Reliability Modeling Toolkit (RMT): Development, Testing, and Tutorial

Reuel Calvin Smith, PhD

Mohammad Modarres, PhD

Center for Risk and Reliability, University of Maryland

RAMS 2024, The 70th Annual Reliability & Maintainability Symposium, Jan 22-25, 2024, Albuquerque, NM

RAMS® 2024

CENTER FOR RISK AND RELIABILITY

Objective of Tutorial

- To demonstrate this toolkit's functions including example problems
- Help graduate-level students and practitioners to apply this toolkit for analyzing physics of failure and accelerated testing problems

Tutorial Outline

- Overview
  - History of the Reliability Modeling Toolkit (RMT)
  - Features and Installation Instructions
- Toolsets and Examples:
  - Physics of Failure and Failure Mechanism
  - Reliability Modeling
  - Accelerated Testing
- RMT Application, Modification, and Testing

What is the Reliability Modeling Toolkit?

- An R library made up of computer scripts
- Solves computationally involved and challenging reliability modeling and analytics including:
  - Probabilistic physics of failure (PPoF)
  - Life probability distribution analysis
  - Accelerated testing: Accelerated life testing (ALT) and accelerated degradation testing (ADT) data analysis
- Free of charge, and is governed by a permissive open-source use license for students, practitioners, and the public

Toolkit Background

- RMT began as a collection of MATLAB, R, and BUGs scripts in 2015

- R was ultimately chosen as RMT's platform for,
  - Open access to the public
  - High use in the engineering industry
  - Extensive function libraries and catalogs
  - Large data analysis and repeated calculations

Toolkit Background (Cont.)

- First released to UMD students in August 2022 as part of reliability engineering course **Probabilistic Physics of Failure and Accelerated Testing**
- Sought testing and feedback
- Feedbacks assisted improvement and further development of RMT tools, course materials*
- Most recent version of RMT currently available for download on GitHub

*M. Modarres, M. Amiri, C. Jackson. *Probabilistic Physics of Failure Approach to Reliability: Modeling, Accelerated Testing, Prognosis and Reliability Assessment*, Maryland, University of Maryland, 2017

RMT Feature Comparisons

RMT

- Free under License GPLv3
- Supports Least Squares, MLE, and Bayesian Estimation
- Twelve Life-Stress models
- Ten Life Distribution models
- ALT, Step-Stress ALT, and ADT
- Data visualization

Minitab

- Available for purchase
- Supports Least Squares and MLE
- Four Life-Stress models
- Five Life distribution models
- ALT
- Data visualization

JMP-SAS

- Available for purchase
- Supports Least Squares, MLE, and Bayesian Estimation
- Multiple Life-Stress models
- Multiple Life distribution models
- Extensive data visualization

Weibull++

- Available for purchase
- Supports Least Squares, MLE, and Bayesian Estimation
- Nine Life-Stress models
- Eleven Life distribution models
- ALT, Step-Stress ALT, ADT, and Step-Stress ADT
- Data visualization

Installation Instructions

1. Download and install latest version of R (https://www.r-project.org/).

2. RStudio recommended as an additional download for interface reasons (https://posit.co/download/rstudio-desktop/). If installing on a Mac, use Anaconda Navigator to install RStudio.

3. Setup R and RStudio then install the “devtools” library

```
install.packages("devtools")
library(devtools)
```

Installation Instructions

4. Install “cmdstanr”

```
install.packages("cmdstanr", repos = c("https://mc-stan.org/r-packages/", getOption("repos")))
```

5. Install “RMT”

   - If Rtools is installed, type the following to build from source

   ```
   devtools::install_github("Center-for-Risk-and-Reliability/RMT", INSTALL_opts = "--install-tests")
   ```

   - If Rtools is not installed, type the following instead

   ```
   devtools::install_github("Center-for-Risk-and-Reliability/RMT", build = FALSE, INSTALL_opts = "--install-tests")
   ```

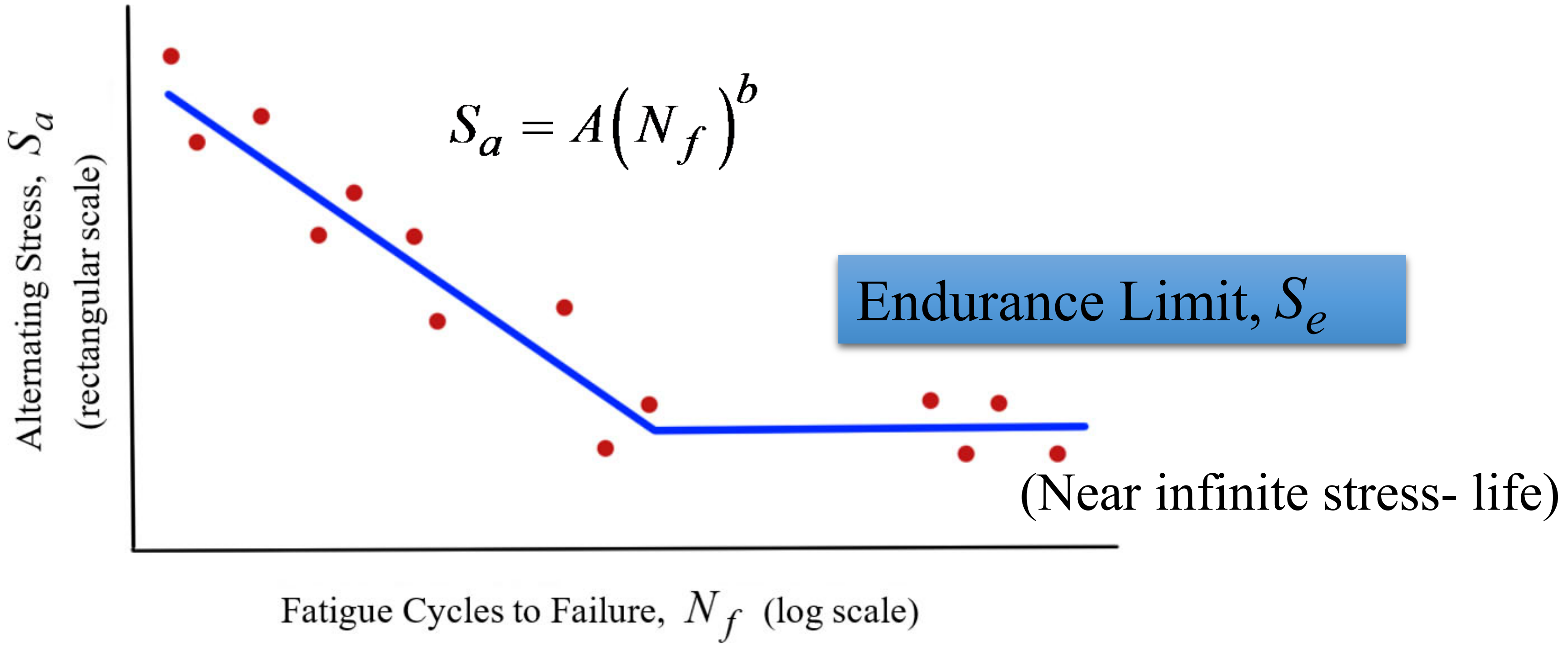
6. Load RMT library in R

```
library(reliabilityRMT)
```

Toolset 1:

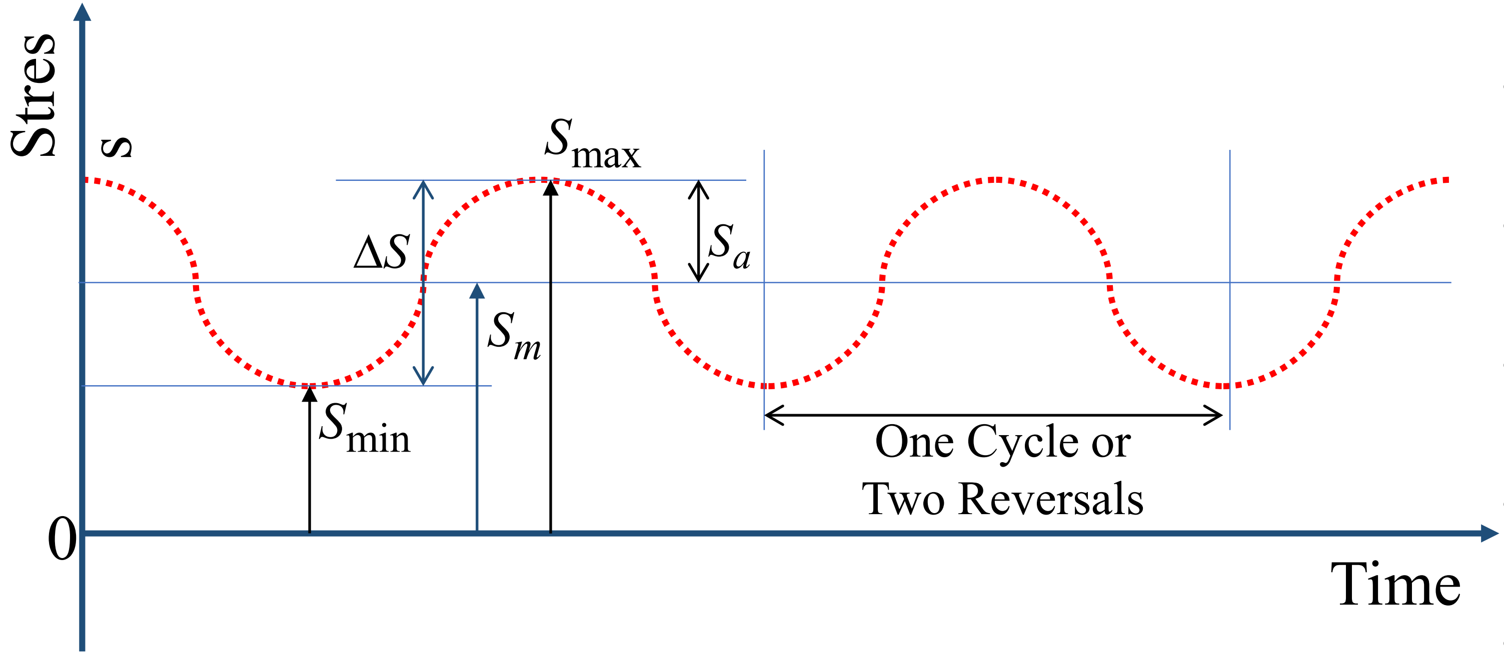
MECHANICS OF FAILURE

Stress-Life: The Ideal S-N Diagram



Stress-Life: The Ideal S-N Diagram

Standard Cyclic Loading Definitions:



- Min Stress – S_{min}
- Max Stress – S_{max}
- Mean Stress – $S_m = \dfrac{S_{max} + S_{min}}{2}$
- Stress Range – $\Delta S = S_{max} - S_{min}$
- Alternating Stress – $S_a = \dfrac{\Delta S}{2}$
- Load Ratio – $R = \dfrac{S_{min}}{S_{max}}$

Stress-Life: S-N Diagram Tools

S-N Diagram Tool

```
SN.diagram(input_type,data,stressunits,options)
```

- "*input_type*" – (1) fully reversed stress/life data points, (2) uniaxial stress range, (3) multiaxial stress range
- "*data*" – input based on "input_type"
- "*stressunits*" – (1) SI stress "MPa" and (2) English stress "ksi"
- "*options*" – Material properties (ultimate strength, BHN, etc.), trace stress or cycles, mean stress relation, model parameters *A* and *b*

Stress-Life: S-N Diagram Tools

Example: Consider the following fully reversed ($R = -1$) stress/life data

Stress Amplitude, S_a (MPa)	Fully Reversed Cycles, N_f	Stress Amplitude, S_a (MPa)	Fully Reversed Cycles, N_f	Stress Amplitude, S_a (MPa)	Fully Reversed Cycles, N_f
340	15×10^3	250	301×10^3	210*	$>10^7$
300	24×10^3	235	290×10^3	210*	$>10^7$
290	36×10^3	230	361×10^3	205*	$>10^7$
275	80×10^3	220	881×10^3	205*	$>10^7$
260	177×10^3	215	1.3×10^6	205*	$>10^7$
255	162×10^3	210	2.5×10^6		

- "*" denotes "runoff" data

Stress-Life: S-N Diagram Tools

- "*input_type*"=1 enter data as LIST

$$data \;<\text{-}\; list\begin{pmatrix} .\text{Fail Stress}, \\ .\text{Fail Cycles}, \\ .\text{Runout Stress}, \\ .\text{Runout Cycles} \end{pmatrix}$$

Define:

```
> datSN1 <- list(
      c(340,300,290,275,260,255,250,235,230,220,215,210),
      c(15e3,24e3,36e3,80e3,177e3,162e3,301e3,290e3,361e3,881e3,1.3e6,2.5e6),
      c(210,210,205,205,205),
      c(1e7,1e7,1e7,1e7,1e7))
```

<u>NOTE</u>: The RMT typically takes input data in list form

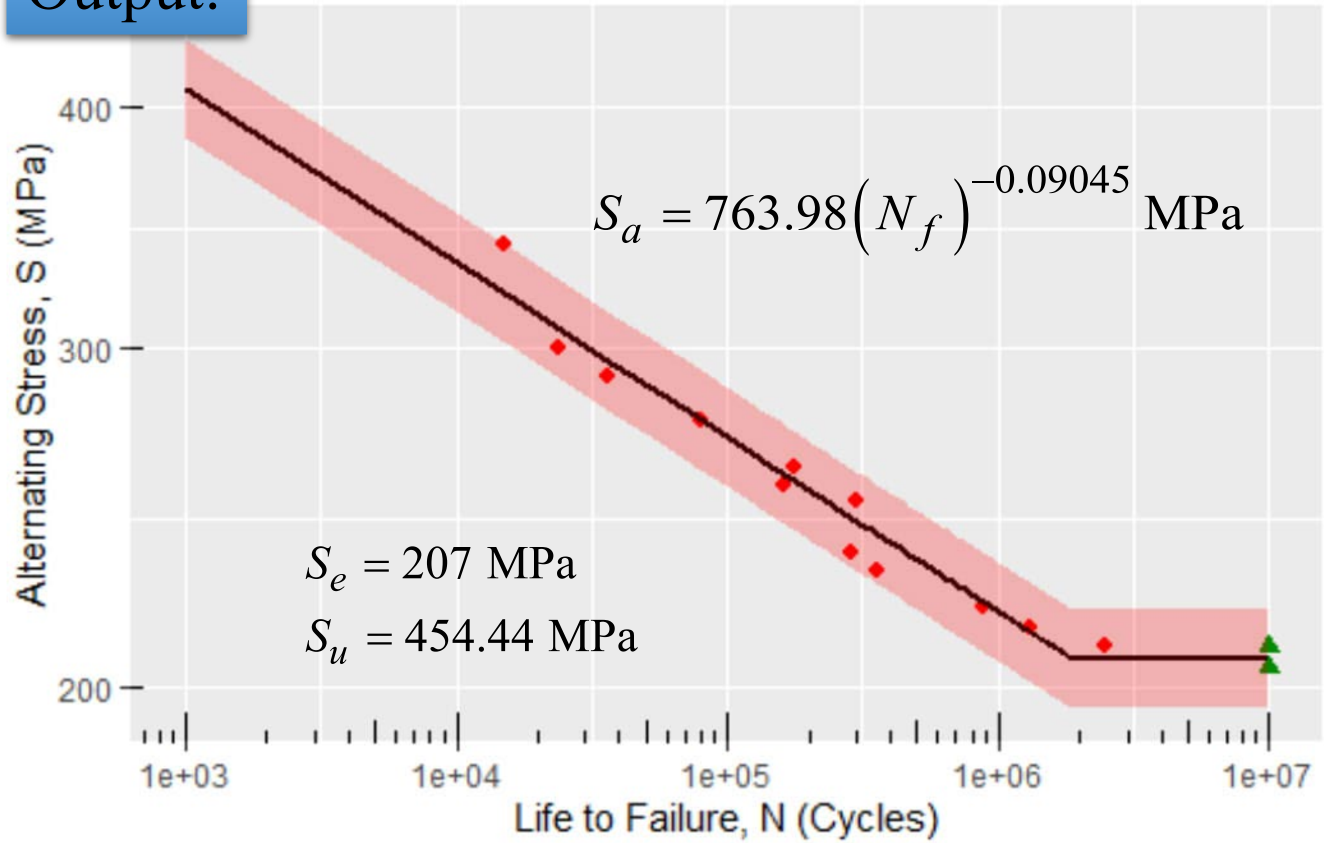
Stress-Life: S-N Diagram Tools

Input:

```
SN.diagram(1,datSN1,1)
```

- "*options*" input may be used as needed. For example…

Output:



$$S_a = 763.98\left(N_f\right)^{-0.09045}\ \text{MPa}$$

$$S_e = 207\ \text{MPa}$$

$$S_u = 454.44\ \text{MPa}$$

Stress-Life: S-N Diagram Tools

- Use "options" to trace a stress or cycle point of interest

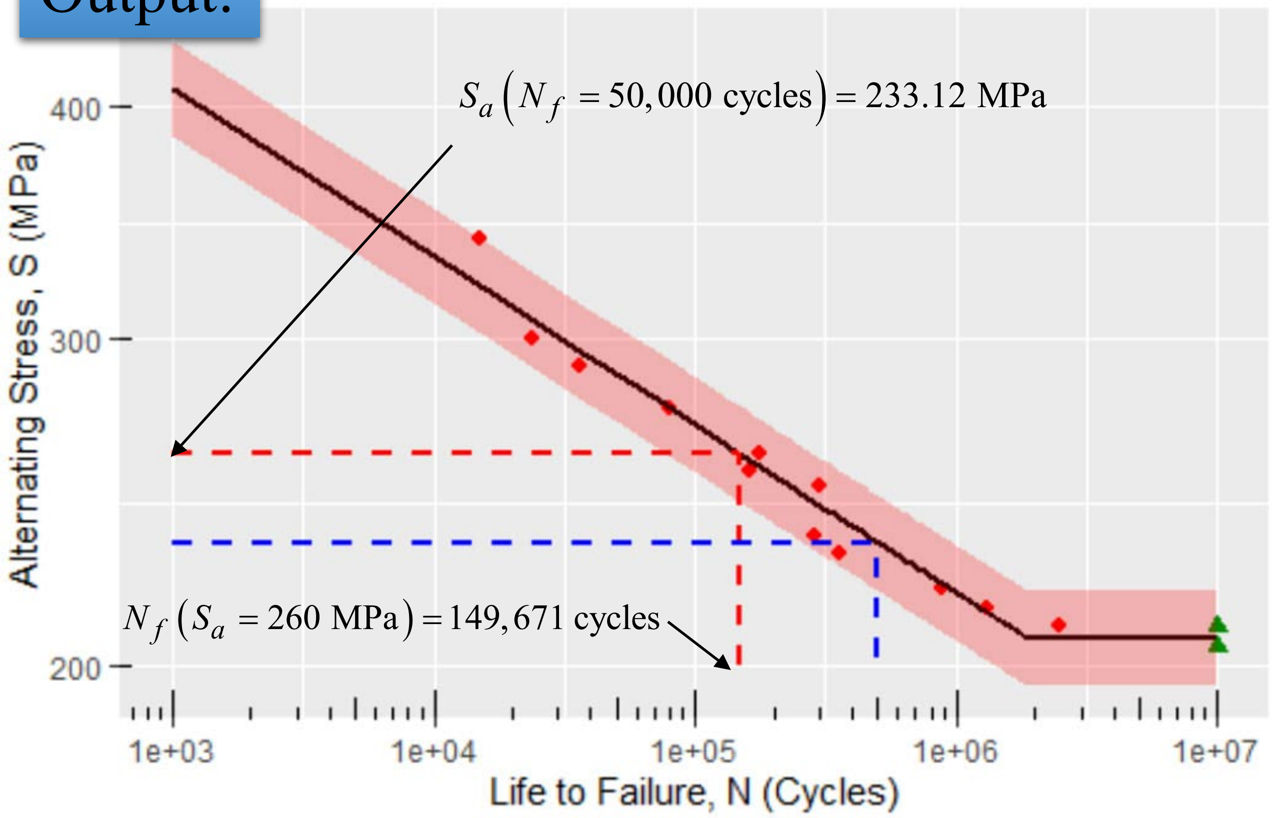
$$S_a\left(N_f = 50{,}000 \text{ cycles}\right) = ?$$

$$N_f\left(S_a = 260 \text{ MPa}\right) = ?$$

Input:

```
SN.diagram(1,datSN1,1,
           list(Ntrace = 5e5,
                Strace = 260))
```

Output:



Strain-Life: Stress-Strain Relationship

- Ramberg-Osgood Equation

$$\varepsilon_{tot} = \underbrace{\frac{\sigma}{E}}_{\text{elastic strain}} + \underbrace{\left(\frac{\sigma}{K'}\right)^{\frac{1}{n'}}}_{\text{plastic strain}} \qquad \Delta\varepsilon_{tot} = \underbrace{\frac{\Delta\sigma}{E}}_{\text{elastic strain}} + \underbrace{2\left(\frac{\Delta\sigma}{2K'}\right)^{\frac{1}{n'}}}_{\text{plastic strain}}$$

- Models stress (σ)/strain (ε) relationship where:
  - E – Modulus of elasticity
  - K' – Cyclic strength coefficient
  - n' – Cyclic hardening component

$\left.\begin{array}{l} K' \\ n' \end{array}\right\}$ *Parameters of interest*

Strain-Life: Strain-Life Relationship

- Coffin-Manson Relationship

$$\varepsilon_{tot} = \underbrace{\frac{\sigma'_f}{E}\left(2N_f\right)^b}_{\text{elastic strain}} + \underbrace{\varepsilon'_f\left(2N_f\right)^c}_{\text{plastic strain}}$$

- Models strain-life relationship where,
  - σ'_f – Fatigue strength coefficient
  - b – Fatigue strength exponent
  - ε'_f – Fatigue ductility coefficient
  - c – Fatigue ductility exponent

Also *__parameters of interest__*

Strain-Life: Relationship and Hysteresis Loop Tools

Stress-Strain Parameters Tool

```
stress_strain.params(dat, E, stressunits, options)
```

- "*dat*" – List of stress, strain, and fatigue cycles (*in that order*)
- "*E*" – Modulus of elasticity
- "*stressunits*" – (1) SI stress "MPa" and (2) English stress "ksi"
- "*options*" – Stress-strain relationship (*other* than Coffin-Manson the default), strain or life trace, loading conditions, and/or mean stress relation

Strain-Life: Relationship and Hysteresis Loop Tools

Example: Consider this stress-strain data from fully reversed fatigue test on steel alloy where $E = 200$ GPa.

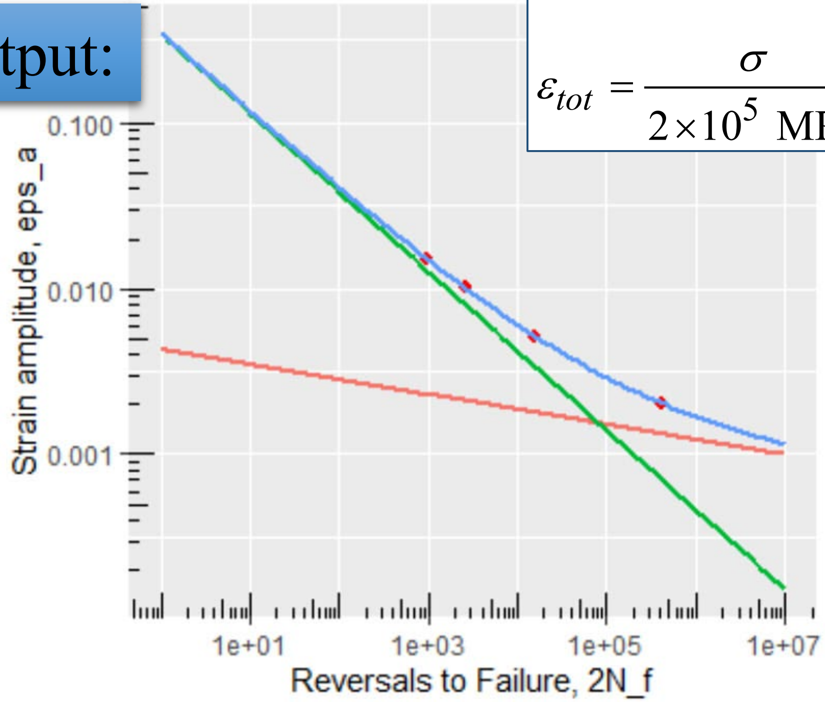
Strain Amplitude, ε_a	Stress Amplitude, S_a (MPa)	Fully Reversed Cycles, N_f
0.00202	261	208,357
0.0051	372	7,947
0.0102	428	1,335
0.0151	444	495

Strain-Life: Relationship and Hysteresis Loop Tools

Input:

```
stress_strain.params(list(c(261, 372, 428, 444),
                          c(0.00202, 0.0051, 0.0102, 0.0151),
                          c(208357,7947,1335,495)),E=200000,1)
```

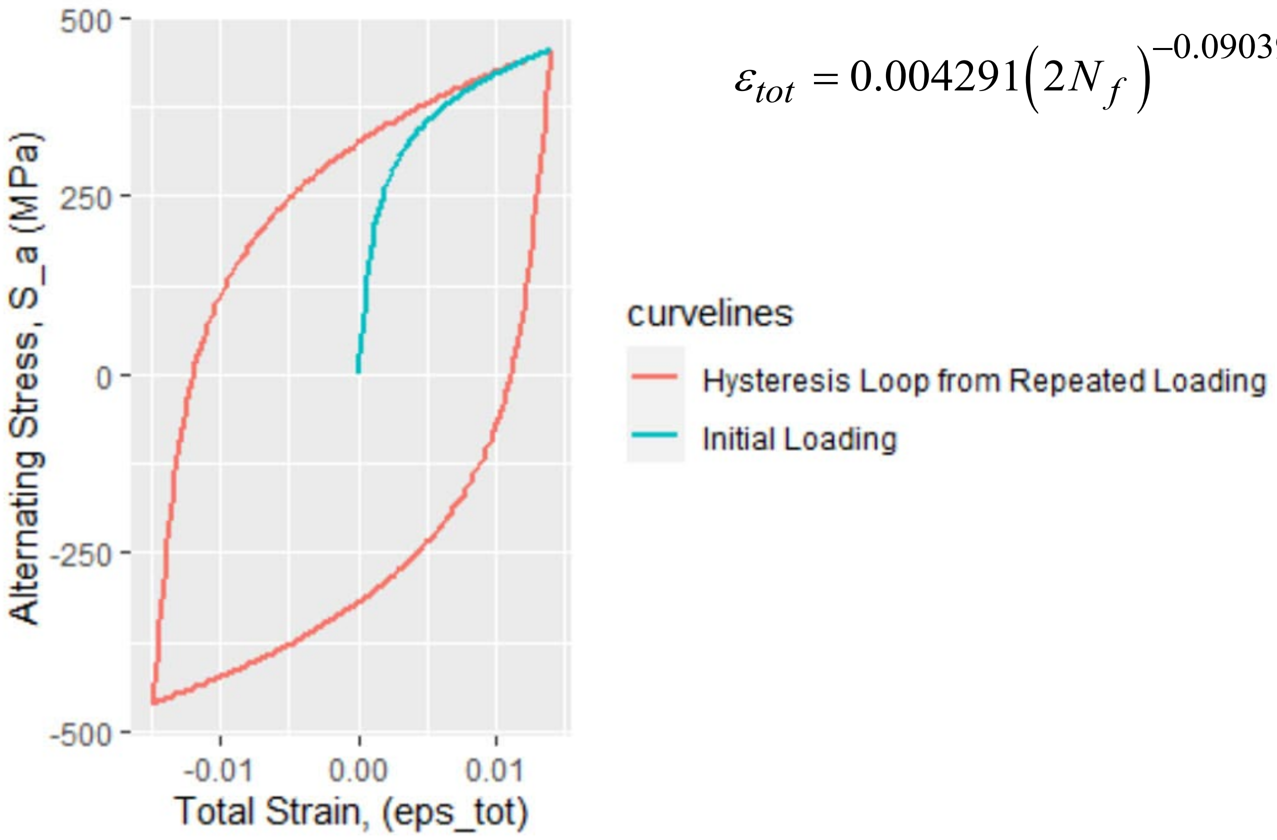
Output:



$$\varepsilon_{tot} = \frac{\sigma}{2\times10^{5}\ \text{MPa}} + \left(\frac{\sigma}{1070.8\ \text{MPa}}\right)^{\frac{1}{0.1926}}$$

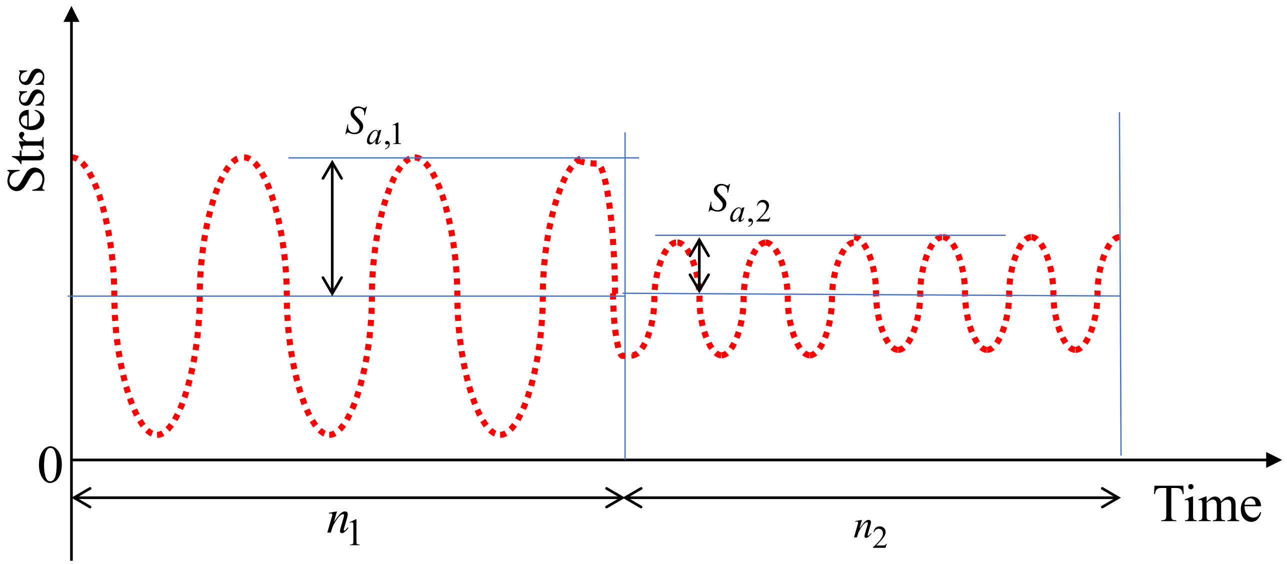
Strain-Life: Relationship and Hysteresis Loop Tools

Output:



$$\varepsilon_{tot} = 0.004291\left(2N_f\right)^{-0.09039} + 0.3493\left(2N_f\right)^{-0.4797}$$

Variable Amplitude Loading: Block Loading Damage Models



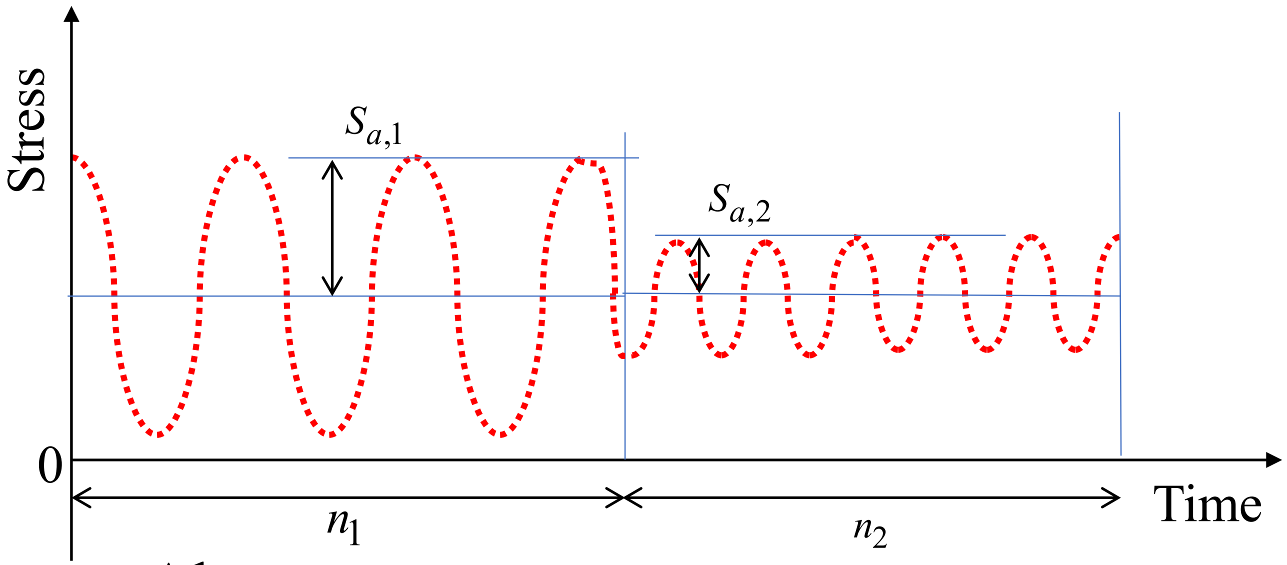
Palmgren-Miner's Linear Damage Rule

$$B_f \left(\sum_{i=1} \frac{n_i}{N_{f,i}} \right)_{\text{one repetition}} = 1$$

$$D_i = B_f \left(\frac{n_i}{N_{f,i}} \right)$$

- Models linear damage based on block loading based fatigue where B_f is the total loading blocks to failure

Variable Amplitude Loading: Block Loading Damage Models



Palmgren-Miner's Linear Damage Rule

$$B_f \left(\sum_{i=1} \frac{n_i}{N_{f,i}} \right)_{\text{one repetition}} = 1$$

$$D_i = B_f \left(\frac{n_i}{N_{f,i}} \right)$$

- Also:
  - n_i – number of operation cycles at a given stress level S_i
  - $N_{f,i}$ – cycles to failure at a constant amplitude S_i (based on S-N diagram)
  - D_i - proportion of damage of the loading block i

Variable Amplitude Loading: Block Loading Tools

Variable Amplitude Loading Modeling with Damage Tool

```
var.amp.loadingdamage.model(dat, damagerule, stressunits)
```

- "*dat*" – List consisting of either,
  - n_i and $N_{f,i}$
  - D_i and $N_{f,i}$ or
  - n_i, $\Delta S_{,i}$, material properties σ'_f and b, and mean stress relation

- "*damagerule*" – Damage model (*Palmgren-Miner's Rule is default*)

- "*stressunits*" – (1) SI stress "MPa" and (2) English stress "ksi"

Variable Amplitude Loading: Block Loading Tools

Example: Take a specimen where is 220 ksi and b is – 0.1 with a stress block plan of,

Block, i	Operation Cycles, N_f	Min Stress, $S_{\min}$ (ksi)	Max Stress, $S_{\max}$ (ksi)
1	200	-80	80
2	1,000	0	100
3	100	-100	0

Since two of the three blocks are not fully reversed we would apply a mean stress correction (**Morrow's mean stress relation** $\frac{S_a}{S_{ar}} + \frac{S_m}{\sigma'_f} = 1$) to find the equivalent stress amplitude S_{ar}.

Variable Amplitude Loading: Block Loading Tools

Input:

```
var.amp.loadingdamage.model(list(ni = c(200,1000,100),
                            sranges = list(c(-80,80),c(0,100),c(-100,0)),
                            sig_f = 220, b = -0.1, corr_rel = "Morrow"),"Miner",2)
```

Output:

```
$damagebyblock
[1] 0.6251178802 0.3745154111 0.0003667087

$cyclestofailurebyblock
[1]    12367.93   103218.89 10541625.96

$reversalstofailurebyblock
[1]    24735.86   206437.78 21083251.93

$repetitionstofailure
[1] 38.65706
```

Block, i	Damage, D_i	Reversals to Failure, $2N_f$
1	0.625	24,735.86
2	0.375	206,437.78
3	0.000367	21,083,251.93

38.6 loading blocks to failure

Variable Amplitude Loading: Block Loading Tools

- Accounts for non-linear damage models as well. For example…

Kwofie-Rahbar Nonlinear Damage Model

$$B_f\left(\sum_{i=1} \frac{n_i}{N_{f,i}} \frac{\ln\left(N_{f,i}\right)}{\ln\left(N_{f,1}\right)}\right)_{\text{one repetition}} = 1$$

$$D_i = B_f\left[\frac{n_i}{N_{f,i}} \frac{\ln\left(N_{f,i}\right)}{\ln\left(N_{f,1}\right)}\right]$$

Input:

```
var.amp.loadingdamage.model(list(ni = c(200,1000,100),
                          sranges = list(c(-80,80),c(0,100),c(-100,0)),
                          sig_f = 220, b = -0.1, corr_rel = "Morrow"),"KwofieRahbar",2)
```

Output:

```
$damagebyblock
[1] 0.5763621277 0.4230576364 0.0005802359

$cyclestofailurebyblock
[1]    12367.93   103218.89 10541625.96

$reversalstofailurebyblock
[1]    24735.86   206437.78 21083251.93

$repetitionstofailure
[1] 35.64203
```

Block, i	Damage, D_i	Reversals to Failure, $2N_f$
1	0.576	24,735.86
2	0.423	206,437.78
3	0.000580	21,083,251.93

35.6 loading blocks to failure

Wear: Overview and Types

- **<u>Definition</u>**: The failure mechanism **wear** W is defined as a volumetric measure of damage produced by repeated contact between two solid surfaces. The material lost is the measure of that damage.

- Can be categorized as follows:
  - Sliding
  - Abrasive
  - Impact
  - Rolling
  - Lubricated
  - Corrosive
  - Thermal

- *Sliding wear is most common*

Wear: Overview and Types

- **<u>Definition</u>: Sliding wear** W is based on the Archard wear equation where we apply factors of contact force P, sliding distance L, and material hardness H.

$$W = k\frac{PL}{H} = Ah$$

  - k – wear coefficient
  - A – constant area of contact between surfaces
  - h – wear depth

- **<u>Definition</u>: Sliding wear velocity or rate** $\dot{W}$ is based on applying sliding velocity $V = L/t$ where t is the time elapsed for wear and,

$$\dot{W} = \frac{W}{t} = k\frac{PV}{H}$$

Wear: Sliding Wear Computation Tools

Sliding Wear Calculator Tool

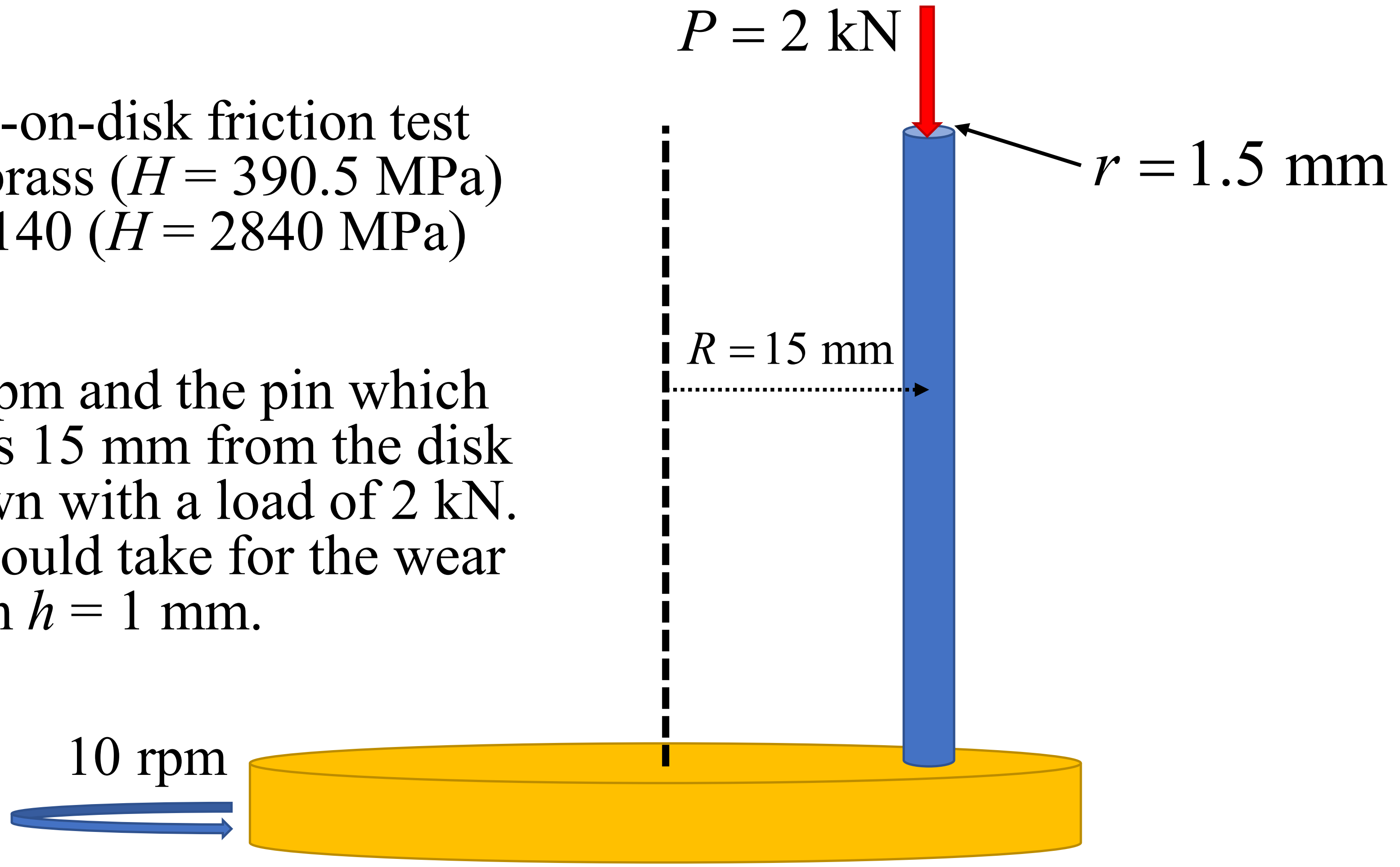
```
wear.sliding(dat, matproperties, units)
```

- "*dat*" – List consisting of area, load, sliding distance, and/or sliding velocity
- "*matproperties*" – wear coefficient *k*, names of contact materials, or the minimum Hardness in scenario
- "*units*" – (1) SI and (2) English

Wear: Sliding Wear Computation Tools

Example: Define a pin-on-disk friction test where the pin is 70-30 brass ($H = 390.5$ MPa) and the disk is Steel 4140 ($H = 2840$ MPa)

The disk rotates at 10 rpm and the pin which has a radius of 1.5 mm is 15 mm from the disk center being pressed down with a load of 2 kN. Let's find how long it would take for the wear depth to reach $h = 1$ mm.



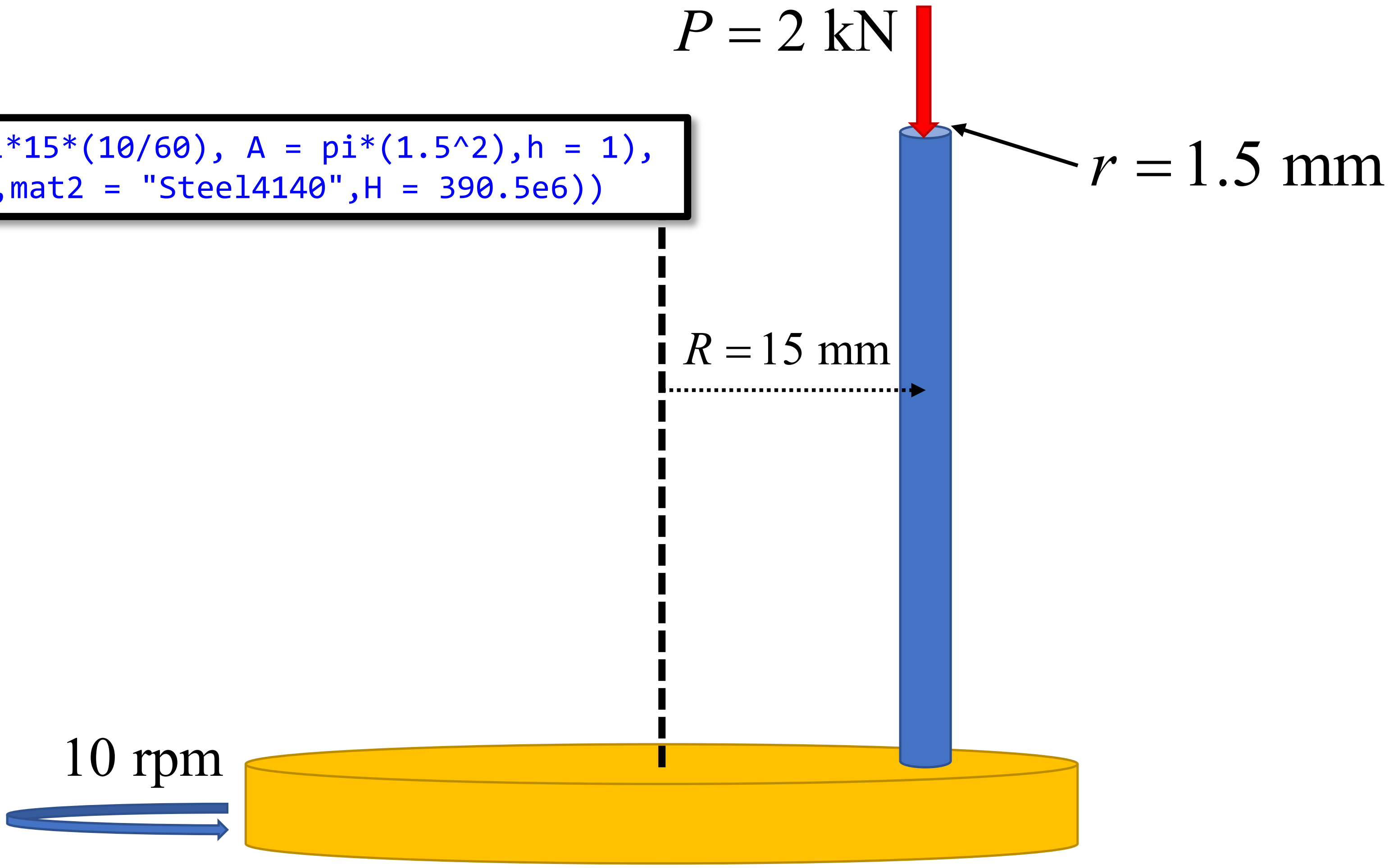
Wear: Sliding Wear Computation Tools

Input:

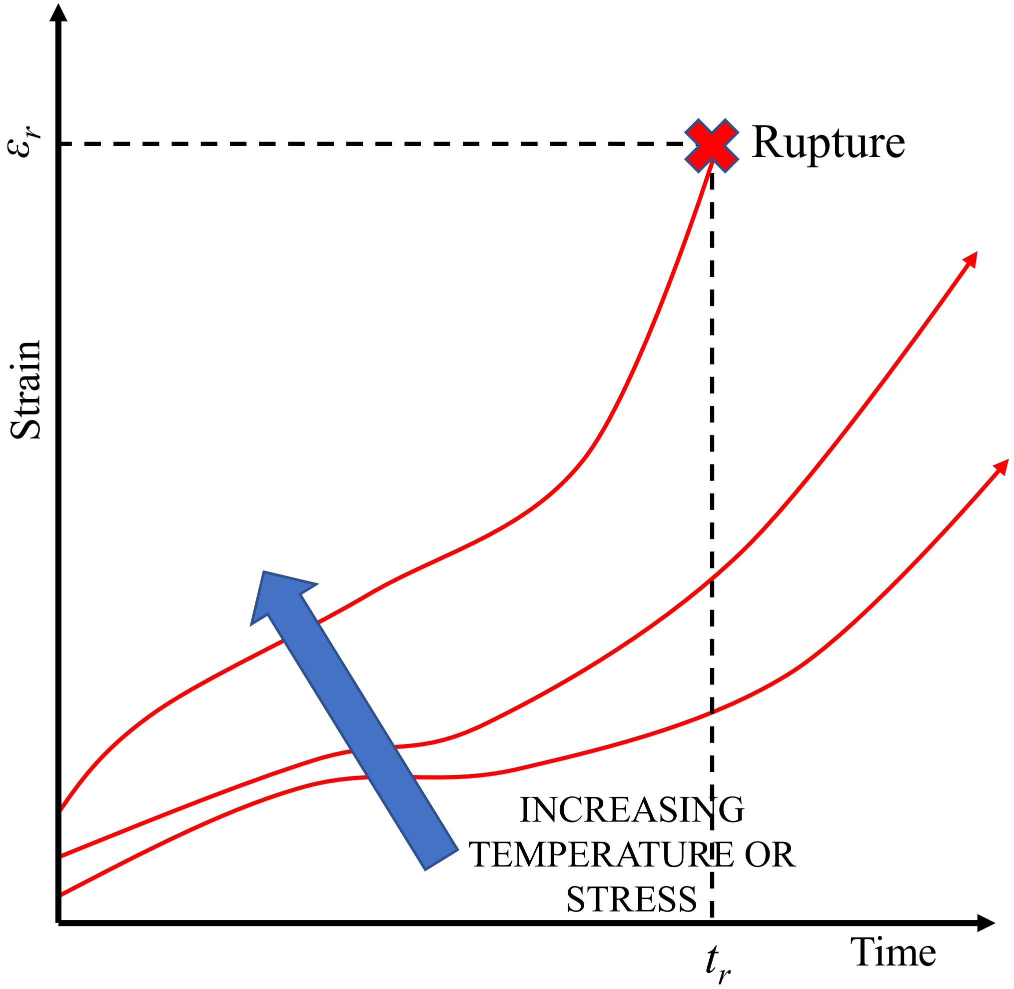
```
wear.sliding(list(P = 2000, V = 2*pi*15*(10/60), A = pi*(1.5^2),h = 1),
             list(mat1="Brass70_30",mat2 = "Steel4140",H = 390.5e6))
```

Output:

$t = 204.33$ seconds
$W = 7.06$ mm^3
$\dot{W} = 0.0346$ mm^3/second



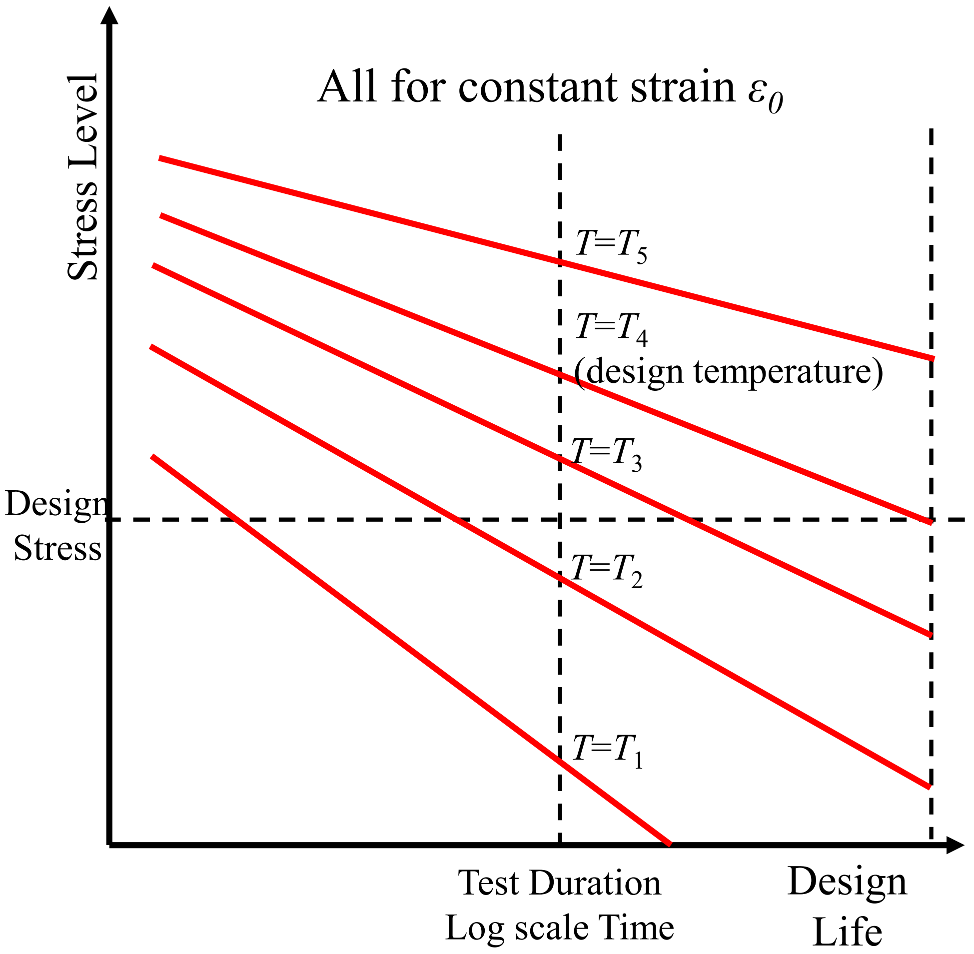
Creep: Overview of Creep



- **<u>Definition</u>**: **Creep** is the accumulation of plastic strain at a given stress at elevated level for a period of time

- Creep induced failure occurs when creep strain produces a deformation that is beyond component's endurance limit

Creep: Overview of Creep



- Creep test data obtained by accelerating temperatures past use level at various constant stress levels
- Data then fit to a given creep model to approximate use life
- Models relationship between temperature T and rupture time t.

Creep: Creep Models

Larson-Miller Creep Model

$$P = \begin{cases} (T + 273.15)(\log_{10} t + C) & \text{S.I.} \\ (T + 459.67)(\log_{10} t + C) & \text{English} \end{cases}$$

- P – Larson-Miller parameter
- T – temperature in Celsius (S.I.) or Fahrenheit (English)
- C – material constant

Creep: Creep Models

Manson-Haferd Creep Model

$$P = \frac{T - T_a}{\log_{10} t - \log_{10} t_a}$$

- P – Manson-Haferd parameter
- T – temperature in Celsius (S.I.) or Fahrenheit (English)
- T_a – material constant in Celsius (S.I.) or Fahrenheit (English)
- $\log_{10} t_a$ – material constant

Creep: Creep Models

Sherby-Dorn Creep Model

$$P = \log_{10} t - 0.43 \frac{E_a}{k_B T} = \log_{10} t - 0.43 \frac{Q}{RT}$$

- P – Sherby-Dorn parameter
- E_a – Activation energy (eV)
- Q – Activation energy (J/mol)
- k_B – Boltzmann constant (8.617×10^{-5} eV/K)
- R – Universal gas constant (8.314 J/mol-K)
- T – temperature in Kelvin

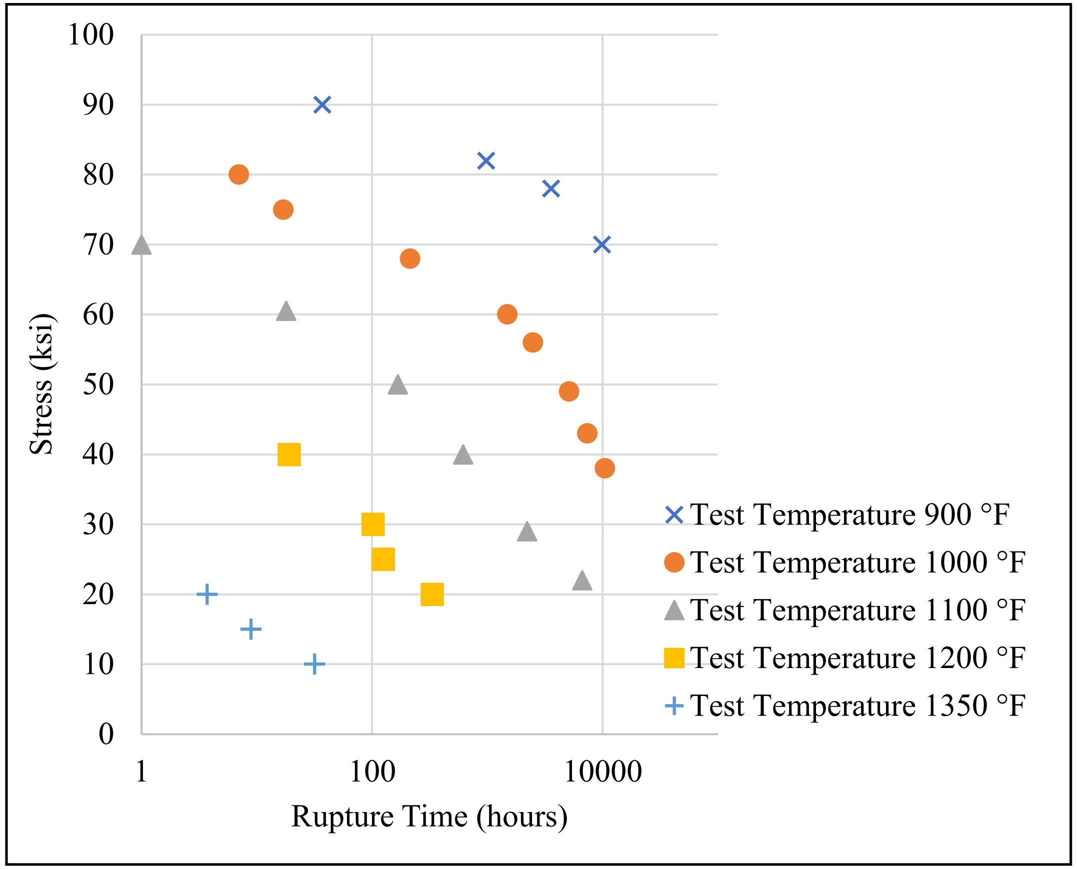
Creep: Creep Modeling Tools

Creep Analyzer Tool

```
creep.analysis(dat, model, creepproperties, stressunits)
```

- "*dat*" – Creep test data in tabular form
- "*model*" – (1) Larson-Miller, (2) Manson-Haferd, (3) Sherby-Dorn
- "*creepproperties*" – material properties based on the creep model, reference temperature, and stress trace
- "*stressunits*" – (1) SI "Celsius" and (2) English "Fahrenheit"

Creep: Creep Modeling Tools



Example: This is data from a creep rupture test series on Cr-Mo-V. Given the following,

Larson-Miller constant	C = 22
Manson-Haferd constants	T_a = 100.13 °F, $\log_{10}t_a$ = 18
Activation energy	Q = 4.6 × 10^5 J/mol

let's find the *Larson-Miller*, *Manson-Haferd*, and *Sherby-Dorn* parameters for test parameters 900 °F at 70 ksi. Then find the rupture time if the temperature is held at 1100 °F at 70 ksi.

Creep: Creep Modeling Tools

Test Temperature (F)	Stress (ksi)	Rupture Time (hrs)
900	90	37
900	82	975
900	78	3581
900	70	9878
1000	80	7
1000	75	17
1000	68	213
1000	60	1493
1000	56	2491
1000	49	5108
1000	43	7390
1000	38	10447
1100	70	1
1100	60.5	18
1100	50	167
1100	40	615
1100	29	2220
1100	22	6637
1200	40	19
1200	30	102
1200	25	125
1200	20	331
1350	20	3.7
1350	15	8.9
1350	10	31.8

CreepData1

- **NOTE**: There are many cases in using the RMT where you would want to read an Excel CSV to enter data rather than form it within R

Define:

```
datCreep1 <- read.csv("https://raw.githubusercontent.com/Center-for-Risk-and-Reliability/RMT/main/CSVExampleData/CreepData1.csv")
```

Creep: Creep Modeling Tools

Input #1:

- Larson-Miller

```
creep.analysis(datCreep1,1,list(C = 22, Strace = 70, Tref = 900),2)
```

Input #2:

- Manson-Haferd

```
creep.analysis(datCreep1,2,list(Strace = 70, Tref = 900,temp_a = 100.13, log10t_a = 18),2)
```

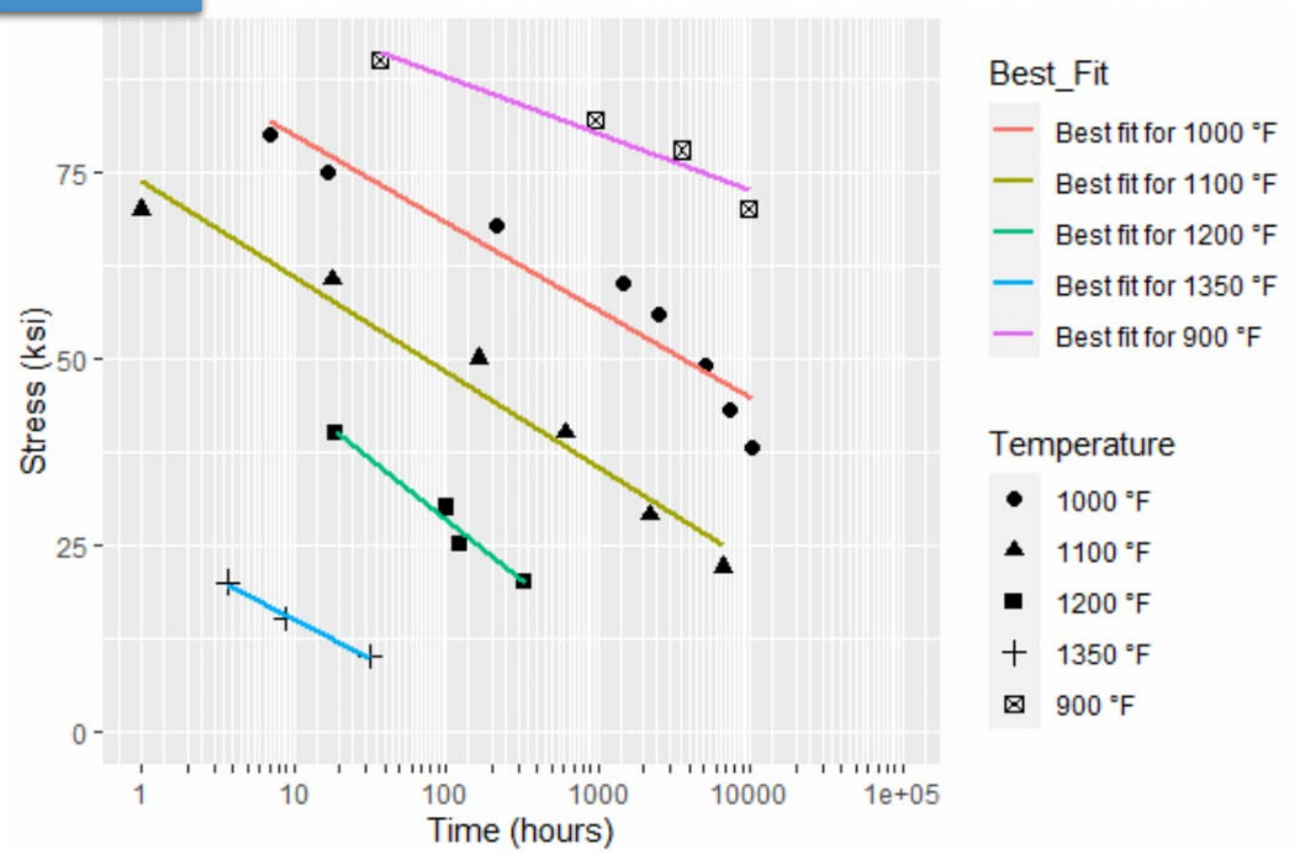
Input #3:

- Sherby-Dorn

```
creep.analysis(datCreep1,3,list(Strace = 70, Tref = 900,Q = 460000),2)
```

Creep: Creep Modeling Tools

Output:



Creep Model	Model Parameter, P	Rupture time, t (hours)
Larson-Miller	35,344.17	4.58
Manson-Haferd	-57.11	3.11
Sherby-Dorn	-27.5	0.9

Remember, this is rupture time if the temperature is held at 1100 °F at 70 ksi

Corrosion: Overview and Types

- **<u>Definition</u>**: **Corrosion** is a material degradation due to chemical or electrochemical reaction to the material's environment

- Can be categorized as follows:
  - Uniform/general corrosion
  - Galvanic corrosion
  - Crevice corrosion
  - Pitting corrosion
  - Environmentally assisted cracking
  - Hydrogen damage
  - Intergranular corrosion
  - Dealloying
  - Erosion corrosion

- Will focus on *pitting corrosion* for our next example

Corrosion: Pitting Corrosion Definitions

- **<u>Definition</u>**: **Pitting corrosion** is described as the gradual growth of small holes or pits on the surface of a material

- Challenging to detect due to initial nanoscale size of most pits – but necessary

- For simplicity, the *Kondo-Wei Model* for corrosion rate assumes a hemispherical shape for its pits

Corrosion: Pitting Corrosion Definitions

Kondo-Wei Model

$$i_{corr} = 2\pi r^2 \frac{dr}{dt} = \frac{MI_{p_0}}{nF\rho}\exp\left(-\frac{E_a}{k_B T}\right)$$

$$= \frac{MI_{p_0}}{nF\rho}\exp\left(-\frac{Q}{RT}\right)$$

- i_{corr} – volumetric pitting corrosion rate
- r – hemispherical radius of pit
- M – molecular weight of material (g/mol)
- I_{p0} – pitting current coefficient (C/s)

  $$I_{p_0} = (1+p)6.5\times10^{-5}$$
- k_B – Boltzmann constant (8.617×10^{-5} eV/K)
- p – number of particles in a group
- n – valence electrons
- F – Faraday Constant 96,514 C/mol
- ρ – material density (g/m^3)
- T – Temperature in Kelvin

- Can also solve for time for a pit to reach final pit radius r_f from initial pit radius r_0

Corrosion: Pitting Corrosion Definitions

- Kondo-Wei Model for time w.r.t. pit radius

$$t = \frac{2\pi}{3}\frac{nF\rho}{MI_{p_0}}\left(r_f^3 - r_0^3\right)\exp\left(\frac{E_a}{kT}\right)$$
$$= \frac{2\pi}{3}\frac{nF\rho}{MI_{p_0}}\left(r_f^3 - r_0^3\right)\exp\left(\frac{Q}{RT}\right)$$

- Can also be expressed in terms of pH of a material where,

$$\frac{2\pi}{3}\frac{nF\rho}{MI_{p_0}} = A \times pH$$

Corrosion: Pitting Corrosion Definitions

- Apply $A \times pH$ with the Kondo-Wei Model,

$$i_{corr} = 2\pi r^2 \frac{dr}{dt} = \frac{2\pi}{3A \times pH} \exp\left(-\frac{E_a}{kT}\right)$$

$$= \frac{2\pi}{3A \times pH} \exp\left(-\frac{Q}{RT}\right)$$

$$t = A \times pH \times \left(r_f^3 - r_0^3\right) \exp\left(\frac{E_a}{kT}\right)$$

$$= A \times pH \times \left(r_f^3 - r_0^3\right) \exp\left(\frac{Q}{RT}\right)$$

- We can use either form for the following tool

Corrosion: Pitting Corrosion Tools

Pitting Corrosion Calculator Tool

```
corrosion.pitting(dat, r_0, r_cr, properties, units)
```

- “*dat*” – Tabular pit corrosion data
- “*r_0*” – Initial pit radius (0 is default)
- “*r_cr*” – Final or critical pit radius
- “*properties*” – material properties based on the Kondo-Wei model if known (M, n, ρ)
- “*units*” – (1) SI and (2) English

Corrosion: Pitting Corrosion Tools

Example: Consider the following data of pit growth by pH and temperature. Assume an initial pit radius r_0 of 0 and estimate the critical time when critical pit radius r_f is 0.15 mm.

	Pit #1 Temp = 395 K pH = 5.5	Pit #2 Temp = 390 K pH = 5	Pit #3 Temp = 500 K pH = 4	Pit #4 Temp = 385 K pH = 4.5	Pit #5 Temp = 390 K pH = 3.5
Time (hours)	Pit radius (mm)	Pit radius (mm)	Pit radius (mm)	Pit radius (mm)	Pit radius (mm)
10	0.01	0.005	0.02	0.018	0.03
100	0.02	0.017	0.04	0.035	0.05
150	0.05	0.018	0.055	0.045	0.065
200	0.07	0.028	0.065	0.062	0.089
500	0.09	0.04	0.078	0.087	0.125
1,000	0.10	0.065	0.09	0.195	0.15

Corrosion: Pitting Corrosion Tools

Time (hours)	Radius (mm)	Pit	Temperature (K)	PH
10	0.01	Pit 1	395	5.5
100	0.02	Pit 1	395	5.5
150	0.05	Pit 1	395	5.5
200	0.07	Pit 1	395	5.5
500	0.09	Pit 1	395	5.5
1000	0.1	Pit 1	395	5.5
10	0.005	Pit 2	390	5
100	0.017	Pit 2	390	5
150	0.018	Pit 2	390	5
200	0.028	Pit 2	390	5
500	0.04	Pit 2	390	5
1000	0.065	Pit 2	390	5
10	0.02	Pit 3	500	4
100	0.04	Pit 3	500	4
150	0.055	Pit 3	500	4
200	0.065	Pit 3	500	4
500	0.078	Pit 3	500	4
1000	0.09	Pit 3	500	4
10	0.018	Pit 4	385	4.5
100	0.035	Pit 4	385	4.5
150	0.045	Pit 4	385	4.5
200	0.062	Pit 4	385	4.5
500	0.087	Pit 4	385	4.5
1000	0.195	Pit 4	385	4.5
10	0.03	Pit 5	390	3.5
100	0.05	Pit 5	390	3.5
150	0.065	Pit 5	390	3.5
200	0.089	Pit 5	390	3.5
500	0.125	Pit 5	390	3.5
1000	0.15	Pit 5	390	3.5

Pitdataexample

- **NOTE**: Pitting corrosion data has a specific entry order by column as all degradation data should be when used by the RMT
  - Column 1 – Time
  - Column 2 – Degradation amount
  - Column 3 – Unit ID
  - Column(s) 4 and up – Stress (or stresses)

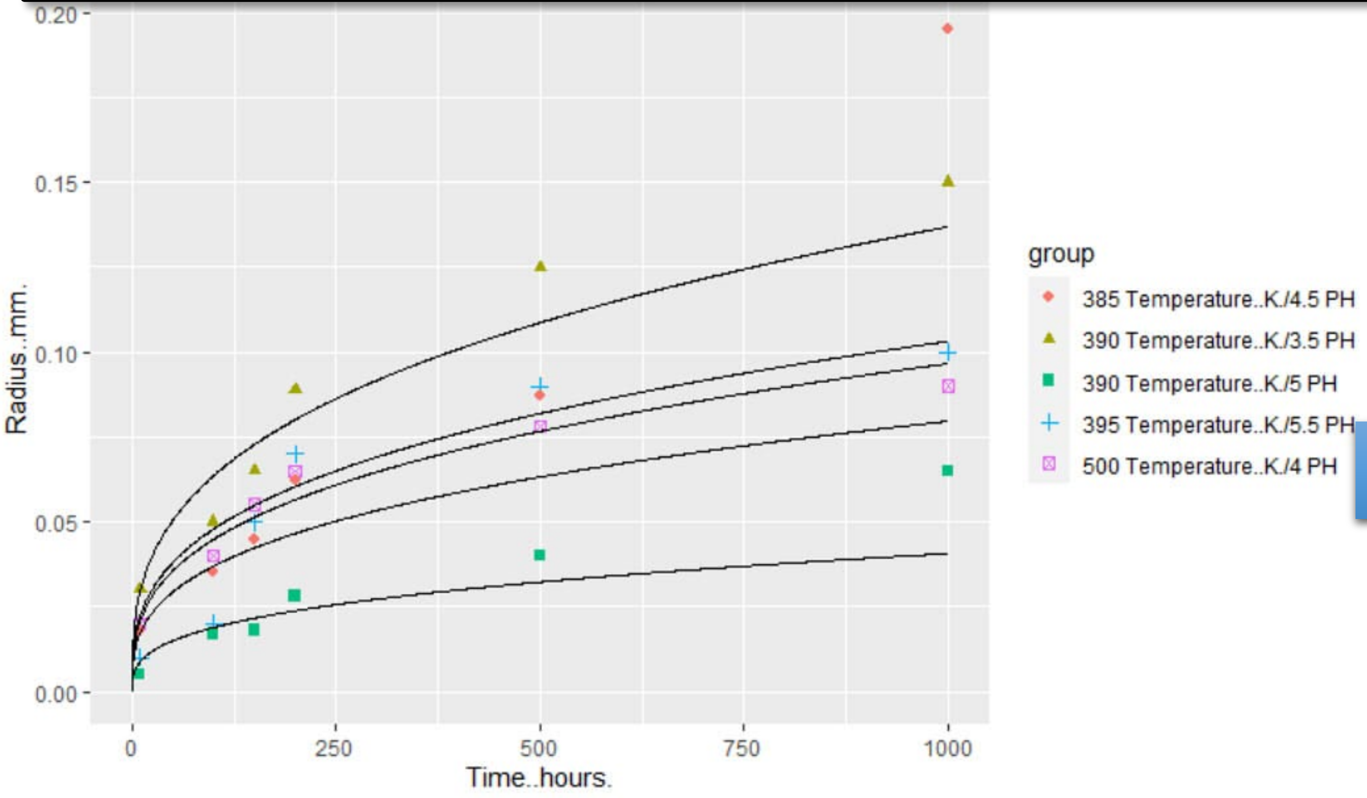
Define:

```
datPitexample  <- read.csv("https://raw.githubusercontent.com/Center-for-Risk-and-Reliability/RMT/main/CSVExampleData/Pitdataexample.csv")
```

Corrosion: Pitting Corrosion Tools

Input:

```
corrosion.pitting(datPitexample, r_0 = 0, r_cr = 0.05, units = 1)
```



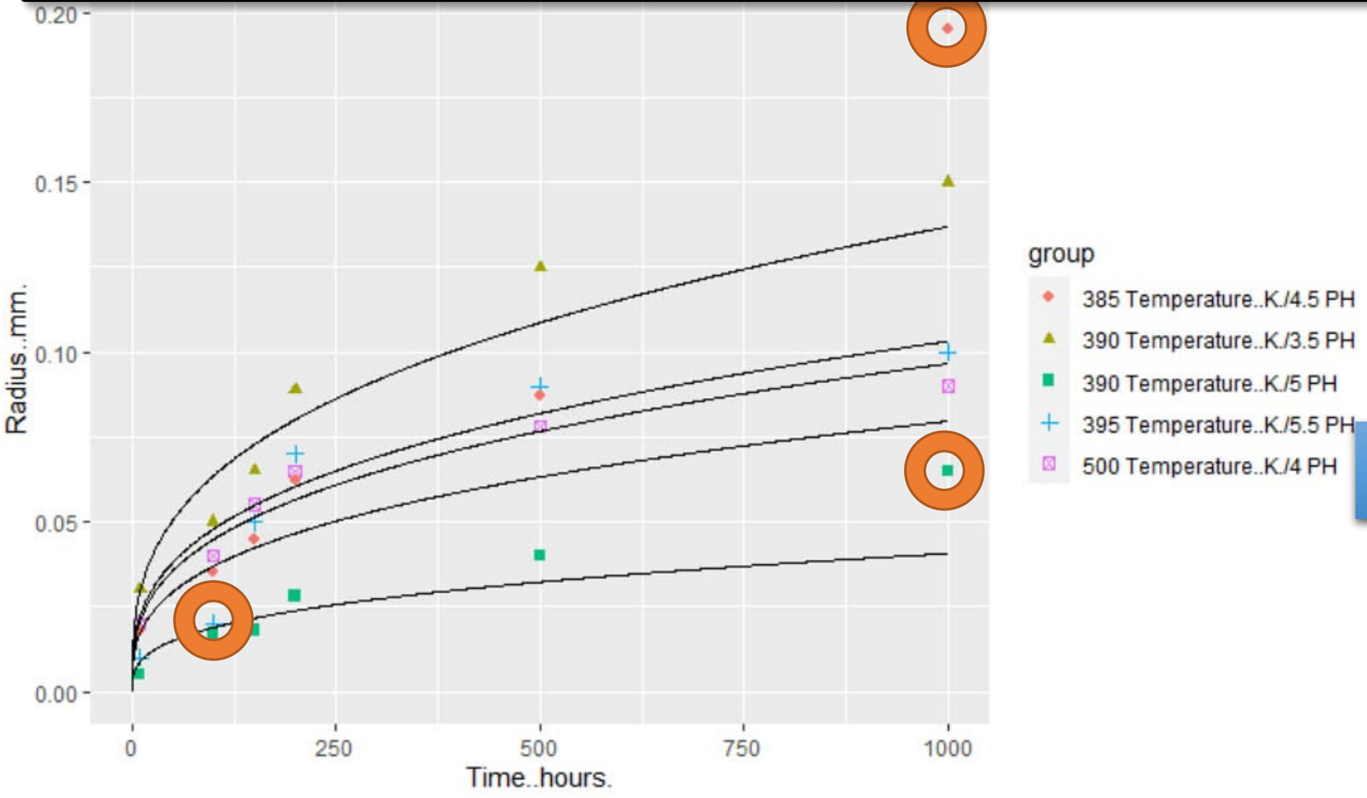
Output:

Pit	A (hr/mm^3)	E_a	Pseudo Critical Time (hours)
Pit 1	359,705	2.79×10^{-6}	247.3
Pit 2	2,975,714	3.29×10^{-6}	1,860
Pit 3	277,065	2.16×10^{-6}	138.5
Pit 4	202,077	2.73×10^{-6}	113.7
Pit 5	111,228	2.57×10^{-6}	48.7

Corrosion: Pitting Corrosion Tools

Input:

```
corrosion.pitting(datPitexample, r_0 = 0, r_cr = 0.05, units = 1)
```

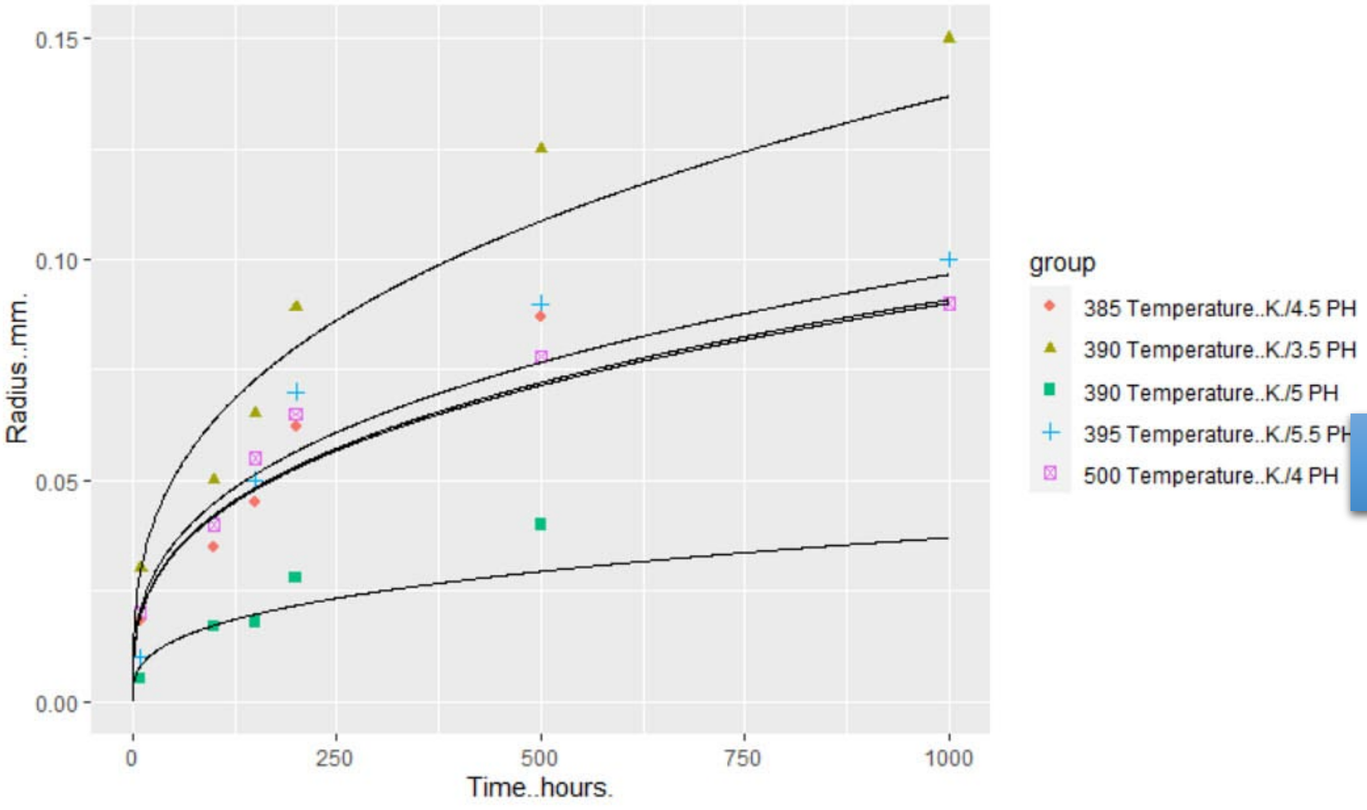


- **NOTE**: Example and model show some significant outliers

Output:

Pit	A	E_a	Pseudo Critical Time (hours)
Pit 1	359,705	2.79×10^{-6}	247.3
Pit 2	2,975,714	3.29×10^{-6}	1,860
Pit 3	277,065	2.16×10^{-6}	138.5
Pit 4	202,077	2.73×10^{-6}	113.7
Pit 5	111,228	2.57×10^{-6}	48.7

Corrosion: Pitting Corrosion Tools



- Removing outlier data and recalculating provides different output for pits 1, 2, and 4

Updated Output:

Pit	A	E_a	Pseudo Critical Time (hours)
Pit 1	**248,791**	**2.71×10^{-6}**	**171.0**
Pit 2	**3,943,294**	**3.35×10^{-6}**	**2,464.8**
Pit 3	277,065	2.16×10^{-6}	138.5
Pit 4	**295,999**	**2.82×10^{-6}**	**166.5**
Pit 5	111,228	2.57×10^{-6}	48.7

Toolset 2:

RELIABILITY MODELING

Nonparametric Probability: Nonparametric and Parametric Reliability Modeling

Nonparametric modeling:

- Strictly based on raw data (failure and censored) and rank of data
- No defined model

Parametric modeling:

- Requires statistical checks to determine fitness
- May adhere to *many* defined models

- **Definition**: To go from raw data to a defined reliability model, requires raw estimators for reliability metrics called **probability plotting positions**

Nonparametric Probability: Probability Plotting Positions

- Nonparametric estimates exist for:
  - Failure probability (cumulative density function, CDF), $F(x)$
  - Reliability (Survival) function, $R(x)$ (or complementary CDF $1 - F(x)$)
  - Hazard function (failure rate), $h(x)$
  - Cumulative hazard function, $H(x)$
  - Probability density (PDF), $f(x)$

- However there is a *large* selection of probability plotting positions to choose from

Nonparametric Probability: Probability Plotting Positions

Plotting Position Model	Plotting Position for CDF, $F_i(x_i)$	Plotting Position Model	Plotting Position for CDF, $F_i(x_i)$
Blom	$F_i(x_i)=\frac{i-0.375}{n+0.25}$	Jenkinson's (Beard's)	$F_i(x_i)=\frac{i-0.31}{n+0.38}$
Mean	$F_i(x_i)=\frac{i}{n+1}$	Bernard & Bos-Levenbach	$F_i(x_i)=\frac{i-0.3}{n+0.2}$
Median	$F_i(x_i)=\frac{i-0.3}{n+0.4}$	Tukey	$F_i(x_i)=\frac{i-1/3}{n+1/3}$
Midpoint	$F_i(x_i)=\frac{i-0.5}{n}$	Grigorten	$F_i(x_i)=\frac{i-0.44}{n+0.12}$
Nelson-Aalen	$F_i(x_i)=1-\exp\left(-\sum_{x_i\le x}\frac{d_i}{n_{i-1}-d_{i-1}-c_{i-1}}\right)$	Kaplan-Meier	$F_i(x_i)=1-\prod_{x_i\le x}\left(1-\frac{d}{j}\right)$

- i – data ranking at failure time x_i
- n – total number of data in set
- d – number of failed units
- c – number of censored units
- j – reverse-rank or the reverse order of cumulative ranking metric i

Nonparametric Probability: Probability Plotting Positions

- Each model has plotting position estimates for the other reliability metrics

Example: These are Blom's plotting position estimates

Blom Plotting Position	Reliability Metric	Plotting Position	Reliability Metric	Plotting Position
Failure probability (CDF) $F_i(x_i) = \frac{i - 0.375}{n + 0.25}$	Reliability	$R_i(x_i) = \frac{n - i + 0.625}{n + 0.25}$	Hazard rate (failure rate)	$h_i(x_i) = \frac{1}{(n - i + 0.625)(x_{i+1} - x_i)}$
	Probability Density (PDF)	$f_i(x_i) = \frac{1}{(n + 0.25)(x_{i+1} - x_i)}$	Cumulative hazard function	$H_i(x_i) = -\ln\left(\frac{n - i + 0.625}{n + 0.25}\right)$

Nonparametric Probability: Probability Estimation Tools

RMT has tools for each of these plotting positions. For example:

Kimball/Blom Non-Parametric Output Tabulation Tool

```
plotposit.blom(i, xi, rc)
```

- "*i*" – the rank of failure or primary event data "*xi*"
- "*xi*" – vector of failure or primary event data in a given set
- "*rc*" – vector right censored data of a given set

Rank can be computed by,

Rank calculator for failure and right censored data Tool

```
rankcalc(xi, rc)
```

Nonparametric Probability: Probability Estimation Tools

For simplicity though we can simply use,

Plotting Position Selector Tool

```
plotposit.select(xi, rc, pp)
```

- "*xi*" – vector of failure or primary event data in a given set
- "*rc*" – vector right censored data of a given set
- "*pp*" – Named probability plotting position model (ex. Enter "Blom" for Blom probability plotting position model)

Nonparametric Probability: Probability Estimation Tools

Example: A simple demonstration using the following time-to-failure data made up of failure times and right censored times

	Failure or Right Censored Times (hours)
Failed	65, 75, 75.2, 87.5, 88.3, 94.2, 101.7, 109.2, 130
Censored	31.7+, 39.2+, 57.2+, 65.8+, 70+, 105.8+, 110+

Define:

```
xFset <- c(65.0,75.0,75.2,87.5,88.3,94.2,101.7,109.2,130.0)
rCset <- c(31.7,39.2,57.2,65.8,70.0,105.8,110.0)
```

Nonparametric Probability: Probability Estimation Tools

- *Solving Method 1*: Compute Rank then compute plotting positions

Input Option 1:

```
rankiset <- rankcalc(xFset, rCset)
plotposit.blom(i = rankiset, xi = xFset, rc = rCset)
```

- *Solving Method 2*: Use `plotposit.select` to do *both* one time

Input Option 2:

```
plotposit.select(xi = xFset, rc = rCset, "Blom")
```

Nonparametric Probability: Probability Estimation Tools

Output:

Failure Time, x_i (hour)	$F_i(x_i)$	$R_i(x_i)$	$h_i(x_i)$	$H_i(x_i)$	$f_i(x_i)$	Rank, i
65	0.038	0.962	0.006	0.039	0.006	1
75	0.128	0.872	0.353	0.137	0.308	2.455
75.2	0.217	0.783	0.006	0.245	0.005	3.909
87.5	0.307	0.693	0.111	0.367	0.077	5.364
88.3	0.397	0.603	0.017	0.505	0.01	6.818
94.2	0.486	0.514	0.016	0.666	0.008	8.273
101.7	0.576	0.424	0.019	0.857	0.008	9.727
109.2	0.687	0.313	0.009	1.163	0.003	11.545
130	0.855	0.145	---	1.933	---	14.273

- Can use output in other RMT tools of course, but you may plot output as is

Nonparametric Probability: Probability Estimation Tools

Non-Parametric Output Tabulation Tool

```
plottable.nonparam(xi, rc, FRhHf, relfcn, alpha, xlabel)
```

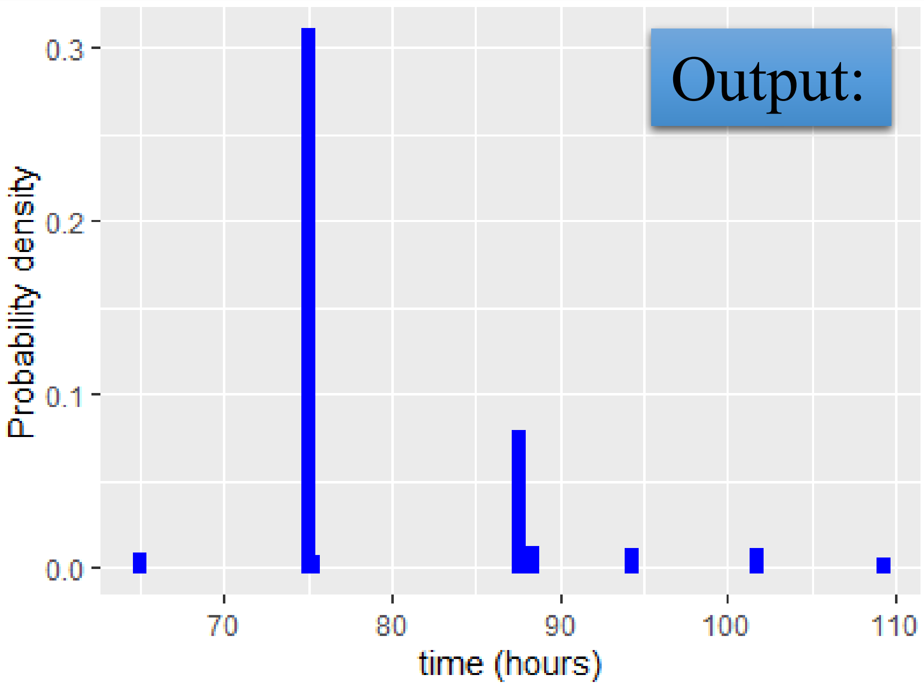
- "*xi*" – vector of failure or primary event data in a given set
- "*rc*" – vector right censored data of a given set
- "*FRhHf*" – output from tabulation of nonparametric computations (`plotposit.select`)
- "*relfcn*" – entry for nonparametric reliability metric to plot: (1) $F(x)$ "unreliability", (2) $R(x)$ "reliability", (3) $h(x)$ "hazard", (4) $H(x)$ "cumulativehazard", (5) $f(x)$ "probabilitydensity"
- "*alpha*" – confidence parameter where 100(1 – *alpha*)% is the confidence bound for $F(x)$, $R(x)$ and $H(x)$
- "*xlabel*" – label for x-axis in plot

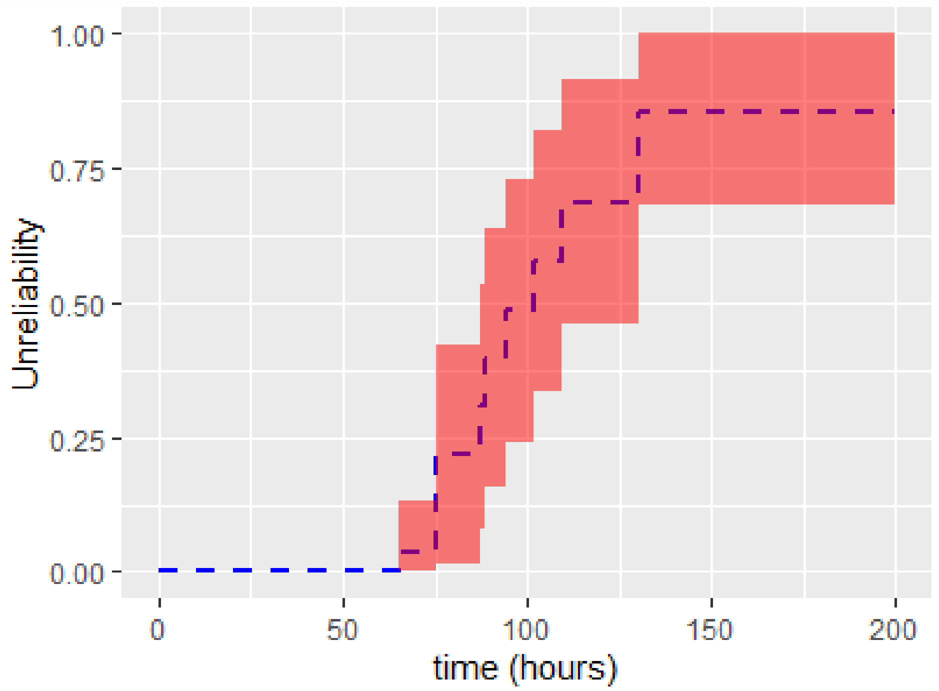
Nonparametric Probability: Probability Estimation Tools

Input:

```
plottable.nonparam(xi = xFset, rc = rCset, outputnonparam,
                   relfcn = “probabilitydensity", 0.05,
                   xlabel = "time (hours)")
```

```
plottable.nonparam(xi = xFset, rc = rCset, outputnonparam,
                   relfcn = "unreliability", 0.05,
                   xlabel = "time (hours)")
```





Least Squares Estimation: Least Squares Estimation and Regression Analysis

- Regression analysis is an important tool in reliability analysis because it is one of the methods used to assign the best distribution to a set of data

- The basis of probability plotting where X-axis and Y-axis are based on the linearization of a CDF

Least Squares Estimation: Least Squares Estimation and Regression Analysis

Example: Take the Weibull CDF,

$$F(x) = 1 - \exp\left[-\left(\frac{x}{\alpha}\right)^{\beta}\right]$$

where α is the scale parameter and β is the shape parameter

- By log-linearization we get,

$$\ln\left\{-\ln\left[1 - F(x)\right]\right\} = \beta \ln x - \beta \ln \alpha$$

<u>Least Squares Parameter (LSQ) Estimates</u>:

$$\hat{\beta} = \text{slope}$$

$$\hat{\alpha} = \exp\left(-\frac{\text{intercept}}{\text{slope}}\right)$$

Least Squares Estimation: Probability Plotting and Least Squares Estimate Tools

Probability Plot Tool

```
probplot.DIST(data, pp, xlabel, confid)
```

- "*dat*" – primary event data, censored status, and stress level in tabular form
- "*pp*" – Named probability plotting position model (ex. Enter "Blom" for Blom probability plotting position model)
- "*xlabel*" – label for x-axis in plot
- "*confid*" – confidence bound between 0 and 1 (0.95 for 95% confidence)

RMT currently has the following distributions set for probability plotting:

- Normal (*DIST* = "*nor*")
- Lognormal (*DIST* = "*logn*")
- Weibull (*DIST* = "*wbl*")
- Three-Parameter Weibull (*DIST* = "*wbl3P*")
- Exponential (*DIST* = "*exp*")
- Two-Parameter Exponential (*DIST* = "*exp2P*")
- Logistic (*DIST* = "*logist*")
- Loglogistic (*DIST* = "*loglogist*")
- Gumbel (*DIST* = "*gumb*")
- Gamma (*DIST* = "*gam*")

Least Squares Estimation: Probability Plotting and Least Squares Estimate Tools

- **NOTE**: Data for probability plotting has a specific entry order by column when used by the RMT
  - Column 1 – Primary event (time usually) data
  - Column 2 – Censored status for Column 1 (0 for right censored, 1 for not censored)
  - Column 3 and up – Stress levels for Column 1

Define:

Set stress at "300 K" for all data in this example

```
time <- c(xFset, rCset)
cens <- c(rep(1,9), rep(0,7))
stress <- rep(300,16)
datset1 <- cbind(time, cens, stress)
```

```
> datset1
      time cens stress
 [1,]  65.0    1    300
 [2,]  75.0    1    300
 [3,]  75.2    1    300
 [4,]  87.5    1    300
 [5,]  88.3    1    300
 [6,]  94.2    1    300
 [7,] 101.7    1    300
 [8,] 109.2    1    300
 [9,] 130.0    1    300
[10,]  31.7    0    300
[11,]  39.2    0    300
[12,]  57.2    0    300
[13,]  65.8    0    300
[14,]  70.0    0    300
[15,] 105.8    0    300
[16,] 110.0    0    300
```

Least Squares Estimation: Probability Plotting and Least Squares Estimate Tools

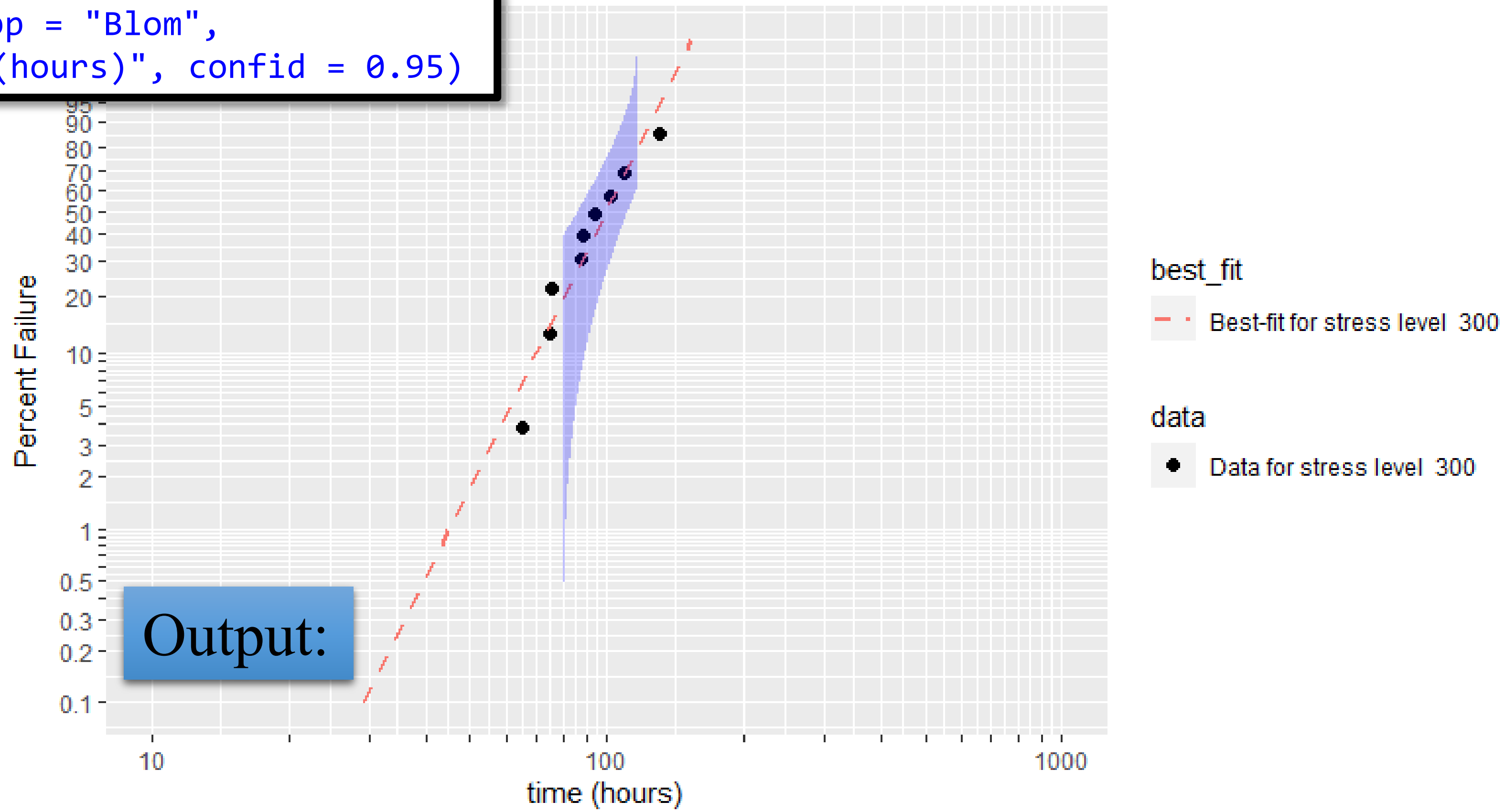
Input:

```
probplot.wbl(dat = datset1, pp = "Blom",
             xlabel = "time (hours)", confid = 0.95)
```

$\hat{\alpha} = 106.34$ hours

$\hat{\beta} = 5.35$

$R^2 = 0.918$



Output:

Maximum Likelihood Estimation: Overview

- An LSQ estimate is a good start, but *maximum likelihood estimates* are the standard in reliability modeling

- **<u>Definition</u>**: **Maximum likelihood estimation** (MLE) handles most data analysis purposes including parameter estimation (for parameter set $\vec{\Theta}$), distribution fitting, and confidence intervals of parameter estimates.

Maximum Likelihood Estimation: Overview

- **<u>Definition</u>**: MLE makes use of a **likelihood** ℓ which is joint distribution that accounts for all n failure data and all m censored data

$$\ell = \prod_{i=1}^{n} f\left(x_i \mid \vec{\Theta}\right) \prod_{j=1}^{m} R\left(x_j \mid \vec{\Theta}\right)$$

- **<u>Definition</u>**: The natural log of a likelihood is a **log-likelihood** Λ which in many cases is easier to work with than just the likelihood

$$\Lambda = \sum_{i=1}^{n} \ln\left[f\left(x_i \mid \vec{\Theta}\right)\right] + \sum_{j=1}^{m} \ln\left[R\left(x_j \mid \vec{\Theta}\right)\right]$$

Maximum Likelihood Estimation: Overview

- But LSQ estimates are <u>*still very necessary*</u> in MLE

- R uses optimization analysis to perform MLE (as most platforms) to satisfy partial differential relations of likelihood (or log-likelihood) w.r.t. every parameter equal to zero

$$0 = \frac{\partial \ell}{\partial \theta_1};\ 0 = \frac{\partial \ell}{\partial \theta_2} \text{ or } 0 = \frac{\partial \Lambda}{\partial \theta_1};\ 0 = \frac{\partial \Lambda}{\partial \theta_2}$$

  - where $\vec{\Theta} = [\theta_1, \theta_2]$

- The built-in *Non-linear Minimization* (*nlm*) R function requires an initial estimate to operate. Random or faulty initials would likely result in a local minimum thus the LSQ would be a good starting point.

Maximum Likelihood Estimation: Distribution Fitting Tools

Maximum Likelihood Estimator for Probability Distributions Tool

```
distribution.MLEest(LSQest,dist,xi,rc,confid,sided)
```

- "*LSQest*" – vector of the initial parameter estimates
- "*dist*" – named probability distribution
- "*xi*" – vector of failure or primary event data in a given set
- "*rc*" – vector right censored data of a given set
- "*confid*" – confidence bound between 0 and 1 (0.95 for 95% confidence)
- "*sided*" – confidence limits for parameters: two-sided, one-sided high, or one-sided low

- **<u>NOTE</u>**: We also have distribution-specific fitting tools in the RMT

Maximum Likelihood Estimation: Distribution Fitting Tools

Weibull Fit Tool

```
fit.wbl(xi,rc,pp,confid,sided)
```

- "*xi*" – vector of failure or primary event data in a given set
- "*rc*" – vector right censored data of a given set
- "*pp*" – Named probability plotting position model (ex. Enter "Blom" for Blom probability plotting position model)
- "*confid*" – confidence bound between 0 and 1 (0.95 for 95% confidence)
- "*sided*" – confidence limits for parameters: two-sided, one-sided high, or one-sided low

Maximum Likelihood Estimation: Distribution Fitting Tools

Example: Continuing from the LSQ part of this example, we want to find the MLE estimate with 95% confidence bounds and compare with the LSQ

	Failure or Right Censored Times (hours)
Failed	65, 75, 75.2, 87.5, 88.3, 94.2, 101.7, 109.2, 130
Censored	31.7+, 39.2+, 57.2+, 65.8+, 70+, 105.8+, 110+

Input:

```
fit.wbl(xi = xFset,rc = rCset,
        pp = "Blom",confid = 0.95,
        sided = "twosided")
```

NOTE: The `fit.wbl` tool computes the LSQ estimate within the code

Maximum Likelihood Estimation: Distribution Fitting Tools

Output:

	$\hat{\alpha}$	$\hat{\beta}$
LSQ Point estimate	*106.34*	*5.35*
MLE Point Estimate	106.11	5.36
Standard Error	1.65	0.33
Lower 95%	93.9	3.32
Upper 95%	119.91	8.67

- MLE also provides the variance-covariance matrix $\Sigma = \begin{bmatrix} 43.80 & 0.72 \\ 0.72 & 1.72 \end{bmatrix}$
- Additional output includes numerical log-likelihood and AIC score for comparison with other fits

Maximum Likelihood Estimation: Distribution Fitting Tools

Weibull Probability Density Function Tool

```
plot.pdf.wbl(xi,rc,pp,confid,sided,xlabel)
```

- "*xi*" – vector of failure or primary event data in a given set
- "*rc*" – vector right censored data of a given set
- "*pp*" – Named probability plotting position model (ex. Enter "Blom" for Blom probability plotting position model)
- "*confid*" – confidence bound between 0 and 1 (0.95 for 95% confidence)
- "*sided*" – confidence limits for parameters: two-sided, one-sided high, or one-sided low
- "*xlabel*" – label for x-axis in plot

Maximum Likelihood Estimation: Distribution Fitting Tools

Weibull Cumulative Distribution Function Tool

```
plot.cdf.wbl(xi,rc,pp,confid,sided,xlabel)
```

- "*xi*" – vector of failure or primary event data in a given set
- "*rc*" – vector right censored data of a given set
- "*pp*" – Named probability plotting position model (ex. Enter "Blom" for Blom probability plotting position model)
- "*confid*" – confidence bound between 0 and 1 (0.95 for 95% confidence)
- "*sided*" – confidence limits for parameters: two-sided, one-sided high, or one-sided low
- "*xlabel*" – label for x-axis in plot

Maximum Likelihood Estimation: Distribution Fitting Tools

Weibull Reliability (Survival) Function Tool

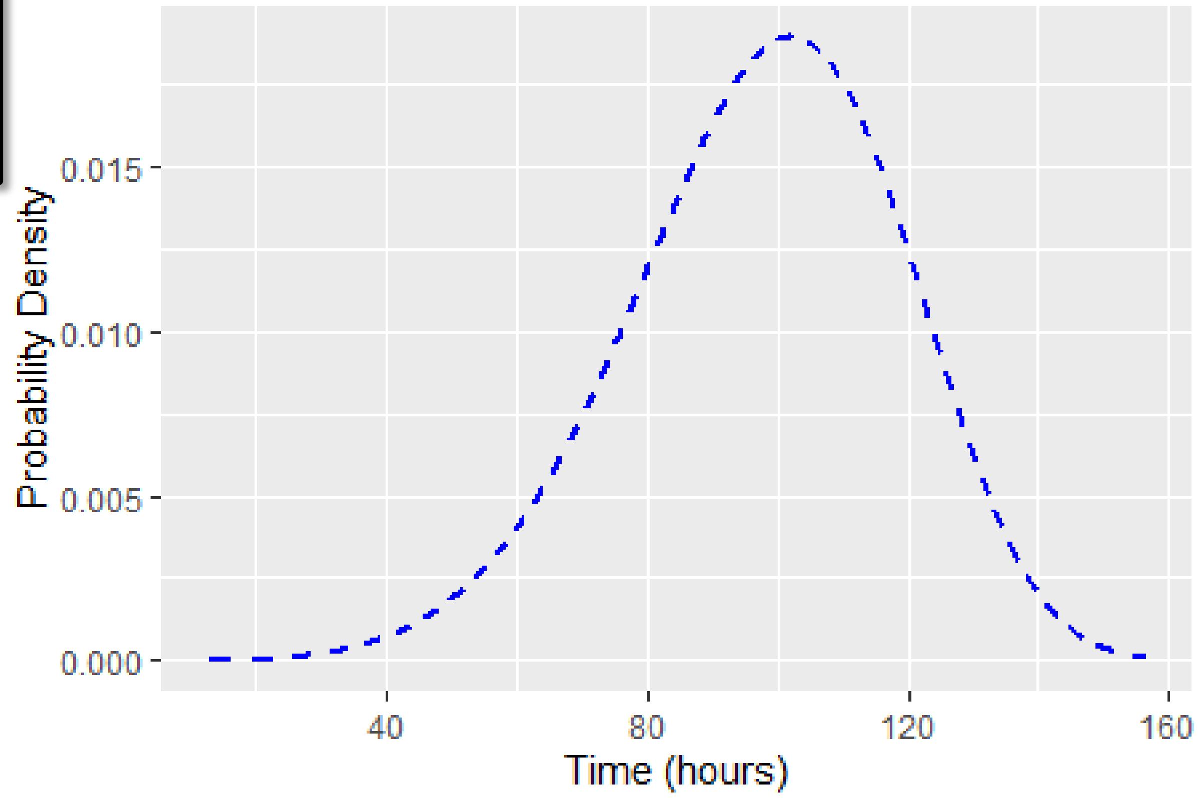
```
plot.reliability.wbl(xi,rc,pp,confid,sided,xlabel)
```

- "*xi*" – vector of failure or primary event data in a given set
- "*rc*" – vector right censored data of a given set
- "*pp*" – Named probability plotting position model (ex. Enter "Blom" for Blom probability plotting position model)
- "*confid*" – confidence bound between 0 and 1 (0.95 for 95% confidence)
- "*sided*" – confidence limits for parameters: two-sided, one-sided high, or one-sided low
- "*xlabel*" – label for x-axis in plot

Maximum Likelihood Estimation: Distribution Fitting Tools

Input:

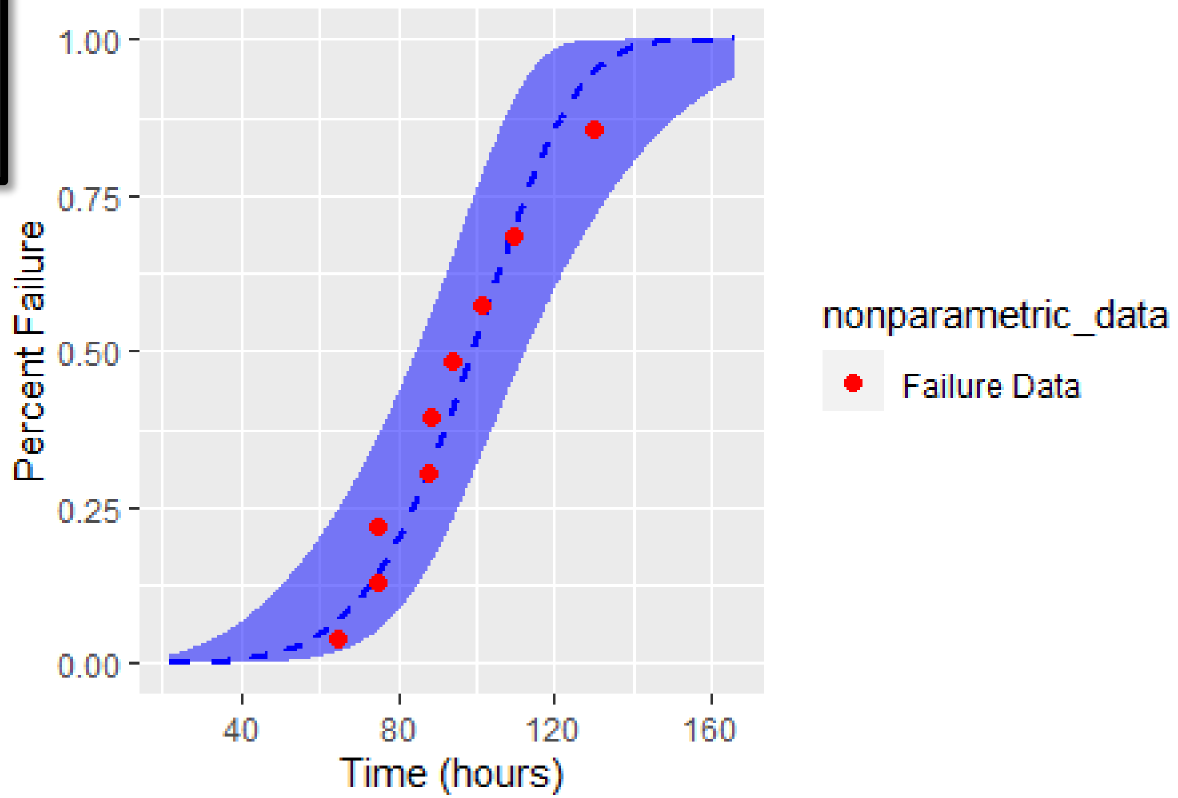
```
plot.pdf.wbl(xi = xFset,rc = rCset,
             pp = "Blom",confid = 0.95,
             sided = "twosided",
             xlabel = "Time (hours)")
```



Maximum Likelihood Estimation: Distribution Fitting Tools

Input:

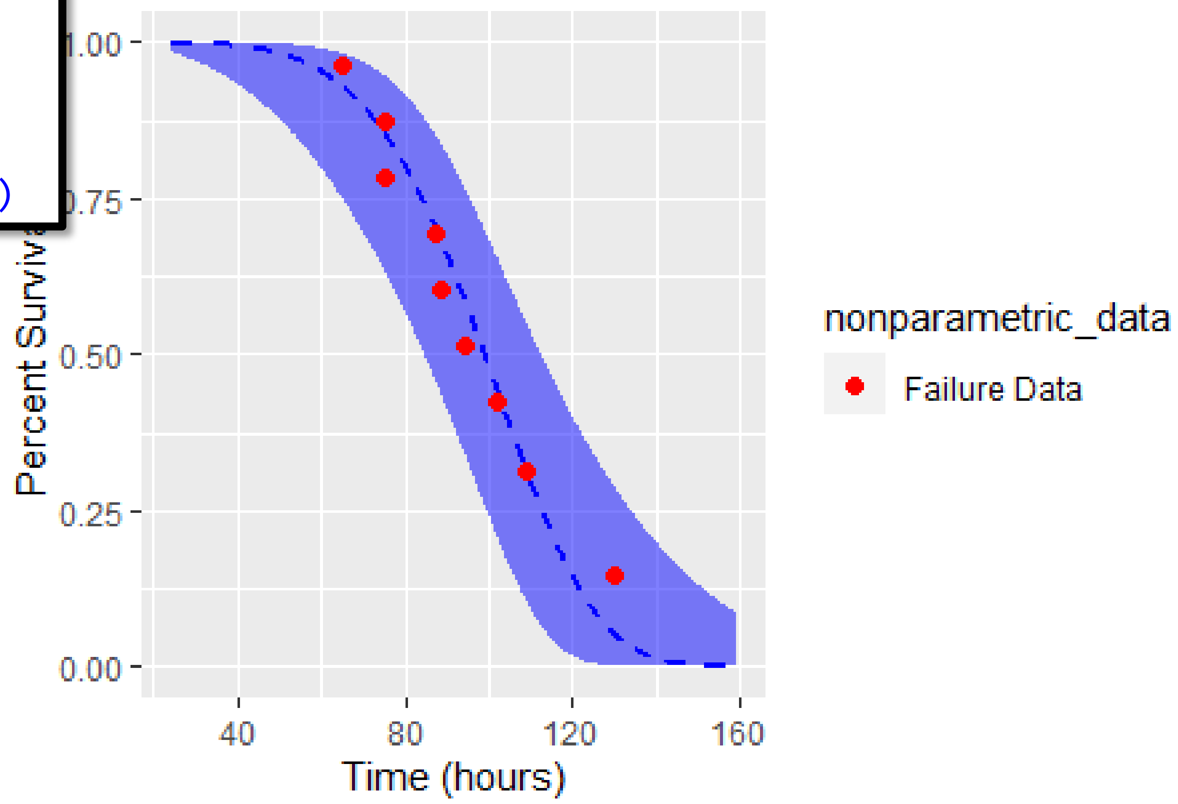
```
plot.cdf.wbl(xi = xFset,rc = rCset,
             pp = "Blom",confid = 0.95,
             sided = "twosided",
             xlabel = "Time (hours)")
```



Maximum Likelihood Estimation: Distribution Fitting Tools

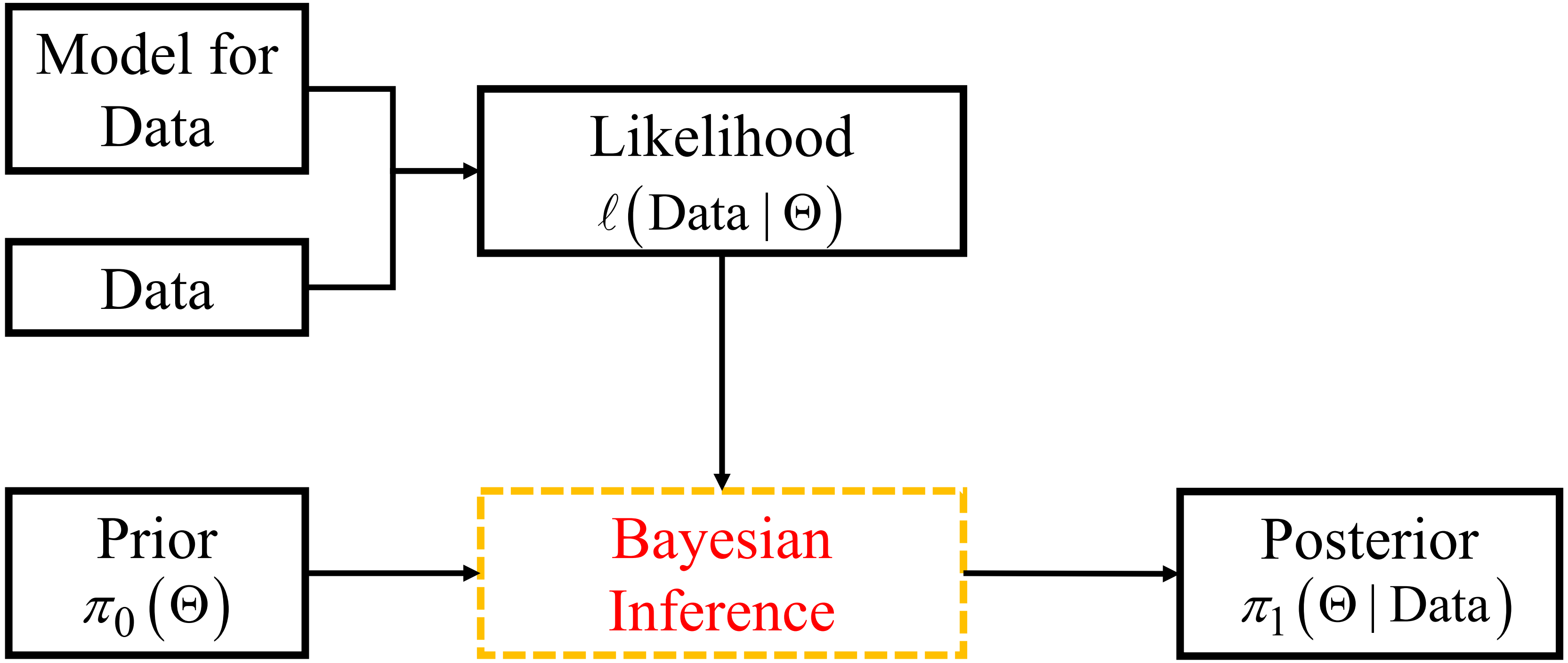
Input:

```
plot.reliability.wbl(xi = xFset,rc = rCset,
                     pp = "Blom",
                     confid = 0.95,
                     sided = "twosided",
                     xlabel = "Time (hours)")
```



Bayesian Estimation: Bayesian Approach

- Bayesian Inference Approach



Bayesian Estimation: Overview of RStan

- All Bayesian operations on RMT are done with the aid of the Rstan computational library

- RStan is an R interface to the Stan code library, widely used for Bayesian statistical inference and sampling

- RStan (and by extension the RMT) makes use of the *No U-Turn Sampler* (aka NUTS) which is an extension of Hamiltonian Monte Carlo, a variant of Metropolis Hastings MCMC that uses Hamiltonian dynamics

Bayesian Estimation: Application of RStan Tools

Bayesian Updater for Probability Distributions Tool

```
distribution.BAYESest(pt_est,dist,xi,rc,confid,priors,nsamples,burnin,nchains)
```

- "*pt_est*" – vector of the initial parameter estimates
- "*dist*" – Named probability distribution
- "*xi*" – vector of failure or primary event data in a given set
- "*rc*" – vector right censored data of a given set
- "*confid*" – confidence bound between 0 and 1 (0.95 for 95% confidence)
- "*nsamples*" – number of MCMC samples or iterations per chain
- "*burnin*" – number of initial MCMC iterations the throw out
- "*nchain*" – number of Markov chains

Bayesian Estimation: Application of RStan Tools

Example: Now we want apply information from the MLE part of this example, where we have the following new data and we want to update the original output through Bayesian analysis

	Failure or Right Censored Times (hours)
Failed	135, 143, 540
Censored	500+, 600+, 600+

Bayesian Estimation: Application of RStan Tools

- From the point estimate and variance covariance output from the MLE analysis, we can extract priors for α and β as normal distributions

$$\vec{M} = \begin{bmatrix} 106.11 \\ 5.36 \end{bmatrix}; \ \Sigma = \begin{bmatrix} 43.80 & 0.72 \\ 0.72 & 1.72 \end{bmatrix}$$

- Define these as R vector data and place each prior in quotes

Define:

```
priorset <- c("normal(106.11,6.61)",
              "normal(5.16,1.311)")
```

- **NOTE**: See Stan documentation to be sure distribution naming and parameter location is correct; otherwise your analysis may be faulty

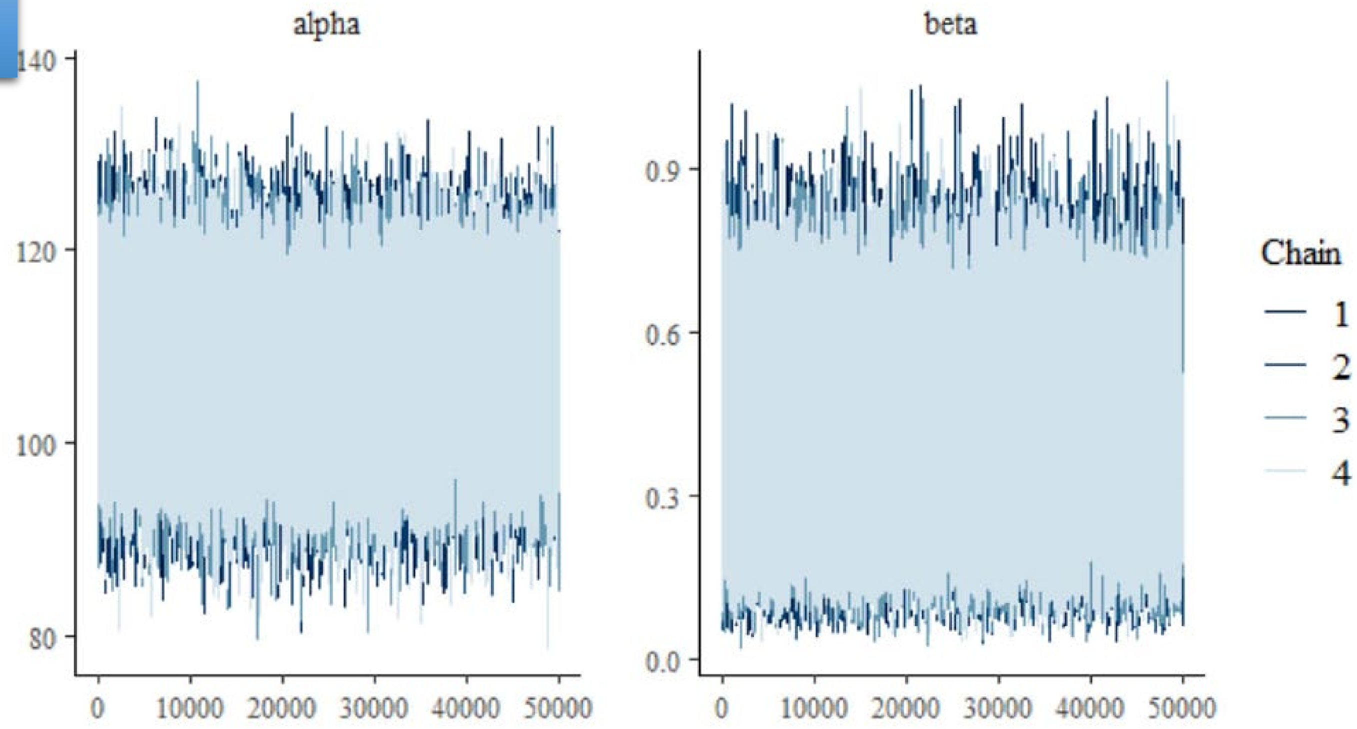
Input:

```
distribution.BAYESest(pt_est = c(100,6),
                      "Weibull",xi = c(135,143,540),
                      rc = c(500,600,600),
                      priors = priorset,
                      nsamples = 50000,
                      burnin = 1000)
```

Bayesian Estimation: Application of RStan Tools

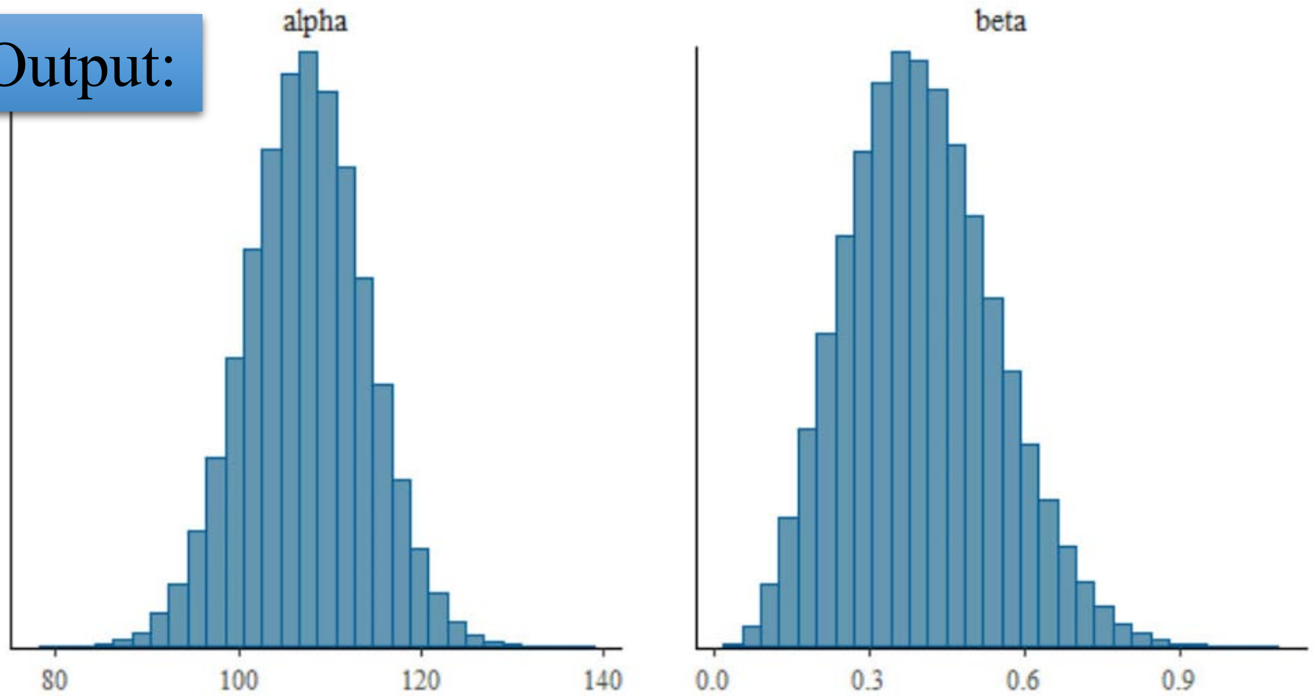
- Output includes several tabular data as well as MCMC trace, posterior histograms, and posterior distribution plots

Output:



Bayesian Estimation: Application of RStan Tools

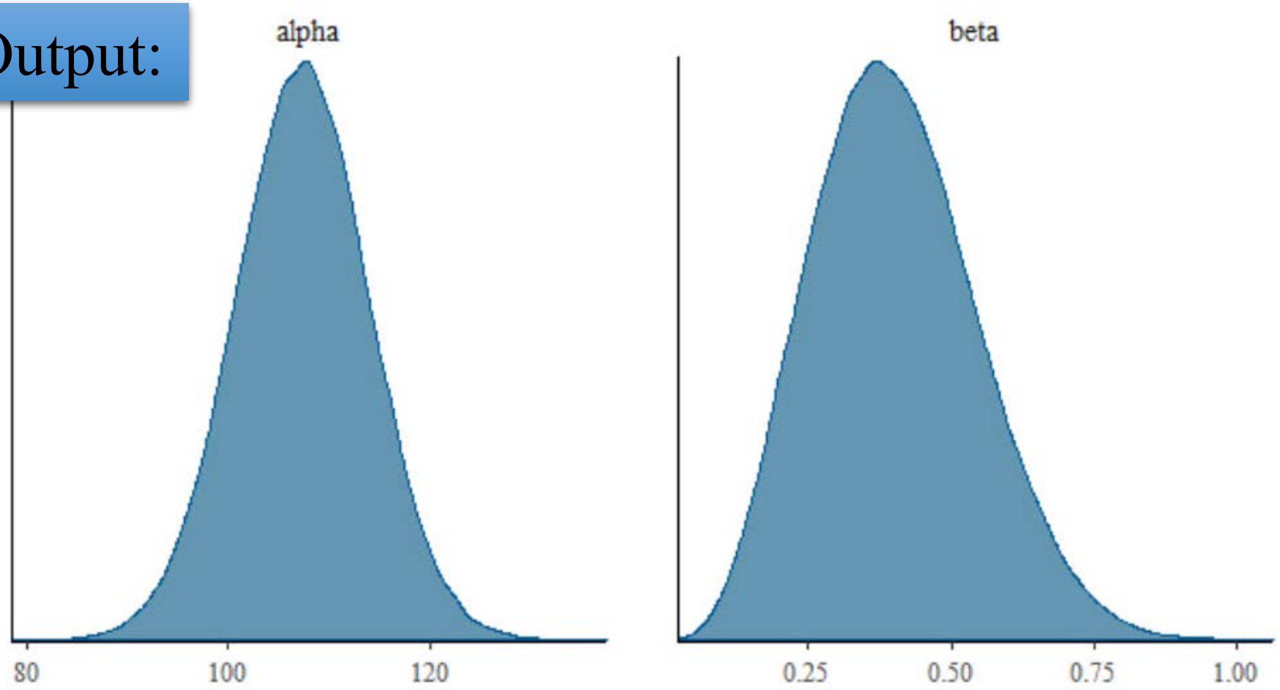
Output:



Posterior	Mean	Standard Deviation	Lower 95%	Median	Upper 95%
$\hat{\alpha}$	107.39	6.56	94.49	107.41	120.25
$\hat{\beta}$	0.4	0.14	0.15	0.39	0.7

Bayesian Estimation: Application of RStan Tools

Output:



Posterior	Mean	Standard Deviation	Lower 95%	Median	Upper 95%
$\hat{\alpha}$	107.39	6.56	94.49	107.41	120.25
$\hat{\beta}$	0.4	0.14	0.15	0.39	0.7

Bayesian Estimation: Application of RStan Tools

- **NOTE**: Take care of any warnings generated by RStan and by extension RMT tools that make use of RStan

Example: A very common warning deals with the `rhat` parameter of any parameter chain.

- If `rhat` is not equal to or close to 1, RStan will warn you as such as it means that convergence didn't occur
- Model may need to be checked or brought to the attention of the developer
- Usually however this may be remedied by modification of the input (burnin, sample number, initial estimates, etc.)

Toolset 3:

ACCELERATED TESTING

Accelerated Life Testing: Overview

- **<u>Definition</u>**: **Accelerated life testing** (ALT) is the methodology where several tests are run at accelerated stress levels S_{acc} and the accelerated life data therein is used to extrapolate the use life at nominal use stress levels S_{use}.

- Many life-stress relations $l(S)$ are available for modeling such data. The RMT considers at least twelve such models.

Accelerated Life Testing: Overview

Life-Stress Model	Life-Stress function, $l(S)$	Life-Stress Model	Life-Stress function, $l(S)$
Linear	$l(S)=b+aS$	Inverse Power	$l(S)=bS^{-a}$
Exponential	$l(S)=b\exp(aS)$	Logarithmic	$l(S)=b+a\ln S$
Arrhenius	$l(S)=b\exp\left(\frac{E_a}{K_B S}\right)$	General Exponential Multi-Stress	$l(S_1,\dots S_n)=\exp(a_0+a_1S_1+\dots,+a_nS_n)$
Eyring	$l(S)=\frac{b}{S}\exp\left(\frac{a}{S}\right)$	Temperature-Humidity	$l(S,H)=A\exp\left(\frac{a}{S}+\frac{b}{H}\right)$
(Alt.) Eyring	$l(S)=\frac{1}{S}\exp\left[-\left(a-\frac{b}{S}\right)\right]$	Generalized Eyring	$l(S,U)=\frac{1}{S}\exp\left[\left(a+\frac{b}{S}\right)+\left(c+\frac{d}{S}\right)U\right]$
Power	$l(S)=bS^a$	Power-Exponential	$l(S,U)=\frac{c}{U^b\exp\left(-\frac{a}{S}\right)}$

- a, b, c, d – model parameters
- E_a – Activation energy (eV)
- k_B – Boltzmann constant 8.617×10^{-5} eV/K
- U – nonthermal stress
- H – humidity
- S – thermal or general stress

Accelerated Life Testing: ALT Analyzer Tools

Life-Stress Model Selector Tool

```
lifestress.select(ls)
```

- "*ls*" – Named life-stress model

Example: Calling the Linear life-stress model pulls a list of functions for life and log-life

Input:

```
lifestress.select("Linear")
```

Output:

```
[[1]]
function(lsparams,S) {
        lsparams[2] + S*lsparams[1]
    }
<bytecode: 0x000001aa0d62e1c0>
<environment: 0x000001aa0daa4258>

[[2]]
function(lsparams,S) {
        log(lsparams[2] + S*lsparams[1])
    }
<bytecode: 0x000001aa0d63c930>
<environment: 0x000001aa0daa4258>
```

- All life-stress models can be called as functions for general use in R or in other RMT tools such as…

Accelerated Life Testing: ALT Analyzer Tools

Acceleration Factor Calculator Tool

```
accelfactor(params,ls,S_acc,S_use)
```

- "*params*" – vector of life-stress model parameters (see documentation for order)
- "*ls*" – Named life-stress model
- "*S_acc*" – accelerated stress or stress vector (for dual or higher stresses)
- "*S_use*" – use stress or stress vector (for dual or higher stresses)

- **Definition**: A common metric in ALT is the **acceleration factor** *AF*, which is a ratio between use life and accelerated life at a given accelerated stress

$$AF = \frac{l\left(S_{\text{use}}\right)}{l\left(S_{\text{accelerated}}\right)}$$

Accelerated Life Testing: ALT Analyzer Tools

Least-Squares Life-Stress Estimator Tool

```
lifestress.LSQest(data,ls,dist,pp,xlabel)
```

Maximum Likelihood Life-Stress Estimator Tool

```
lifestress.MLEest(pt_est,ls,dist,xi,S_xi,rc,S_rc,confid,sided)
```

Bayesian Life-Stress Estimator Tool

```
lifestress.BAYESest(pt_est,ls,dist,xi,S_xi,rc,S_rc,confid,priors,nsamples,burnin,nchains)
```

NEW INPUT VARIABLES FOR STRESS VECTORS

- "*S_xi*" – stress vector (or list of vectors) corresponding with failure data "*xi*"
- "*S_rc*" – stress vector (or list of vectors) corresponding with right censored data "*rc*"

Accelerated Life Testing: ALT Analyzer Tools

- All distributions have parameters that are replaced by governing life-stress model to effectively conduct ALT analysis

Example: Normal, lognormal, Weibull, and Exponential distributions require the following replacements

Standard Life Distribution (PDF)	Parameter to replace	Life-Stress Distribution (PDF)
Normal $f(x)=\frac{1}{\sigma\sqrt{2\pi}}\exp\left[-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right]$	Mean $\mu=l(S)$	$f(x)=\frac{1}{\sigma\sqrt{2\pi}}\exp\left[-\frac{1}{2}\left(\frac{x-l(S)}{\sigma}\right)^2\right]$
Lognormal $f(x)=\frac{1}{\sigma_t x\sqrt{2\pi}}\exp\left[-\frac{1}{2}\left(\frac{\ln x-\mu_t}{\sigma_t}\right)^2\right]$	Log-mean $\mu_t=\ln l(S)$	$f(x)=\frac{1}{\sigma_t x\sqrt{2\pi}}\exp\left[-\frac{1}{2}\left(\frac{\ln x-\ln l(S)}{\sigma_t}\right)^2\right]$
Weibull $f(x)=\frac{\beta}{\alpha}\left(\frac{x}{\alpha}\right)^{\beta-1}\exp\left[-\left(\frac{x}{\alpha}\right)^{\beta}\right]$	Scale parameter $\alpha=l(S)$	$f(x)=\frac{\beta}{l(S)}\left(\frac{x}{l(S)}\right)^{\beta-1}\exp\left[-\left(\frac{x}{l(S)}\right)^{\beta}\right]$
Exponential $f(x)=\lambda\exp(-\lambda x)$	Failure rate $\lambda=\frac{1}{l(S)}$	$f(x)=\frac{1}{l(S)}\exp\left[-\left(\frac{1}{l(S)}\right)x\right]$

Accelerated Life Testing: ALT Tool Demo

Example: Let's demonstrate a full ALT analysis using these tools. Start by defining a stress-life fatigue test where the material samples produced this failure (and right censored) data

Stress Amplitude, S_a (ksi)	Fully Reversed Cycles, N_f	Stress Amplitude, S_a (ksi)	Fully Reversed Cycles, N_f
78.9	45,000	59.61	7,800,000
74.02	240,000	59.61	10,000,000
68.16	800,000	58.63	26,000,000+
63.27	1,500,000	57.65	12,000,000+
62.05	2,700,000	57.41	22,000,000+

Then some components made with that material were also fatigue ALT tested and produced new data

Stress Amplitude, S_a (ksi)	Fully Reversed Cycles, N_f	Stress Amplitude, S_a (ksi)	Fully Reversed Cycles, N_f
58.7	1,400,000	57.2	10,000,000+
59.6	2,900,000	55.3	10,000,000+
56.2	9,000,000	45.1	10,000,000+

Accelerated Life Testing: ALT Tool Demo

- Prepare the data used for your prior ALT evaluation

Define:

```
time_prior <- c(45000, 240000, 800000, 1500000, 2700000, 7800000, 10000000, 26000000, 12000000, 22000000)
cens_prior <- c(rep(1,7), rep(0,3))
stress_prior <- c(78.9, 74.02, 68.16, 63.27, 62.05, 59.61, 59.61, 58.63, 57.65, 57.41)
prior_dat <- cbind(time_prior,cens_prior,stress_prior)
```

- Then prepare the data for the Bayesian updating step (only need to establish vectors)

Define:

```
timeF_new <- c(2900000, 1400000, 9000000)
stressF_new <- c(59.6, 58.7, 56.2)
timeRc_new <- c(10000000, 10000000, 10000000)
stressRc_new <- c(57.2, 55.3, 45.1)
```

Accelerated Life Testing: ALT Tool Demo

STEP 1: LSQ estimation

- **NOTE**: ALT requires some assessment on what life-stress model to go with as well as life distribution (*we can't just pick one*)

- Select between *Lognormal* and *Weibull* life distributions and between *linear*, *exponential*, and *inverse-power* life-stress models

Input:

```
lifestress.LSQest(prior_dat,ls="Linear",dist="Weibull",pp="Blom",xlabel="Fully Reversed Cycles")
```

Output:

R^2	Linear life-stress	Exponential life-stress	Inverse-Power life-stress
Weibull life	0.5065	0.9646	0.9654
Lognormal life	0.5269	0.9678	***0.9679***

- By coefficient of determination, we go with the *Lognormal-Inverse-Power model*

Accelerated Life Testing: ALT Tool Demo

Input:

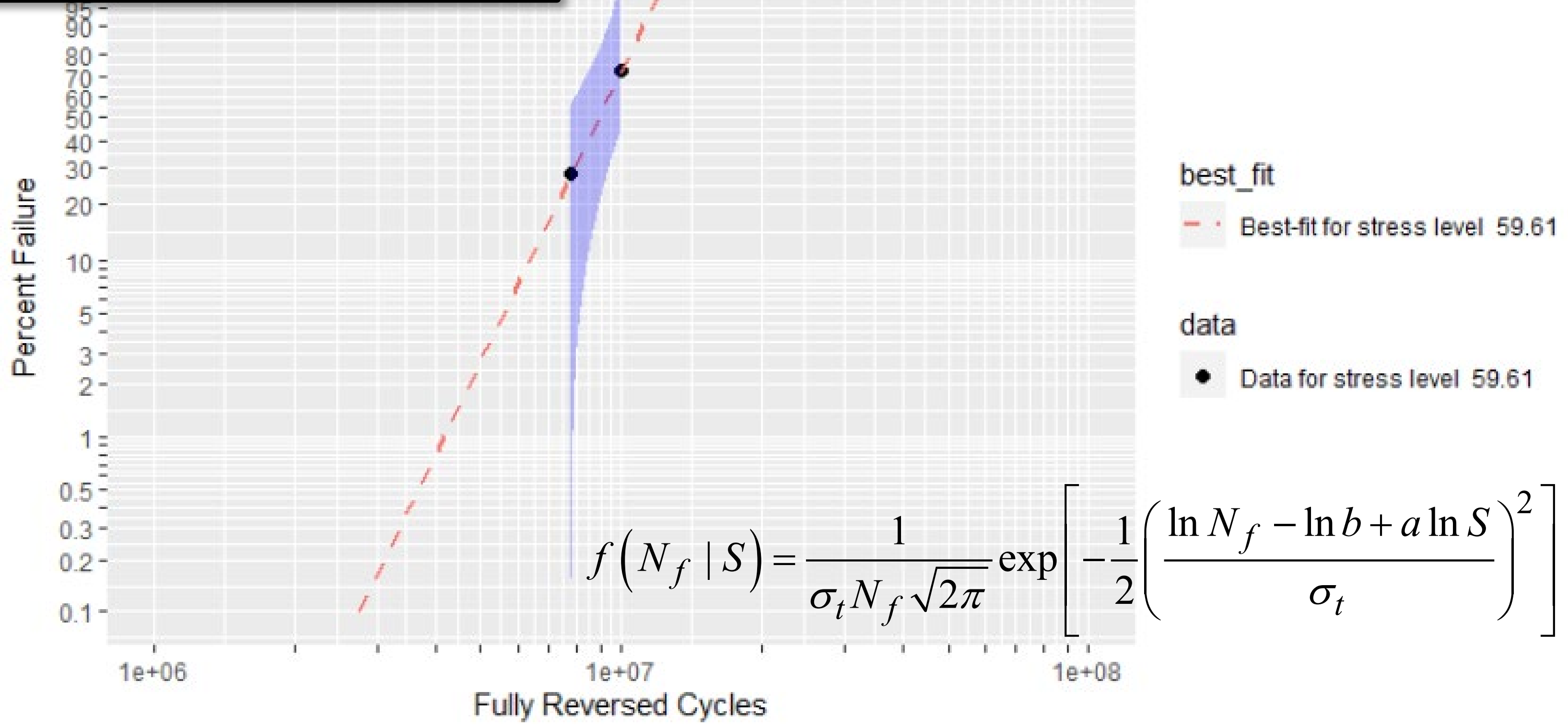
```
lifestress.LSQest(prior_dat,ls="InversePower",dist="Lognormal
",pp="Blom",xlabel="Fully Reversed Cycles")
```

Output:

$$\hat{\sigma}_t = 0.211$$

$$\hat{a} = 16.79$$

$$\hat{b} = 4.08\times 10^{36}$$



$$f\left(N_f \mid S\right) = \frac{1}{\sigma_t N_f \sqrt{2\pi}} \exp\left[-\frac{1}{2}\left(\frac{\ln N_f - \ln b + a\ln S}{\sigma_t}\right)^2\right]$$

Accelerated Life Testing: ALT Tool Demo

STEP 2: MLE estimation

Input:

```
lifestress.MLEest(c(0.211,16.79,4.074e36),"InversePower","Lognormal",
time_prior[1:7], stress_prior[1:7], time_prior[8:10], stress_prior[8:10],
0.9)
```

Output:

	$\hat{\sigma}_t$	$\hat{a}$	$\hat{b}$
Point Estimate	0.488	19.58	6.23×10^{41}
Standard Error	0.0431	0.506	9.62×10^{50}
Lower 90%	0.309	16.94	1.05×10^{37}
Upper 90%	0.773	22.21	3.68×10^{46}

Accelerated Life Testing: ALT Tool Demo

STEP 3: Bayesian updating

- Again, obtain prior data from the MLE output

Define:

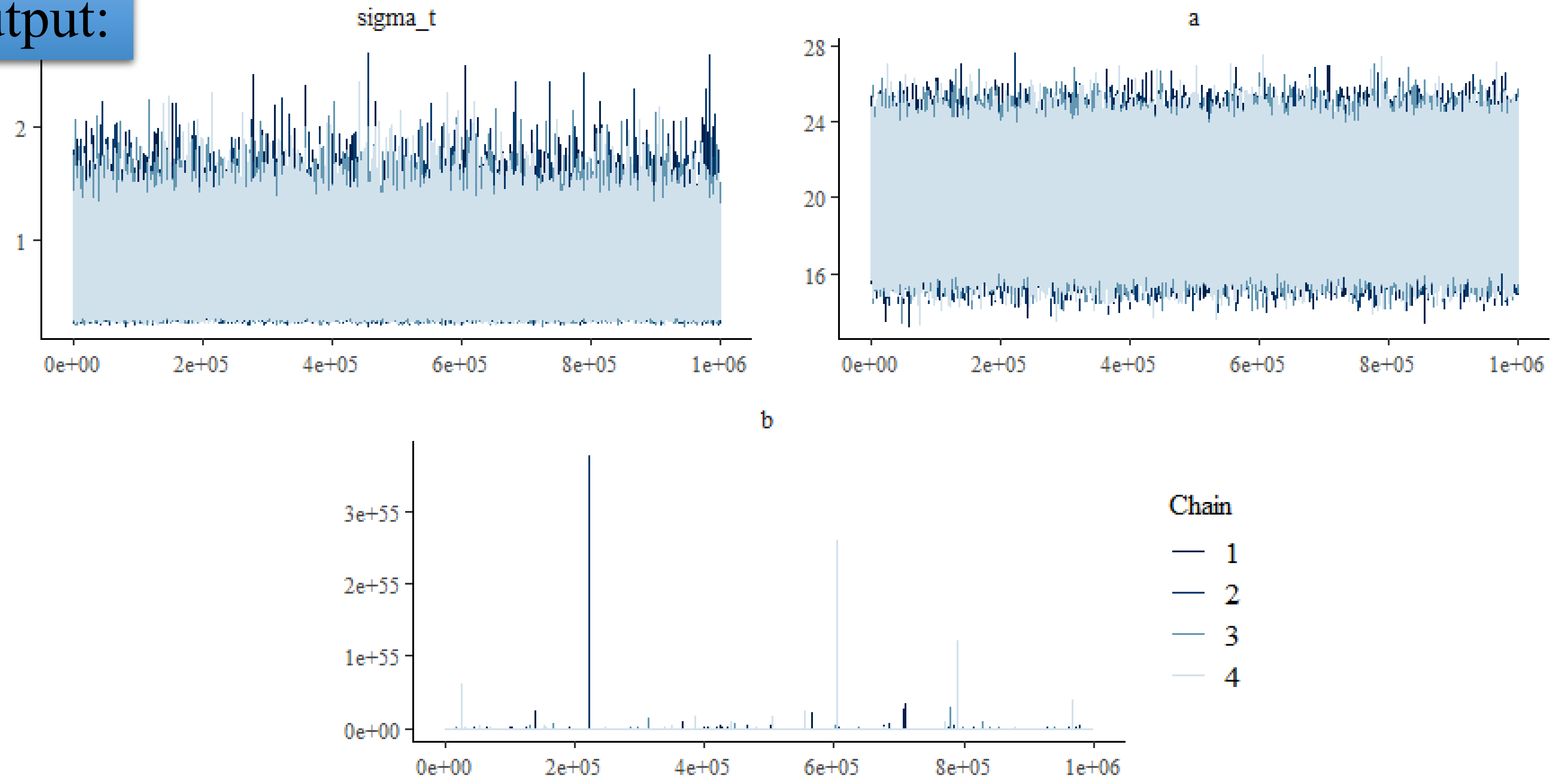
```
priorsALT <- c("lognormal(-0.715395,0.358001)",
               "lognormal(2.9654,0.10560)",
               "lognormal(96.2361,8.57329)")
```

Input:

```
lifestress.BAYESest(est_prior,ls="InversePower",dist="Lognormal",timeF_new,stressF_new,timeRc_new,
                    stressRc_new,0.95,priorsALT,100000,1000,4)
```

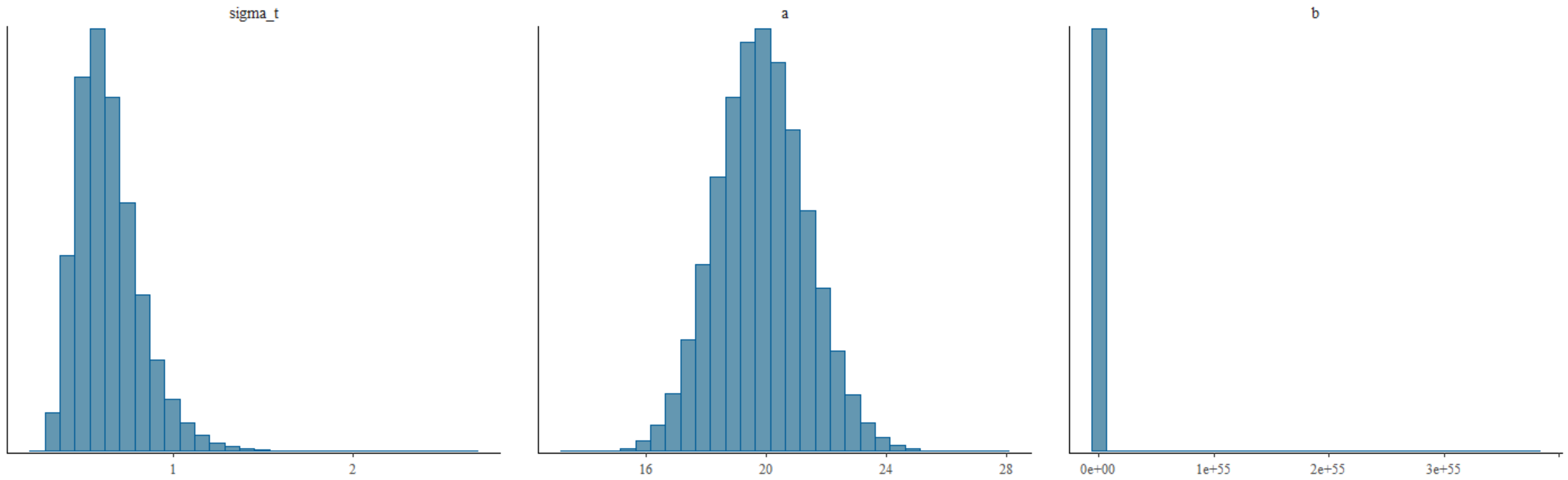
Accelerated Life Testing: ALT Tool Demo

Output:



Accelerated Life Testing: ALT Tool Demo

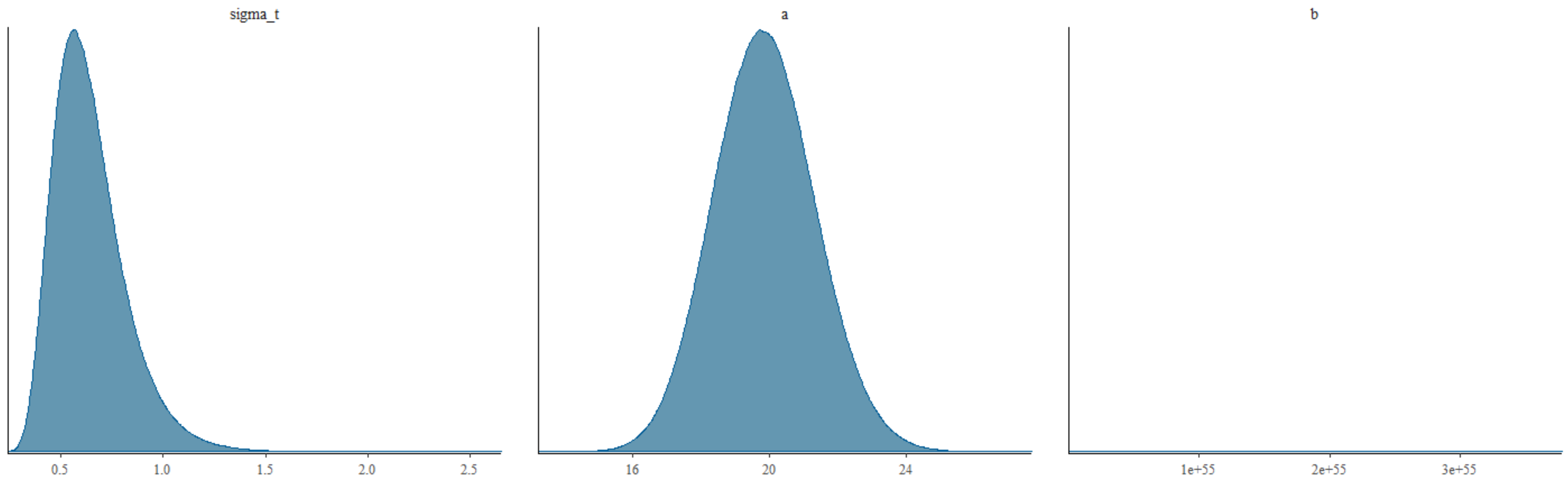
Output:



	Mean	Standard Deviation	Lower 95%	Median	Upper 95%
$\hat{\sigma}_t$	0.64482	0.17834	0.38156	0.61640	1.07084
$\hat{a}$	19.853	1.4819	17.023	19.832	22.831
$\hat{b}$	7.211×10^{48}	1.286×10^{51}	5.439×10^{36}	4.856×10^{41}	9.047×10^{46}

Accelerated Life Testing: ALT Tool Demo

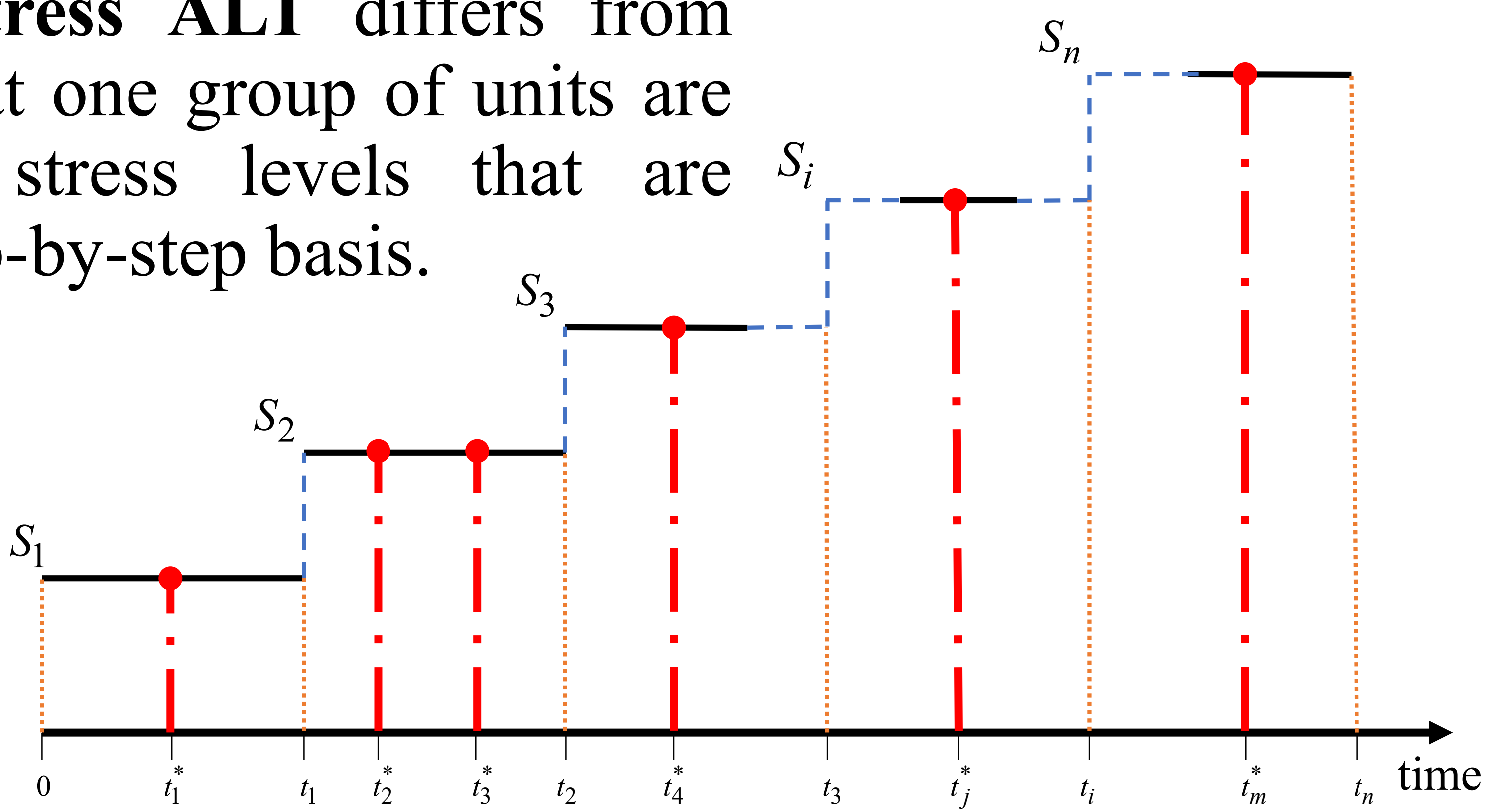
Output:



	Mean	Standard Deviation	Lower 95%	Median	Upper 95%
$\hat{\sigma}_t$	0.64482	0.17834	0.38156	0.61640	1.07084
$\hat{a}$	19.853	1.4819	17.023	19.832	22.831
$\hat{b}$	7.211×10^{48}	1.286×10^{51}	5.439×10^{36}	4.856×10^{41}	9.047×10^{46}

Step-Stress Accelerated Life Testing: Overview

- **<u>Definition</u>: Step-Stress ALT** differs from standard ALT in that one group of units are tested at various stress levels that are accelerated on a step-by-step basis.



- $STEP_i$ – Occurs between times t_{i-1} and t_i
- t_j^* - Time of unit failure j
- t_j^{*+} - Time of unit censoring j

Step-Stress Accelerated Life Testing: Step-Stress ALT Analyzer Tools

Least-Squares Step-Stress Estimator Tool

```
stepstress.LSQest(data,stepstresstable,ls,dist,pp,xlabel)
```

Maximum Likelihood Step-Stress Estimator Tool

```
stepstress.MLEest(pt_est,dat,stepstresstable,ls,dist,confid,sided)
```

Bayesian Step-Stress Estimator Tool

```
stepstress.BAYESest(pt_est,dat,stepstresstable,ls,dist,confid,priors,nsamples,burnin,nchains)
```

NEW INPUT VARIABLES FOR STEP-STRESS TABLE

- "*stepstresstable*" – a table that defines the step-stress test conditions where: the *First column(s)* is(are) the stress (or stresses in a dual case) per step and the *Last column* is the time duration per step

Step-Stress Accelerated Life Testing: Step-Stress ALT Tool Demo

Example: Let's demonstrate a full Step-Stress ALT analysis using these tools. Start with the following: The wear life of a bearing N (in cycles) can be defined by the maximum shear stress model,

$$N = C\left(\frac{\tau_{yp}}{\tau_{\max}}\right)^{n}$$

- C and n – model parameters
- $\tau_{\max}$ – maximum shear stress in surface vicinity
- τ_{yp} – material shear stress yielding point

A set of bearings designed to operate at maximum shear stress $\tau_{\max}$ of 200 psi with a material shear stress yielding point τ_{yp} of 1400 psi are subjected to three-step step-stress tests.

Step #	Maximum Shear Stress, $\tau_{\max}$ (psi)	Test Period (cycles)
1	250	2500
2	750	1000
3	1500	200

Step-Stress Accelerated Life Testing: Step-Stress ALT Tool Demo

Two tests were performed on an initial batch of bearings,

	Failure or Right Censored Times (cycles)
Failed	2800, 3100, 3300, 3520, 3600, 3660
Censored	3700+

And again on a prototype grade set of bearings.

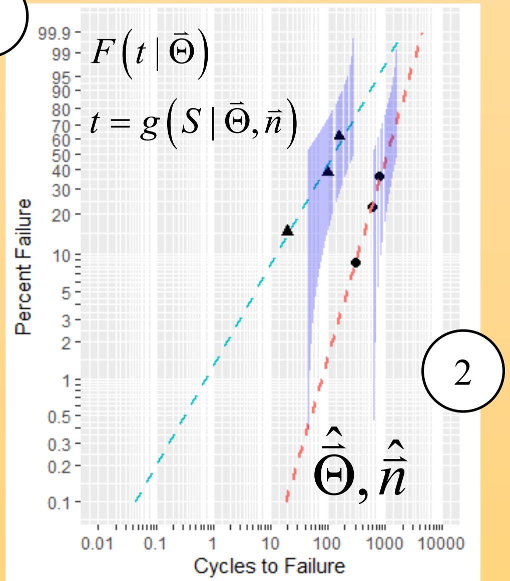
	Failure or Right Censored Times (cycles)
Failed	2000,2530, 3400, 3580, 3600, 3600, 3650
Censored	3699+, 3699+, 3699+

Step-Stress Accelerated Life Testing: Step-Stress ALT Tool Demo

<u>STEP 1</u>: LSQ estimation

- Based on an iterative optimization procedure

1



Initialize LSQ estimates

$k = 1$

2

Time equivalent times

$$t_j^e = t_j + \sum_{i<j} \frac{t_i}{AF_{i \to j}}$$

5

$k = k + 1$

4

$\bar{n}_{i \to j}^{adj(k)}$

2

t_i^e, t_j^e

3

$$AF_{i \to j}^{\mathrm{adj}(k)} = \frac{D\%_{fail,j}}{D\%_{fail,i}} \times AF_{i \to j} = \frac{D\%_{fail,j}}{D\%_{fail,i}} \times \frac{t_i^e}{t_j^e}$$

Step-Stress Accelerated Life Testing: Step-Stress ALT Tool Demo

- Setup life-stress model as *inverse power law*

$$N = C\left(\frac{\tau_{yp}}{\tau_{\max}}\right)^{n} = C\left(\frac{\tau_{\max}}{\tau_{yp}}\right)^{-n} \Leftrightarrow l(S) = bS^{-a}$$

- Redefine the step-stress table stress using *maximum shear stress over maximum yield point shear stress ratio* as stress S

$$S = \frac{\tau_{\max}}{\tau_{yp}},\ a = n,\ b = C$$

Step #	Maximum Shear Stress, $\tau_{\max}$ (psi)	Shear stress ratio	Test Period (cycles)
1	250	0.1786	2500
2	750	0.5357	1000
3	1500	1.0714	200

Step-Stress Accelerated Life Testing: Step-Stress ALT Tool Demo

- Upload (or form) the data tables and test to see which life distribution is best suited for step-stress analysis

Define:

```
datStepStress_Bearing_Wear_Prior<-read.csv("https://raw.githubusercontent.com/Center-for-Risk-and-Reliability/RMT/main/CSVExampleData/StepStressExample_Wear_Cycle_Prior.csv")

tableStepStress_Bearing_Wear_Prior<-read.csv("https://raw.githubusercontent.com/Center-for-Risk-and-Reliability/RMT/main/CSVExampleData/StepStressTableExample_Wear_Cycle_Prior.csv")
```

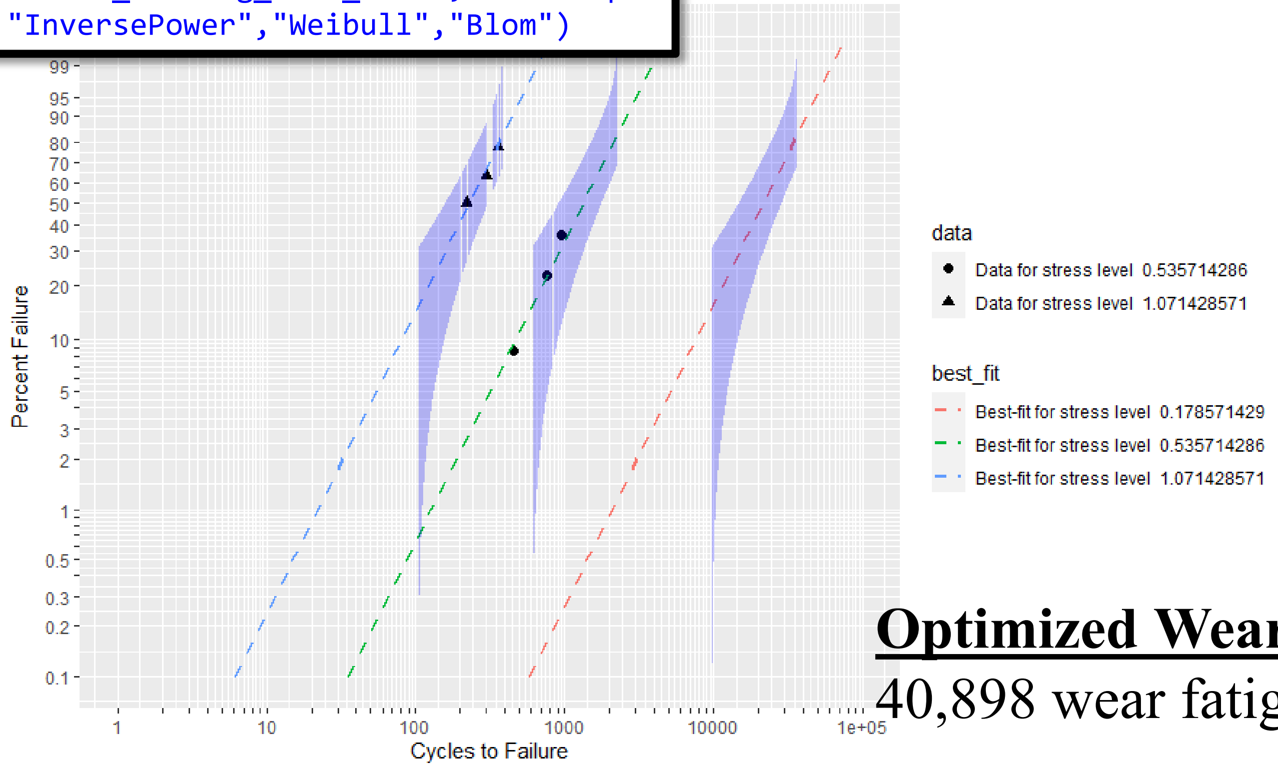
- Between Weibull and Lognormal the data best showed that *Weibull* had the best fit by coefficient of determination

- We go with we go with the *Weibull-Inverse-Power model*

Step-Stress Accelerated Life Testing: Step-Stress ALT Tool Demo

Input:

```
stepstress.LSQest(datStepStress_Bearing_Wear_Prior,tableStepStress_Bearing_Wear_Prior,"InversePower","Weibull","Blom")
```



Output:

$\hat{\beta} = 1.810$

$\hat{n} = 2.530$

$\hat{C} = 334.701$

<u>Optimized Wear Life LSQ estimate</u>:
40,898 wear fatigue cycles

Step-Stress Accelerated Life Testing: Step-Stress ALT Tool Demo

STEP 2: MLE estimation

- Operates on step-time equivalence such that…

$$F\left(t_i, S_i\right) = F\left(t_i^e, S_{i+1}\right)$$

- Relates time at Step i t_i to equivalent time t_i^e as seen from Step $i + 1$. Or,

$$t_{i-1}^e = \left(t_{i-1} - t_{i-2} + t_{i-2}^e\right) AF_{i-1\to i}^{-1}$$

Step-Stress Accelerated Life Testing: Step-Stress ALT Tool Demo

Input:

```
stepstress.MLEest(c(1.810,2.530,334.701),datStepStress_Bearing_Wear_Prior
,tableStepStress_Bearing_Wear_Prior,"InversePower","Weibull",confid =
0.9)
```

Output:

	$\hat{\beta}$	$\hat{n}$	$\hat{C}$
Point Estimate	1.859	3.053	304.858
Standard Error	0.489	0.616	85.013
Lower 90%	0.591	0.370	103.078
Upper 90%	5.845	5.7436	901.632

Wear Life MLE Mean estimate:
103,054 wear fatigue cycles

Step-Stress Accelerated Life Testing: Step-Stress ALT Tool Demo

STEP 3: Bayesian updating

- Bayesian estimation takes a simpler likelihood form where all failure and censored data are computed as seen from the last step

$$\ell = \prod_{i=1}^{n} f_N\left(x^*_{f_{N1}}, x^*_{f_{N2}}, \ldots x^*_{f_{Nn}} \mid \vec{\Theta}\right) \times$$

$$\prod_{j=1}^{m} R_N\left(x^*_{c_{N1}}, x^*_{c_{N2}}, \ldots x^*_{c_{Nm}} \mid \vec{\Theta}\right)$$

where $x^*_{f_{Nn}}$ is the n-th number of failure and $x^*_{c_{Nm}}$ is the m-th censored unit

Step-Stress Accelerated Life Testing: Step-Stress ALT Tool Demo

- Again, obtain prior data from the MLE output

Define:

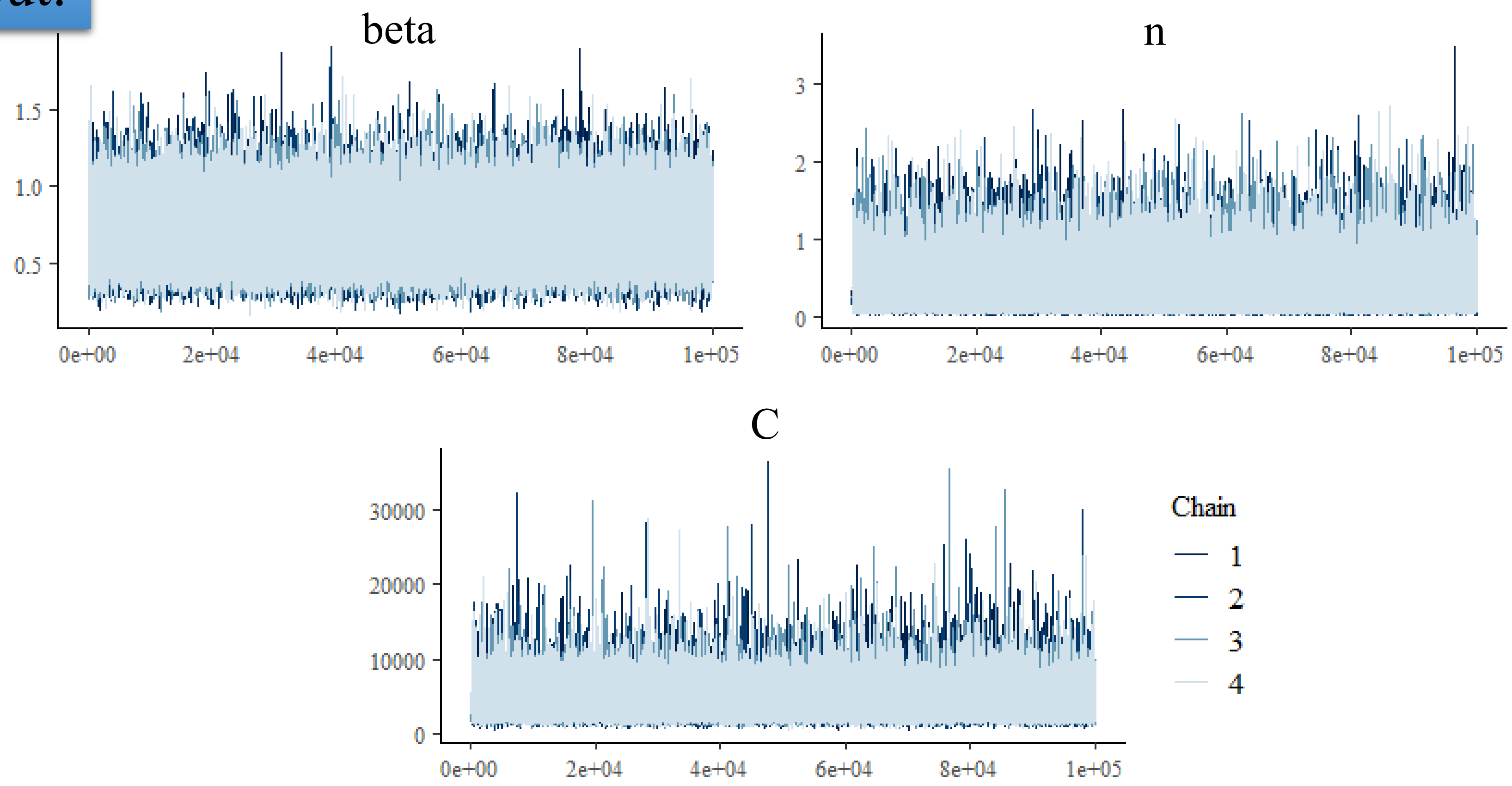
```
priorsStepStress <- c("lognormal(0.620,0.484)",
                      "lognormal(0.361,0.281)",
                      "lognormal(5.720,4.463)")
```

Input:

```
stepstress.BAYESest(pt_est,datStepStress_Bearing_Wear_Posterior,tableStepStress_Bearing_Wear_Prior
,"InversePower","Weibull",0.95, priorsStepStress,100000,1000,4)
```

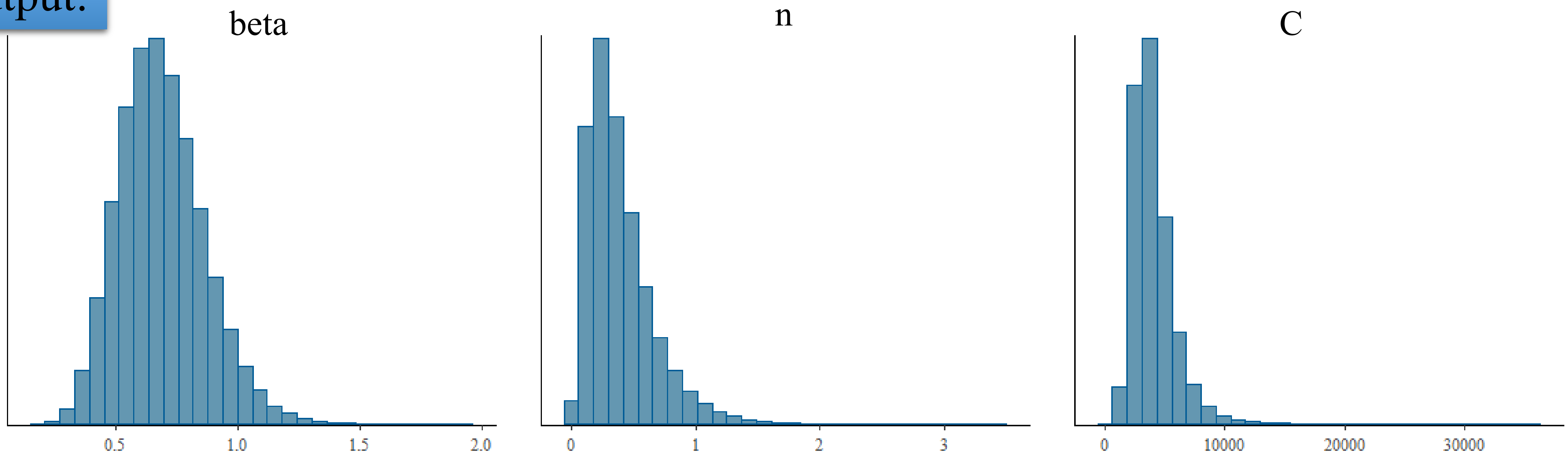
Step-Stress Accelerated Life Testing: Step-Stress ALT Tool Demo

Output:



Step-Stress Accelerated Life Testing: Step-Stress ALT Tool Demo

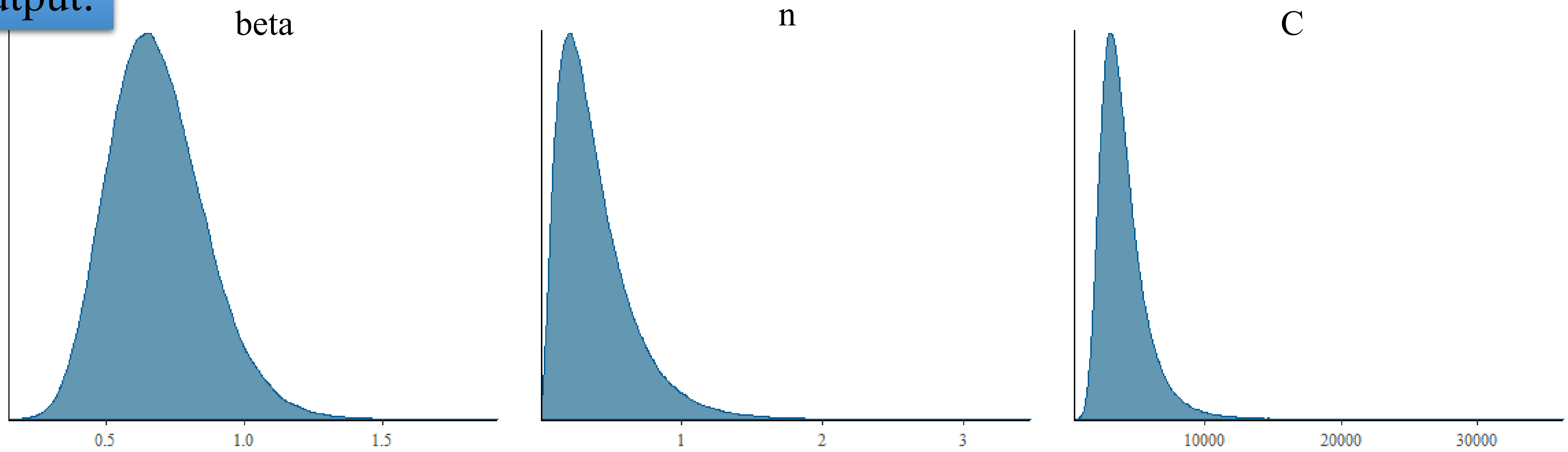
Output:



	Mean	Standard Deviation	Lower 95%	Median	Upper 95%
$\hat{\beta}$	0.688	0.173	0.390	0.674	1.068
$\hat{n}$	0.392	0.266	0.0724	0.318	1.079
$\hat{C}$	3996	1743	1772	3631	8374

Step-Stress Accelerated Life Testing: Step-Stress ALT Tool Demo

Output:



	Mean	Standard Deviation	Lower 95%	Median	Upper 95%
$\hat{\beta}$	0.688	0.173	0.390	0.674	1.068
$\hat{n}$	0.392	0.266	0.0724	0.318	1.079
$\hat{C}$	3996	1743	1772	3631	8374

Toolset 4:

Application, Modification, and Testing for the Reliability Modeling Toolkit

Application, Modification, and Testing for the RMT: Overview

- As an open-source toolkit meant to be applied and potentially modified by others, the RMT includes a testing framework to allow users to perform validation of code functionality and modifications

- The basis of the code testing is the development of test cases for different code functions and the encoding of those test cases into a testing framework called "testthat"

Application, Modification, and Testing for the RMT: Implementation

Expect function, the key component of a test from “testthat”

```
expect_equal(function(input)), expected_output)
```

- “*function()*” – The function from the RMT under test
- “*input*” – input data the function is expected to process, preferably representing some edge case
- “*expected_output*” – the expected and separately validated correct output from the function and input

Application, Modification, and Testing for the RMT: Implementation

- Ideally several tests are made for each function, covering test cases that could be broken by modifications to the code or other errors

- The encoding of test cases imbues the RMT with a form of self validation

- If a function is modified in error or a 3rd party library/dependency's behavior changes in an undesirable way, for example from an update, a broken test case allows a user or code developer to catch and hopefully correct the problem before erroneous results are generated

Application, Modification, and Testing for the RMT: Execution

In order to run tests, the following should be executed in the R shell:

1. Install the “testthat” library, and load “testthat” and the RMT

```
install.packages("testthat")
library(testthat)
library(reliabilityRMT)
```

2. Run all unit tests:

```
test_package("reliabilityRMT")
```

3. If all tests pass, information like the following is printed to the console:

```
[ FAIL 0 | WARN 0 | SKIP 0 | PASS 45 ]
```

Application, Modification, and Testing for the RMT: Execution

4. If any tests do not pass, information like the following is printed to the console to inform the user which test(s) failed and for what reason, in order to assist with troubleshooting:

```
[ FAIL 1 | WARN 0 | SKIP 0 | PASS 44 ]
══ Failed tests ════════════════════════════════════════════════════════════════
── Failure (test-rankadj.R:2:3): rank adjustment with duplicates is correct ────
rankadj(c(70, 71, 75, 78, 78, 89), c(80, 80, 84)) (`actual`)
not equal to c(1, 2, 3, 4, 7) (`expected`).

  `actual[2:5]`: 2 3 4 6
`expected[2:5]`: 2 3 4 7
```

Closing Remarks

- Educational tools like those that make up the RMT can aid in bridging the connection between learning reliability methodology and applying and adjusting reliability methodology

- The RMT is continuously undergoing development, and the development staff is always seeking to bring more tools into the package for education and engineering field applications

Closing Remarks

- More information on the RMT is available on the Github site and R documentation

- For tips in the RMT's Bayesian applications, please also make use of RStan help guides

- Also feel free to contact the RMT developers for feedback and assistance

Contact Information

Reuel Smith

smithrc@umd.edu

Sherief Magdy Elsibaie

elsibaie@umd.edu

Mohammad Modarres

modarres@umd.edu

Acknowledgements

- We would like to thank the **Teaching Learning and Transformation Center (TLTC)** of the UMD for funding the project in its initial year (TLTC Grant #2938082)

References

- R. Smith, *Reliability Modeling Toolkit*, R Package Version 1.2.6, 2023. https://github.com/Center-for-Risk-and-Reliability/RMT
- R. Smith, M. Modarres, "Tools for Analysis of Accelerated Life and Degradation Test Data," *IEEE Accelerated Stress Testing & Reliability Conference*, (Sept) 2016.
- V. Krivstov, *RARE*, University of Maryland, 2011.
- M. Modarres, M. Amiri, C. Jackson. *Probabilistic Physics of Failure Approach to Reliability: Modeling, Accelerated Testing, Prognosis and Reliability Assessment*, Maryland, University of Maryland, 2017.
- Simplilearn. *What is R: Overview, its Applications and what is R used for?*, https://www.simplilearn.com/what-is-r-article, (7 Jun) 2023.
- J. Banantine, J. Corner, J. Handrock. *Fundamentals of Metal Fatigue Analysis*, New Jersey Prentice Hall, 1990.
- M.M. Khonsari, M. Amiri, *Introduction to Thermodynamics of Mechanical Fatigue*, CRC Press, Taylor & Francis Group, Boca Raton, FL., 2012.
- W. Ramberg, W.R. Osgood, "Description of stress-strain curves by three parameters," *Technical Note No. 902*, National Advisory Committee for Aeronautics, Washington D.C., 1943.
- L.F. Coffin Jr., "A Study of the Effect of Cyclic Thermal Stresses on a Ductile Metal," *Transactions of ASME*, vol. 76, 1954, pp. 931-950.
- S.S. Manson, "Behavior of Materials Under Conditions of Thermal Stress," *Heat Transfer Symposium*, University of Michigan Engineering Research Institute, 1953, pp. 9–95.

References

- A. Palmgren, "Durability of Ball Bearings," *ZVDI*, Vol. 68, No. 14, 1924, pp. 339-341.
- M. A. Miner, "Cumulative Damage in Fatigue," *Journal of Applied Mechanics*, Vol. 12, Trans. ASME Vol. 67, 1945, pp. A159-A164.
- J. Morrow, “Fatigue properties of metals, section 3.2.” *Fatigue Design Handbook*, No. AE-4. SAE, Warrendale, PA. 1968.
- S. Kwofie and N. Rahbar, "A fatigue driving stress approach to damage and life prediction under variable amplitude loading," *International Journal of Damage Mechanics*, vol. 22 (3), 2012, pp. 393–404.
- L. Molent, R. Jones, "The influence of cyclic stress intensity threshold on fatigue life scatter," *International Journal of Fatigue*, vol. 82, 2016, pp. 748–756.
- P. Paris, F. Erdogan, "A Critical Analysis of Crack Propagation Laws," *Journal of Basic Engineering*, vol. 85(2), 1963, pp. 528-534.
- R.J. Allen, G.S. Booth, T. Jutla, "A review of fatigue crack growth characterization by Linear Elastic Fracture Mechanics (LEFM). Part II – Advisory documents and applications within National Standards," *Fatigue & Fracture of Engineering Materials and Structures*, vol. 11 (2), 1988, pp. 71–108.
- K. Walker, "The Effect of Stress Ratio During Crack Propagation and Fatigue for 2024-T3 and 7075-T6 Aluminum," *Effects of Environment and Complex Load History on Fatigue Life ASTM STP 462*, 1970, pp. 1-14.
- R. G. Forman, V. E. Kearney, R. M. Eagle, "Numerical Analysis of Crack Propagation in Cyclic Loaded Structures," *Journal of Basic Engineering*, vol. 89, 1967, pp. 459-464.
- A. J. McEvily, J. Groeger, "On the threshold for fatigue crack growth," *Advances in Research on the Strength and Fracture of Materials*, Elsevier, 1978, pp. 1293–1298,

References

- R.G. Forman, V. Shivakumar, W. Cardinal, L.C. Williams, P.C. McKeighan, "Fatigue Crack Growth Database for Damage Tolerance Analysis," *DOT/FAA/AR-05/15*, 2005,
- J. F. Archard, "Contact and rubbing of flat surfaces," *J. Appl. Phys*., vol. 24, 1953, pp. 981-988.
- F.R. Larson, J. Miller, "A Time-Temperature Relationship for Rupture and Creep Stresses," *Transactions ASME*, vol. 74, 1952, pp. 765-771.
- S.S. Manson, A.M. Haferd, "A Linear Time-Temperature Relation for Extrapolation of Creep and Stress-Rupture Data," *NASA-TN-2890*, 1953.
- O.D. Sherby, R.L. Orr, J.E. Dorn "Creep correlations of metals at elevated temperatures," *Trans. AIMME*, vol. 200, 1954, pp. 71-80.
- H. Al-Ethari, *Lecture #19 Creep in Metals*. 2023, URL https://www.uobabylon.edu.iq/eprints/publication_4_29484_1037.pdf
- Y. Kondo, "Prediction of Fatigue Crack Initiation Life Based on Pit Growth," *Corrosion*, Vol. 45, No. 1, 1989, pp. 7-11.
- Y. Kondo, R.P. Wei, "Approach on Quantitative Evaluation of Corrosion Fatigue Crack Initiation Condition," *International Conference on Evaluation of Materials Performance in Severe Environments, EVALMAT 89*, Vol. 1, Iron and Steel Institute of Japan, 1989, pp. 135-142.
- G. Blom, *Statistical Estimates and Transformed Beta Variables*, Wiley, New York, N.Y., 1958, pp.68–75 and pp.143–146.
- W. Weibull, (1939) "A statistical Theory of Strength of Materials," *Ingeniors Vetenskaps*, Academy Handlingar, Stockholm, vol. 151, 1939, pp 1–45.

References

- A. Benard, E.C. Bos-Levenbach, "The Plotting of Observations on Probability," *Statististica Neerlandica*, vol. 7, 1953, pp. 163–173.
- A. Hazen, "Storage to be provided in impounding reservoirs for municipal water supply," *Transactions of the American Society of Civil Engineers*. vol. 1308 (77), 1934, pp. 1547–1550.
- L.R. Beard, "Statistical Analysis in Hydrology," *Transactions of the American Society of Civil Engineers*, vol. 108, 1943, pp. 1110–1160.
- J.W. Tukey, "The Future of Data Analysis," *Annals of Mathematical Statistics*, vol. 33(1), 1962, pp. 21–24.
- I.I. Gringorten, "A Plotting Rule for Extreme Probability," *Journal of Geophysical Research*, vol. 68, 1963, pp. 813–-814
- W. Nelson, "Theory and Applications of Hazard Plotting for Censored Failure Data," *Technometrics*, vol. 42(1), 1972, pp. 12–25.
- E.L. Kaplan, P. Meier, "Nonparametric estimation from incomplete observations". *J. Amer. Statist. Assoc*. vol. 53 (282), 1958, pp. 457–481.
- R, *Non-Linear Minimization*, stats (version 3.6.2), 2023.
- T. Bayes, "An Essay Towards Solving a Problem in the Doctrine of Chances," *Philosophical Transactions*, 53, 1763, pp. 370–418.
- B. Carpenter "The fundamental abstractions underlying BUGS and Stan as probabilistic programming languages." *Statistical Modeling, Causal Inference, and Social Science*. Accessed December 5, 2022. https://statmodeling.stat.columbia.edu/2017/09/07/fundamental-abstractions-underlying-bugs-stan-probabilistic-programming-languages/.

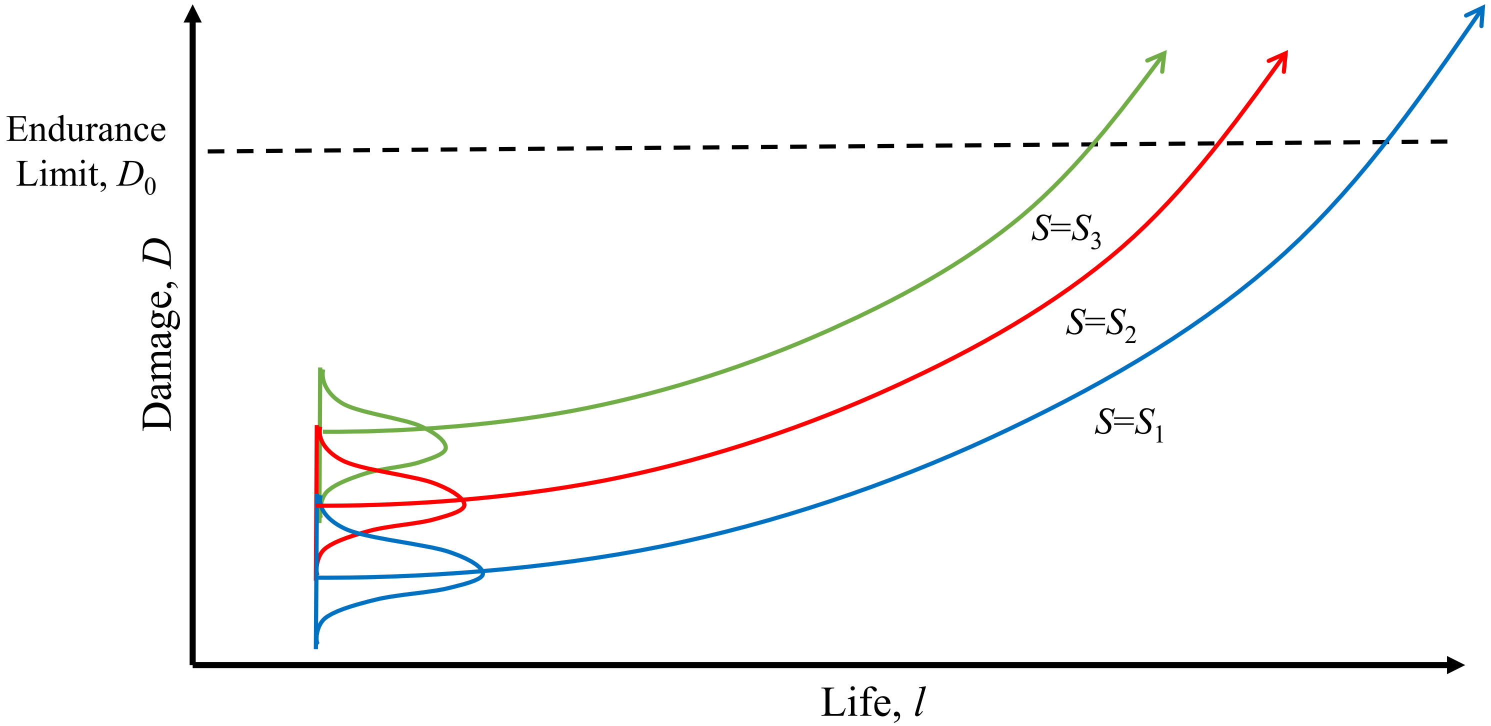
References

- M. D. Homan and A. Gelman, “The No-U-turn sampler: adaptively setting path lengths in Hamiltonian Monte Carlo,” *J. Mach. Learn. Res.*, vol. 15, no. 1, pp. 1593–1623, Jan. 2014.
- S. Duane, A. D. Kennedy, B. J. Pendleton, D. Roweth, “Hybrid Monte Carlo,” *Physics Letters B*, vol. 195 (2) (1987) 216–222. doi:10.1016/0370-2693(87)91197-X
- W. Nelson, "Accelerated Life Testing - Step-Stress Models and Data Analyses," *IEEE Transactions on Reliability*, Vol. R-29(2), 1980, pp. 103-108.
- F. Mitsuo, *Reliability and Degradation of Semiconductor Lasers and LEDs*, Artech House, 1991
- M.S. Hamada, A. Wilson, R. Shane, H.C. Martz. *Chapter 8: Using Degradation Data to Assess Reliability. Bayesian Reliability*, Springer, 2008.
- W. Bissi, A. G. Serra Seca Neto, and M. C. F. P. Emer, “The effects of test driven development on internal quality, external quality and productivity: A systematic review,” *Information and Software Technology*, vol. 74, pp. 45–54, Jun. 2016, doi: 10.1016/j.infsof.2016.02.004.
- H. Munir, M. Moayyed, and K. Petersen, “Considering rigor and relevance when evaluating test driven development: A systematic review,” *Information and Software Technology*, vol. 56, no. 4, pp. 375–394, Apr. 2014, doi: 10.1016/j.infsof.2014.01.002
- D. Fucci, H. Erdogmus, B. Turhan, M. Oivo, and N. Juristo, “A Dissection of the Test-Driven Development Process: Does It Really Matter to Test-First or to Test-Last?,” *IEEE Transactions on Software Engineering*, vol. 43, no. 7, pp. 597–614, Jul. 2017, doi: 10.1109/TSE.2016.2616877.
- R, *devtools*, 2023.

EXTRA SLIDES

Accelerated Degradation Testing: Overview

- **Definition**: **Accelerated degradation testing** (ADT) is a methodology where nominal life is obtained from damage or degradation modeling (at accelerated stress levels) using a given endurance limit D_0 as measure of end of life. ADT is handy because it can be mostly a nondestructive form of testing.



- Many damage-life relations $D(l)$ are available for modeling such data. The RMT considers at least eight such models.

Accelerated Degradation Testing: Overview

Damage-Life Model	**Damage-Life function, *D*(*l*)**	**Life-Stress Model**	**Life-Stress function, *l*(*S*)**
Linear	$D(l) = a + bl$	Logarithmic	$D(l) = \dfrac{1}{1 + bl^a}$
Exponential	$D(l) = b\exp(al)$	Lloyd-Lipow	$D(l) = a - \dfrac{b}{l}$
Square-Root	$D(l) = (a + bl)^2$	Mitsuo	$D(l) = a + b\ln(l)$
Power	$D(l) = bl^a$	Hamada et. al.	$D(l) = \left\{1 + \beta_1 \left[l\exp\left[\beta_3 11605\left(\dfrac{1}{T_{use}} - \dfrac{1}{T}\right)\right]\right]^{\beta_2}\right\}^{-1}$

- $a, b, \beta_1, \beta_2, \beta_3$ – model parameters
- T_{use} – use level temperature (K)
- T – temperature (K)

Accelerated Degradation Testing: ADT Analyzer Tools

Least-Squares Accelerated Degradation Testing Estimator Tool

```
adt.full.LSQ(dat,lifedam,D0,Tuse)
```

- "*dat*" – Tabular degradation data
- "*lifedam*" – damage-life model
- "*D0*" – given endurance limit
- "*Tuse*" – Use level of temperature (only applied for Hamada et. al. model)

Accelerated Degradation Testing: ADT Analyzer Tools

Remember: Degradation data "*dat*" follows a specific column-by-column order of entry

- Column 1 – Time
- Column 2 – Degradation amount
- Column 3 – Unit ID
- Column(s) 4 and up – Stress (or stresses)

Accelerated Degradation Testing: ADT Analyzer Tools

Example: Consider this sliding wear test data where we are given an endurance limit of 50 microns

	Unit 1	Unit 2	Unit 3	Unit 4	Unit 5	Unit 6	Unit 7	Unit 8	Unit 9	Unit 10	Unit 11	Unit 12
Cycles (x100)	Weight lost from wear (μg)											
2	3.2	2.7	2.1	2.6	7.5	7.5	7	7.8	12.5	11	13	11.7
5	4.1	3.4	2.7	3.5	7.8	8.1	8.9	8.9	15.4	13.9	15.1	13.7
10	4.5	3.8	3.1	4	8.2	9.8	9.4	10	17.2	16.1	18.6	16.7
20	4.7	3.9	3.3	4	10.6	10.9	11.1	11.5	20.5	18.6	20.2	17.5
50	5.8	5.4	4	5.2	12.6	14.8	12.4	13.7	24.1	22.2	23.9	22.3
100	6.8	5.7	4.6	6.1	13.3	16.1	13.5	16.2	27	27.8	29.7	25.3
200	7.7	6.3	5.7	6.7	12.9	17.3	16.7	16.2	29.4	31	31.5	32
500	9.6	8.4	6.6	8.5	14.8	20.2	17.3	21	37.9	36.6	39.6	38.2
Applied Weight (g)	10	10	10	10	50	50	50	50	100	100	100	100

Accelerated Degradation Testing: ADT Analyzer Tools

A1 | Cycles (x100)

	A	B	C	D
1	Cycles (x1	Weight Lo	Unit Numl	Applied Weight (g)
2	2	3.2	Unit 1	10
3	5	4.1	Unit 1	10
4	10	4.5	Unit 1	10
5	20	4.7	Unit 1	10
6	50	5.8	Unit 1	10
7	100	6.8	Unit 1	10
8	200	7.7	Unit 1	10
9	500	9.6	Unit 1	10
10	2	2.7	Unit 2	10
11	5	3.4	Unit 2	10
12	10	3.8	Unit 2	10
13	20	3.9	Unit 2	10
14	50	5.4	Unit 2	10
15	100	5.7	Unit 2	10
16	200	6.3	Unit 2	10
17	500	8.4	Unit 2	10
18	2	2.1	Unit 3	10
19	5	2.7	Unit 3	10
20	10	3.1	Unit 3	10
21	20	3.3	Unit 3	10
22	50	4	Unit 3	10
23	100	4.6	Unit 3	10
24	200	5.7	Unit 3	10
25	500	6.6	Unit 3	10
26	2	2.6	Unit 4	10
27	5	3.5	Unit 4	10
28	10	4	Unit 4	10
29	20	4	Unit 4	10
30	50	5.2	Unit 4	10
31	100	6.1	Unit 4	10
32	200	6.7	Unit 4	10
33	500	8.5	Unit 4	10
34	2	7.5	Unit 5	50

Degradation_Data_1_Mass_Loss_by

Ready

- ***Remember***: Degradation data is almost always large, so entry by way of CSV is always encouraged when using the RMT

Define:

```
datADTexample1 <- read.csv("https://raw.githubusercontent.com/Center-for-Risk-and-Reliability/RMT/main/CSVExampleData/Degradation_Data_1_Mass_Loss_by_Weight_gms_Example_5_2.csv")
```

- ***Remember ALSO***: Like ALT, we can't just pick a damage-life model for data just like that

Accelerated Degradation Testing: ADT Analyzer Tools

Accelerated Degradation Testing Rank System Tool

```
adt.rank(dat)
```

- "*dat*" – Tabular degradation data

- Tool provides a unit-by-unit rank check of all relevant models as well as an average rank

Input:

```
adt.rank(datADTexample1)
```

Output:

```
              Average Rank
Linear                   3
Exponential              5
Square-Root              4
Power                    1
Logarithmic              2
Lloyd-Lipow              6
```

- We'll go with the *Power damage-life model* for this example

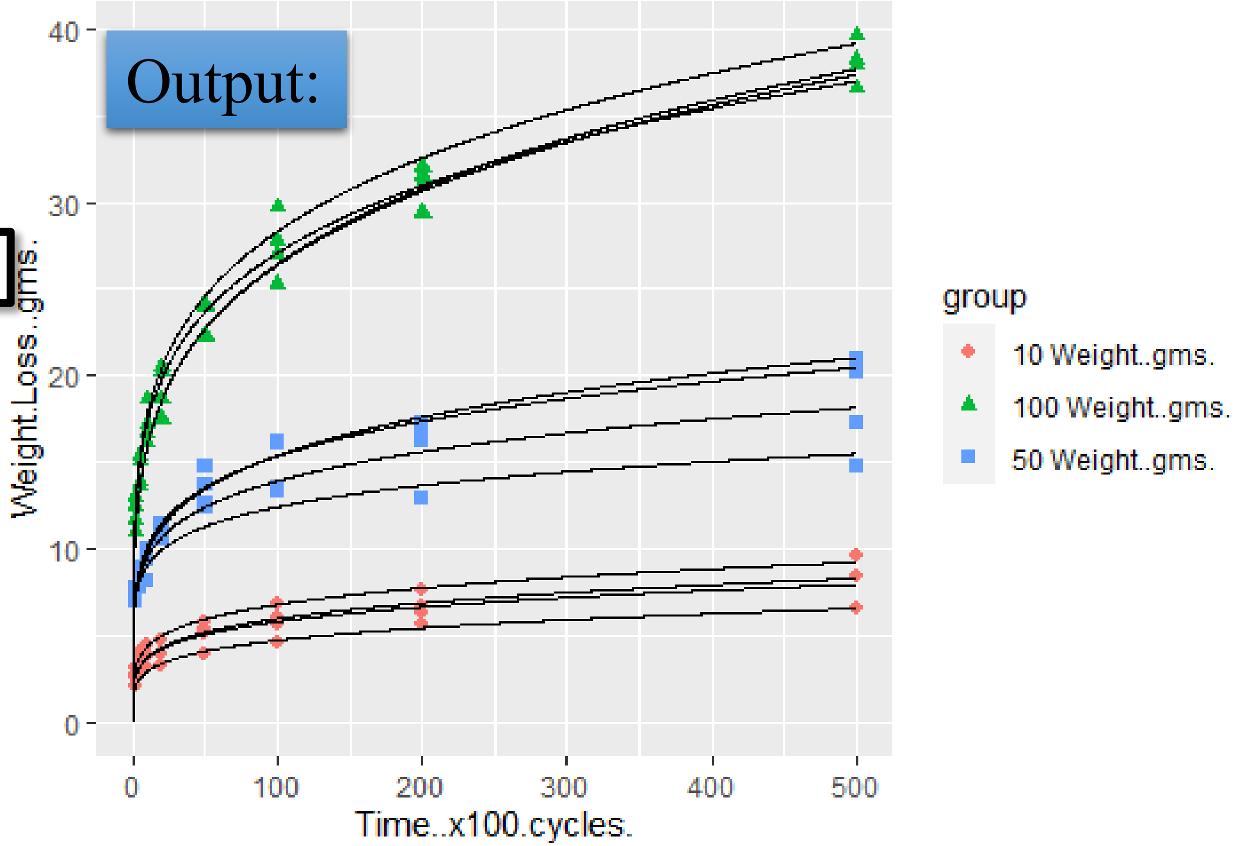
Accelerated Degradation Testing: ADT Analyzer Tools

- Proceed with the `adt.full.LSQ` tool,

Input:

```
adt.full.LSQ(datADTexample1,"Power",50)
```

Output:

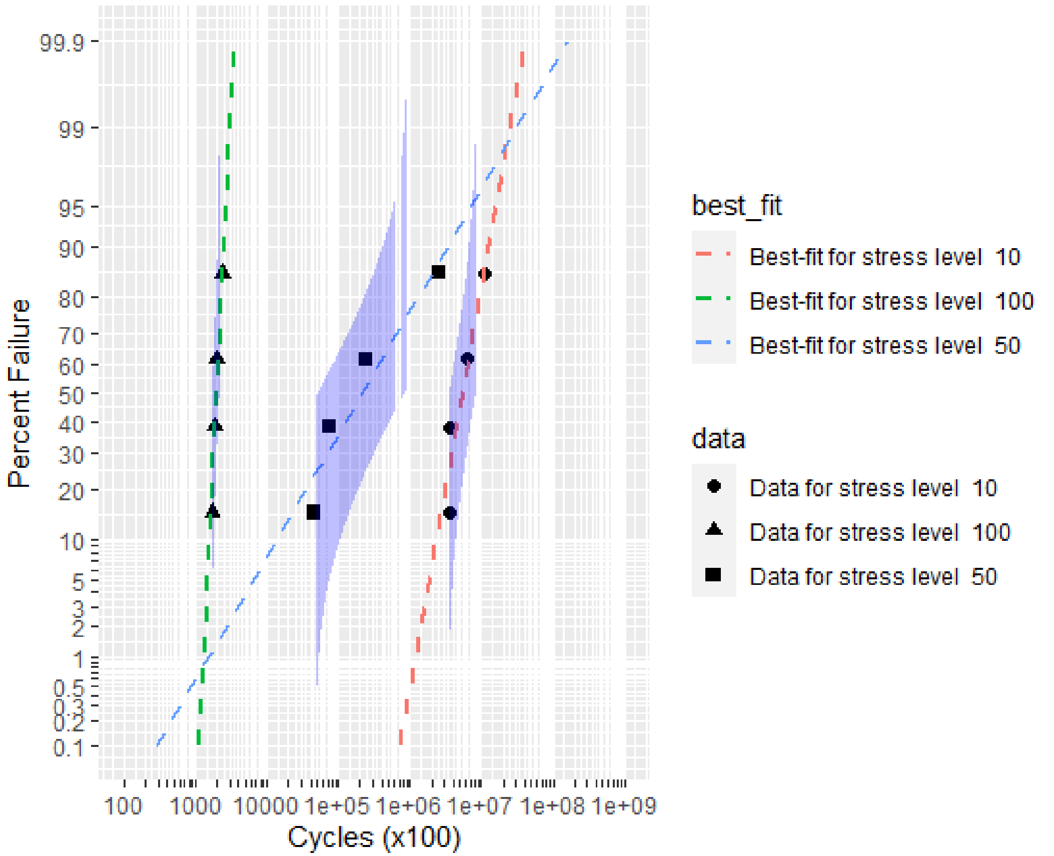


- Output includes unit-by-unit model parameters, pseudo-times, and coefficient of determination
- Also provides tabular output for use with probability plotting tools

Accelerated Degradation Testing: ADT Analyzer Tools

Output:

Unit #	Pseudo Failure Cycle (x100)	Applied Weight (microns)
Unit 1	3,637,648	10
Unit 2	6,281,169	10
Unit 3	11,294,602	10
Unit 4	3,602,944	10
Unit 5	2,571,094	50
Unit 6	44,446	50
Unit 7	237,904	50
Unit 8	75,202	50
Unit 9	2,387	100
Unit 10	1,815	100
Unit 11	1,688	100
Unit 12	1,929	100



Accelerated Degradation Testing: ADT Analyzer Tools

- Of course the MLE estimate is preferred, however for ADT analysis, the likelihood takes the following form that accounts for n units of degradation with m_i readings for each unit i,

$$\ell = \prod_{i=1}^{n} \int_{\Theta} \prod_{j=1}^{m_i} \frac{\phi\left(z_{ij}\right)}{\sigma_{\varepsilon}} f\left(\vec{\Theta} \mid \mu_{\vec{\Theta}}, \Sigma_{\vec{\Theta}}\right) d\vec{\Theta} \text{ where } z_{ij} = \frac{y_{ij} - D\left(l_{ij} \mid \vec{\Theta}\right)}{\sigma_{\varepsilon}}$$

where,

- $f\left(\vec{\Theta} \mid \mu_{\vec{\Theta}}, \Sigma_{\vec{\Theta}}\right)$ – joint distribution (such as a multivariate normal distribution)
- $\vec{\Theta}$ – damage-life model parameters as a vector
- $\mu_{\vec{\Theta}}$ – mean vector for parameters $\vec{\Theta}$
- $\Sigma_{\vec{\Theta}}$ – variance-covariance matrix for parameters $\vec{\Theta}$

Accelerated Degradation Testing: ADT Analyzer Tools

Maximum Likelihood Estimate Accelerated Degradation Testing Estimator Tool

```
adt.full.MLE(dat,lifedam,dist,D0,Tuse,confid,sided)
```

- "*dat*" – Tabular degradation data
- "*lifedam*" – damage-life model
- "*dist*" – named probability distribution ("Normal" or "Lognormal")
- "*D0*" – given endurance limit
- "*Tuse*" – Use level of temperature (only applied for Hamada et. al. model)
- "*confid*" – confidence bound between 0 and 1 (0.95 for 95% confidence)
- "*sided*" – confidence limits for parameters: two-sided, one-sided high, or one-sided low

Accelerated Degradation Testing: ADT Analyzer Tools

Example: Refine the LSQ estimate for this sliding wear test data using the MLE tool

	Unit 1	Unit 2	Unit 3	Unit 4	Unit 5	Unit 6	Unit 7	Unit 8	Unit 9	Unit 10	Unit 11	Unit 12
Cycles (x100)	Weight lost from wear (μg)											
2	3.2	2.7	2.1	2.6	7.5	7.5	7	7.8	12.5	11	13	11.7
5	4.1	3.4	2.7	3.5	7.8	8.1	8.9	8.9	15.4	13.9	15.1	13.7
10	4.5	3.8	3.1	4	8.2	9.8	9.4	10	17.2	16.1	18.6	16.7
20	4.7	3.9	3.3	4	10.6	10.9	11.1	11.5	20.5	18.6	20.2	17.5
50	5.8	5.4	4	5.2	12.6	14.8	12.4	13.7	24.1	22.2	23.9	22.3
100	6.8	5.7	4.6	6.1	13.3	16.1	13.5	16.2	27	27.8	29.7	25.3
200	7.7	6.3	5.7	6.7	12.9	17.3	16.7	16.2	29.4	31	31.5	32
500	9.6	8.4	6.6	8.5	14.8	20.2	17.3	21	37.9	36.6	39.6	38.2
Applied Weight (g)	10	10	10	10	50	50	50	50	100	100	100	100

Accelerated Degradation Testing: ADT Analyzer Tools

- Assuming a lognormal fit to the damage data, proceed with the MLE step

Input:

```
adt.full.MLE(datADTexample1,"Power", "Lognormal",50)
```

Output:

	$\hat{\sigma}_{\varepsilon}$	$\hat{a}$	$\hat{b}$
Point Estimate	0.638	0.191	5.43
Standard Error	0.0208	0.0106	0.224
Lower 90%	0.554	0.119	4.115
Upper 90%	0.735	0.263	7.182

$$\mu_{\vec{\Theta}} = \begin{bmatrix} 0.191 \\ 5.430 \end{bmatrix} \qquad \Sigma_{\vec{\Theta}} = \begin{bmatrix} 1.351\times10^{-3} & -2.534\times10^{-2} \\ -2.534\times10^{-2} & 0.6002 \end{bmatrix}$$

Accelerated Degradation Testing: ADT Analyzer Tools

Bayesian Accelerated Degradation Testing Updater Tool

```
adt.full.BAYES(pt_est,dat,lifedam,dist,D0,Tuse,confid,priors,nsamples,burnin,nchains)
```

- "*pt_est*" – vector of the initial parameter estimates
- "*dat*" – Tabular degradation data
- "*lifedam*" – damage-life model
- "*dist*" – Named probability distribution
- "*D0*" – given endurance limit
- "*Tuse*" – Use level of temperature (only applied for Hamada et. al. model)
- "*confid*" – confidence bound between 0 and 1 (0.95 for 95% confidence)
- "*nsamples*" – number of MCMC samples or iterations per chain
- "*burnin*" – number of initial MCMC iterations the throw out
- "*nchain*" – number of Markov chains