Analysis of Repairable Systems with mcotear

Jake Warren

8/22/2019

Abstract

This document is intended to introduce the use of the `mcotear` package for analysis of data on repairable systems.

Getting started with `mcotear`

Assuming the `mcotear` package is installed on your computer, the first step would be to load the package. The package can be loaded using the `library` function as shown below. There are many ways you can learn more about the package, one is through listing all the functions in the package by using the command `ls("package:mcotear")`. For all the functions in the package you can visit the help package using the help. An example would be running the command `?ttt` (or equivalently `help(ttt)`) to see the help page/file for the `ttt` (Total Time on Test) function.

Additional, most, if not all of the functions in the package are documented with examples. If a function has an example you can see that example documented at the bottom of the function's help page. There you would see code that would execute the function. If you want to see what is actually produced by the example you could copy that code and run it in your R session, or you could run the example without having to copy/paste or retype the whole example by using the `example` function. For example, running the command `example(ttt)` in your R session would run the example documented for the `ttt` function.

Lastly, in the package I documented some of the references I used in order to create this package. You can see/access those references using the command `browseVignettes("mcotear")`.

During the loading of the `mcotear` package via the `library` function you should see via a message returned that the `ggplot2` package is loaded. In the last part of the following code block I set a `ggplot2` black and white color theme for use throughout this paper.

```
library(mcotear)
## Loading required package: ggplot2
## Registered S3 methods overwritten by 'ggplot2':
##   method         from
##   [.quosures     rlang
##   c.quosures     rlang
##   print.quosures rlang
## Loading required package: gridExtra
## Loading required package: goftest
# Loads the package

#ls("package:mcotear")
# Lists all of the functions/objects in the package.

#?ttt
#example(ttt)


#browseVignettes("mcotear")
# browseVignettes shows you some of the package
# documentation. I included some of the papers
# I used to create these functions within the
```

```r
# documentation.

theme_set(theme_bw())
# Sets ggplot2 color theme
```

Objects in `mcotear`

In the `mcotear` package that are few objects that get loaded for use along with the functions. The objects that are loaded are `amsaa`, `cbPalette`, and `cbbPalette`. The first object `amsaa` is a data set that came from one of the references documented in package and accessible via `browseVignettes("mcotear")`. The data set contains simulated data consisting of failure times for three repairable systems (you can see this in the help documentation for `amsaa` using `?amsaa`).

```r
amsaa
##    Failure System  Time
## 1        1     S1   4.3
## 2        2     S1   4.4
## 3        3     S1  10.2
## 4        4     S1  23.5
## 5        5     S1  23.8
## 6        6     S1  26.4
## 7        7     S1  74.0
## 8        8     S1  77.1
## 9        9     S1  92.1
## 10      10     S1 197.2
## 11       1     S2   0.1
## 12       2     S2   5.6
## 13       3     S2  18.6
## 14       4     S2  19.5
## 15       5     S2  24.2
## 16       6     S2  26.7
## 17       7     S2  45.1
## 18       8     S2  45.8
## 19       9     S2  75.7
## 20      10     S2  79.7
## 21      11     S2  98.6
## 22      12     S2 120.1
## 23      13     S2 161.8
## 24      14     S2 180.6
## 25      15     S2 190.8
## 26       1     S3   8.4
## 27       2     S3  32.5
## 28       3     S3  44.7
## 29       4     S3  48.4
## 30       5     S3  50.6
## 31       6     S3  73.6
## 32       7     S3  98.7
## 33       8     S3 112.2
## 34       9     S3 129.8
## 35      10     S3 136.0
## 36      11     S3 195.8
```

The other objects `cbPalette`, and `cbbPalette` are hexidecimal color code values for color-blind friendly color palettes. These color-blind friendly color palettes come from the "Cookbook for R", at http://www.

cookbook-r.com/Graphs/Colors_(ggplot2)/. Again, this is documented in the `cbPalette`, and `cbbPalette` help pages/files.

```
cbPalette
## [1] "#999999" "#E69F00" "#56B4E9" "#009E73" "#F0E442" "#0072B2" "#D55E00"
## [8] "#CC79A7"
cbbPalette
## [1] "#000000" "#E69F00" "#56B4E9" "#009E73" "#F0E442" "#0072B2" "#D55E00"
## [8] "#CC79A7"
```

Repairable Systems Analysis Example

```
(m <- power_law_process(
  t = split(amsaa$Time, amsaa$System),
  T = list(200,200,200),                   # If any one was failure truncated
  alpha = 0.05,
  fail.trunc = FALSE,                      # then make this true
  iter = 10
))
## $estimates
##    lambda      beta
## 3.5255156 0.6153363
##
## $beta.convergence
## [1] 0.6153363
##
## $lambda.convergence
## [1] 0.4605471
# Fit Power-Law NHPP (AMSAA-Crow Model)
# the .converge items can be ignored since
# we have time terminated data.
# If the data were failure terminated
# then iterative methods need to be used
# to solve for the parameters, so the
# .converge items are included so you can
# assess if they if the reached convergence.
# So, the iter parameter in the function is
# only applicable to failure terminated data,
# when we would make fail.trunc = TRUE.
# Also, note that the final lambda estimate
# is a transformation of the lambda.converge,
# so they will not match.
# Additionally, alpha is not implemented right
# now. I plan to include confidence intervals
# for the parameters in a future update, which
# will then depend upon alpha.
```

```
common_beta(
  t = split(amsaa$Time, amsaa$System),
  T = list(200,200,200),
  fail.trunc = FALSE)
## $`Test Statistic`
## [1] 1.762104
```

```
##
## $`P-Value`
## [1] 0.4143468
##
## $df
## [1] 2
# Large p-value, do not reject null hypothesis
# of a common shape parameter
```

```
trend_test(
  t = split(amsaa$Time, amsaa$System),
  T = list(200,200,200),
  fail.trunc = FALSE)
## $`Mil-HDBK`
##                test statistic df        pval
## 1 Mil-HDBK Combined  117.0092 72 0.001269251
## 2      Mil-HDBK TTT  117.0092 72 0.001269251
##
## $Laplace
##               test statistic        pval
## 1 Laplace Combined -3.012036 0.002595015
## 2      Laplace TTT -3.012036 0.002595015
##
## $`Anderson-Darling`
##               test statistic        pval
## 1 Anderson-Darling  5.498379 0.001663774
# Low p-values across all tests indicates we do
# not have a Homogeneous Poisson Process
# (failure rate is not constant).
```

```
power_law_mcf(t = amsaa$Time, m$est[1], m$est[2])
##        t  power_mcf
## 1    0.1  0.1116703
## 2    4.3  1.1299785
## 3    4.4  1.1460771
## 4    5.6  1.3294172
## 5    8.4  1.7061477
## 6   10.2  1.9226592
## 7   18.6  2.7826040
## 8   19.5  2.8646999
## 9   23.5  3.2132301
## 10  23.8  3.2384096
## 11  24.2  3.2717931
## 12  26.4  3.4517434
## 13  26.7  3.4758271
## 14  32.5  3.9227507
## 15  44.7  4.7727438
## 16  45.1  4.7989792
## 17  45.8  4.8446768
## 18  48.4  5.0121089
## 19  50.6  5.1510962
## 20  73.6  6.4868174
## 21  74.0  6.5084881
## 22  75.7  6.6000905
```

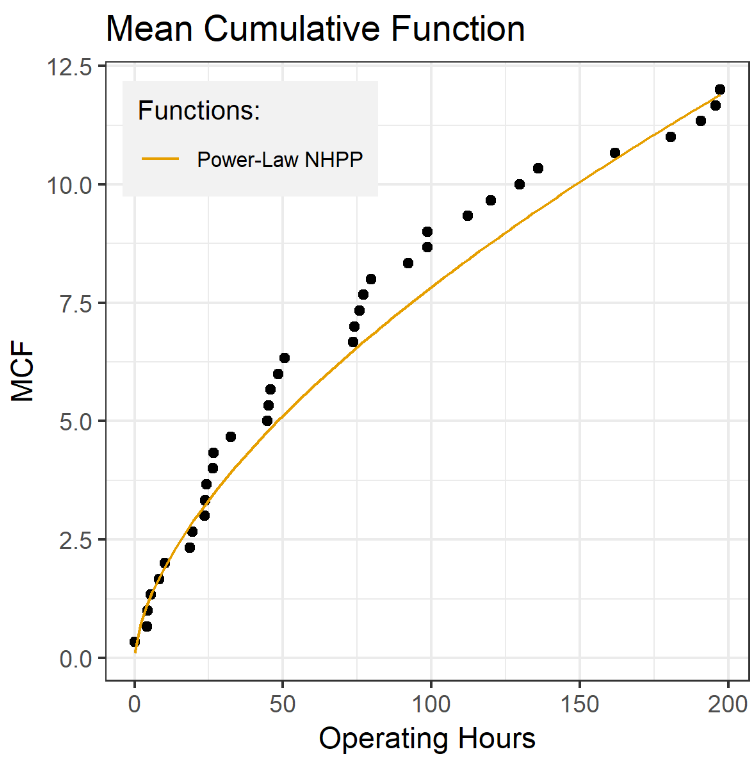
```
## 23  77.1  6.6749350
## 24  79.7  6.8125595
## 25  92.1  7.4465349
## 26  98.6  7.7656680
## 27  98.7  7.7705134
## 28 112.2  8.4083154
## 29 120.1  8.7678331
## 30 129.8  9.1970511
## 31 136.0  9.4649409
## 32 161.8 10.5326738
## 33 180.6 11.2697508
## 34 190.8 11.6572637
## 35 195.8 11.8443028
## 36 197.2 11.8963433
# power_law_mcf(t = list(amsaa$Time), m$est[1], m$est[2])
# power_law_mcf(t = split(amsaa$Time, amsaa$System), m$est[1], m$est[2])
# Any one of these work. These values themselves
# are not real useful. This function is more
# useful in plotting the function versus the
# observed/emperical mcf estimates so you can
# visually assess if the model fits the data.
```

```
df_mcf <- mcf(t = split(amsaa$Time, amsaa$System), by = NULL)
head(df_mcf)
##      t       mcf
## 1  0.1 0.3333333
## 2  4.3 0.6666667
## 3  4.4 1.0000000
## 4  5.6 1.3333333
## 5  8.4 1.6666667
## 6 10.2 2.0000000
names(df_mcf)[1] <- "Time"
head(df_mcf)
##   Time       mcf
## 1  0.1 0.3333333
## 2  4.3 0.6666667
## 3  4.4 1.0000000
## 4  5.6 1.3333333
## 5  8.4 1.6666667
## 6 10.2 2.0000000
# Get the nonparametric mcf estimates.
# I show the first 6 rows so you can see
# what it looks like, and you can see the
# column names. Then I change the name
# of "t" to "Time" so it matches the name
# in the amsaa data set.
```

```
amsaa1 <- merge(amsaa, df_mcf, by = "Time")
head(amsaa1)
##   Time Failure System       mcf
## 1  0.1       1     S2 0.3333333
## 2  4.3       1     S1 0.6666667
## 3  4.4       2     S1 1.0000000
## 4  5.6       2     S2 1.3333333
```

```
## 5  8.4       1     S3 1.6666667
## 6 10.2       3     S1 2.0000000
# Merge the nonparametric mcf estimates
# with amsaa into a new data.frame amsaa1
```

```r
#dev.new(height = 4, width = 4)
ggplot(amsaa1, aes(x = Time, y = mcf)) +
  geom_point() +
  labs(
    x = "Operating Hours", y = "MCF",
    title = "Mean Cumulative Function") +
  stat_function(
    fun = function(x){
      power_law_mcf(t = x, lambda = m$e[[1]], beta = m$e[[2]])$power_mcf
    },
    mapping = aes(colour = "Power-Law NHPP")
  ) +
  scale_colour_manual("Functions:",
    breaks = c("Power-Law NHPP"),
    values = c("Power-Law NHPP" = cbPalette[2]),
    guide = guide_legend(
      override.aes = list(
        linetype = c("solid")
    ))
  ) +
  theme(legend.position = c(.225,.875),
    legend.background = element_rect(fill="grey95"),
    legend.key = element_rect(fill="grey95"),
    legend.text = element_text(size=8),
    legend.title=element_text(size=10)
  )
```



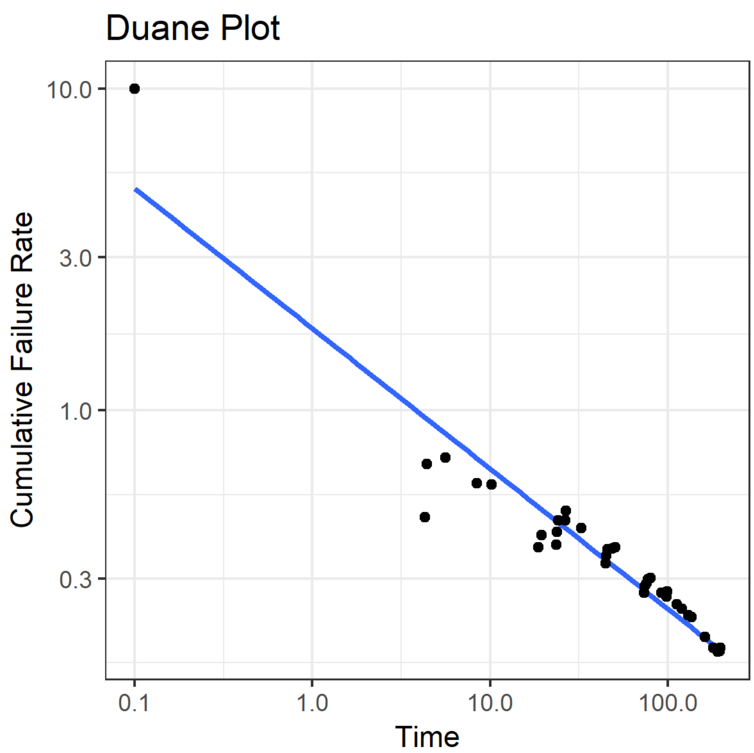

```
# setwd("C:/Users/ThinkPad_User_003/Desktop/Analysis of Repairable Systems")
# ggsave("AMSAA_MCF.png", dpi = 300, height = 4, width = 4)
# Mean Cumulative Function Plot
# Looks like the data fits pretty well.
# I show how you can set your working directoy
# and save the plot to file.
```

```
df_rocof <- rocof(t = split(amsaa$Time, amsaa$System), by = NULL)
head(df_rocof)
##         t     mtbf      rocof
## S21   0.1 0.100000 10.0000000
## S11   4.3 2.150000  0.4651163
## S12   4.4 1.466667  0.6818182
## S22   5.6 1.400000  0.7142857
## S31   8.4 1.680000  0.5952381
## S13 10.2 1.700000  0.5882353
names(df_rocof)[1] <- "Time"
head(df_rocof)
##      Time     mtbf      rocof
## S21   0.1 0.100000 10.0000000
## S11   4.3 2.150000  0.4651163
## S12   4.4 1.466667  0.6818182
## S22   5.6 1.400000  0.7142857
## S31   8.4 1.680000  0.5952381
## S13 10.2 1.700000  0.5882353
# Get the nonparametric rocof and mtbf estimates.
# I show the first 6 rows so you can see
# what it looks like, and you can see the
# column names. Then I change the name
```

```r
# of "t" to "Time" so it matches the name
# in the amsaa data set. The rocof and mtbf
# are inverses of one another. You can use
# either of these to create a Duane plot.
```

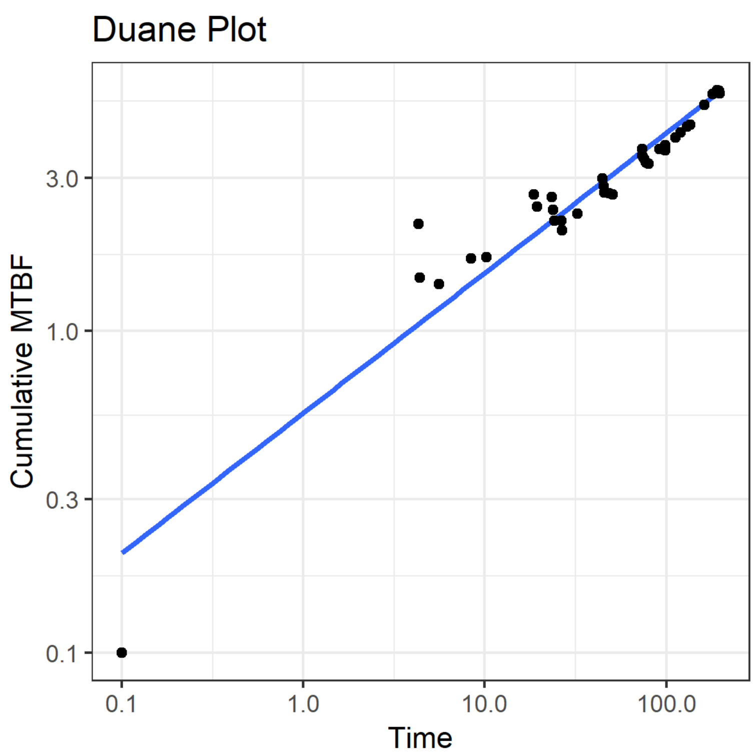
```r
amsaa2 <- merge(amsaa1, df_rocof, by = "Time")
head(amsaa2)
##   Time Failure System       mcf     mtbf      rocof
## 1  0.1       1     S2 0.3333333 0.100000 10.0000000
## 2  4.3       1     S1 0.6666667 2.150000  0.4651163
## 3  4.4       2     S1 1.0000000 1.466667  0.6818182
## 4  5.6       2     S2 1.3333333 1.400000  0.7142857
## 5  8.4       1     S3 1.6666667 1.680000  0.5952381
## 6 10.2       3     S1 2.0000000 1.700000  0.5882353
# Merge our data set with the df_rocof by the Time variable
```

```r
ggplot(amsaa2,
  aes(
    x = Time,
    y = rocof)) +
  scale_x_log10() +
  scale_y_log10() +
  geom_smooth(method='lm', se = FALSE) +
  geom_point() +
  labs(y = "Cumulative Failure Rate") +
  scale_colour_manual(values = cbPalette) +
  ggtitle("Duane Plot")
```



```r
# Create a Duane Plot. A Duane Plot is another
# tool to visually assess if the data appears to follow
# a Power NHPP by assessing if the data appears to follow
# a straight line when plotted on a log-log scale.
# If the data followed a horizontal straight line
# that is an indication of a HPP rather than a NHPP.
# Here since I plotted the rocof vs time, you can
# see that it appears as though the rocof is decreasing,
# or in other words the failure rate is decreasing, or
# the time between failures is increasing.
```

```r
ggplot(amsaa2,
  aes(
    x = Time,
    y = mtbf)) +
  scale_x_log10() +
  scale_y_log10() +
  geom_smooth(method='lm', se = FALSE) +
  geom_point() +
  labs(y = "Cumulative MTBF") +
  scale_colour_manual(values = cbPalette) +
  ggtitle("Duane Plot")
```

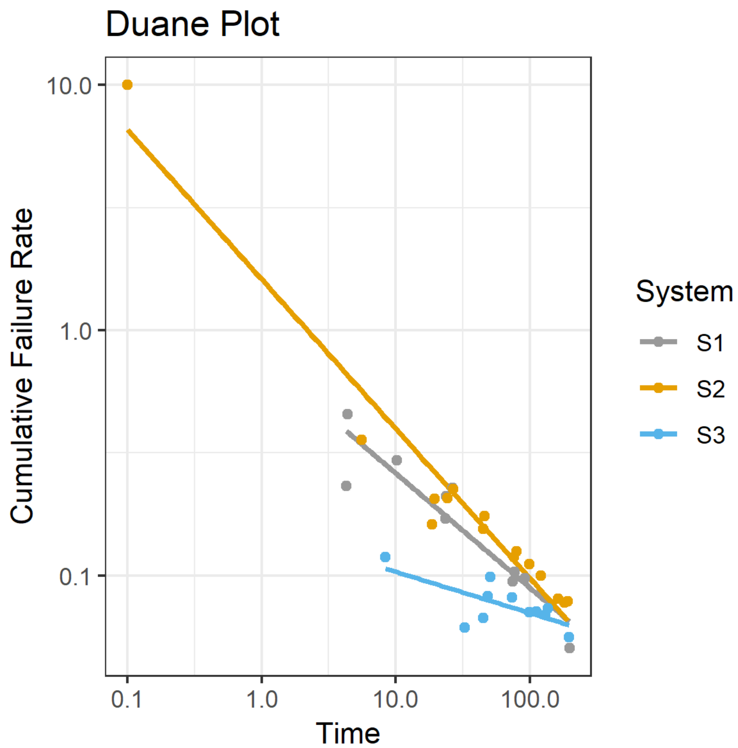



```r
# This is the same Duane Plot, just now using the
# mtbf instead of the rocof. These are equivalent,
# and lead to the same conclusions.
```

```r
t <- split(amsaa$Time, amsaa$System)
df_rocof2 <- rocof(t = t, by = names(t))
head(df_rocof2)
```

```r
##   by    t  mtbf     rocof
## 1 S1  4.3 4.300 0.2325581
## 2 S1  4.4 2.200 0.4545455
## 3 S1 10.2 3.400 0.2941176
## 4 S1 23.5 5.875 0.1702128
## 5 S1 23.8 4.760 0.2100840
## 6 S1 26.4 4.400 0.2272727
names(df_rocof2)[1:2] <- c("System", "Time")
amsaa3 <- merge(amsaa1, df_rocof2, by = c("Time", "System"))
head(amsaa3)
##    Time System Failure        mcf     mtbf       rocof
## 1   0.1     S2       1  0.3333333  0.10000 10.00000000
## 2  10.2     S1       3  2.0000000  3.40000  0.29411765
## 3 112.2     S3       8  9.3333333 14.02500  0.07130125
## 4 120.1     S2      12  9.6666667 10.00833  0.09991674
## 5 129.8     S3       9 10.0000000 14.42222  0.06933744
## 6 136.0     S3      10 10.3333333 13.60000  0.07352941
# The rocof and mtbf estimates we got before were
# based on grouping all systems together.
# Instead I could do it by System, and create
# a Duane plot with the Systems separated to
# see if they all seem to follow a similar pattern.
# This would be a visual assessment of a common beta.
```

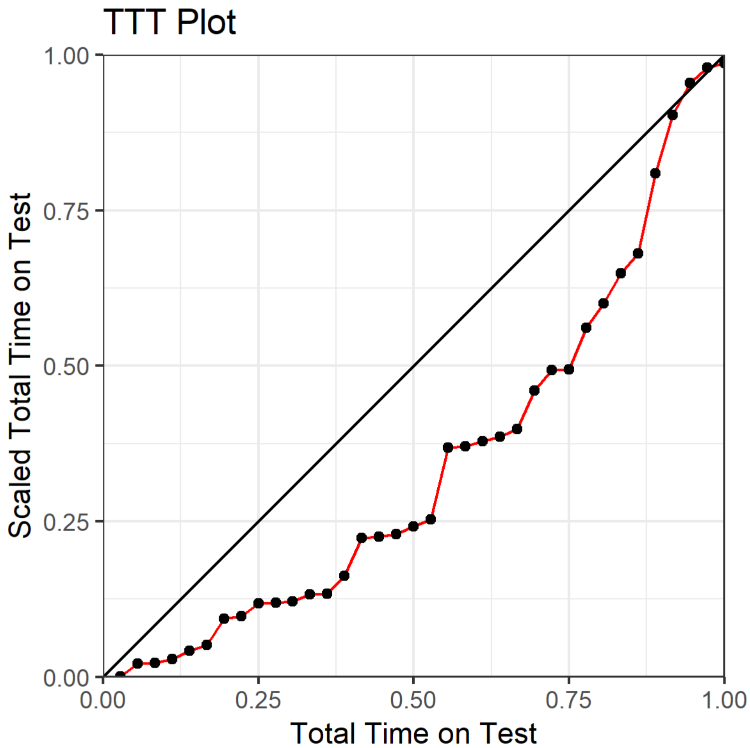
```r
ggplot(amsaa3,
  aes(
    x = Time,
    y = rocof,
    colour = System)) +
  scale_x_log10() +
  scale_y_log10() +
  geom_smooth(method='lm', se = FALSE) +
  geom_point() +
  labs(y = "Cumulative Failure Rate") +
  scale_colour_manual(values = cbPalette) +
  ggtitle("Duane Plot")
```




```r
# This is a Duane Plot by system. We see that
# the slopes are not exactly equal, but they
# are all indicating a decreasing failure rate.
# The common_beta test/function already told us
# that there was no statistical difference between
# the systems' failure rates.
```

```r
df_ttt <- ttt(
  t = split(amsaa2$Time, amsaa2$System),
  T = list(200,200,200))
# This ttt function is for the total time on test (TTT).
# The TTT is used in certain trend tests (the
# Military Handbook Test, and Laplace Centroid Tests
# both have TTT versions, see the trend_test function).
# The TTT can also be used in plots to visually assess
# if the data follow a Power Law Process.
```

```r
ggplot(df_ttt, aes(x = ttt, y = scaled_ttt)) +
   geom_line(colour = "red") + geom_point() +
   geom_abline(intercept = 0, slope = 1) +
   labs(
     x = "Total Time on Test",
     y = "Scaled Total Time on Test") +
   scale_x_continuous(limits = c(0, 1), expand = c(0, 0))  +
   scale_y_continuous(limits = c(0, 1), expand = c(0, 0)) +
   theme(plot.margin = margin(t = 10, r = 10, unit = "pt")) +
   coord_fixed() +
   ggtitle("TTT Plot")
```

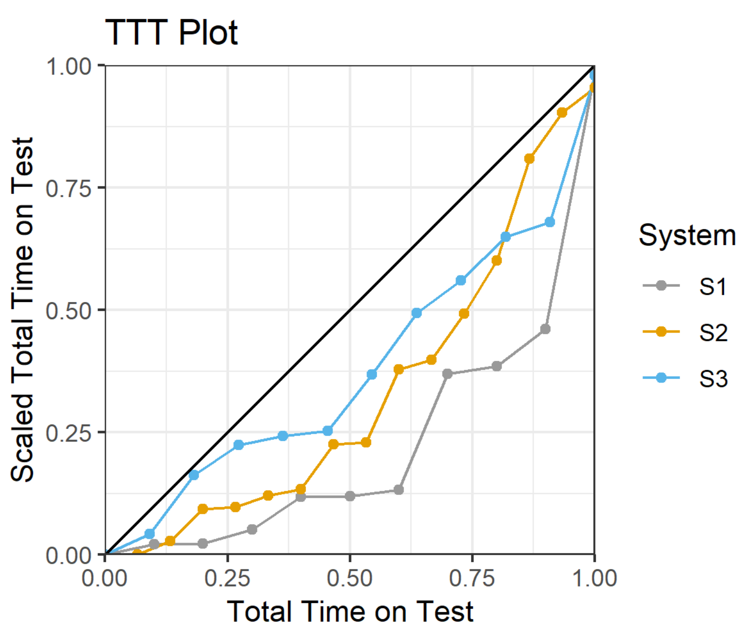



```
# A power-law process is appropriate if the TTT plot lies
# close to the diagonal or is a curve that is either concave
# up or concave down. If there is no pattern, or a curve that
# shifts between being concave up and concave down, the
# power-law process is inadequate. We seem to have a pattern
# with a concave up curve. Concave up indicates decreasing
# failure rate. Concave down would indicate increasing
# failure rate, and no curve (i.e. following the straight line)
# would indicate a HPP.
```

```
t <- split(amsaa2$Time, amsaa2$System)
T <- list(200,200,200)
ttt_by_system <- purrr::map2(t, T, ttt)
ttt_by_system <- Map(rbind, ttt_by_system, data.frame(0,0,0))
df_ttt_by_system <- dplyr::bind_rows(ttt_by_system, .id = "System")
head(df_ttt_by_system)
##   System    t ttt scaled_ttt
## 1     S1  4.3 0.1     0.0215
## 2     S1  4.4 0.2     0.0220
## 3     S1 10.2 0.3     0.0510
## 4     S1 23.5 0.4     0.1175
## 5     S1 23.8 0.5     0.1190
## 6     S1 26.4 0.6     0.1320
names(df_ttt_by_system)[2] <- "Time"
# In the ttt function I did not include a "by"
# arguement that lets you get the TTT per
# system, for example. But, if you install
# the purrr and dplyr packages you can easily get
# the TTT per system as I show above. You could
# also just run the ttt function 3 times by only
```

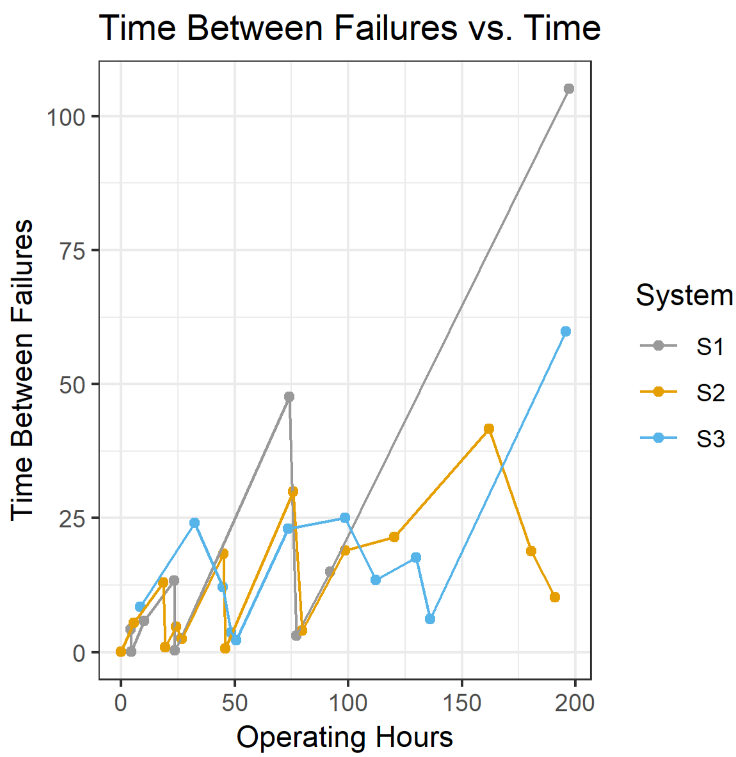
```r
# passing in the t and T for each system separately,
# but if you had 10 systems, 100 systems, etc. you
# probably wouldn't want to do that. You could follow
# this appoach in the other functions too (mcf and rocof),
# but the "by" variable in those was included to
# make it a litter easier. (Note that the Map, step
# I included is really not necessary, but I included
# it so the ttt plot lines would start at the origin.)
```

```r
ggplot(df_ttt_by_system, aes(x = ttt, y = scaled_ttt, colour = System)) +
   geom_line() + geom_point() +
   geom_abline(intercept = 0, slope = 1) +
   labs(
     x = "Total Time on Test",
     y = "Scaled Total Time on Test") +
   scale_x_continuous(limits = c(0, 1), expand = c(0, 0))  +
   scale_y_continuous(limits = c(0, 1), expand = c(0, 0)) +
   theme(plot.margin = margin(t = 10, r = 10, unit = "pt")) +
   coord_fixed() +
   scale_colour_manual(values = cbPalette) +
   ggtitle("TTT Plot")
```

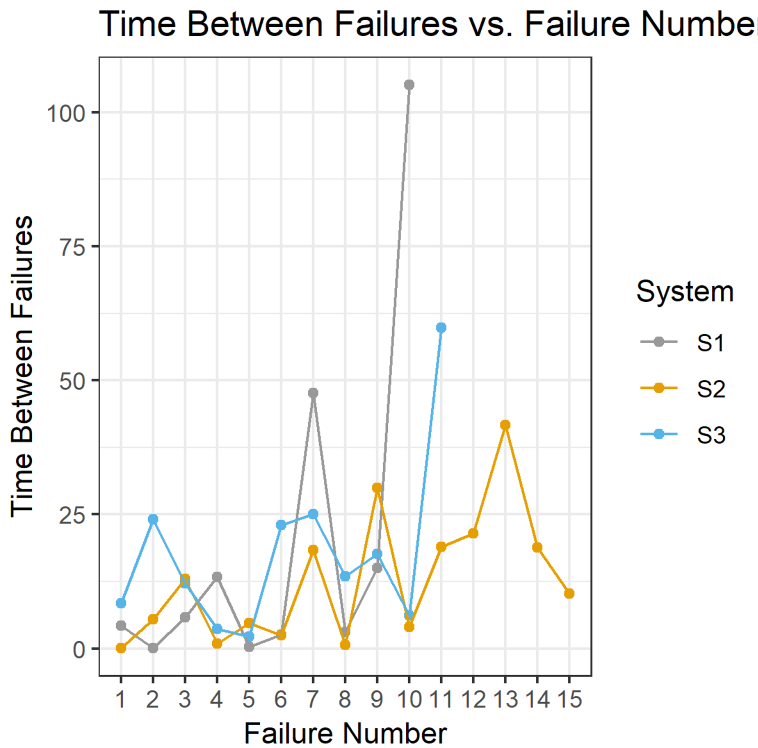



```r
# Plot Time Between Failures vs Time
ggplot(amsaa,
  aes(
    x = Time,
    y = ifelse(diff(c(0,amsaa$Time)) < 0, amsaa$Time, diff(c(0,amsaa$Time))),
    colour = System)) +
  geom_point() + geom_line() +
```

```r
  labs(
    x = "Operating Hours",
    y = "Time Between Failures",
    main = "Time Between Failures vs Time"
  ) + scale_colour_manual(values = cbPalette) +
  ggtitle("Time Between Failures vs. Time")
```




```r
# Plot Time Between Failures vs Failure Number
ggplot(amsaa,
  aes(
    x = Failure,
    y = ifelse(diff(c(0,amsaa$Time)) < 0, amsaa$Time, diff(c(0,amsaa$Time))),
    colour = System)) +
  geom_point() + geom_line() +
  labs(
    x = "Failure Number",
    y = "Time Between Failures",
    main = "Time Between Failures vs Time"
  ) + scale_colour_manual(values = cbPalette) +
  ggtitle("Time Between Failures vs. Failure Number") +
  scale_x_continuous(breaks = 1:15) +
  theme(panel.grid.minor.x = element_blank())
```


Time Between Failures vs. Failure Number
Time Between Failures
100
75
50
25
0
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
Failure Number
System
S1
S2
S3