DSS Prototype Analysis

Alvin Murphy

Table of contents

1 Hypothesis

Hypotheses are "innocent until proven guilty." We'll assume that SpaceX and others have proven that DevSecOps tech can meet hard-real-time requirements but nothing available in the body of knowledge documents this.

Hypothesis: Modern DevSecOps architectures can be designed to meet hard-real-time latency (μ) requirements using modern computing environments and computing infrastructure.

$H_0 : \mu \leq 500ms$ with jitter within latency bounds
$H_a : \mu > 500ms$ with jitter exceeding latency bounds

Murphy, Alvin C. and Moreland Jr, James D. 'Integrating AI Microservices into Hard-Real-Time SoS to Ensure Trustworthiness of Digital Enterprise Using Mission Engineering'. 1 Jan. 2021 : 38 – 54.

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

2 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 = TRU
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 = TR
cci_Data <- read.csv('DSS_SpanData-odu_cci-2022-06-28 17_47_20_s10-5-1.csv', header = TRUE
```

2.1 Review and Tag MacBook Air (2017) Data

```
summary(macData)
```

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

A data.frame: 6 × 2

	Trace.ID <chr>	Trace.name <chr>
1	9ee3577fb1b427bc4fc17fecc5154d7d	dss-prototype: /TE
2	f05ddc4dc13aff5c3098011b2a402401	dss-prototype: /tracks
3	2bd901fbbfc9ee8dfa7c9629d93a1567	dss-prototype: /IAD
4	69a48381a14e79da08aaa2353f7db4b2	dss-prototype: /RIC
5	e83037dcb9438c04dc12fba373b5502f	dss-prototype: /WA
6	7e381cd880adb670bb9627ca47020938	dss-prototype: /TE

A data.frame: 6 × 2

	Start.time <chr>	Duration <chr>
1	2022-05-02 10:25:01.366	36.0 ms
2	2022-05-02 10:25:00.309	43.3 ms
3	2022-05-02 10:24:58.818	464 ms
4	2022-05-02 10:24:57.307	494 ms
5	2022-05-02 10:24:56.128	139 ms
6	2022-05-02 10:24:55.081	30.3 ms

2.1.1 Add Source Indicator to MacBook Data

```
macDataPlat <- macData

macDataPlat$platform = "2017-macbook"
macDataPlat$env = 0
```

2.2 Tag Linux PC (2012) Data

```
linpcDataPlat <- linpcData

linpcDataPlat$platform = "2012-linpc"
linpcDataPlat$env = 1
```

2.3 Tag Raspberry Pi 4 (2020) Data

```
rpi4DataPlat <- linpcData

rpi4DataPlat$platform = "2020-rpi4"
rpi4DataPlat$env = 2
```

2.4 Tag AWS EC2 t2.micro Data

```
awsEC2DataPlat <- awsEC2Data

awsEC2DataPlat$platform = "2022-aws-ec2"
awsEC2DataPlat$env = 3
```

2.5 Tag ODU CCI Data

```
cciDataPlat <- cci_Data

cciDataPlat$platform = "2022-odu-cci"
cciDataPlat$env = 4
```

2.6 Merge Data Files

Here we merge data from all platforms.

```
spanData = rbind(macDataPlat, linpcDataPlat, rpi4DataPlat,
                 awsEC2DataPlat, cciDataPlat)

# Mclust components
    # cci = 1
    # mac = 9
    # linpc = 1
    # rpi4 = 1
    # awsEC2 = 9

# summary(spanData)
# head(spanData[, c(1,2,3)])
```

```r
# head(spanData[, c(4,5,6)])
# spanData
```

3 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.

```r
# install.packages("tidyverse")
library(tidyverse)
```

```r
## 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"
)
```

3.1 Add Additional Column Descriptors

```
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]
```

```r
    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)
    }

    spanMetrics[index,4] = as.numeric(duration)

    # Convert time

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

# Convert Duration columns from char to numeric
spanMetrics$Duration = as.numeric(spanMetrics$Duration)

glimpse(spanMetrics)
```

```
Rows: 500
Columns: 9
$ Trace.ID   <chr> "9ee3577fb1b427bc4fc17fecc5154d7d", "f05ddc4dc13aff5c309801…
$ Trace.name <chr> "/TE", "/tracks", "/IAD", "/RIC", "/WA", "/TE", "/tracks", …
$ Start.time <chr> "2022-05-02 10:25:01.366", "2022-05-02 10:25:00.309", "2022…
$ Duration   <dbl> 36.0, 43.3, 464.0, 494.0, 139.0, 30.3, 30.0, 478.0, 546.0, …
$ platform   <chr> "2017-macbook", "2017-macbook", "2017-macbook", "2017-macbo…
$ env        <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,…
$ useCase    <chr> "Trial Engage (Internal)", "Get Stored Local DSS Tracks (In…
$ useCaseNum <dbl> 2, 1, 4, 5, 3, 2, 1, 4, 5, 3, 2, 1, 4, 5, 3, 2, 1, 4, 5, 3,…
$ ext        <lgl> FALSE, FALSE, TRUE, TRUE, FALSE, FALSE, FALSE, TRUE, TRUE, …
```

```
# Assess hypothesis threshold

for(index in 1:nrow(spanMetrics)) {         # for-loop over rows
    if(spanMetrics[index,4] > 500) {
        spanMetrics[index,10] = FALSE
    } else {
    spanMetrics[index,10] = TRUE
    }
}
```

```
glimpse(spanMetrics)
# view(spanMetrics)
```

```
Rows: 500
Columns: 10
$ Trace.ID   <chr> "9ee3577fb1b427bc4fc17fecc5154d7d", "f05ddc4dc13aff5c309801…
$ Trace.name <chr> "/TE", "/tracks", "/IAD", "/RIC", "/WA", "/TE", "/tracks", …
$ Start.time <chr> "2022-05-02 10:25:01.366", "2022-05-02 10:25:00.309", "2022…
$ Duration   <dbl> 36.0, 43.3, 464.0, 494.0, 139.0, 30.3, 30.0, 478.0, 546.0, …
$ platform   <chr> "2017-macbook", "2017-macbook", "2017-macbook", "2017-macbo…
$ env        <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,…
$ useCase    <chr> "Trial Engage (Internal)", "Get Stored Local DSS Tracks (In…
$ useCaseNum <dbl> 2, 1, 4, 5, 3, 2, 1, 4, 5, 3, 2, 1, 4, 5, 3, 2, 1, 4, 5, 3,…
$ ext        <lgl> FALSE, FALSE, TRUE, TRUE, FALSE, FALSE, FALSE, TRUE, TRUE, …
$ V10        <lgl> TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, FALSE, TRUE…
```

```
summary(spanMetrics)

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

# head(spanMetricsA[, c(2,3,4,5)])
# head(spanMetricsA[, c(6,7,8,9)])

# spanMetricsA
```

```
   Trace.ID          Trace.name         Start.time          Duration      
 Length:500         Length:500         Length:500         Min.   :   4.29  
 Class :character   Class :character   Class :character   1st Qu.:   7.42  
 Mode  :character   Mode  :character   Mode  :character   Median :  21.65  
                                                          Mean   : 198.15  
                                                          3rd Qu.: 381.00  
                                                          Max.   :2000.00  
   platform              env       useCase           useCaseNum    ext         
 Length:500         Min.   :0   Length:500         Min.   :1    Mode :logical  
 Class :character   1st Qu.:1   Class :character   1st Qu.:2    FALSE:300      
 Mode  :character   Median :2   Mode  :character   Median :3    TRUE :200      
                    Mean   :2                      Mean   :3                   
                    3rd Qu.:3                      3rd Qu.:4                   
                    Max.   :4                      Max.   :5                   
    V10         
 Mode :logical  
 FALSE:56       
 TRUE :444      
```

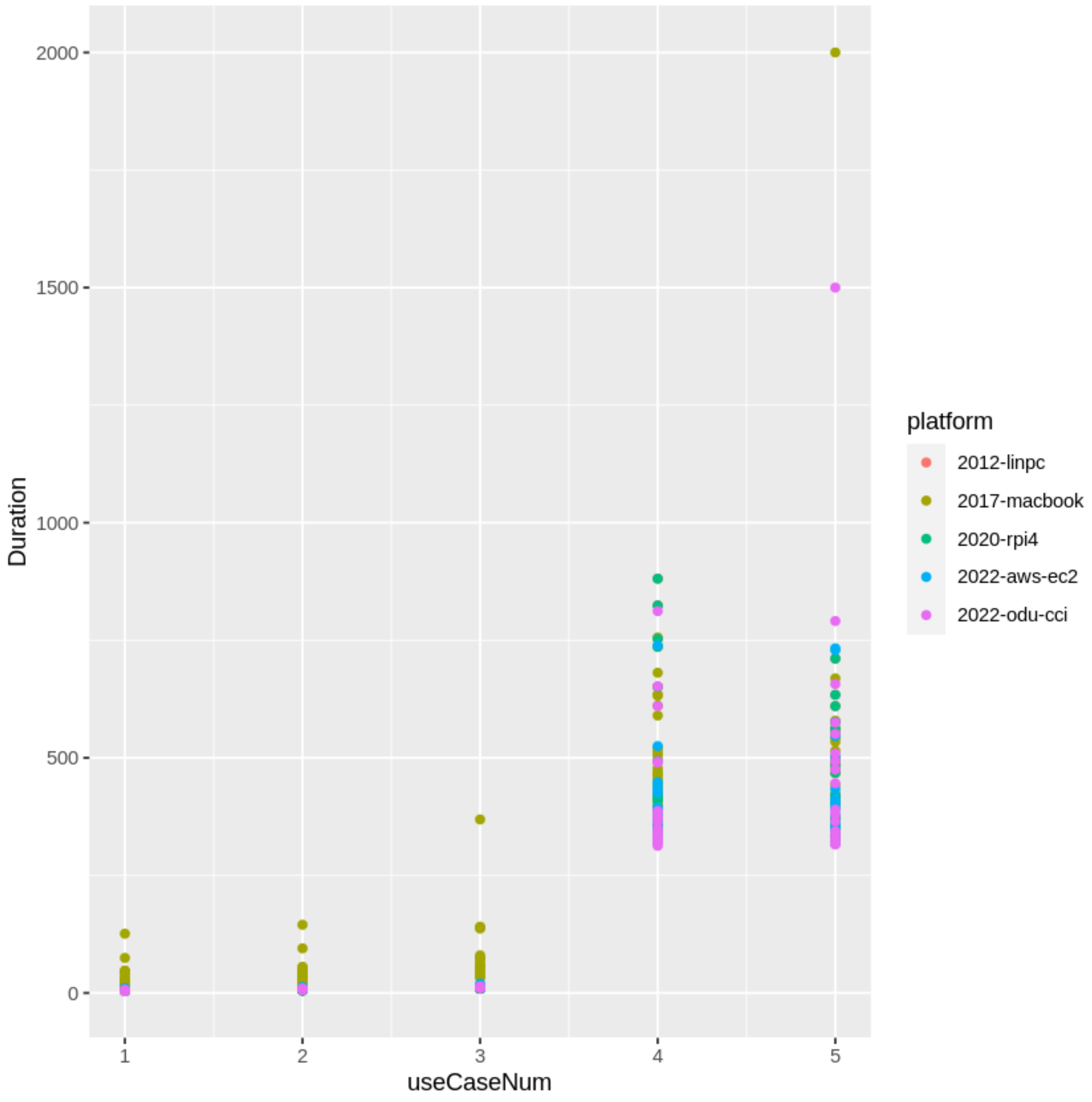
```r
# binom.test(444, 500, p = 0.95,
#             alternative = c("greater"),
#             conf.level = 0.95)

binom.test(444, 500, p = 0.95, alternative = "less")
```

```
    Exact binomial test

data:  444 and 500
number of successes = 444, number of trials = 500, p-value = 2.384e-08
alternative hypothesis: true probability of success is less than 0.95
95 percent confidence interval:
 0.000000 0.910444
sample estimates:
probability of success 
                 0.888 
```

3.2 Exploratory Analysis Plots

```r
# spanMetricsNum <- spanMetricsA %>%
#     dplyr::select(useCaseNum, env, ext, Duration)
#     # dplyr::select(Duration, useCaseNum, env)
```

```r
qplot(useCaseNum, Duration, data = spanMetricsA, colour = platform)
```



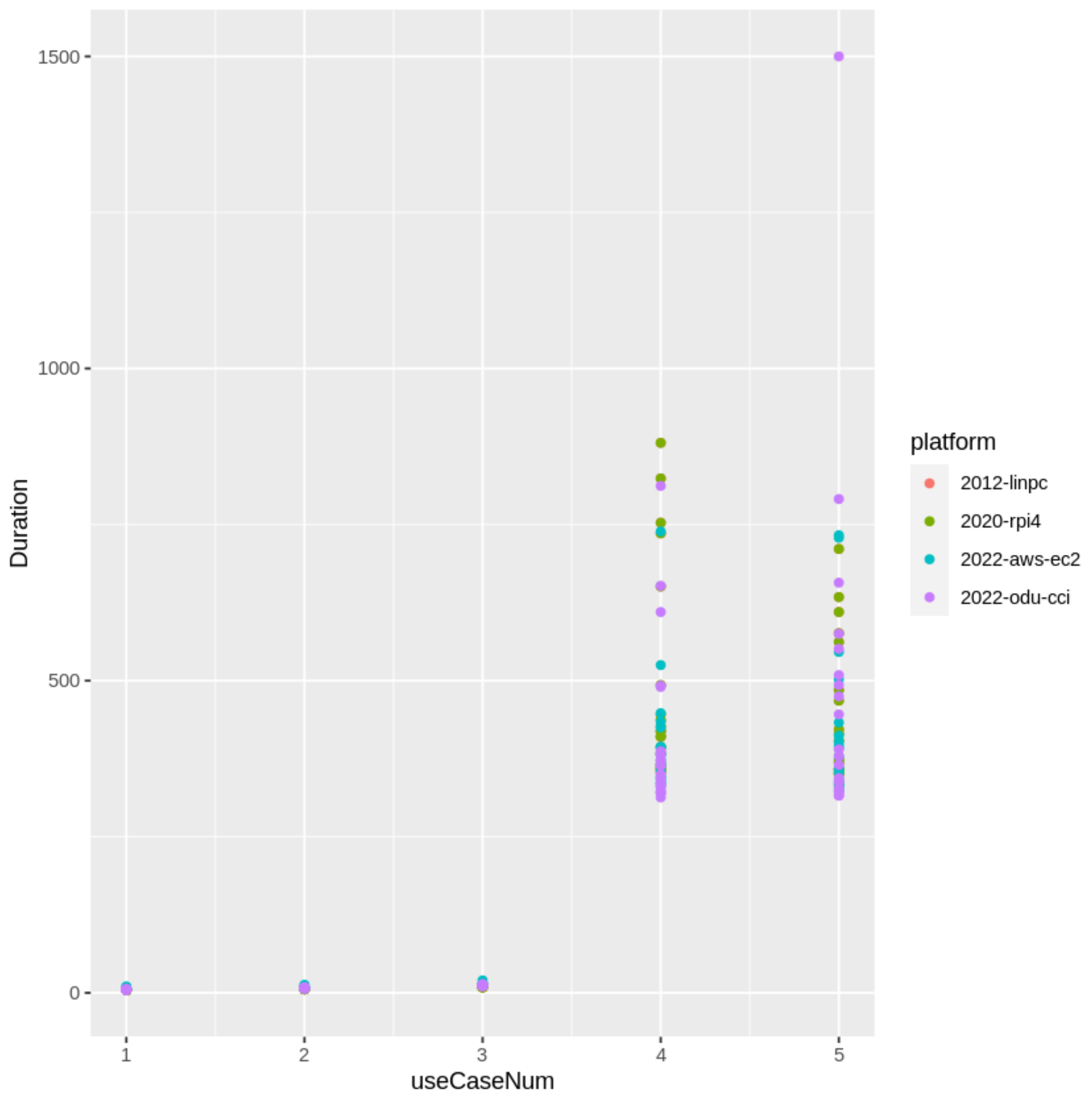
3.3 Remove Macbook Data from Development Platform

Here we remove the data from the Macbook development platform. The qplot shows that the **Mac implementation of Docker** adds latency within the Docker environment. In non-linux based plaforms, a Docker desktop running a virtual machine is required to provided that Docker capability that is native to Linux platforms. The Mac is considered to be the development environment and not representative of the integration and production environments.

https://dev.to/ericnograles/why-is-docker-on-macos-so-much-worse-than-linux-flh
https://collabnix.com/how-docker-for-mac-works-under-the-hood/

```
noMacSpan <- spanMetricsA[!spanMetricsA$env == 0,]
```

```
qplot(useCaseNum, Duration, data = noMacSpan, colour = platform)
```

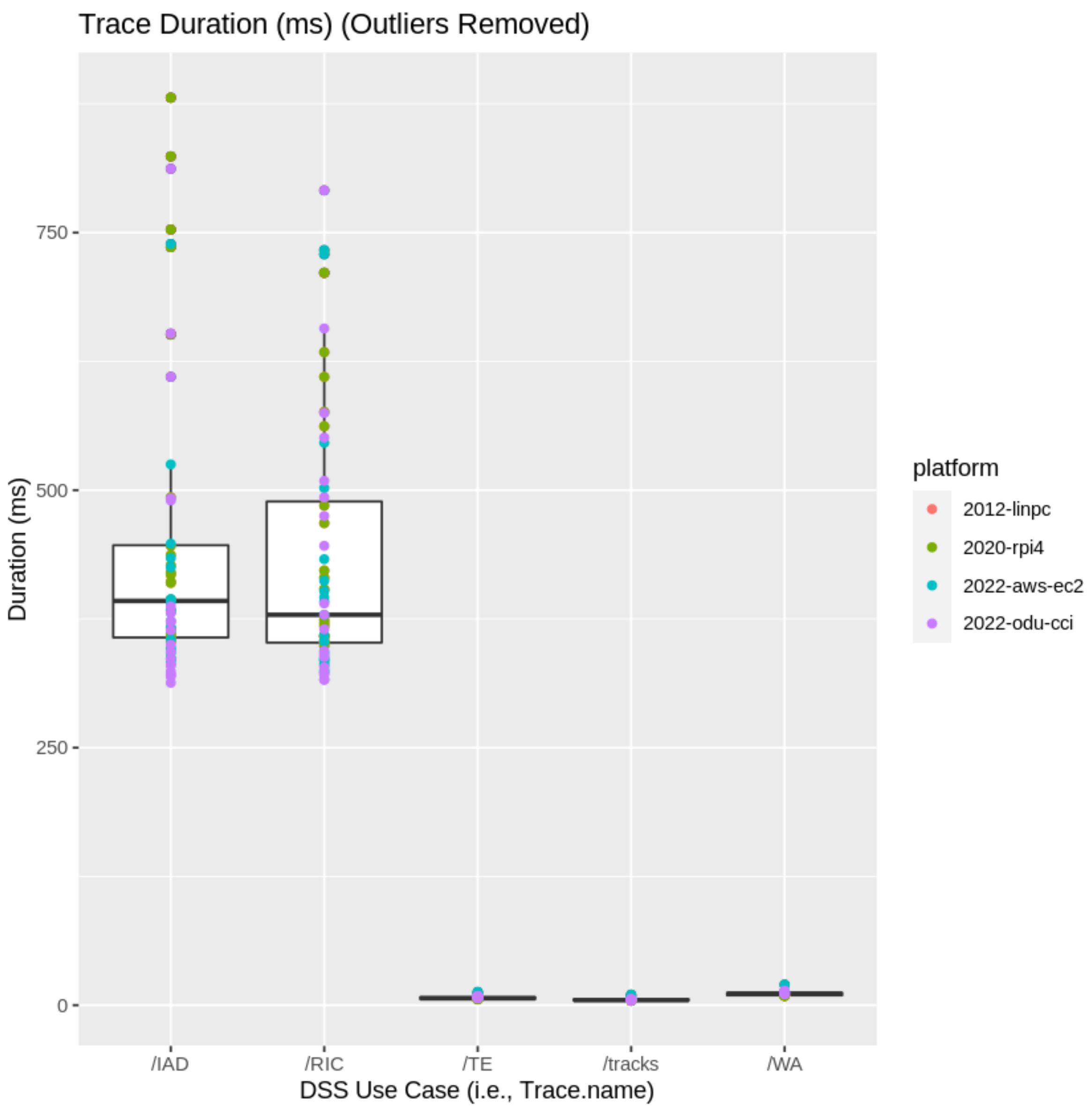


```
# par(mfrow=c(2,1))
# hist(spanMetricsA$Duration, counts = 5)

# spanMetricsA %>%
#     ggplot(aes(Trace.name, Duration)) +
#     stat_boxplot(notch="FALSE") + geom_point() +
#     ggtitle("Duration of Endpoint Responses from Trace")
# # notch went outside hinges. Try setting notch=FALSE.
```

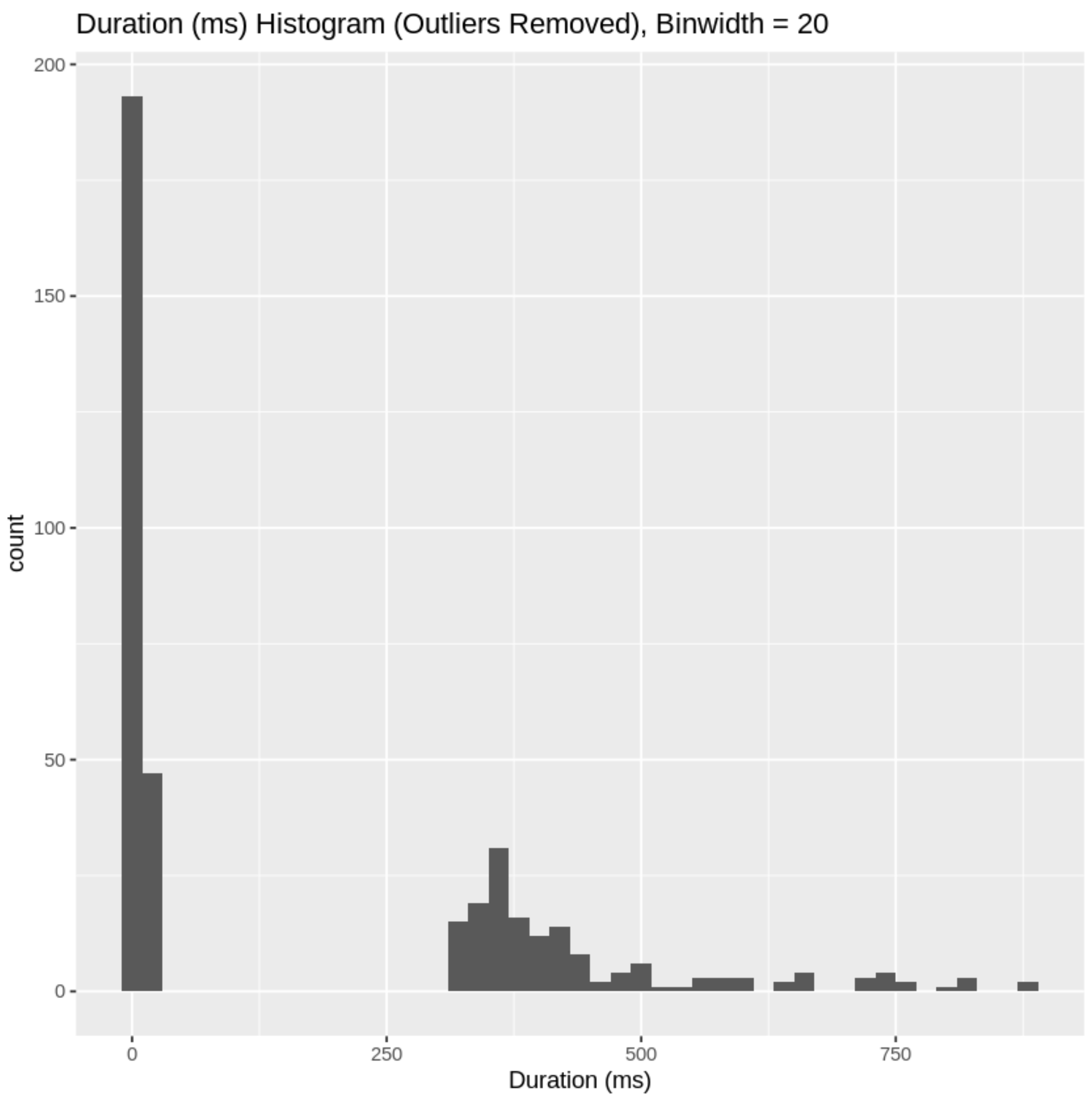
```
# Remove outliers
aSpan <- noMacSpan
outliers <- boxplot(aSpan$Duration, plot = FALSE)$out
outliers

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

1500

```
aSpan %>%
    ggplot(aes(Trace.name, Duration)) +
    stat_boxplot(notch="FALSE") + geom_point(aes(colour = platform)) +
    ggtitle("Trace Duration (ms) (Outliers Removed)") +
    ylab("Duration (ms)") +
    xlab("DSS Use Case (i.e., Trace.name)")
# notch went outside hinges. Try setting notch=FALSE.
```



```
aSpan %>%
    ggplot(aes(Duration)) + geom_histogram(binwidth = 20) +
    ggtitle("Duration (ms) Histogram (Outliers Removed), Binwidth = 20") +
    xlab("Duration (ms)")
```



```
summary(aSpan)
sd(aSpan$Duration)
```

```
   Trace.ID           Trace.name          Start.time            Duration
Length:399         Length:399         Length:399         Min.   :  4.290
Class :character   Class :character   Class :character   1st Qu.:  6.045
Mode  :character   Mode  :character   Mode  :character   Median : 11.100
```

```
                                                          Mean   :180.874
                                                          3rd Qu.:364.500
                                                          Max.   :881.000
   platform              env          useCase            useCaseNum
 Length:399         Min.   :1.000   Length:399         Min.   :1.000
 Class :character   1st Qu.:1.500   Class :character   1st Qu.:2.000
 Mode  :character   Median :2.000   Mode  :character   Median :3.000
                    Mean   :2.496                      Mean   :2.995
                    3rd Qu.:3.000                      3rd Qu.:4.000
                    Max.   :4.000                      Max.   :5.000
    ext             V10
 Mode :logical   Mode :logical
 FALSE:240       FALSE:34
 TRUE :159       TRUE :365
```

229.448905568277

```
# dnorm_aSpan <- aSpan
# dnorm_aSpan$Duration <- dnorm(dnorm_aSpan$Duration,mean=180.874,sd=229.4489)

# dnorm_aSpan %>%
#     ggplot(aes(Duration)) + geom_histogram() +
#     ggtitle("Duration (ms) Histogram (w dnorm, Binwidth = 20") +
#     xlab("Duration (ms)")

# shapiro.test(dnorm_aSpan$Duration)
```

```
# ggpairs(spanMetricsNum, title="correlogram with ggpairs()")
```

3.3.1 mclust

Used mclust to verify the separation of internal and external models as indicated from the useCaseNum vs. Duration plot; i.e. use cases 4 and 5 use an external API.

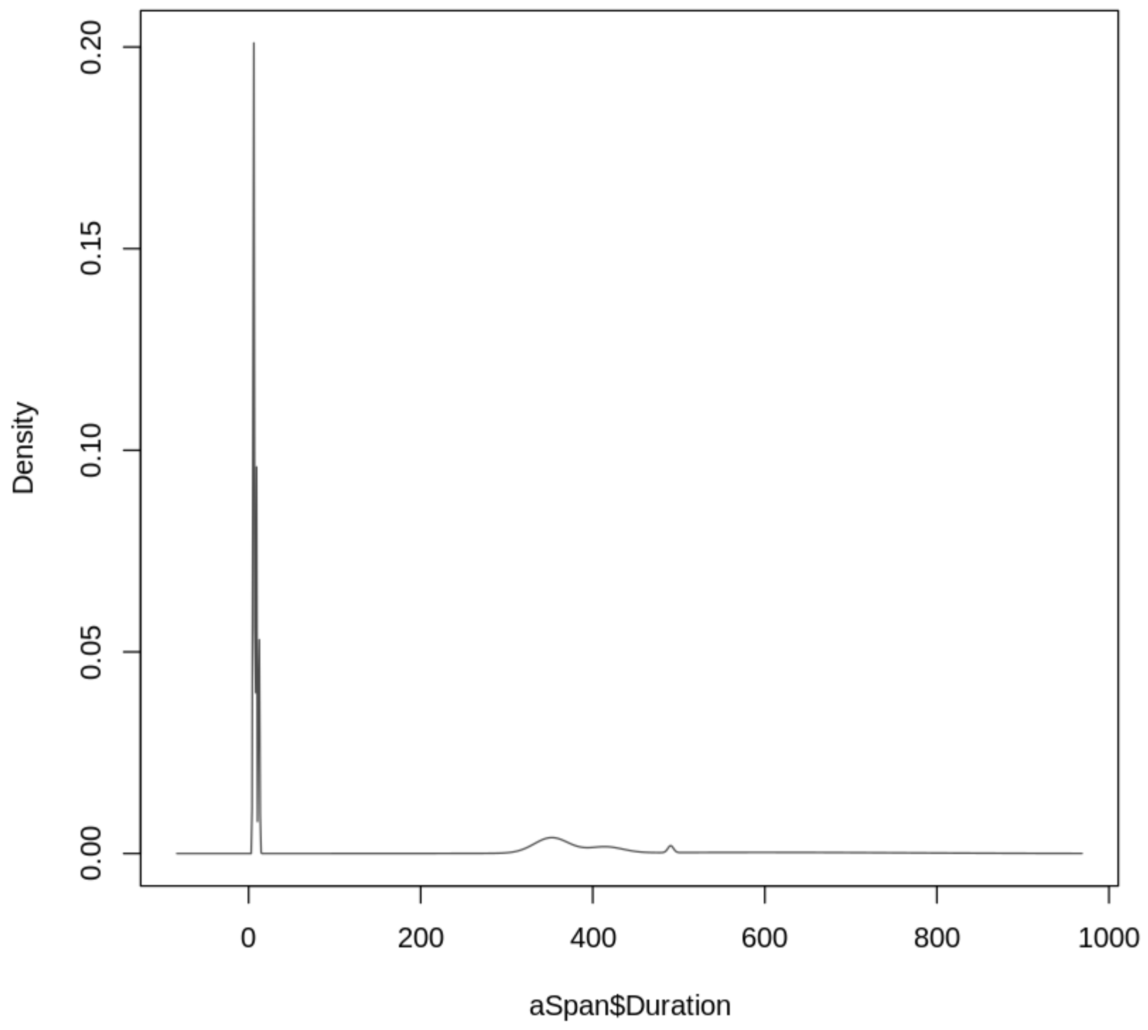
The library mclust is a contributed R package for model-based clustering, classification, and density estimation based on finite normal mixture modelling. It provides functions for parameter estimation via the EM algorithm for normal mixture models with a variety of covariance structures, and functions for simulation from these models.

Scrucca L., Fop M., Murphy T. B. and Raftery A. E. (2016) mclust 5: clustering, classification and density estimation using Gaussian finite mixture models The R Journal 8/1, pp. 289-317

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

3.3.2 Mclust Univariate Analysis of Duration

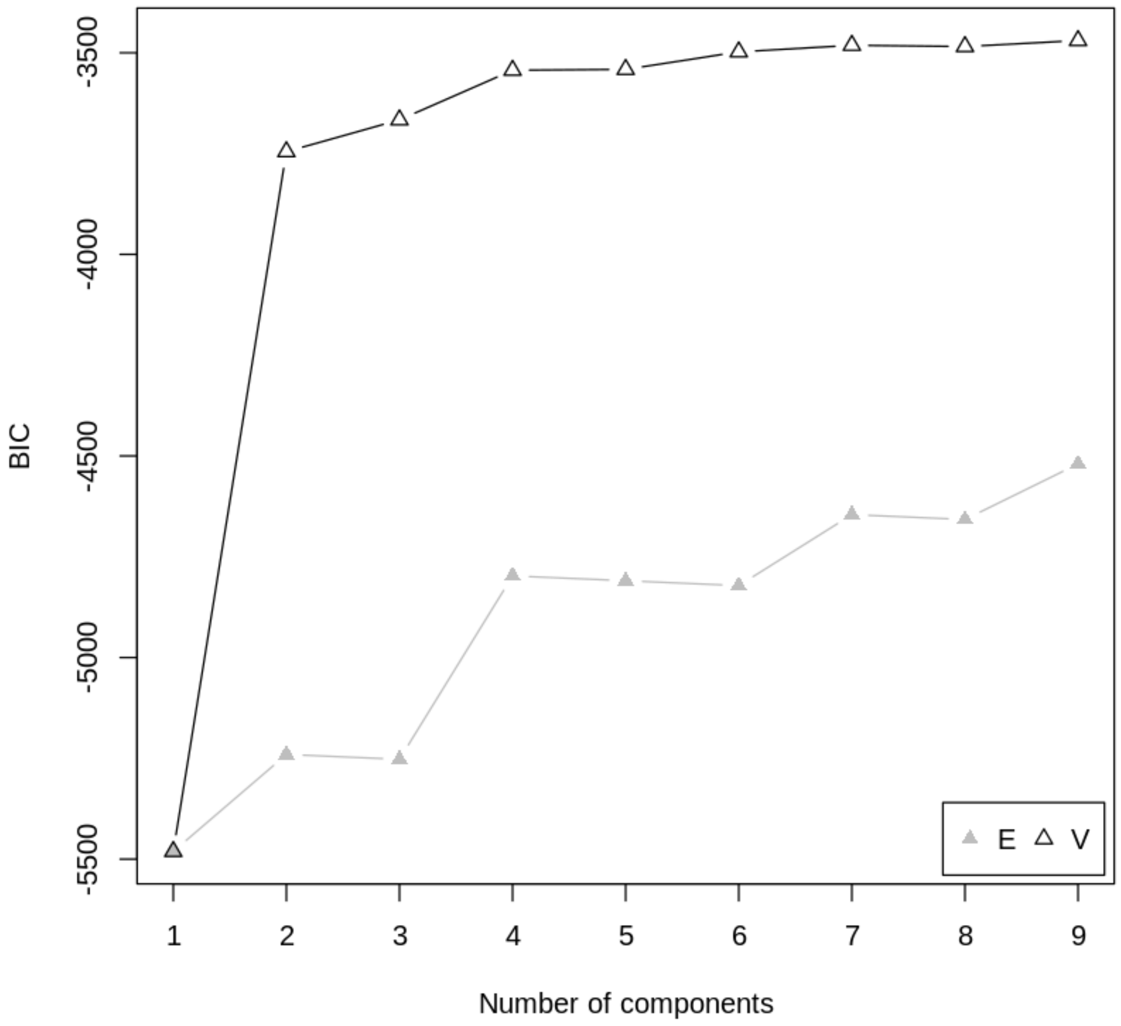
```
mod4 <- densityMclust(aSpan$Duration)
```



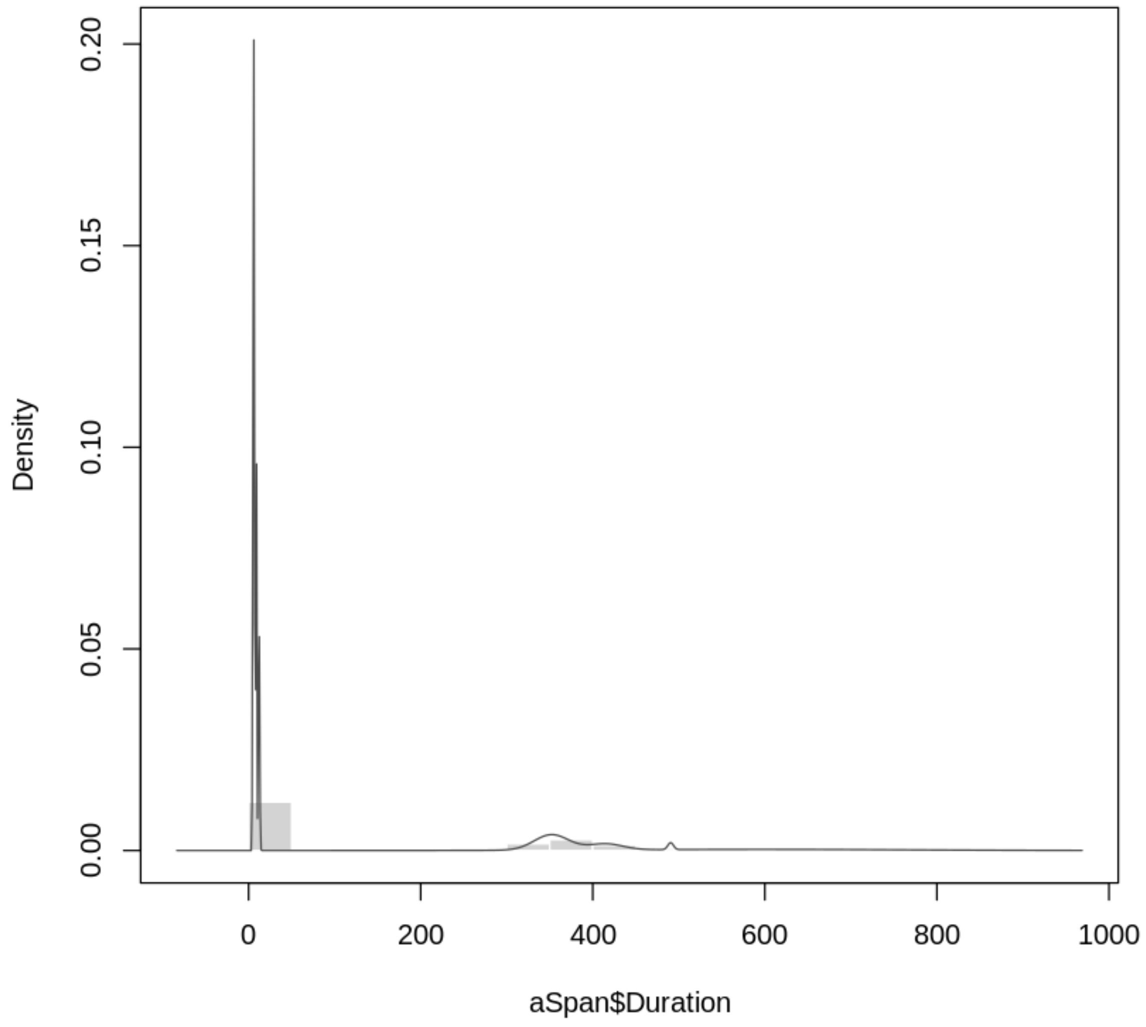
```r
summary(mod4)
```

```
-------------------------------------------------------
Density estimation via Gaussian finite mixture modeling
-------------------------------------------------------

Mclust V (univariate, unequal variance) model with 9 components:
 log-likelihood   n df       BIC       ICL
      -1656.953 399 26 -3469.619 -3511.198
```

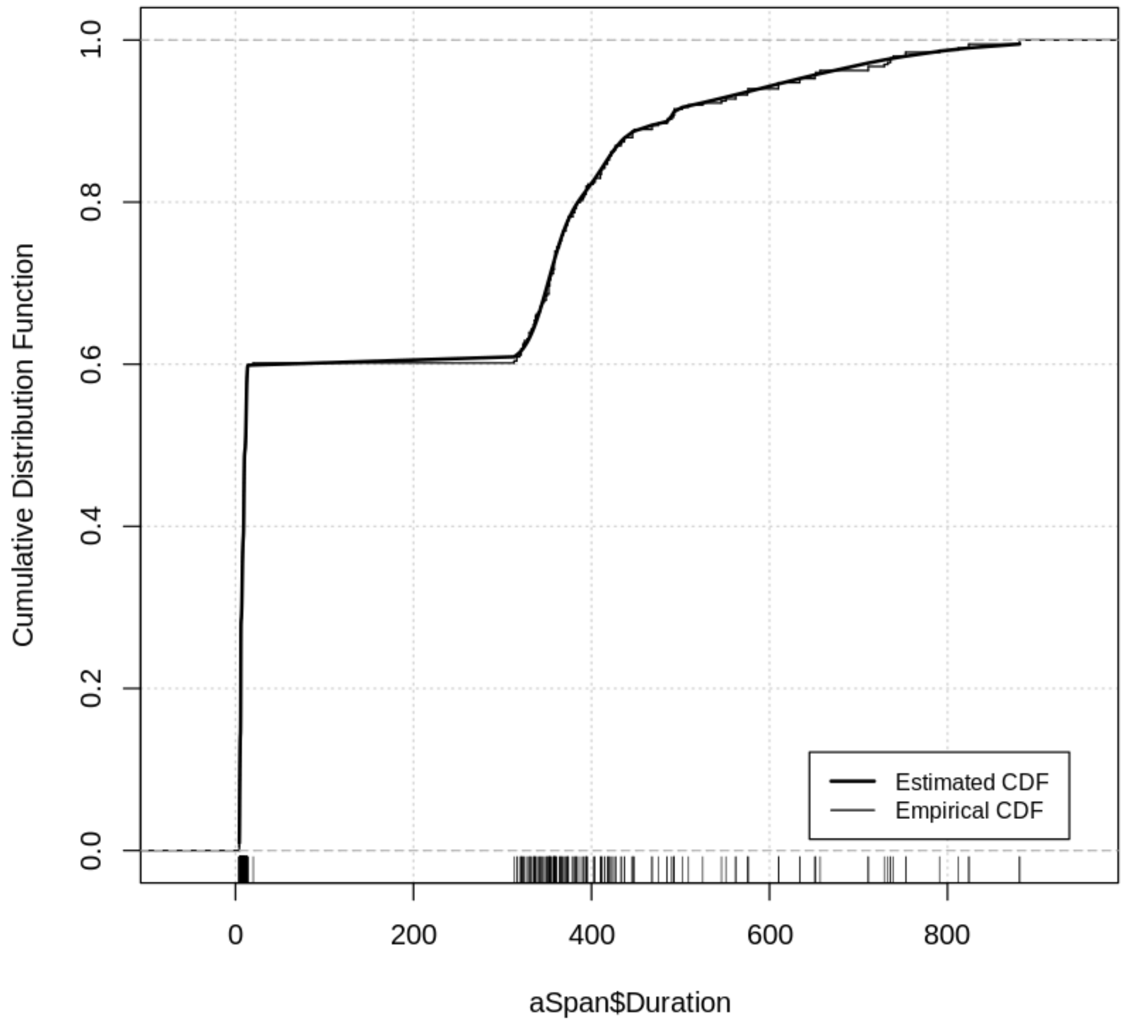
```
plot(mod4, what ="BIC")
```



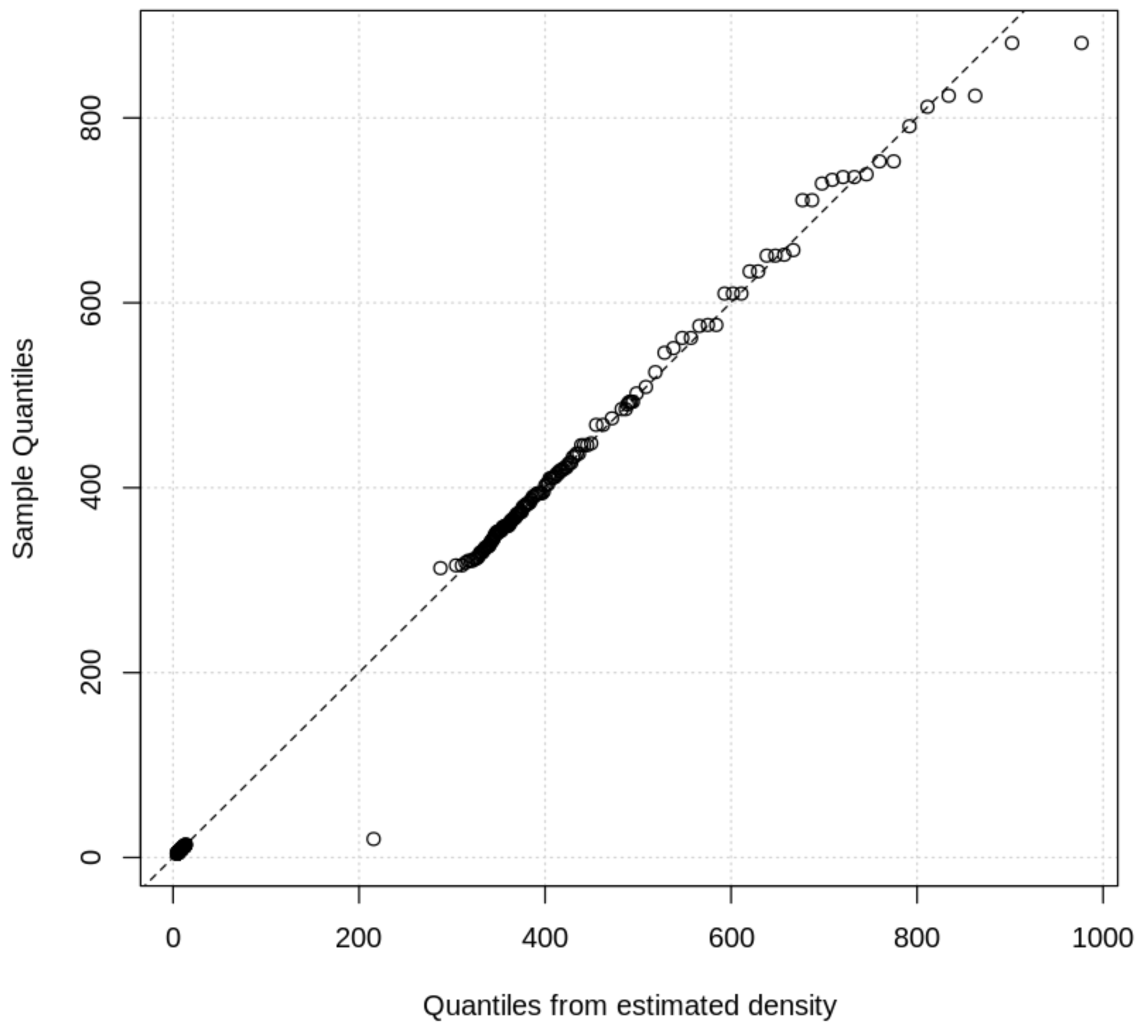
```
plot(mod4, what = "density", data = aSpan$Duration, breaks = 20)
```



```
plot(mod4, what = "diagnostic", type = "cdf")
```



```
plot(mod4, what = "diagnostic", type = "qq")
```

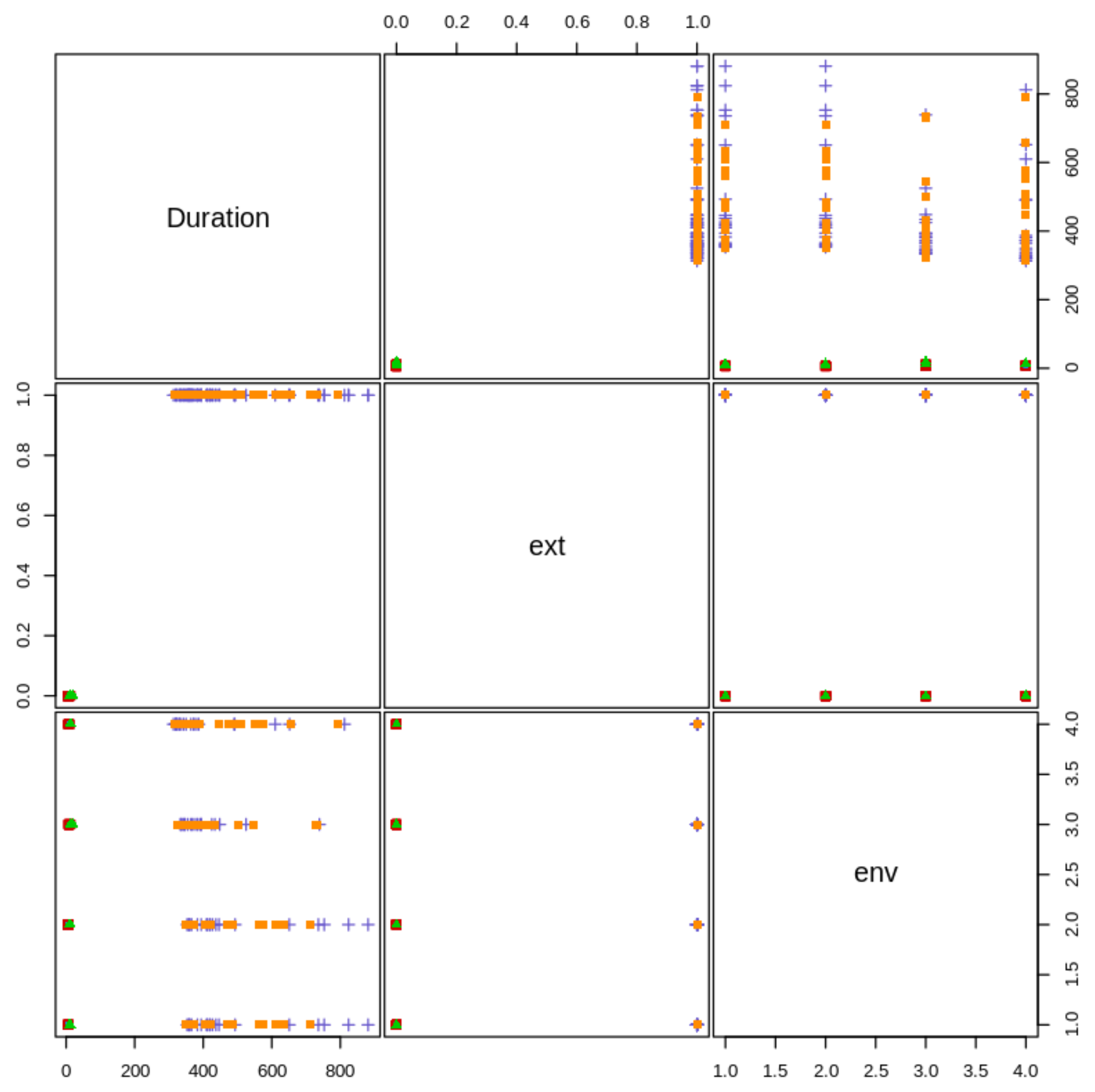


3.3.3 Mclust Multivariate Analysis

```
uc <- aSpan$useCaseNum # Trace.name is char, used uc num conversion

X <- aSpan %>%
    # dplyr::select(useCaseNum, env, ext, Duration)
    dplyr::select(Duration, ext, env)
    # dplyr::select(Duration)

head(X)
clPairs(X, uc)
```

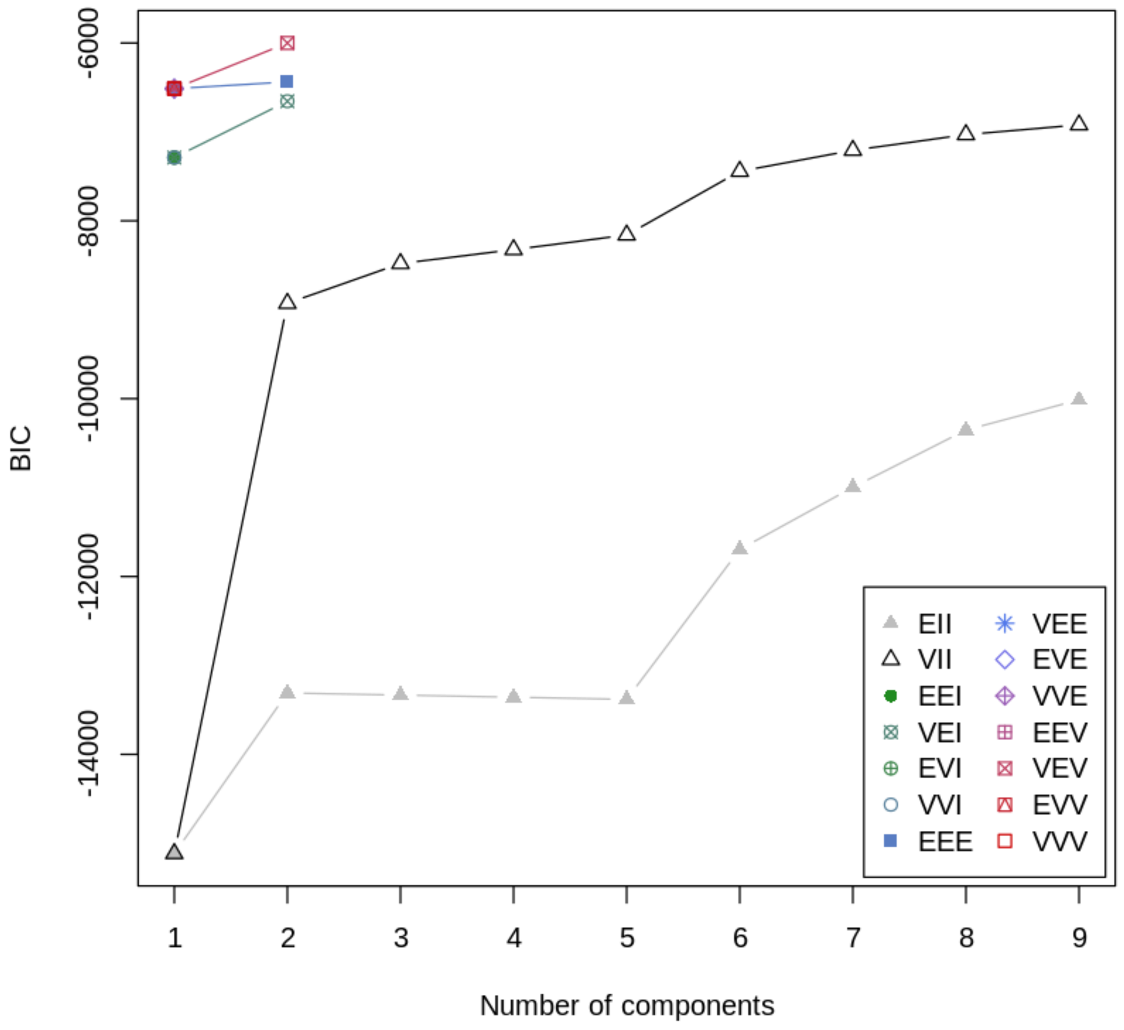
A data.frame: 6 × 3

	Duration <dbl>	ext <lgl>	env <dbl>
21	4.90	FALSE	1
22	4.43	FALSE	1
23	4.67	FALSE	1
24	4.68	FALSE	1
25	4.73	FALSE	1
26	4.89	FALSE	1



```
# spanMclust <- Mclust(aSpan)
# spanMclust <- Mclust(X)
# summary(spanMclust)
# plot(spanMclust, what = c("classification"))
```

```
BIC <- mclustBIC(X)
plot(BIC)
```



```r
summary(BIC)
```

```
Best BIC values:
             VEV,2      EEE,2      EEE,1
BIC      -6000.192 -6437.9228 -6512.9492
BIC diff     0.000  -437.7311  -512.7575
```

Note that 2 is included within the list of best Bayesian Information Criterion (BIC) values.

```
# mod1 <- Mclust(X, x = BIC)
# summary(mod1, parameters = TRUE)
```

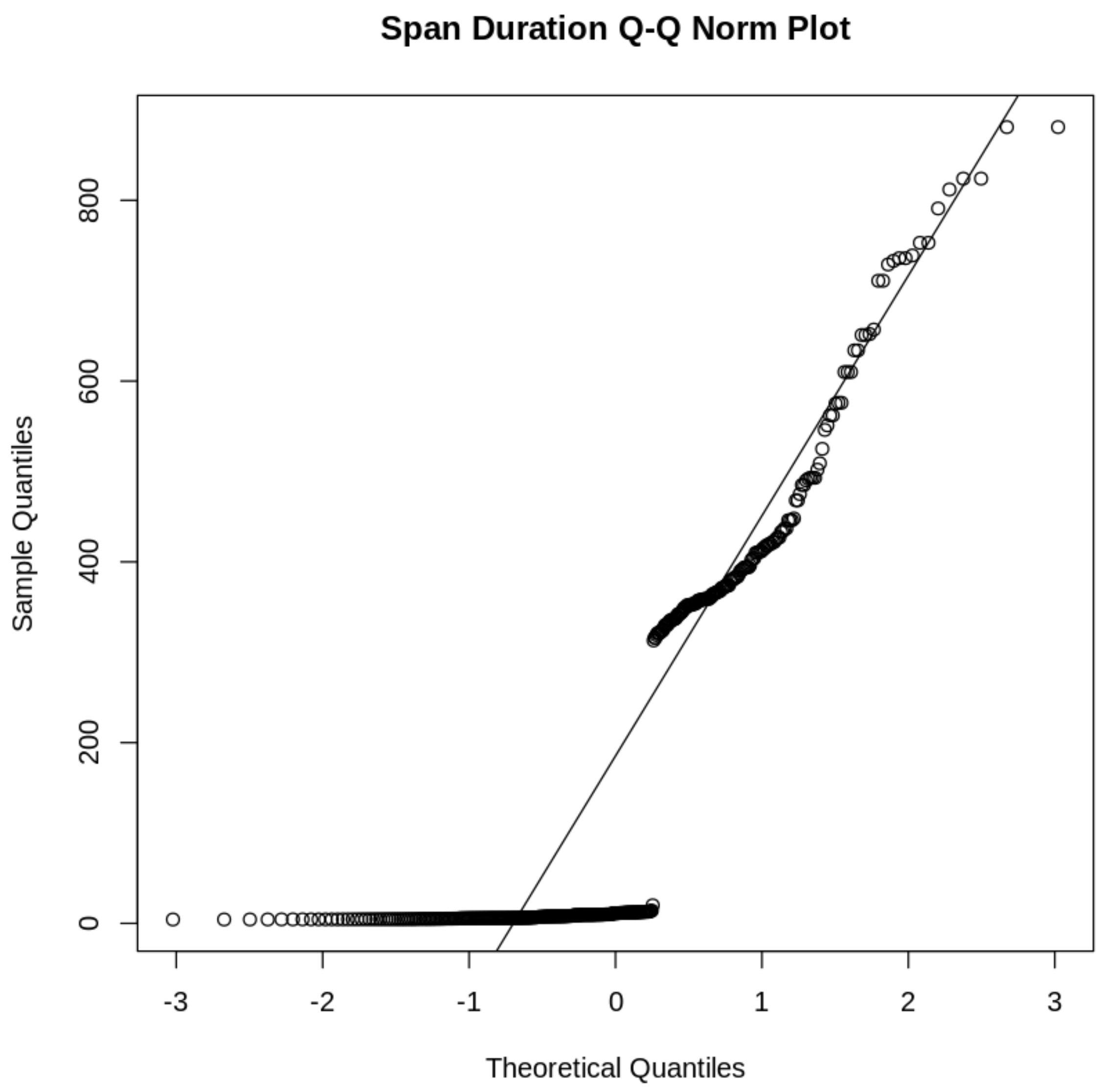
```
# plot(mod1, what = "classification")
```

```
# plot(mod1, what = "uncertainty")
```

```
# ICL <- mclustICL(X)
# summary(ICL)
# plot(ICL)
```

```
# LRT <- mclustBootstrapLRT(X, modelName = "VEV")
# LRT
```

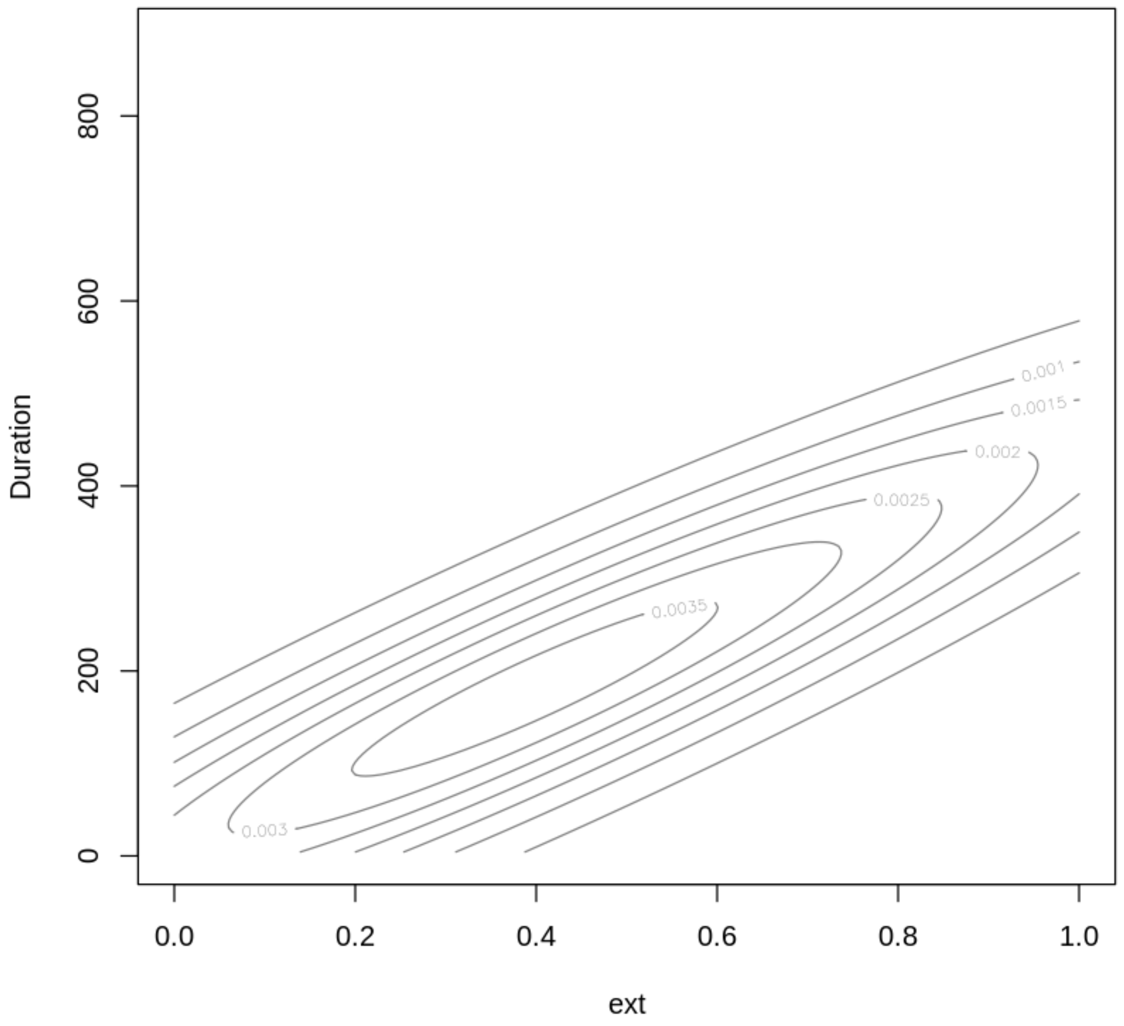
```
qqnorm(aSpan$Duration, main="Span Duration Q-Q Norm Plot")
qqline(aSpan$Duration)
```



```
# head(aSpan)
```

```
aSpan_Density <- aSpan %>%
    # dplyr::select(useCaseNum, env, ext, Duration)
    dplyr::select(ext, Duration)
    # dplyr::select(Duration)
```

```
mod5 <- densityMclust(aSpan_Density)
```



4 Separating Internal from External Data

4.1 Internal Data

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

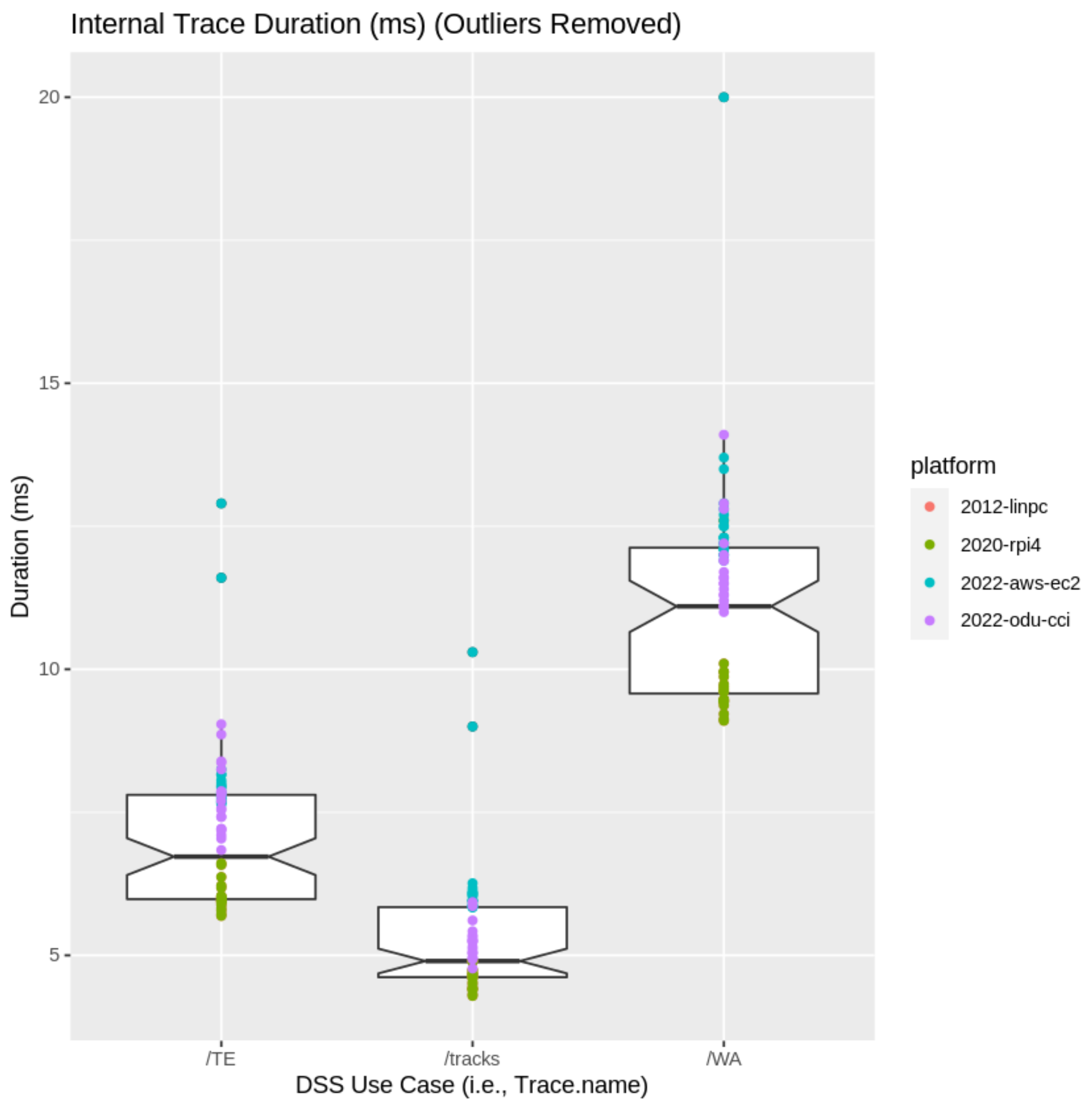
```
# Remove Outliers
# outliers <- boxplot(internalSpanData$Duration, plot = FALSE)$out
# outliers
# iSpan <- iSpan[-which(iSpan$Duration %in% outliers),]

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

# iSpan
```

```
summary(iSpan)
sd(iSpan$Duration)
```

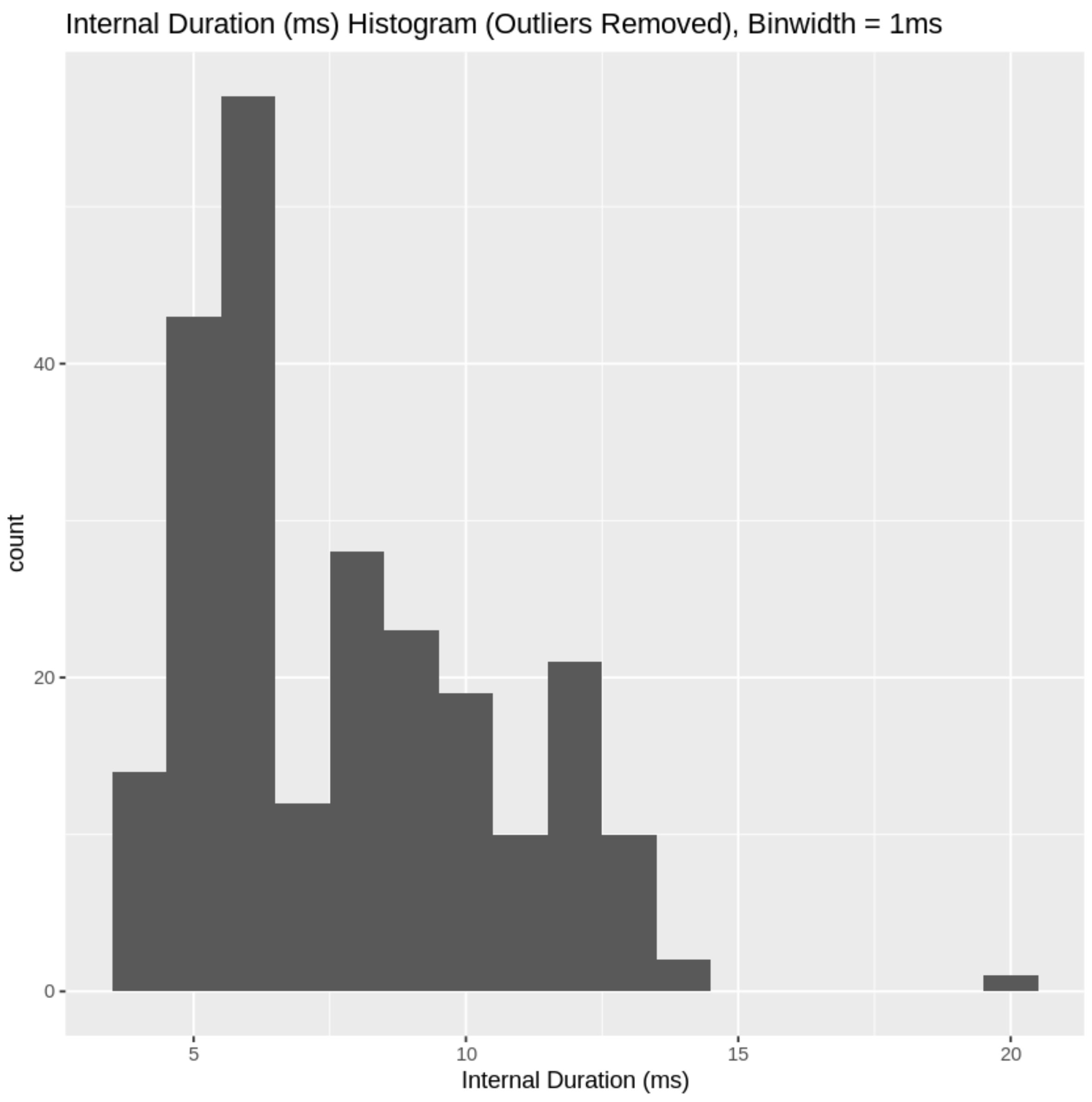
```
  Trace.ID          Trace.name         Start.time           Duration
 Length:240         Length:240         Length:240         Min.   : 4.290
 Class :character   Class :character   Class :character   1st Qu.: 5.713
 Mode  :character   Mode  :character   Mode  :character   Median : 7.070
                                                          Mean   : 7.745
                                                          3rd Qu.: 9.610
                                                          Max.   :20.000
  platform               env          useCase            useCaseNum
 Length:240         Min.   :1.00   Length:240         Min.   :1
 Class :character   1st Qu.:1.75   Class :character   1st Qu.:1
 Mode  :character   Median :2.50   Mode  :character   Median :2
                    Mean   :2.50                      Mean   :2
                    3rd Qu.:3.25                      3rd Qu.:3
                    Max.   :4.00                      Max.   :3
    ext               V10
 Mode :logical   Mode:logical
 FALSE:240       TRUE:240
```

2.77664210997812

```
iSpan %>%
    ggplot(aes(Trace.name, Duration)) +
    stat_boxplot(notch="TRUE") + geom_point(aes(colour = platform)) +
    ggtitle("Internal Trace Duration (ms) (Outliers Removed)") +
    ylab("Duration (ms)") +
    xlab("DSS Use Case (i.e., Trace.name)")
```



```
iSpan %>%
    ggplot(aes(Duration)) + geom_histogram(binwidth = 1) +
    ggtitle("Internal Duration (ms) Histogram (Outliers Removed), Binwidth = 1ms") +
    xlab("Internal Duration (ms)")
```



```
# dnorm_iSpan <- iSpan
# dnorm_iSpan$Duration <- dnorm(dnorm_iSpan$Duration,mean=7.745,sd=2.776)

# dnorm_iSpan %>%
#     ggplot(aes(Duration)) + geom_histogram() +
#     ggtitle("Duration (ms) Histogram (w dnorm, Binwidth = auto") +
#     xlab("Duration (ms)")
# shapiro.test(dnorm_iSpan$Duration)
```

Note that the histogram plot indicates that the data is not normally distrbuted and will need a transformation to enable application of statistics.

4.2 External Data

```
RIC_SpanData = subset(aSpan, useCaseNum == 5)
IAD_SpanData = subset(aSpan, useCaseNum == 4)

externalSpanData <- rbind(RIC_SpanData, IAD_SpanData)
```

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

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

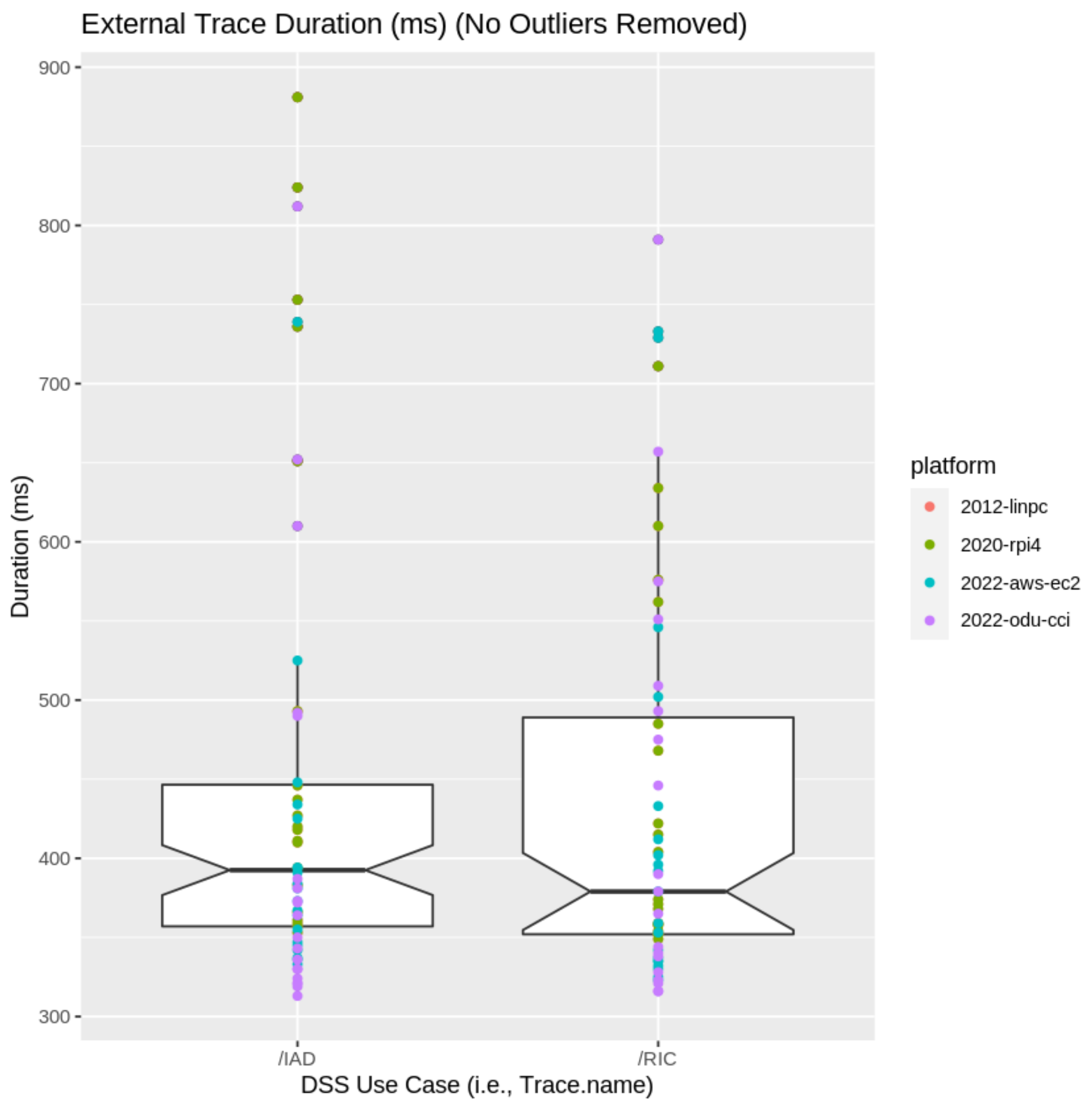
```
summary(eSpan)
sd(eSpan$Duration)
```

```
   Trace.ID           Trace.name          Start.time            Duration
 Length:159         Length:159         Length:159         Min.   :313.0
 Class :character   Class :character   Class :character   1st Qu.:353.0
 Mode  :character   Mode  :character   Mode  :character   Median :387.0
                                                          Mean   :442.2
                                                          3rd Qu.:485.0
                                                          Max.   :881.0
   platform              env          useCase             useCaseNum
 Length:159         Min.   :1.000   Length:159         Min.   :4.000
 Class :character   1st Qu.:1.500   Class :character   1st Qu.:4.000
 Mode  :character   Median :2.000   Mode  :character   Median :4.000
                    Mean   :2.491                      Mean   :4.497
                    3rd Qu.:3.000                      3rd Qu.:5.000
                    Max.   :4.000                      Max.   :5.000
   ext              V10
 Mode:logical   Mode :logical
 TRUE:159       FALSE:34
```

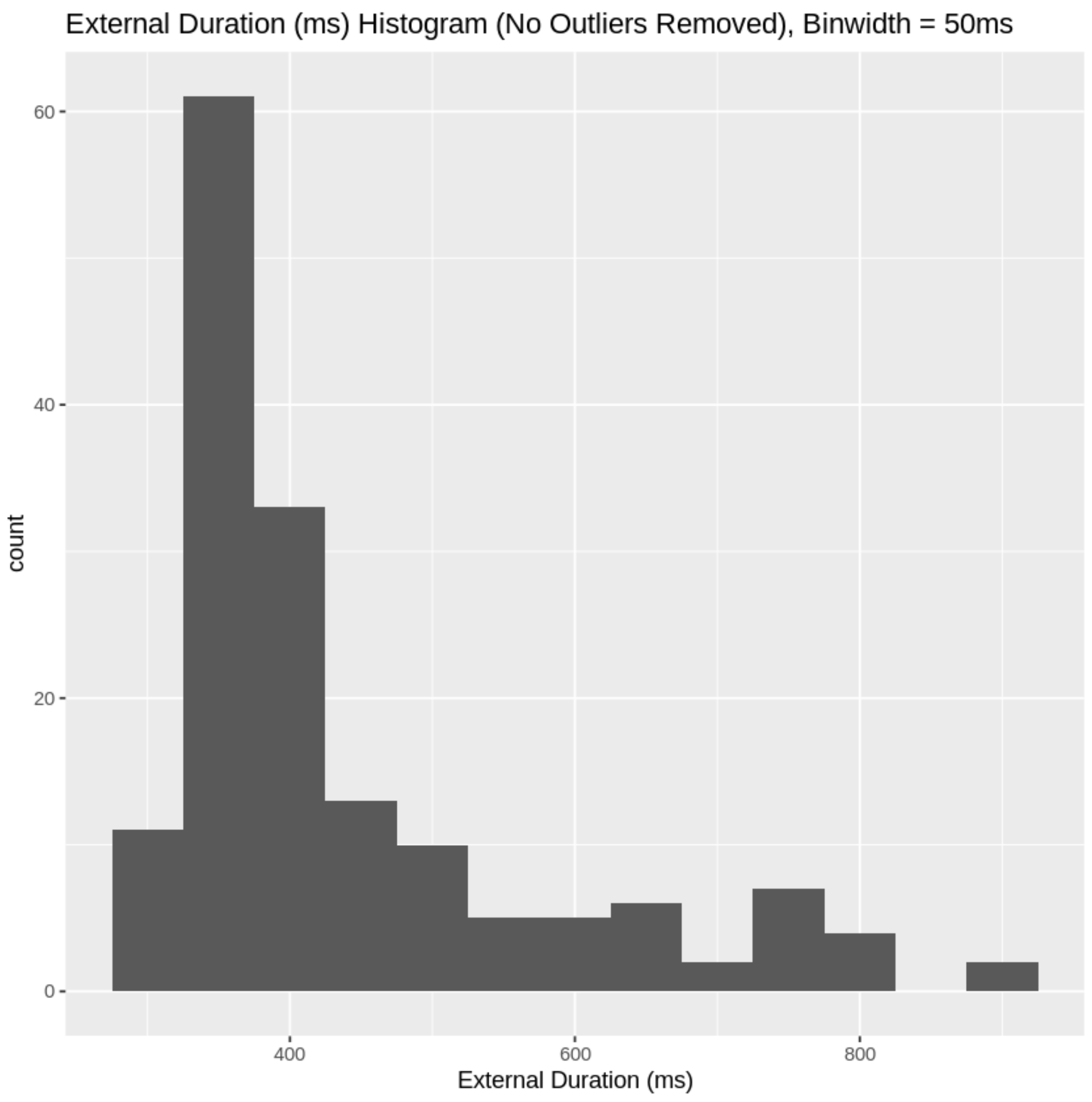
```
                TRUE :125
```

135.466366505137

```
eSpan %>%
    ggplot(aes(Trace.name, Duration)) +
    stat_boxplot(notch="TRUE") + geom_point(aes(colour = platform)) +
    ggtitle("External Trace Duration (ms) (No Outliers Removed)") +
    ylab("Duration (ms)") +
    xlab("DSS Use Case (i.e., Trace.name)")
```



```
eSpan %>%
    ggplot(aes(Duration)) + geom_histogram(binwidth = 50) +
    ggtitle("External Duration (ms) Histogram (No Outliers Removed), Binwidth = 50ms") +
    xlab("External Duration (ms)")
```



Note that the histogram plot of the external data indicates that the data is not normally distributed and will need a transformation to enable application of statistics.

5 Transformation and Normality Testing of the Data

The histograms of the internal and external span data imply that a log transform is needed; however, we need to look at cube and sqrt transforms. A Box-Cox transformation may also need to be explored. Once that data has been transformed, we shall use a Shapiro-Wilk test to assess whether or not the data is normally distributed.

5.1 Box-Cox Transformation

Box and Cox (1964) developed a family of transformations designed to reduce nonnormality of the errors in a linear model. Applying this transform often reduces non-linearity as well, and heteroscedascity.

The idea is to transform the response variable Y to a replacement response variable $Y_i^{(\lambda)}$, leaving the right-hand side of the regression model unchanged, so that the regression residuals become normally-distributed. Note that the regression coefficients will also change, because the response variable has changed; therefore, the regression coefficients must be interpreted with respect to the transformed variable. Also, any predictions made with the model have to be back-transformed, to be interpreted in the original units.

The standard (simple) Box-Cox transform is:

$$Y_i^{(\lambda)} = \begin{cases} \frac{Y_i^\lambda - 1}{\lambda}, & (\lambda \neq 0) \\ log(Y_i), & (\lambda = 0) \end{cases}$$

Box, G. E. P., & Cox, D. R. (1964). An Analysis of Transformations. Journal of the Royal Statistical Society, Series B (Metholodogical), 26(2), 211-252.

http://www.css.cornell.edu/faculty/dgr2/_static/files/R_html/Transformations.html

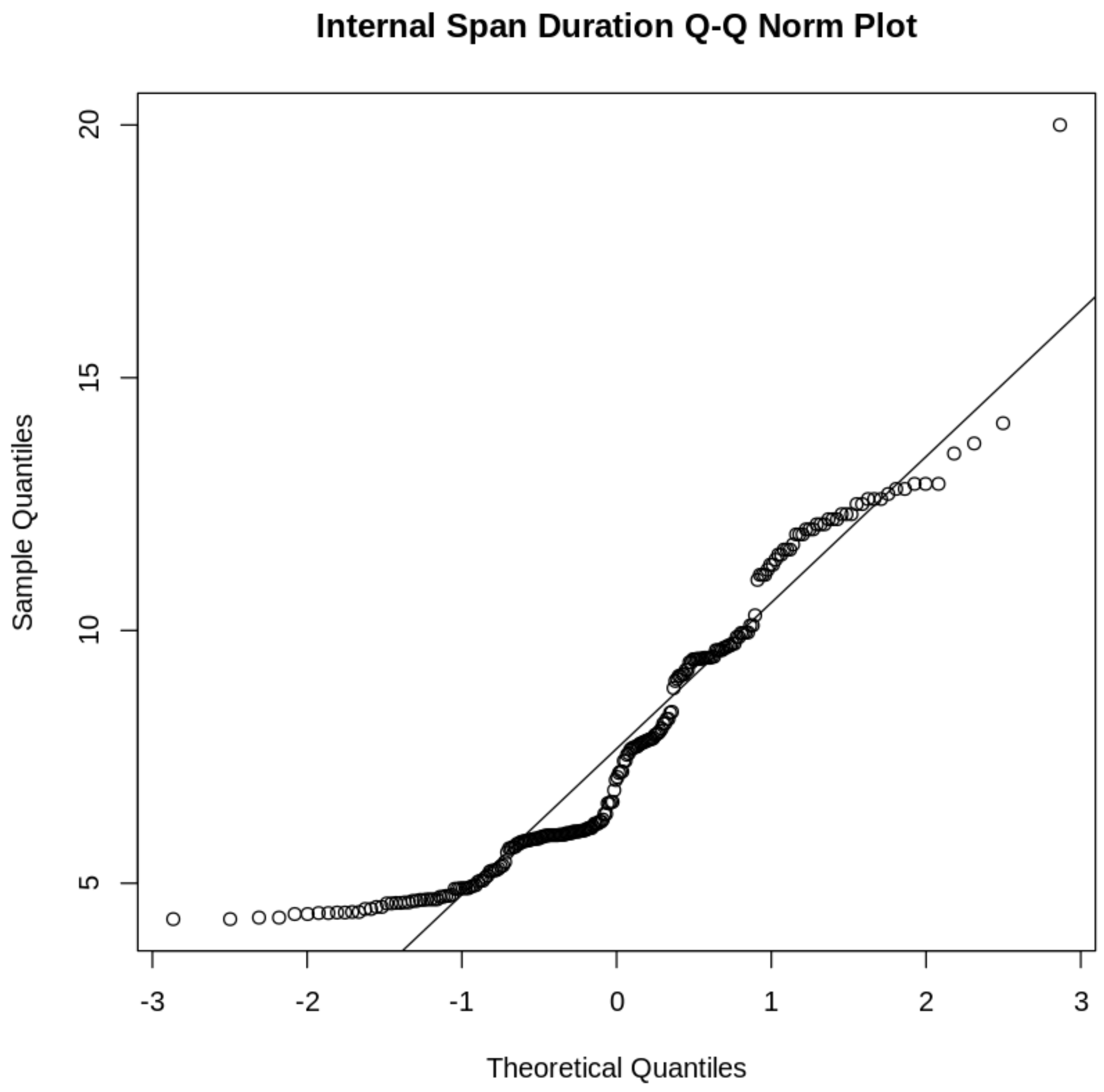
5.2 Shapiro-Wilk Test

The null-hypothesis of this test is that the population is normally distributed. Thus, if the p value is less than the chosen alpha level, then the null hypothesis is rejected and there is evidence that the data tested are not normally distributed. On the other hand, if the p value is greater than the chosen alpha level, then the null hypothesis (that the data came from a normally distributed population) can not be rejected (e.g., for an alpha level of .05, a data set with a p value of less than .05 rejects the null hypothesis that the data are from a normally distributed population).

https://en.wikipedia.org/wiki/Shapiro–Wilk_test

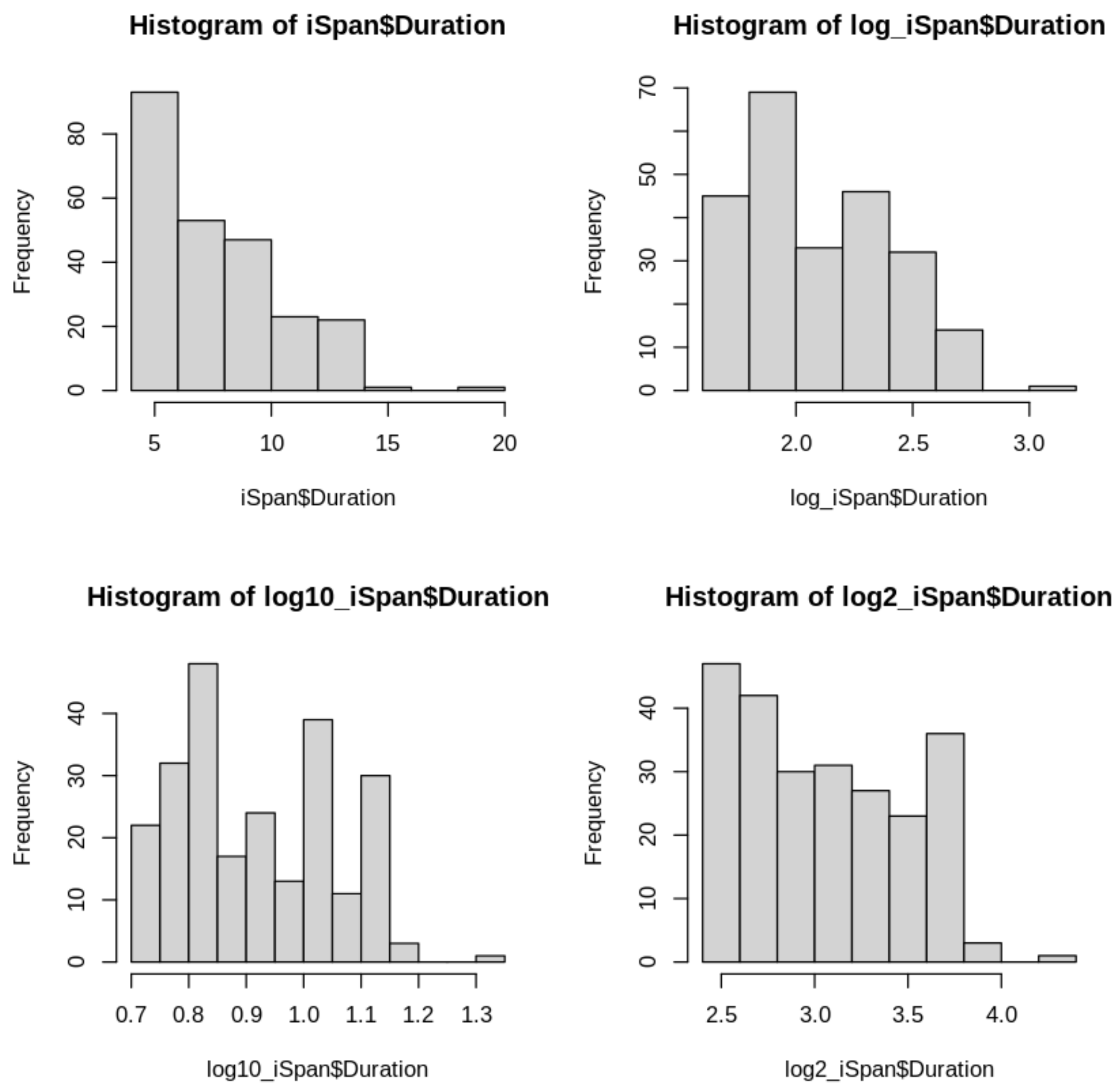
5.3 Data Transformations and Hypothesis Testing (Internal Data)

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



5.3.1 Sqrt-Log-Cube Transformations

```r
sqrt_iSpan <- iSpan
sqrt_iSpan$Duration=sqrt(sqrt_iSpan$Duration)

log_iSpan <- iSpan
log_iSpan$Duration=log(log_iSpan$Duration + 1) # Natural Log
log10_iSpan <- iSpan
log10_iSpan$Duration=log10(log10_iSpan$Duration + 1) # Log Base 10
log2_iSpan <- iSpan
log2_iSpan$Duration=log2(log2_iSpan$Duration + 1) # Log Base 2

cube_iSpan <- iSpan
cube_iSpan$Duration=cube_iSpan$Duration^(1/3)

par(mfrow=c(2,2))
hist(iSpan$Duration, counts = 10)
hist(log_iSpan$Duration)
hist(log10_iSpan$Duration)
hist(log2_iSpan$Duration)
```

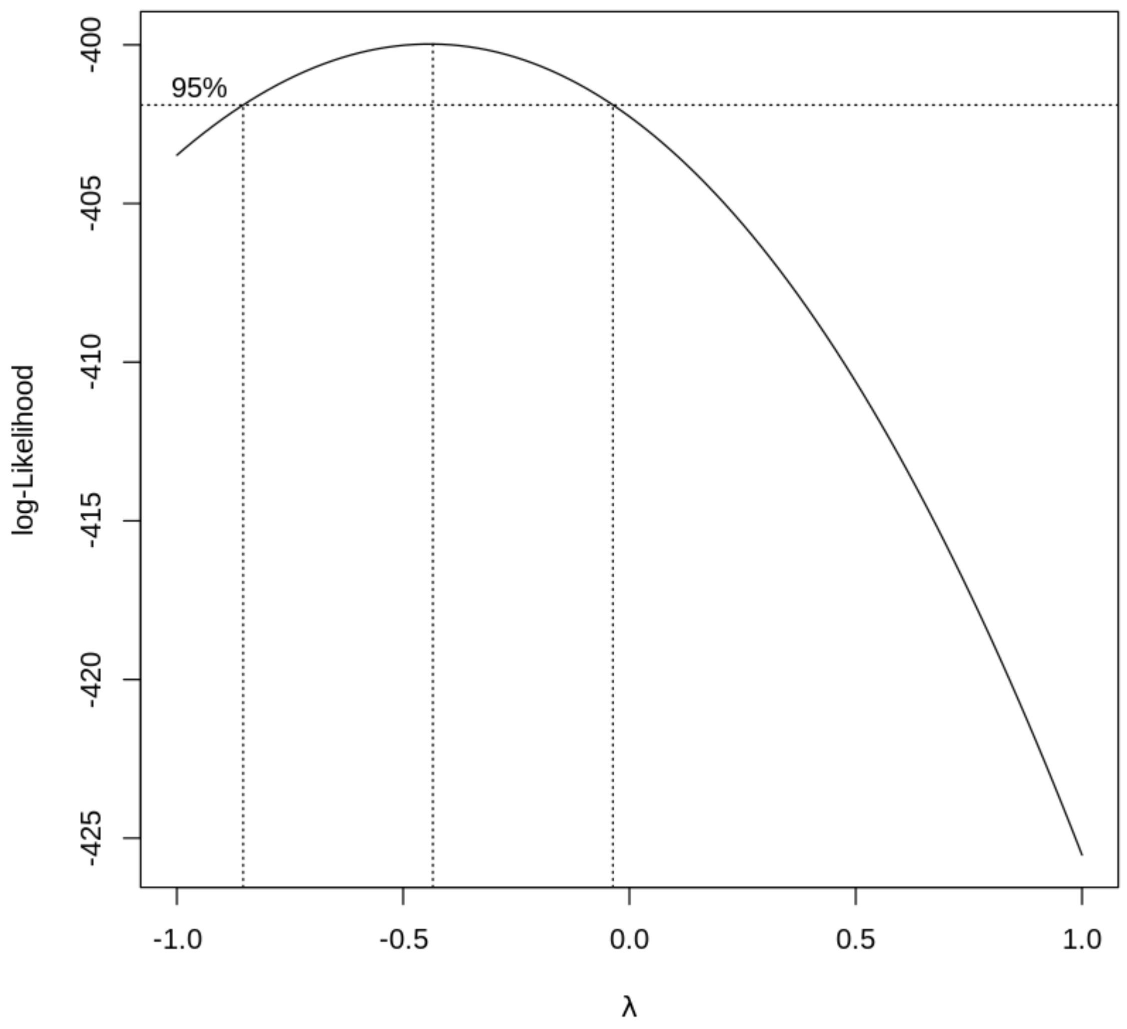


5.3.2 Box-Cox Transformation

```
library("MASS")
```

```
bc_iSpan = iSpan
x <- bc_iSpan$Duration
```

```
bc = boxcox(lm(x ~ 1), seq(-1,1,.1))
# bc = boxcox(lm(x ~ bcData$useCaseNum))
lambda <- bc$x[which.max(bc$y)]
new_x_exact <- (x ^ lambda - 1) / lambda
```


log-Likelihood
λ