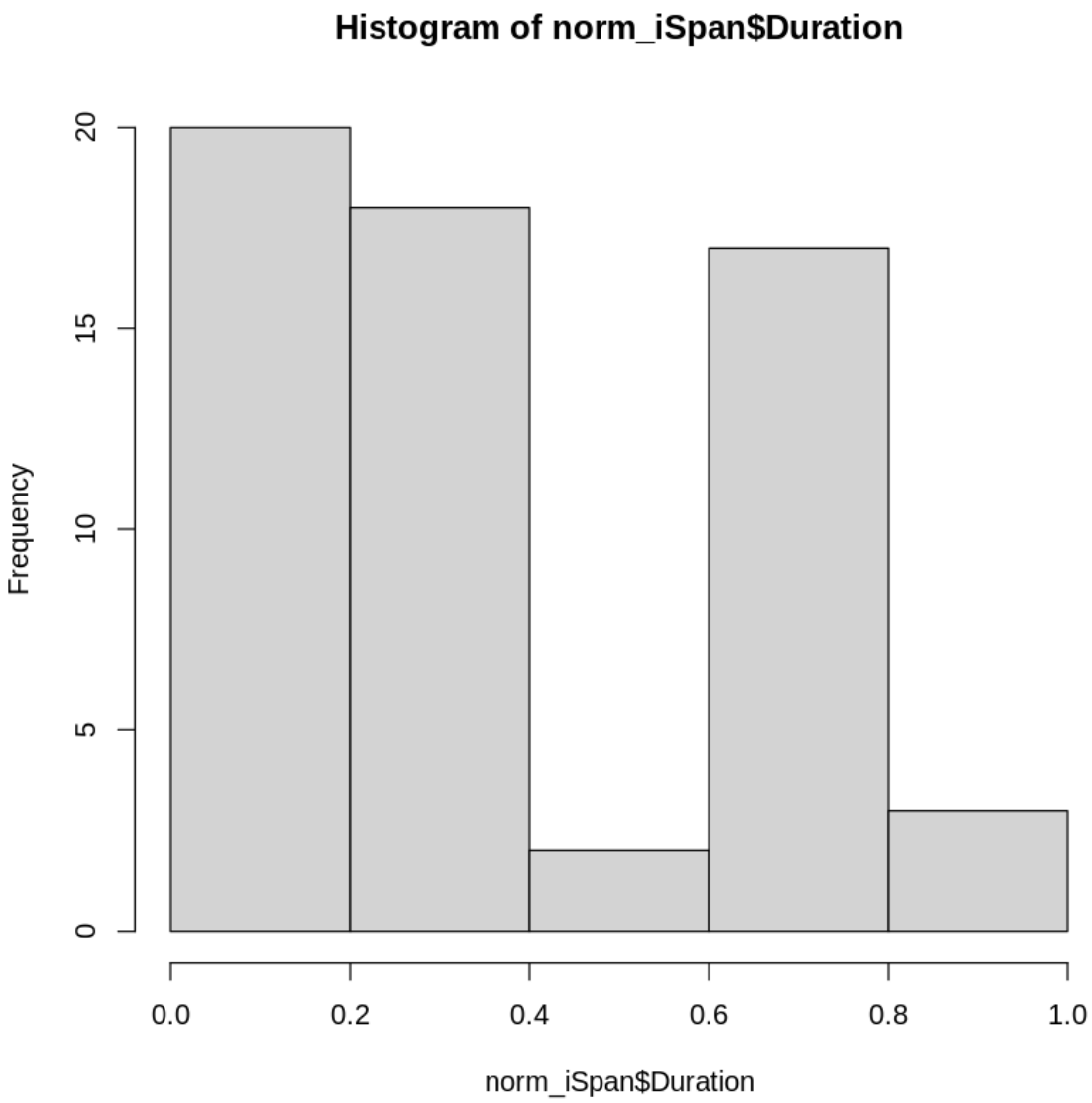
```r
# create min-max-norm function
min_max_norm <- function(x) {
    (x - min(x)) / (max(x) - min(x))
  }

#apply Min-Max normalization
norm_iSpan <- iSpan
norm_iSpan$Duration <- (iSpan$Duration - min(iSpan$Duration)) /
    (max(iSpan$Duration) - min(iSpan$Duration))

summary(norm_iSpan)
```

```
  Trace.ID           Trace.name  Start.time           Duration
Length:60          /IAD   : 0   Length:60          Min.   :0.00000
Class :character   /RIC   : 0   Class :character   1st Qu.:0.06163
Mode  :character   /TE    :20   Mode  :character   Median :0.30707
                   /tracks:20                      Mean   :0.37203
                   /WA    :20                      3rd Qu.:0.69185
                                                   Max.   :1.00000
                                    useCase     useCaseNum    ext
Assess Weapons (Internal)              :20   Min.   :1    Mode :logical
Get Dulles Airport Data (External)     : 0   1st Qu.:1    FALSE:60
Get Richmond Airport Data (External)   : 0   Median :2
Get Stored Local DSS Tracks (Internal):20   Mean   :2
Trial Engage (Internal)                :20   3rd Qu.:3
                                             Max.   :3

Trace.useCaseNum
1:20
2:20
3:20
4: 0
5: 0
```

```r
log_iSpan <- iSpan
log_iSpan$Duration=log(log_iSpan$Duration + 1) # Natural Log
inv_iSpan <- iSpan
inv_iSpan$Duration = (1 / inv_iSpan$Duration)
```

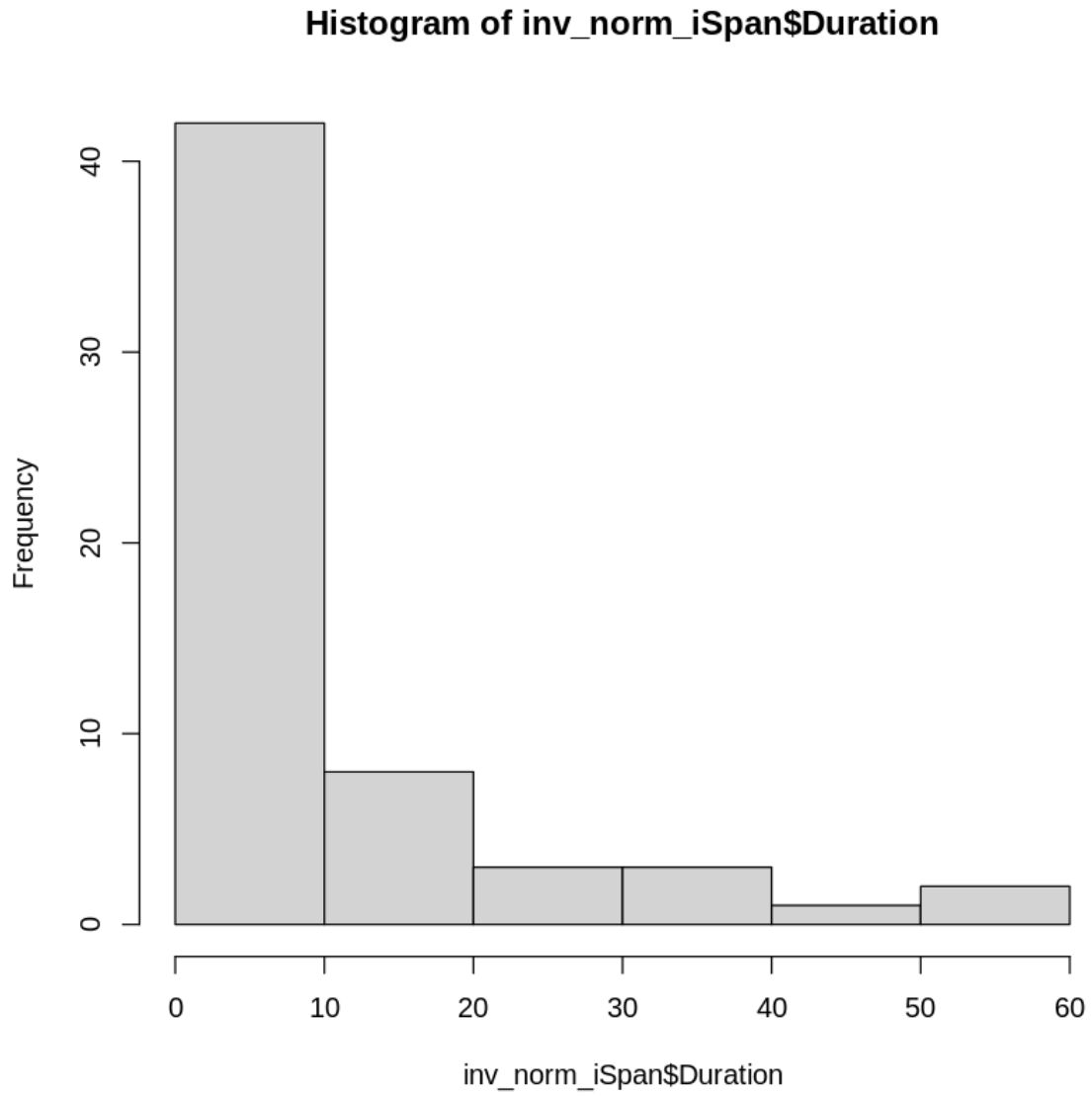
```r
shapiro.test(log_iSpan$Duration)
shapiro.test(inv_iSpan$Duration)
shapiro.test(norm_iSpan$Duration)

hist(norm_iSpan$Duration)
```

```
    Shapiro-Wilk normality test

data:  log_iSpan$Duration
W = 0.89819, p-value = 0.0001131


    Shapiro-Wilk normality test

data:  inv_iSpan$Duration
W = 0.88952, p-value = 5.576e-05


    Shapiro-Wilk normality test

data:  norm_iSpan$Duration
W = 0.8888, p-value = 5.268e-05
```

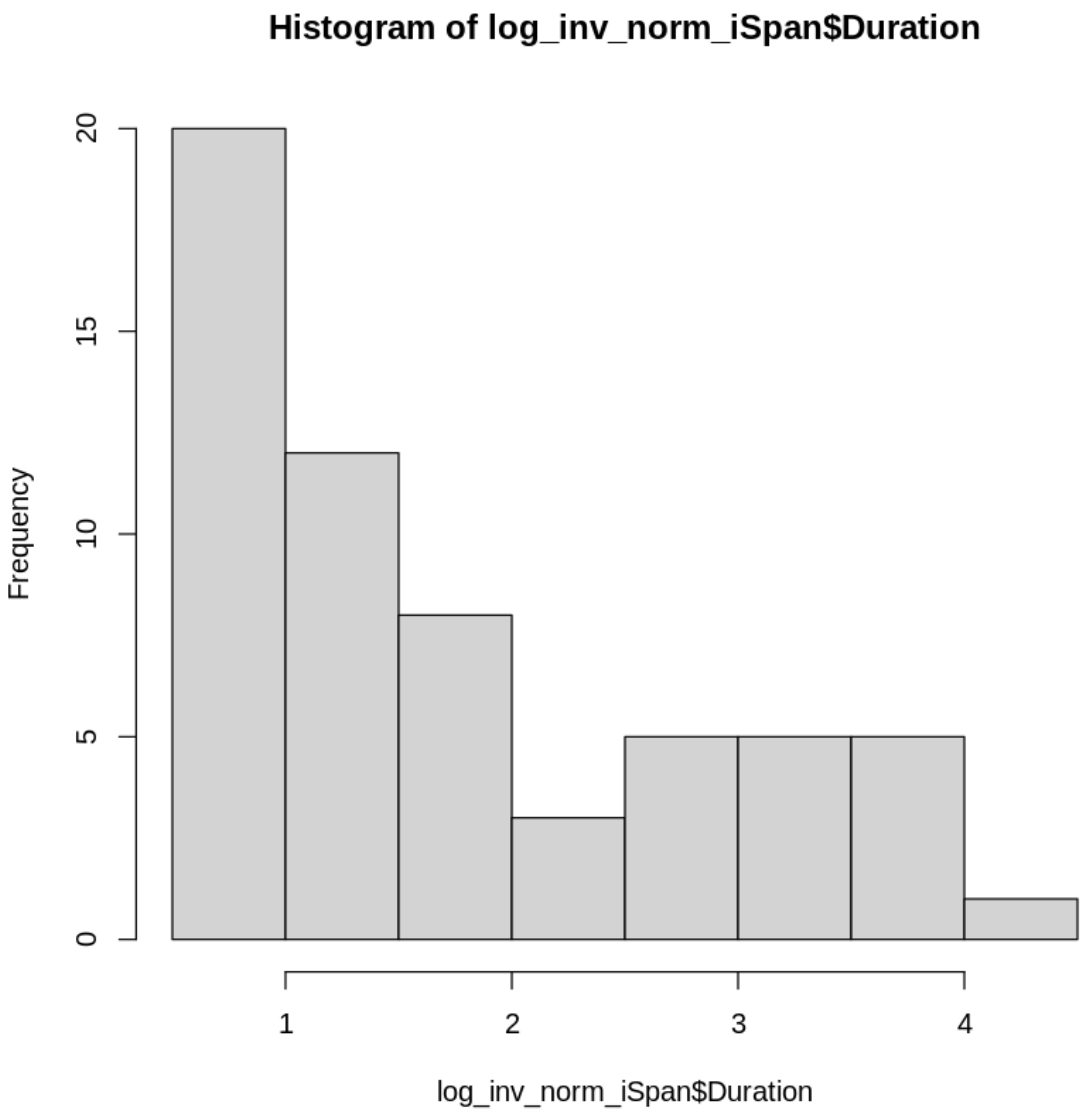



```
# log_norm_iSpan <- norm_iSpan
# log_norm_iSpan$Duration=log(log_norm_iSpan$Duration + 1)
# hist(log_norm_iSpan$Duration)
# shapiro.test(log_norm_iSpan$Duration)
# inv_log_norm_iSpan <- log_norm_iSpan
# inv_log_norm_iSpan$Duration = (1 / inv_log_norm_iSpan$Duration)
# hist(inv_log_norm_iSpan$Duration)
```

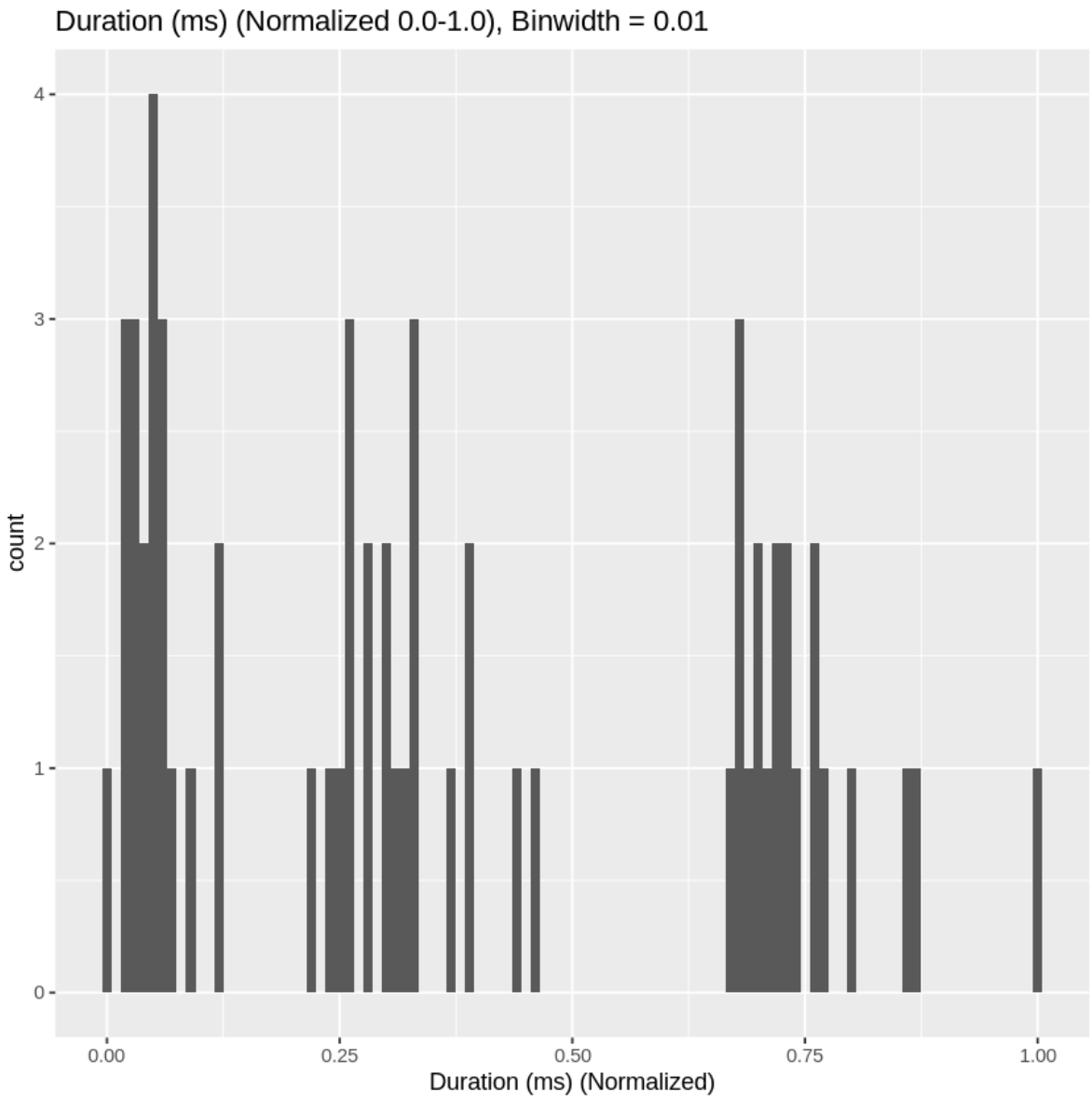
```r
inv_norm_iSpan <- norm_iSpan
inv_norm_iSpan$Duration = (1 / inv_norm_iSpan$Duration)
hist(inv_norm_iSpan$Duration)

log_inv_norm_iSpan <- inv_norm_iSpan
log_inv_norm_iSpan$Duration=log(log_inv_norm_iSpan$Duration + 1)
hist(log_inv_norm_iSpan$Duration)
shapiro.test(log_inv_norm_iSpan$Duration)
```




```
    Shapiro-Wilk normality test

data:  log_inv_norm_iSpan$Duration
W = NaN, p-value = NA
```




```
norm_iSpan %>%
# aSpan %>%
    ggplot(aes(Duration)) + geom_histogram(binwidth = 0.01) +
    ggtitle("Duration (ms) (Normalized 0.0-1.0), Binwidth = 0.01") +
    xlab("Duration (ms) (Normalized)")
```


Duration (ms) (Normalized 0.0-1.0), Binwidth = 0.01
count
4
3
2
1
0
0.00
0.25
0.50
0.75
1.00
Duration (ms) (Normalized)