An Automated Script for the Software Failure and Reliability Assessment Tool (SFRAT)

Lance Fiondella
Department of Electrical and Computer Engineering
University of Massachusetts Dartmouth
e-mail: lfiondella@umassd.edu

February 17, 2019

Executive Summary

The Software Failure and Reliability Assessment Tool (SFRAT) automates the application of several software reliability growth models (SRGM). While a graphical user interface is the preferred method for many users to conduct software reliability analysis, this document describes a configurable script that further automates the application of SRGM. This document provides instruction on how to access, download, configure, and execute the script. To further familiarize the user with software reliability models and the inferences they enable, a verbose option is provided, which produces additional output to complement tables and figures with interpretations. The reports generated by the automated script can serve as an additional artifact to document testing progress at regular intervals.

Acknowledgment

This work was supported by the National Aeronautics and Space Administration (NASA) and SARP under Grant Number (#80NSSC18K0154) and NSF CAREER award (#1749635).

Contents

List of Figures

1 Introduction

Software reliability models were proposed over forty years ago. However their application has been limited because the mathematical learning curve and computational requirements deter most potential users. Some researchers have developed computational tools to overcome this limitation, but their maintenance has been complicated by lack of access to source code. To address these challenges, the SFRAT (Software Failure and Reliability Assessment Tool) was developed as an open source and open architecture platform for users to download and researcher to contribute additional models and functionality. The tool was developed in the R statistical programming language and possesses a simple graphical user interface. While a graphical user interface is the method choice for many users to utilize SFRAT functionality, frequent use of the SFRAT can benefit from further automation. This document describes a configurable automated report generation script for the SFRAT, which can greatly reduce the time dedicated working with the tool. Instead, the script can be modified to specified what the users would do within the graphical user interface during a typical session with the software and run this script to produce a pdf document suitable for reporting or inclusion in a test database as an artifact documenting testing progress at periodic intervals. A verbose option is provided. When enabled, it provides additional interpretation of results, in order to help new users understand SFRAT output more easily.

The remainder of the manual is organized as follows: Section 3 describes the system requirements, required packages, as well as input data format before executing the script. Section 4 provides a detailed description of how to download, install, and run the script.

2 Prerequisite knowledge

Reader should follow the below information to successfully execute the script and interpret results:

1. Users are advised to review the SFRAT user guide to understand the functionality and options available in the tool.
   The user's guide is available at *https://sasdlc.org/lab/assets/projects/srt.html*.

2. Users need not have to download SFRAT based on the information provided in the user's guide. The script folder will contain all the necessary files including SFRAT to execute the script. However, if the user has already installed SFRAT files, then the path to SFRAT should be set as the working directory while using R studio. These steps are explained in Section 4.

3 Requirements, Installation, and Data Format

This section describes the necessary requirements to run the script. This section also briefly illustrates the input data format to execute the script.

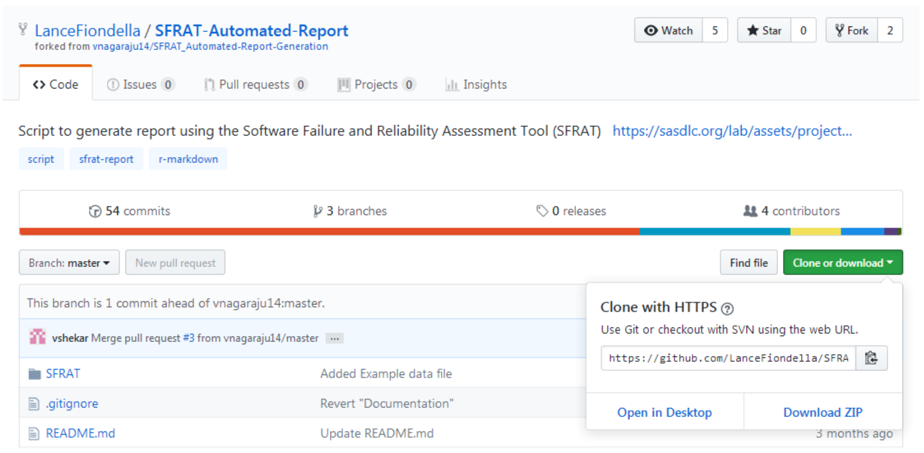
3.1 System requirements

To execute the script, the following requirements should be satisfied:

- *Operating System*: Windows 7 (64-bit) or newer, Mac OS X 10.9 or newer, Linux capable of running R.
- *Perl*: Perl 5 version 16 or newer - may be pre-installed on OSX/Linux; for Windows, Perl may be downloaded from *http://strawberryperl.com/*.
- *LaTeX*: Some user-chosen LaTeX editor (e.g. MikTex, TexStudio, Texmaker).
- *pandoc*: Newer versions of R do not support installing pandoc as a package, so it must be downloaded separately from *https://pandoc.org/installing.html*.
- *R*: R (ver. 3.0 or newer) and RStudio(ver. 0.99.482 or newer) are required, downloadable from *http://rstudio.com*.

3.2 Script Download and Installation

The source code of the SFRAT and automated script can be downloaded from the GitHub repository available at: *https://github.com/LanceFiondella/SFRAT-Automated-Report*. Figure 1 shows script download from GitHub Website.


LanceFiondella / SFRAT-Automated-Report
forked from vnagaraju14/SFRAT_Automated-Report-Generation

Script to generate report using the Software Failure and Reliability Assessment Tool (SFRAT) https://sasdlc.org/lab/assets/project...

54 commits 3 branches 0 releases 4 contributors

This branch is 1 commit ahead of vnagaraju14:master.

vshekar Merge pull request #3 from vnagaraju14/master

SFRAT	Added Example data file
.gitignore	Revert "Documentation"
README.md	Update README.md

Clone with HTTPS
Use Git or checkout with SVN using the web URL.
https://github.com/LanceFiondella/SFRA

Figure 1: Script download from the GitHub Repository

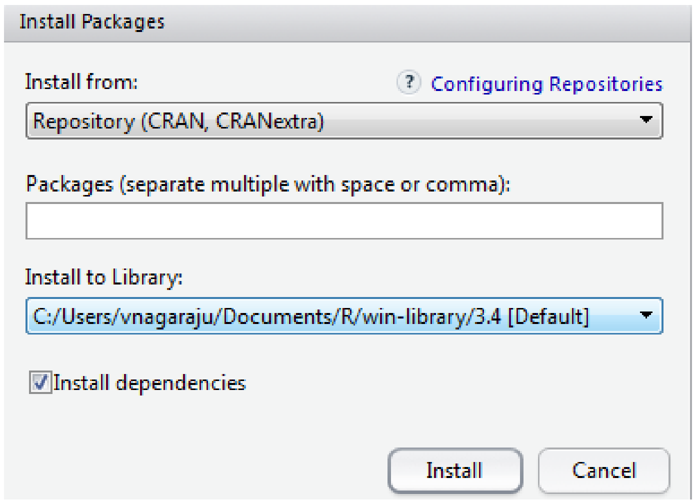
To execute the script successfully, some additional R packages that must be installed, including

- **shiny** - A web application framework for R.
- **gdata** - A package that provides data manipulation tools.
- **ggplot**- A graphical package capable of creating elegant and complex plots.
- **DT** - A package that provides an R interface to the JavaScript library DataTables.
- **rootSolve**- A package used to find the roots of n nonlinear (or linear) equations.
- **knitr**- A package providing a general-purpose tool for dynamic report generation in R using Literate Programming techniques.
- **rmarkdown**- A package allowing converting R Markdown documents into a variety of formats including HTML, MS Word, PDF, and Beamer.
- **markdown**- A package for authoring HTML, PDF, and MS Word documents.
- **readxl**- A package to import excel files.
- **formatR**- A package designed to reformat R code to improve readability.

To automatically install all the required packages, perform one of the below actions using RStudio

- Open **installscript.R** using RStudio and select *"Source"* option on the top right corner.
- Run *Source('installscript.R')* in the RStudio console.
- Execute *Rscript installscript.R* using the command line.

Alternatively, the user can install the packages manually using the Install packages menu item in RStudios Tools menu. This will bring up the dialog box in Figure 2. Specify the package(s) you want to install in the first line of the dialog box. It is not necessary to change the installation location, so the second line can be left blank. Make sure that the "*Install dependencies*" box is checked. If it is not, the packages you install may not work expected.


Install Packages

Install from: Repository (CRAN, CRANextra)

Packages (separate multiple with space or comma):

Install to Library: C:/Users/vnagaraju/Documents/R/win-library/3.4 [Default]

[✓] Install dependencies

Figure 2: Install Packages Dialog Box

3.3 Input data format

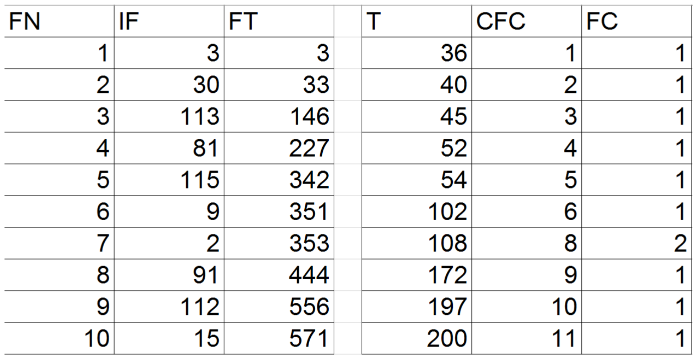
The script reads input data in excel (.xlsx) file format which is compatible with the SFRAT input file format. An input data file contains the failure data collected during the software testing phase in one of the following formats:

- Failure time data (FT): records the time of individual failures. A vector of individual failure times $\mathbf{T} = \langle t_1, t_2, \ldots, t_n \rangle$, where, n is the number of faults observed.

- Inter-failure data (IF): records the length of time between two successive failures. It is equivalent to the time between $(i-1)^{st}$ and i^{th} failure, defined as $t_i - t_{i-1}$.

- Failure count data (FC): records the length of each interval and number of failures observed within that interval $\langle \mathbf{T}, \mathbf{K} \rangle = \langle (t_2, k_2), \ldots, (t_n, k_n) \rangle$ where, t_i is the time at which the i^{th} interval ended and k_i is the number of faults detected in that interval.

The input data should be entered in one of the three formats shown in Figure 3. Any of these types of failure data can be read by SFRAT [1]. An example of each data format is shown in Figure 3 below. Figure 3 shows part of a set of input data in the form of times between successive failures, which are inter-failure times, failure times, and failure counts data labeled as "IF", "FT", and "FC". The column labeled "FN" indicates the failure number for the failure time and inter-failure data, whereas the column labeled "T" denotes the time interval for the failure count data and "CFC" denotes the cumulative failure count, which is the total number of failures observed over time. The tool automatically converts the data to failure time and inter-failure data formats, so that models can be applied.



FN	IF	FT		T	CFC	FC
1	3	3		36	1	1
2	30	33		40	2	1
3	113	146		45	3	1
4	81	227		52	4	1
5	115	342		54	5	1
6	9	351		102	6	1
7	2	353		108	8	2
8	91	444		172	9	1
9	112	556		197	10	1
10	15	571		200	11	1

Figure 3: Inter-failure, failure times, and failure counts data formats for Excel

4 SFRAT report generation script

This section discusses the usage of the automated SFRAT script. The inputs to the script are the user-defined input are normally provided through the four different tabs of the SFRAT GUI. Therefore, it is recommended that the user familiarize themselves with the SFRAT interface.

The script runs the SFRAT with the user-defined inputs inside the script, and then generates a PDF report based on the SFRAT output. Manually configuring the settings via the user interface consumes time and attention. The script reduces the time burden and is a repeatable alternative to generate reports automatically. The report generation script is written in the R programming language and contains the following files, which are available for download from the GitHub repository:

1. **report-specifications.R**: An R file that contains the variables needed to generate a report (typically specified through the user interface). The user must also specify the location of the failure data in this file.

2. **SFRATReport.Rmd**: A markdown file in R used to graphically generate the report. The report title, date, time, and output file format can be modified using this file. The markdown file is used as a template when placing the tool's output into a readable PDF report.

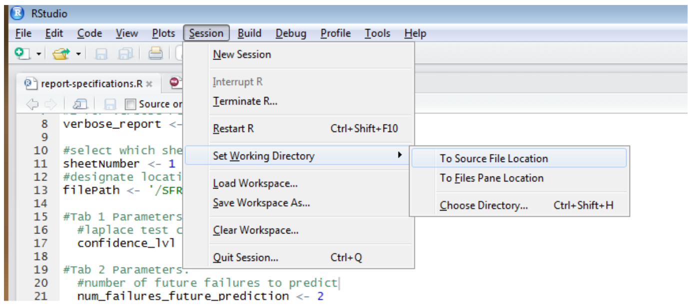
3. **SFRAT**: This folder contains all the files essential to run the SFRAT tool, which are invoked by the script. As noted in Section 2, if the SFRAT has been previously downloaded, then the previously downloaded SFRAT files or folder within the script download can be use. The user can remove one of the folders if they wish to conserve storage. The working directory should be set to the path of the SFRAT folder selected for this purpose.

4. **installscript.R**: An R file to check system compatibility and install the required packages on R Studio needed to execute the script, as described in Section 3.2.

Next, launch R Studio and open **report-specifications.R** and **SFRATReport.Rmd**. Set the working directory to the location where the SFRAT folder is saved. If SFRAT was downloaded previously, then the directory can be set via **Session** → **Set Working Directory** → **Choose Directory** to browse to the location of SFRAT. Otherwise, follow **Session** → **Set Working Directory** → **To Source File Location** in order to set the working directory. Figure 4 illustrates the process of setting the working directory.

4.1 report-specifications.R

Each variable in *report-specifications.R* file represents distinct input parameters in the GUI of the SFRAT [1]. Setting found in the GUI are configurable within the script. A sample of the code in the *report-specifications.R* available for user modification is:



```
verbose_report <-

#select which she
sheetNumber <- 1
#designate locati
filePath <- '/SFR

#Tab 1 Parameters
  #laplace test c
  confidence_lvl

#Tab 2 Parameters:
  #number of future failures to predict
  num_failures_future_prediction <- 2
```

Figure 4: Set Working Directory on R Studio

```
verbose_report <- 1
filePath <- '/SFRAT/model_testing/model_data.xlsx'
sheetNumber <- 1
colors <- c("navy","red","green","firebrick4","magenta")
confidence_lvl <- 0.9
num_failures_future_prediction <- 2
models_to_apply <- c('DSS', 'GM', 'Wei','GO','JM')
mission_time <- 600
num_failures_to_predict <- 2
additional_time_software_will_run <- 4116
desired_reliability <- .9
reliability_interval_length<- 600
percent_data_for_PSSE <- .9
```

A brief description of each parameter can be found in the script, which is organized according to different tabs in the SFRAT GUI. For the sake of clarity, the values above are taken from the SFRAT user's guide [1].

Tab 1 - Select, Analyze, and Filter Data:

This tab allows the user to upload the input data file, choose a specific data set, and view the data as cumulative failures, failure intensity, and times between failures. This tab also assesses the reliability growth of the data using trend tests, including the Laplace trend test (LTT) and running arithmetic average (RAA).

- **verbose_report** - Enabling this option produces verbose text output in the PDF report, providing a brief description of each result in the report. This option is enabled by indicating 1 on the right hand side of this variable.

- **filepath** - The user specifies the path to the location where the data file is saved. The script will access the sample file located within downloaded content at '/SFRAT/model_testing/model_data.xlsx', which consists of 10 failure times and 21 failure count data sets.

- **sheetNumber** - If the Excel file has more than one data set (one data set per sheet), this option allows the user to specify the data to be analyzed by indicating the sheet number. The first sheet is selected by default.

- **colors** - This specifies the set of colors used to plot different models in the report. The number of colors specified should be equal to the number of models selected, otherwise default colors will be automatically selected by the tool.

- **confidence_lvl** - This allows the user to specify a confidence level between 0 and 1 for the Laplace trend test to quantify the desired level of significance for reliability growth.

Tab 2 - Set Up and Apply Models:

This tab allows the user to specify a set of models to apply to the input data and make predictions. The outputs are a visual model fit of cumulative failures, failure intensity, times between failures, and assessment of reliability growth for the specified mission time.

- **num_failures_future_prediction** - The user can specify the number of failures they would like to predict beyond the end of testing.

- **models_to_apply** - This allows the user to specify which software reliability growth models they would like to apply to make predictions. The available models are the Delayed s-shape (DSS), geometric (GM), Goel-Okumoto (GO), Jelinski-Moranda (JM), and Weibull (Wei). Models must be specified with the acronyms given here. **Note**: model acronyms are case sensitive.

- **mission_time** - This specifies the required mission/operation time for which the predictions should be made.

Tab 3 -Query Model Results:

This tab uses the model results from Tab 2 to make numerical predictions.

- **num_failures_to_predict** - The number of failures to predict beyond the end of testing. This is similar to *num_failures_future_prediction* on Tab 2.

- **additional_time_software_will_run** - The additional time beyond the end of testing to predict the number of future failures.

- **desired_reliability** - The target reliability between 0 to 1 in order to estimate the time required to achieve such reliability.

- **reliability_interval_length** - The mission time beyond testing to estimate reliability. This is similar to *mission_time* on Tab 2.

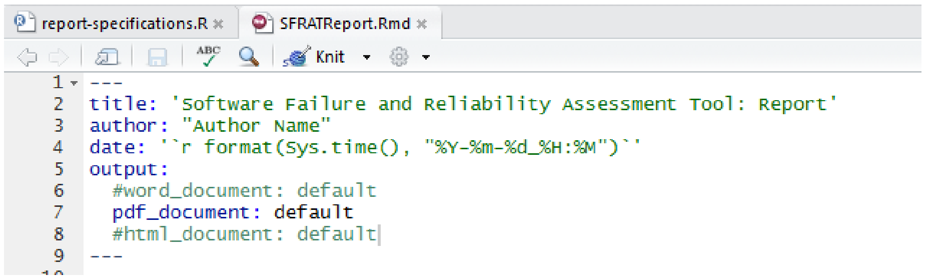
Tab 4 - Evaluate Models:

This tab compares all the applied models based on goodness-of-fit measures to identify the model that best characterize the input data.

- **percent_data_for_PSSE** - The percentage of data to be used for model fitting. The remaining data is used to assess model prediction using the predictive sum of squares error goodness-of-fit measure.

4.2 SFRATReport.Rmd

The Markdown file allows the user to modify the header of the report, including title, author name, date, and time as well as the output format of the report. Figure 5 shows a screenshot of the code available for user modification.



```
1  ---
2  title: 'Software Failure and Reliability Assessment Tool: Report'
3  author: "Author Name"
4  date: '`r format(Sys.time(), "%Y-%m-%d_%H:%M")`'
5  output:
6    #word_document: default
7    pdf_document: default
8    #html_document: default
9  ---
```

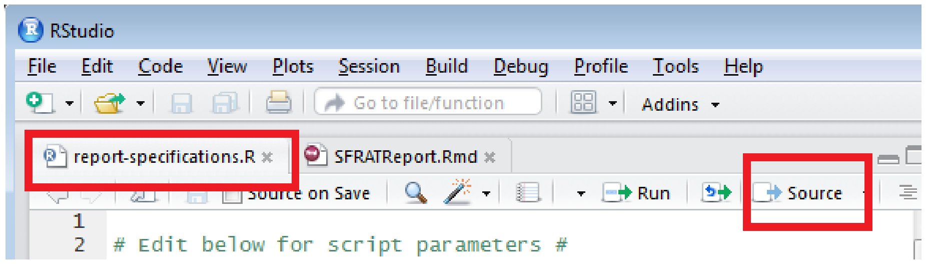
Figure 5: Markdown document

Figure 5 shows code between lines 2 through 8, which can be modified by the user. Modifying other sections of the code may produce errors. In Figure 5, lines 2 and 3 are simple text that should be specified within single and double quotes respectively. *'date'* on line 4 automatically generates the system date and time within the report in the format YYYY-MM-DD_hr:min, which are also be used to name the output file in the format *SFRAT report_dataName_YYYY-MM-DD.pdf*. Lines 5-8 allows the user to specify the output file format. Figure 5 specifies PDF format. However, other formats can be specified by commenting line 7 and uncommenting the required format such as Word or HTML.

4.3 Output Report

Once all the input parameters and required format are specified, the user can run the script in one of the two following ways:

- **Command Line**: The script may be run via the command line by navigating to the folder where the *report-specifications.R* file is saved. Run R and enter *source('report-specifications.R')*.
- **R Studio**: Open the *report-specifications.R*, set the working directory, and click on *'Source'* on the top right corner or type *source('report-specifications.R')* in the console. Figure 6 shows this step:

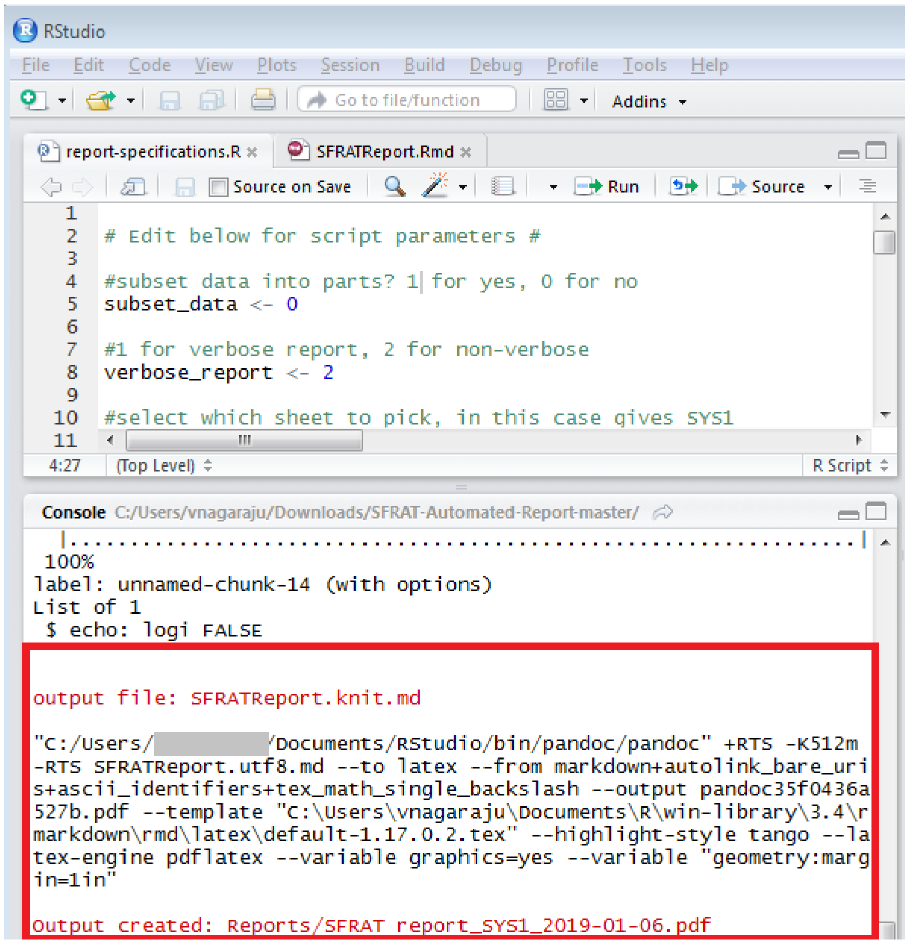


```
# Edit below for script parameters #
```

Figure 6: Source *report-specifications.R*

If the script runs without interruption, Figure 7 shows the message that will be displayed in the console.

In Figure 7, the last line indicates the file name and location of the output file, which is in the **/Reports** folder within the source file location.

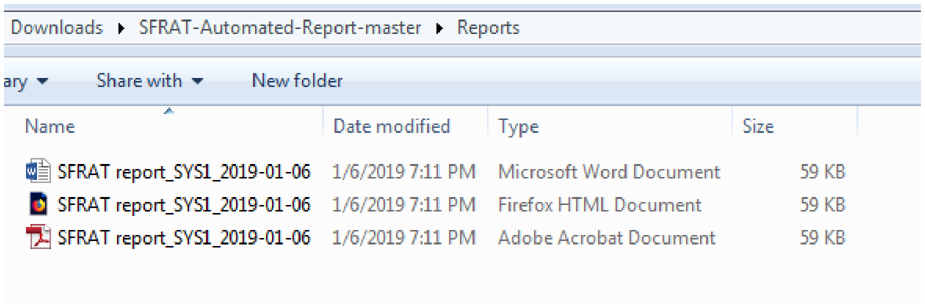

RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
report-specifications.R
SFRATReport.Rmd
Source on Save
Run
Source
4:27
(Top Level)
R Script
Console C:/Users/vnagaraju/Downloads/SFRAT-Automated-Report-master/

```
# Edit below for script parameters #

#subset data into parts? 1 for yes, 0 for no
subset_data <- 0

#1 for verbose report, 2 for non-verbose
verbose_report <- 2

#select which sheet to pick, in this case gives SYS1
```

```
  |.................................................................|
 100%
label: unnamed-chunk-14 (with options)
List of 1
 $ echo: logi FALSE


output file: SFRATReport.knit.md

"C:/Users/[redacted]/Documents/RStudio/bin/pandoc/pandoc" +RTS -K512m
-RTS SFRATReport.utf8.md --to latex --from markdown+autolink_bare_uri
s+ascii_identifiers+tex_math_single_backslash --output pandoc35f0436a
527b.pdf --template "C:\Users\vnagaraju\Documents\R\win-library\3.4\r
markdown\rmd\latex\default-1.17.0.2.tex" --highlight-style tango --la
tex-engine pdflatex --variable graphics=yes --variable "geometry:marg
in=1in"

Output created: Reports/SFRAT report_SYS1_2019-01-06.pdf
```

Figure 7: Script successful execution message on the console

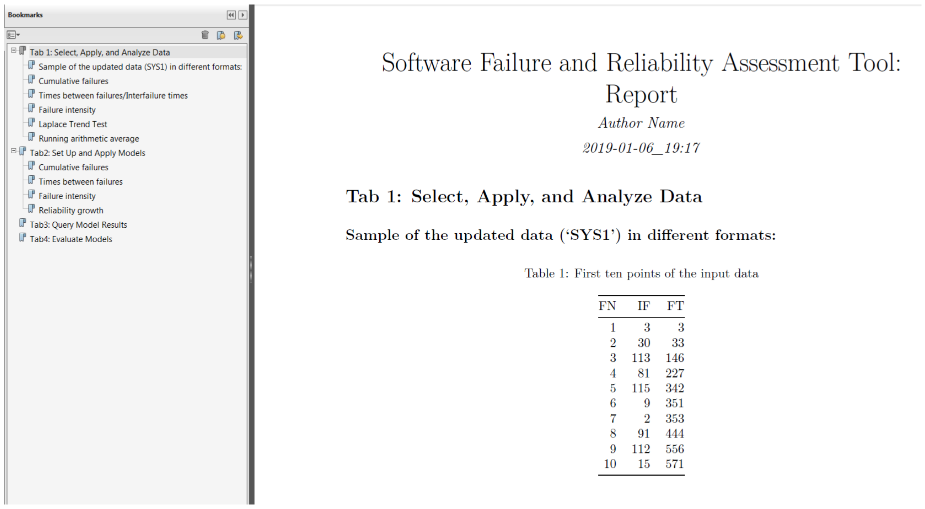
Figure 8 shows all possible output formats saved in the Report file.



Name	Date modified	Type	Size
SFRAT report_SYS1_2019-01-06	1/6/2019 7:11 PM	Microsoft Word Document	59 KB
SFRAT report_SYS1_2019-01-06	1/6/2019 7:11 PM	Firefox HTML Document	59 KB
SFRAT report_SYS1_2019-01-06	1/6/2019 7:11 PM	Adobe Acrobat Document	59 KB

Figure 8: Different output formats

Figure 9 shows an example of the non-verbose report.



Tab 1: Select, Apply, and Analyze Data
- Sample of the updated data (SYS1) in different formats:
- Cumulative failures
- Times between failures/Interfailure times
- Failure intensity
- Laplace Trend Test
- Running arithmetic average

Tab2: Set Up and Apply Models
- Cumulative failures
- Times between failures
- Failure intensity
- Reliability growth

Tab3: Query Model Results

Tab4: Evaluate Models

Software Failure and Reliability Assessment Tool: Report

Author Name

2019-01-06_19:17

Tab 1: Select, Apply, and Analyze Data

Sample of the updated data ('SYS1') in different formats:

Table 1: First ten points of the input data

FN	IF	FT
1	3	3
2	30	33
3	113	146
4	81	227
5	115	342
6	9	351
7	2	353
8	91	444
9	112	556
10	15	571

Figure 9: Non-verbose report

In Figure 9, the left side shows output organized into different tabs similar to the SFRAT. If the verbose option is specified, each figure and table is accompanied by a corresponding description. A complete sample of a verbose report is given in the Appendix.

References

[1] L. Fiondella, "Software Failure and Reliability Assessment Tool (SFRAT)," http://sasdlc.org/lab/projects/srt.html, 2016.

5 Appendix

This section shows screenshot of each page of the verbose report generated using SYS1 data set.

Cumulative failures

The following figure shows the SYS1 data as the cumulative number of failures (FN) detected as a function of cumulative test time (FT). An increasing trend indicates periods where more faults were detected. Ideally, the cumulative number of failures should level off to a horizontal line, indicating that no new faults have been detected.

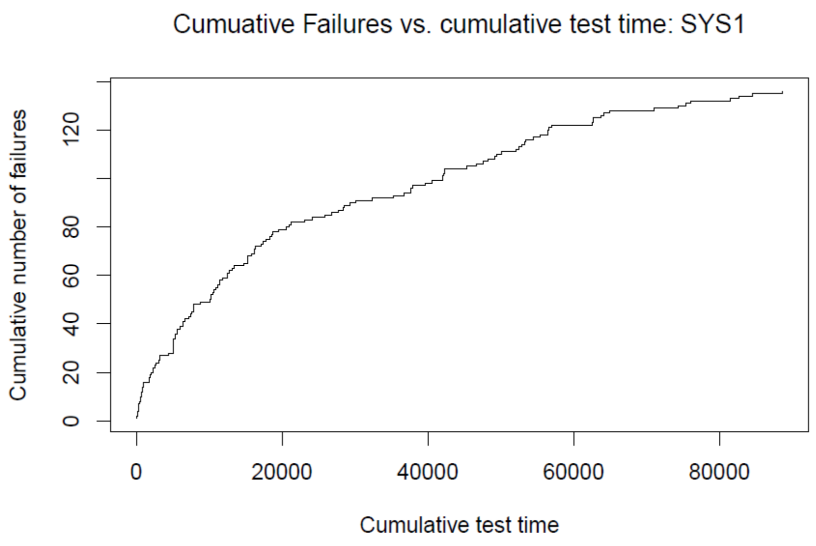


Figure 10: Verbose report: Cumulative Failures

Times between failures/Interfailure times

The following figure shows the SYS1 times between failures (IF) as a function of cumulative test time (FT). An increasing trend indicates periods where fewer faults were detected. Ideally, the time between failures should increase, indicating that no new faults have been detected.

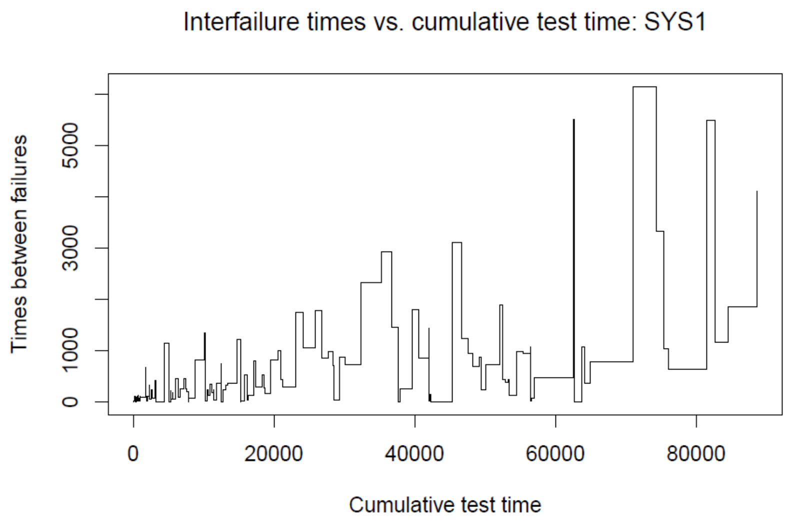


Figure 11: Verbose report: Times between failures

Failure intensity

The following figure shows the SYS1 data as the number of failures detected per unit time as a function of cumulative test time (FT). A decreasing trend indicates periods where fewer faults were detected. Ideally, the failure intensity should decrease, indicating that no new faults have been detected.

A decrease in failure intensity indicates increase in reliability of the software subjected to testing. Ideally, the failure intensity should go to zero.

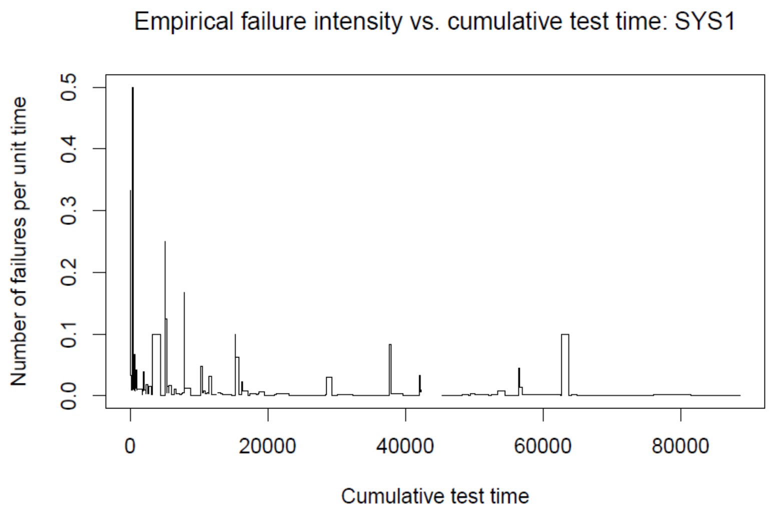


Figure 12: Verbose report: Failure intensity

Laplace Trend Test

The following figure shows the Laplace test statistic for reliability growth as a function of cumulative test time (FT). A decreasing trend indicates reliability growth, while an increasing trend indicates reliabilty deterioration. The Laplace test statistic on the y-axis corresponds to the critical values of a normal distribution. This means that if the trend falls below a specific level, then we cannot reject the null hypothesis that the failure data suggests reliability growth at a specified level of confidence. The six black dot-dash style lines correspond to the 90%, 95%, 99%, 99.9%, 99.9999%, and 99.999999% respectively. The red line is user-specified and has been set to 90%. The level of confidence is a subjective choice made by the analyst. Reliability growth is desired because software reliability growth models assume curves that exhibit increasing time between failures. If reliability growth is not present than the model fitting step may fail or produce predictions that are inaccurate. Therefore, the Laplace test statistic provides an objective quantitative measure for the analyst to decide if predictions may or may not be accurate.

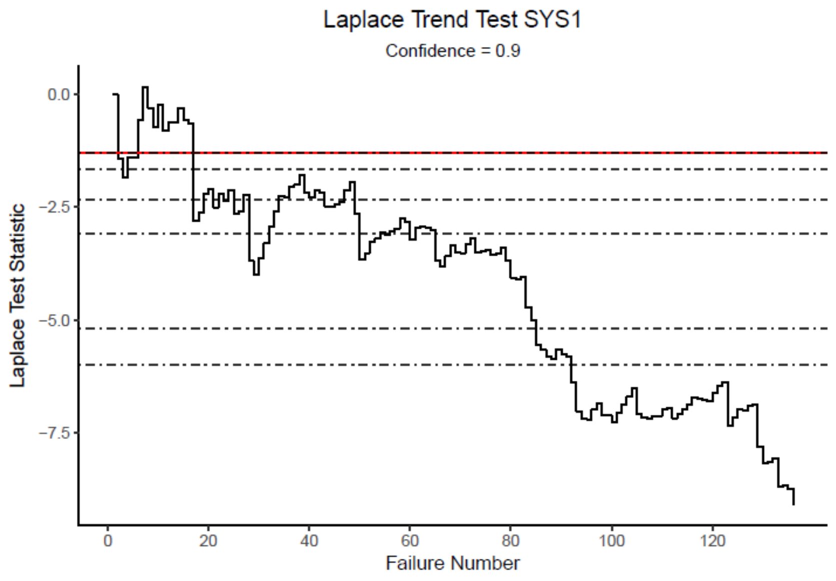


Figure 13: Verbose report: Laplace trend test

Running arithmetic average

The running arithmetic average plots the average of the first k times between failure. An increasing trends indicates reliability growth, while a decreasing trend indicates reliability deterioration. This is intuitive because if the time between failures is increasing, then later failures will increase the average.

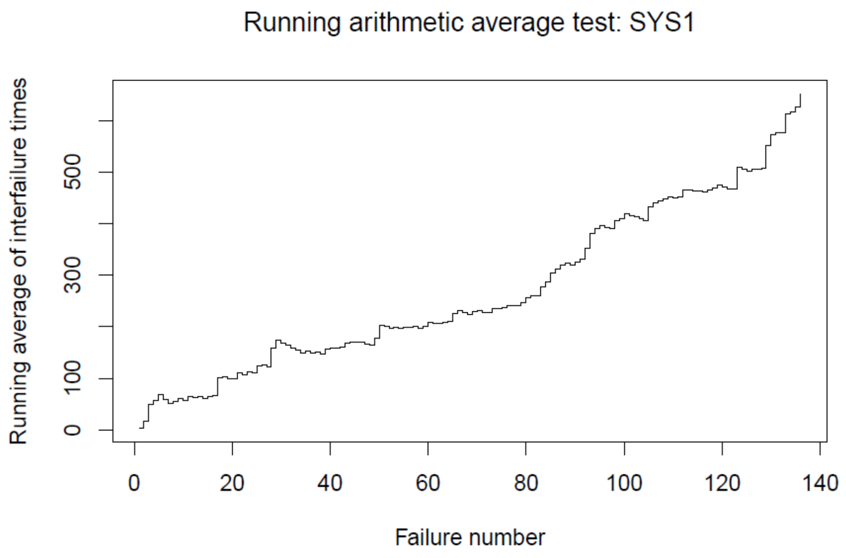


Figure 14: Verbose report: Running arithmetic average

Tab2: Set Up and Apply Models

Cumulative failures

The following figure shows the fit of delayed s-shaped, geometric, Weibull, Goel-Okumoto, Jelinski-Moranda models to the cumulative number of failures detected in the SYS1 data.

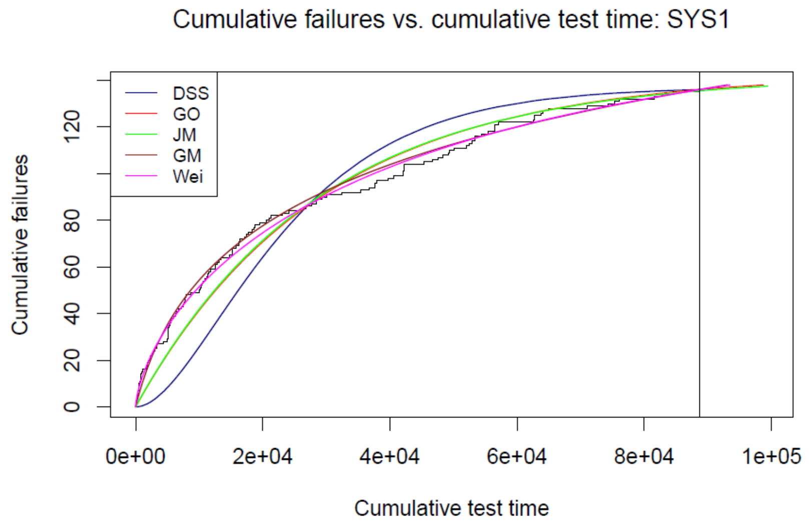


Figure 15: Verbose report: Model fit - Cumulative failures

Times between failures

The following figure shows the fit of delayed s-shaped, geometric, Weibull, Goel-Okumoto, Jelinski-Moranda models to the times between failures detected in the SYS1 data.

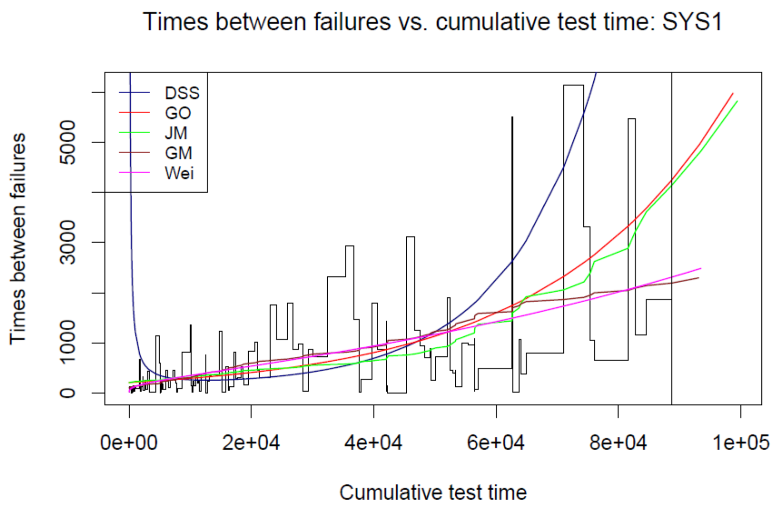


Figure 16: Verbose report: Model fit - Times between failures

Failure intensity

The following figure shows the fit of delayed s-shaped, geometric, Weibull, Goel-Okumoto, Jelinski-Moranda models to the failure intensity of the SYS1 data.

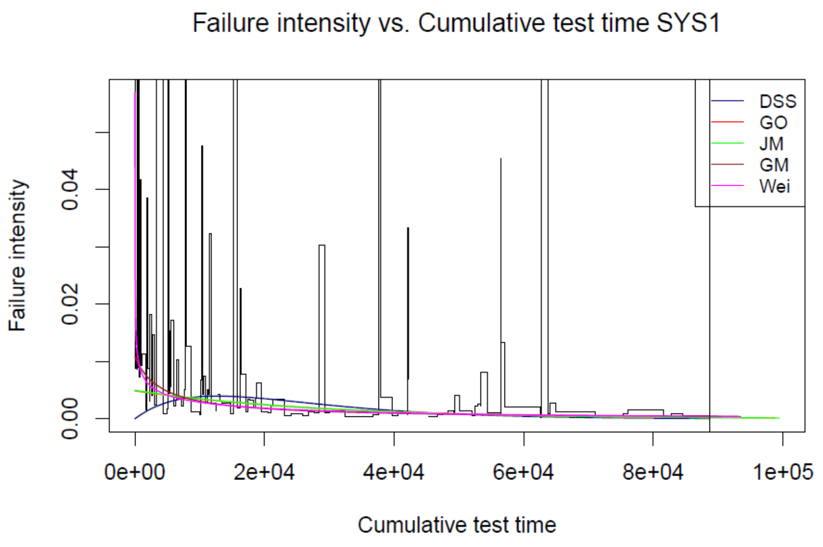


Figure 17: Verbose report: Model fit - Failure intensity

Reliability growth

The following figure shows the reliability growth curve of the fit of delayed s-shaped, geometric, Weibull, Goel-Okumoto, Jelinski-Moranda models to the SYS1 data. The data itself does not display. This plot indicates a models prediction that the software will be reliable (exhibit zero failures) for a duration of 600 time units as a function of cumulative test time (FT). Selecting a model upon which to base a reliability assessment is a subjective choice made by the analyst. Statistical measures of goodness of fit, reported on page 12 of this report can be used to decide this decision making process. If the Laplace test statistic does not exhibit reliability growth, than a conservative approach is to document this as the reason why no reliability estimate is provided at the time of preparing a report.

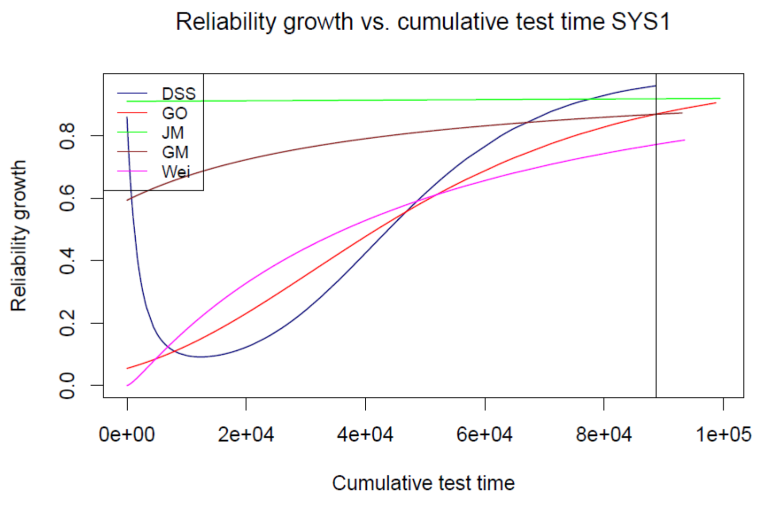
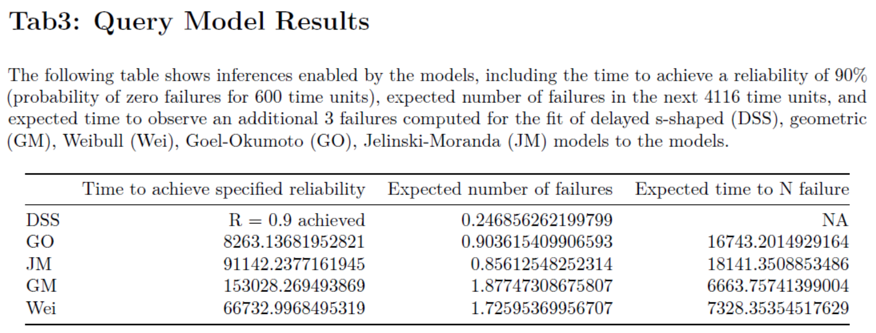


Figure 18: Verbose report: Reliability Growth



Tab3: Query Model Results

The following table shows inferences enabled by the models, including the time to achieve a reliability of 90% (probability of zero failures for 600 time units), expected number of failures in the next 4116 time units, and expected time to observe an additional 3 failures computed for the fit of delayed s-shaped (DSS), geometric (GM), Weibull (Wei), Goel-Okumoto (GO), Jelinski-Moranda (JM) models to the models.

	Time to achieve specified reliability	Expected number of failures	Expected time to N failure
DSS	R = 0.9 achieved	0.246856262199799	NA
GO	8263.13681952821	0.903615409906593	16743.2014929164
JM	91142.2377161945	0.85612548252314	18141.3508853486
GM	153028.269493869	1.87747308675807	6663.75741399004
Wei	66732.9968495319	1.72595369956707	7328.35354517629

Figure 19: Verbose report: Tab3 output - Predictions

Tab4: Evaluate Models

The following table shows the measures of goodness of fit computed for the delayed s-shaped, geometric, Weibull, Goel-Okumoto, Jelinski-Moranda The Akaike Information Criterion (AIC) is an information theoretic measure. Lower values are preferred. The GM model achieved the lowest AIC value on the SYS1 data. A difference of 2.0 or more in the AIC values of two models indicates the model with the lower AIC score is preferred with statistical significance. The Predictive Sum of Squares Error (PSSE) used 90% of the SYS1 data to fit the models and computed the sum of the squares between the differences of the remaining 10% of the data not used to fit the models. Lower values are preferred. The GM model achieved the lowest PSSE value on the SYS1 data. The measures of goodness of fit can help select a model, but the choice is ultimately a subjective choice made by the analyst.

	Akaike Information Criterion (AIC)	"Predictive sum of squares error (PSSE)" ~ 0.9
DSS	2075.15	296.35
GO	1953.61	23.07
JM	1950.53	*19.6
GM	*1937.03	84.33
Wei	1938.16	74.94

Figure 20: Verbose report: Tab4 output - Goodness of fit comparison