Unit. 4 Software Reliability Models

1. **Introduction**
2. **Data Collection and Analysis**
3. **Reliability Growth Models**
4. **Modeling Process**
5. **Revisiting the Rayleigh Model**

Reading: TB-Chapter 15 (15.1-15.7)

1. Introduction

Definitions of Software Reliability

Definition 1: Software reliability is the probability of failure-free operation of a software system for a specified time in a specified environment.

-Key considerations:

1. Probability of failure-free operation
2. Length of time of failure-free operation
3. Definition with respect to a given execution environment

Exercise 4.1

An office secretary turns on his/her PC every morning at 8:30am and turns it off at 4:30pm before leaving for home.

Calculate the reliability of the PC considering that he/she comes to the office for 200 days in a year and observes that the PC crashes five times on different days in a year for a few years.

Definitions of Software Reliability (ctd.)

Definition 2: Failure intensity is a measure of the reliability of a software system.

-Key considerations:

1. Failure intensity is expressed as the number of failures observed per time unit.
2. The lower the failure intensity of a software system, the higher its reliability.

Exercise 4.2

Assume that during system testing of a software system, test engineers are observing failures at the rate of 2 failures per eight hours of system execution. Calculate the failure intensity of the system.

Definitions of Software Reliability (ctd.)

-Failure intensity **λ** is studied in relation with the *cumulative failure* **μ** *:*

- The rate of rising of the graph of the cumulative number of failures is the rate at which failures are being observed (i.e. Failure intensity).
- Smaller rate of rising of the cumulative graph indicates that failures are occurring infrequently, and hence more reliable system.

-Both **λ** and **μ** are functions of time:

- Time: Data tracking is done either in terms of ***precise CPU execution time*** or on a ***calendar-time*** basis.
- In principle, ***execution-time tracking is for small projects*** while ***calendar-time is common for commercial*** development.

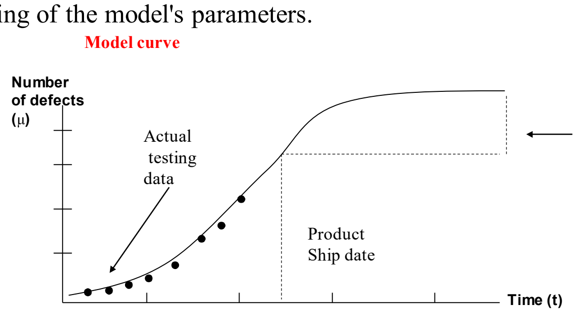
-λ being the rate of rising of the graph of **μ** gives:

$$\lambda(t) = \frac{d\mu(t)}{dt} \qquad \mu(t) = \int_0^t \lambda(x).dx$$

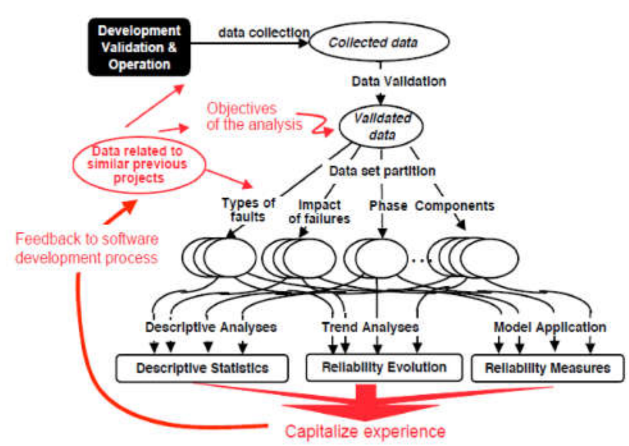
Software Reliability Models

-Software reliability models are ***statistical models*** which can be used to ***make predictions about a software system's failure rate***, given the failure history of the system.

-The models make assumptions about the fault discovery and removal process. These assumptions determine the form of the model and the meaning of the model's parameters.



Global reliability analysis method



2. Data Collection and Analysis

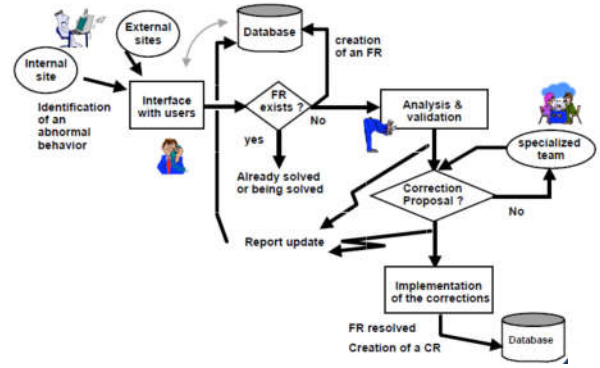
Data to be collected

- Background information
• Product itself: software size, language, functions, current version, workload
• Usage environment: verification and validation methods, tools, etc.

-Data relative to failures and corrections
• Date of occurrence, nature of failures, consequences
• Type of faults, fault location

-Usually, recorded through
• Failure Reports (FR)
• Correction Reports (CR)

Life cycle of Failure and Correction Reports (FRs/CRs)



Data Validation

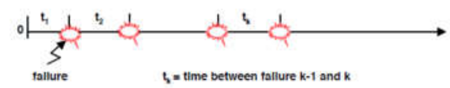
-Objectives
• Check the validity and usability of the information recorded
• Keep only genuine software faults in the database

-Elimination of:
• Duplicated data (FR reporting of the same failure)
• FR proposing a correction related to an already existing FR
• False FR (signaling a false or non identified problem)
• FR proposing an improvement
• Incomplete FRs or FRs containing inconsistent data (Unusable)
• FR related to a hardware failure

Data pre-processing for reliability analysis

-Two kinds of data sets can be extracted from FRs and CRs

• Time between failures



• Failure Count (or Grouped data)
-Number of failures per unit of time, n(k)
-Cumulative number of failures N(k)

-Time between failures can be measured as:
• Execution time
• Wall clock or Calendar time
• Number of executions

3. Reliability Models

Notions of Reliability Growth

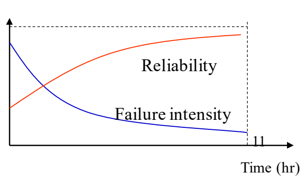
-In contrast to Rayleigh, which models the defect pattern of the entire development process, reliability growth models are usually ***based on data from the formal testing phases***.

•In practice, such models are applied during the final testing phase when the ***development is virtually completed***.

•The rationale is that defect arrival or failure patterns during such testing are good ***indicators of the product's reliability*** when it is used by customers.

•During such post-development testing, when failures occur and defects are identified and fixed, the software becomes more stable, and reliability grows over time.

•Hence models that address such a process are called **reliability growth models**.



Reliability Growth Models

-Examples of models currently being used include the following:

Times Between Failures (TBF) Models	Failures Counts (FC) Models
Geometric	Generalized Poisson
Jelinski-Moranda	Generalized Poisson-User specified interval weighting
Littlewood-Verrall Linear	Nonhomogeneous Poisson (NHPP)
Littlewood-Verrall Quadratic	Schneidewind
Musa Basic	Schneidewind- ignore first "s-1" test intervals
Musa-Okumuto	Schneidewind – total failures in first "s-1" test intervals
Nonhomogeneous Poisson (NHPP)	Shick-Wolverton
	Yamada S-shaped

-Selection depends on
• Objectives
Development follow-up, evaluation of operational MTTF and residual failure rate
• Trend displayed by the data set

Reliability Modeling Assumptions

-Reliability models are developed based on the following assumptions:

1. Program failures occur independently
2. During test, the software is operated in a similar manner as the anticipated operational usage.
3. The set of inputs per test run is randomly selected
4. All failures are observed
5. The fault causing a failure is immediately fixed or else its reoccurrence is not counted again.

Basic Reliability Models

-The **basic** and **logarithmic models** are two fundamental reliability models widely used.

-In these models, failure intensity λ as a measure of reliability is expressed as a function of execution time τ, i.e., $\lambda(\tau)$.

-Intuition behind the model: *as the cumulative failure count increases, the failure intensity decreases.*

-Basic model: *the decrease in failure intensity after a failure and fixing the corresponding failure is constant.*

-Logarithmic model: *the decrease in failure intensity after a failure and fixing the corresponding failure is smaller than the previous decrease.*

Basic Reliability Models (ctd.)

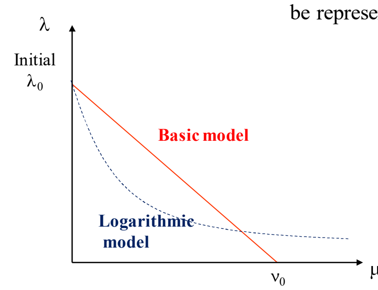
Modeling Parameters

- μ: mean number of failures observed
- λ: mean failure intensity
- λ_0: initial failure intensity observed at the beginning of system-level testing (ST)
- ν_0: total number of system failures expected to be observed over infinite time (starting from the beginning of ST)

-Θ: decrease in failure intensity in the logarithmic model

Basic Reliability Models (ctd.)

Basic (execution) Model



-The failure decrement (in the basic model) can be represented by the following linear equation:

$$\lambda(\mu) = \lambda_0\left(1 - \frac{\mu}{\upsilon_0}\right)$$

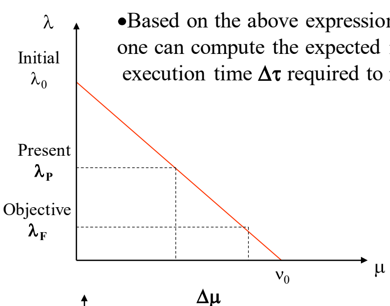
-Since, $\lambda(\tau) = \frac{d\mu(\tau)}{d\tau}$, we have:

$$\frac{d\mu(\tau)}{d\tau} = \lambda_0\left(1 - \frac{\mu}{\upsilon_0}\right)$$

-By solving the above differential equation, we have

$$\mu(\tau) = \nu_0 \times \left(1 - e^{-\frac{\lambda_0}{\nu_0}\tau}\right)$$ and $$\lambda(\tau) = \lambda_0 \times e^{-\frac{\lambda_0}{\nu_0}\tau}$$

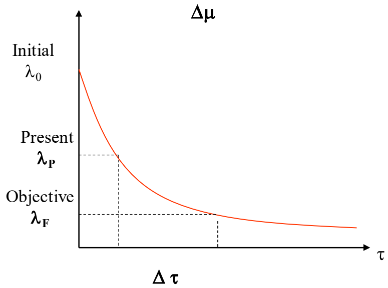
•Based on the above expressions, given some failure intensity objective, one can compute the expected number of failures $\Delta\mu$ and the additional execution time $\Delta\tau$ required to reach that objective.



$$\Delta\mu = \frac{\nu_0}{\lambda_0} \times (\lambda_1 - \lambda_2)$$

Where:

- λ_1 is the current failure intensity
- λ_2 is the failure intensity objective

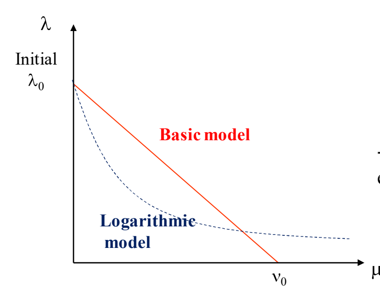


$$\Delta\tau = \frac{\nu_0}{\lambda_0} \times \ln\left(\frac{\lambda_1}{\lambda_2}\right)$$

Basic Reliability Models (ctd.)

Logarithmic Model

-The nonlinear drop in failure intensity is captured by a decay parameter Θ associated with a negative exponential function as follows: $\lambda(\mu) = \lambda_0 e^{-\theta\mu}$



-Since, $\lambda(\tau) = \frac{d\mu(\tau)}{d\tau}$, we have:

$$\frac{d\mu(\tau)}{d\tau} = \lambda_0 e^{-\theta\mu(\tau)}$$

-By solving the above differential equation, we have

$$\mu(\tau) = \frac{\ln(\lambda_0\theta\tau + 1)}{\theta}$$

$$\lambda(\tau) = \frac{\lambda_0}{\ln(\lambda_0\theta\tau + 1)}$$

Exercise 4.3

Assume that a program will experience 100 failures in infinite time. It has now experienced 50 failures. The failure intensity was 10 failures/CPU hr.

1. Calculate the current failure intensity.
2. Calculate the number of failures experienced after 10 and 100 CPU hr of execution.
3. Calculate the failure intensities at 10 and 100 CPU hr of execution.
4. Calculate the expected number of failures that will be experienced and the execution time between a current failure intensity of 3.68 failures/CPU hr and an objective of 0.000454 failure/CPU hr.

Exercise 4.4

It has been estimated that the safety-critical subsystem of a patient monitoring system will experience a total of $v_0 = 120$ failures (in infinite time) . Suppose that healthcare regulations require a failure intensity of $\lambda_{obj} = 0.001$ failures/CPU hr for such critical component before the product could be released.

1. Considering that the testing starts with an initial failure intensity of $\lambda_0 = 20$ failures/CPU hr, calculate the number of failures and the amount of execution time required to reach the failure intensity objective.

The effort required per hr of execution time is 6 person hr (for failure identification). Each failure requires (additionally) 2 person hr on the average to verify and determine its nature.

2. Calculate the total cost of testing assuming a loaded salary of $40/hr and additional cost of $50/hr for resources (e.g. Computer) and overheads.
3. Calculate the total duration of the testing activity in calendar time, assuming a standard work week of 40 hrs, and a 2 members (full time) test team.

4. Modeling Process

-To model software reliability, the following steps should be followed:

1. **Examine the data:** plot the data points against time in the form of a scatter diagram, analyze the data informally and gain an insight into the nature of the process being modeled.

2. **Select a model** or several models to fit the data based on an understanding of the test process, the data, and the assumptions of the models.

3. **Estimate the parameters of the model** using statistical techniques such as maximum likelihood or least-squares methods.

4. **Obtain the fitted model** by substituting the estimates of the parameters into the chosen model.

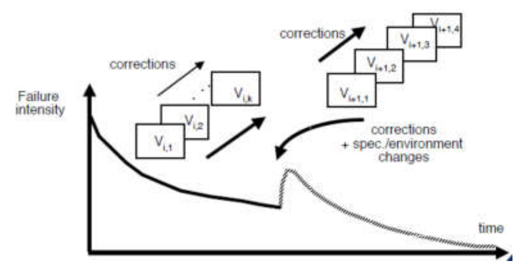
5. **Perform a goodness-of-fit test** and assess the reasonableness of the model. If the model does not fit, a more reasonable model should be selected.

6. **Make reliability predictions based on the fitted model.**

Trend analysis

-Objectives:

- Analyze software reliability evolution
- Identify periods of reliability growth and decrease



-Trend indicators

- Empirical (arithmetical) means
- Laplace factor

Empirical mean trend test: Running arithmetic average

-One of the simplest trend tests that can be applied to determine whether a set of failure data exhibits reliability growth.

-May be applied to both time between failures data and failure counts data.
●*For failure counts data, the test may only be applied to data in which the test intervals are of equal length.*

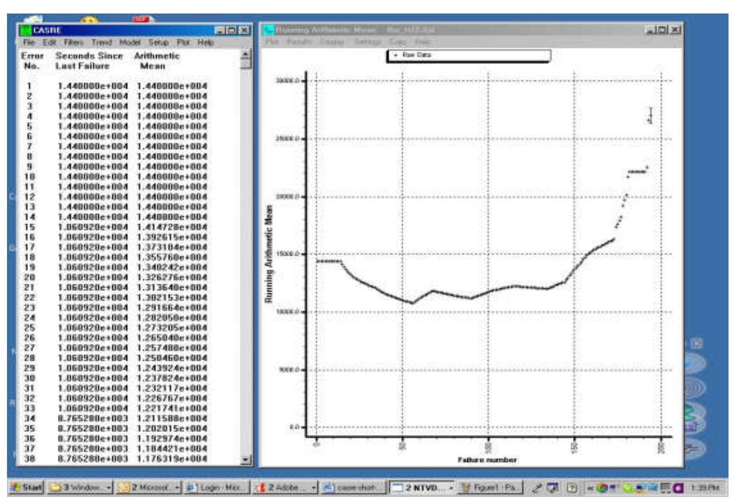
Running Arithmetic Average for Time between failures

m_k : arithmetic mean of the times to failures (from failure 1 to k)

$$m_k = \frac{t_1 + t_2 + \cdots t_k}{k}$$

m_k constitutes a globally increasing series ⇒reliability growth
m_k constitutes a globally decreasing series ⇒ reliability decrease

Example: Running Arithmetic Average - Time Between Failures Data



Error No.	Seconds Since Last Failure	Arithmetic Mean
1	1.440000e+004	1.440000e+004
2	1.440000e+004	1.440000e+004
3	1.440000e+004	1.440000e+004
4	1.440000e+004	1.440000e+004
5	1.440000e+004	1.440000e+004
6	1.440000e+004	1.440000e+004
7	1.440000e+004	1.440000e+004
8	1.440000e+004	1.440000e+004
9	1.440000e+004	1.440000e+004
10	1.440000e+004	1.440000e+004
11	1.440000e+004	1.440000e+004
12	1.440000e+004	1.440000e+004
13	1.440000e+004	1.440000e+004
14	1.440000e+004	1.440000e+004
15	1.060920e+004	1.414728e+004
16	1.060920e+004	1.392615e+004
17	1.060920e+004	1.373104e+004
18	1.060920e+004	1.355760e+004
19	1.060920e+004	1.340242e+004
20	1.060920e+004	1.326276e+004
21	1.060920e+004	1.313640e+004
22	1.060920e+004	1.302153e+004
23	1.060920e+004	1.291664e+004
24	1.060920e+004	1.282050e+004
25	1.060920e+004	1.273205e+004
26	1.060920e+004	1.265040e+004
27	1.060920e+004	1.257480e+004
28	1.060920e+004	1.250460e+004
29	1.060920e+004	1.243924e+004
30	1.060920e+004	1.237824e+004
31	1.060920e+004	1.232117e+004
32	1.060920e+004	1.226767e+004
33	1.060920e+004	1.221741e+004
34	8.765280e+003	1.211588e+004
35	8.765280e+003	1.202015e+004
36	8.765280e+003	1.192974e+004
37	8.765280e+003	1.184421e+004
38	8.765280e+003	1.176319e+004

Running Arithmetic Average for Failure Counts

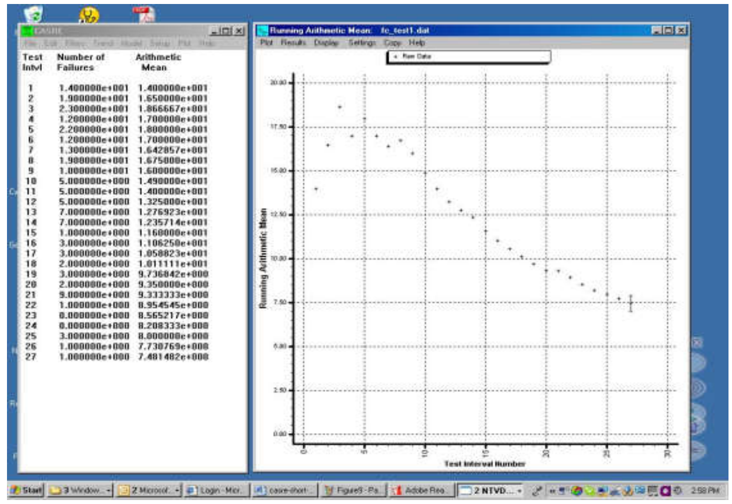
-For failure counts data, the running arithmetic average after the k^{th} test interval has been completed, r_k, is given by:

$$r_k = \frac{n_1 + \cdots + n_k}{k}$$

- Where n_k is the number of failures that have been observed in the k^{th} test interval.

r_k constitutes a globally decreasing series ⇒reliability growth
r_k constitutes a globally increasing series ⇒ reliability decrease

Example: Running Arithmetic Average - Failure Counts Data



Goodness-of-fit Test

Goodness-of-fit - Kolmogorov-Smirnov Test

- Uses the absolute vertical distance between two CDFs to measure goodness of fit.

-The test statistics is computed as:

$$D(n) = \underset{x}{Max}\left(\left| F^*(x) - F(x) \right|_x\right)$$

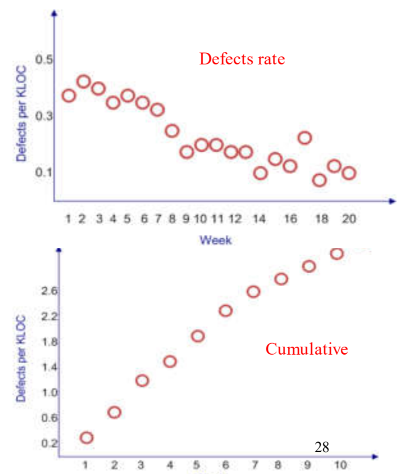
- Where F*(x) is the observed normalized cumulative distribution at each time point and F(x) is the expected normalized cumulative distribution at each time point, based on the model.
- Normalization is done with respect to the maximum cumulative value.

-If D(n) is less than the established criteria (from the Kolmogorov empirical table), the model fits the data adequately

Exercise 4.5: Modeling based on sample defect data

Week	Defects Arrival/KLOC	Defects/KLOC cumulative
1	.353	.353
2	.436	.789
3	.415	1.204
4	.351	1.555
5	.380	1.935
6	.366	2.301
7	.308	2.609
8	.254	2.863
9	.192	3.055
10	.219	3.274
11	.202	3.476
12	.180	3.656
13	.182	3.838
14	.110	3.948
15	.155	4.103
16	.145	4.248
17	.221	4.469
18	.095	4.564
19	.140	4.704
20	.126	4.830

Based on the data, estimate the number of defects 4 years after releasing the system.



Exercise 4.5: Modeling based on sample defect data (ctd.)

-Based on the trend of the data (an overall decreasing trend), we hypothesize the basic execution or exponential model.

-Model parameters can be estimated using different techniques such as non-linear regression or non-linear least squares.

-Using non-linear least square methods, we obtain:
- Cumulative failure rate: ν_0=6.597
- Initial failure rate: λ_0=0.469

-Fitting the estimated parameters into the basic execution (or exponential) distribution gives:

$$\lambda(t) = 0.469 \times e^{-0.0712t}$$

$$\mu(t) = 6.597 \times (1 - e^{-0.0712t})$$

Where t is the week number since the start of system test

Kolmogorov-Smirnov goodness-of-fit test for sample data

Week	Observed Defects/KLOC cumulative (A)	Model Defects/KLOC cumulative (B)	F(x)	F*(x)	|F*(x)-F(x)|
1	.353	.437	.07314	.09050	.01736
2	.789	.845	.16339	.17479	.01140
3	1.204	1.224	.24936	.25338	.00392
4	1.555	1.577	.32207	.32638	.00438
5	1.935	1.906	.40076	.39446	.00630
6	2.301	2.213	.47647	.45786	.01861
7	2.609	2.498	.54020	.51691	.02329
8	2.863	2.764	.59281	.57190	.02091
9	3.055	3.011	.63259	.62311	.00948
10	3.274	3.242	.67793	.67080	.00713
11	3.476	3.456	.71984	.71522	.00462
12	3.656	3.656	.75706	.75658	.00048
13	3.838	3.842	.79470	.79510	.00040
14	3.948	4.016	.81737	.83098	.01361
15	4.103	4.177	.84944	.86438	.01494
16	4.248	4.327	.87938	.89550	.01612
17	4.469	4.467	.92515	.92448	.00067
18	4.564	4.598	.94482	.95146	.00664
19	4.704	4.719	.97391	.97659	.00268
20	4.830	4.832	1.0000	1.0000	.00000

Compute the Kolmogorov-Smirnov statistic to check whether the data fit adequately the hypothesized model (B).

The Kolmogorov statistic is computed as D(n)=xF*(x)-F(x)|, where ***F*(x) is the observed normalized cumulative distribution*** at each time point , ***F(x) is the expected normalized cumulative distribution*** at each time point, based on the model.

If D(n) is less than the established criteria, the model fits the data adequately.

The Kolmogorov-Smirnov test statistic for **n=20**, and p **value=.05** is **0.29408**. Because the maximum value of D(n) is 0.02329, which is less than .29408, the test indicates that the model is adequate.

5. Revisiting the Rayleigh Model

- Defect Arrival Rate (PDF) – the number of defects to arrive **during** time t:

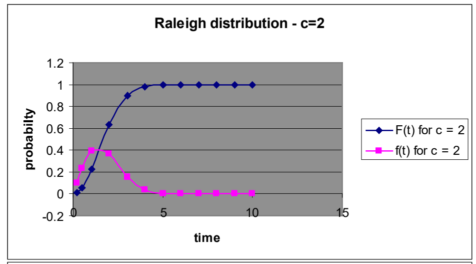
$$f(t) = K\left(\frac{2t}{c^2}\right)e^{-\left(\frac{t}{c}\right)^2}$$

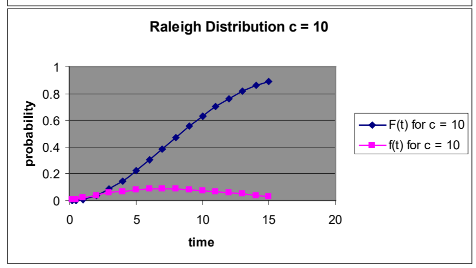
- Cumulative Defects (CDF) -- the total number of defects to arrive **by** time t:

$$F(t) = K\left(1 - e^{-\left(\frac{t}{c}\right)^2}\right)$$

- Where the c parameter is a function of the time t_{max} that the curve reaches its peak
  - $c = t_{max}\sqrt{2}$
  - K=total number of injected defects

Plotting the graphs



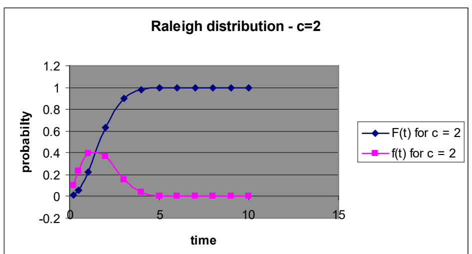


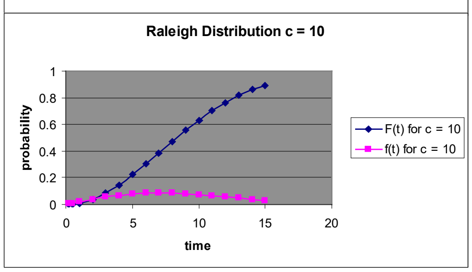
F(t) = probability of 1 defect arriving by time t (K=1 – e.g., total number of defects)

f(t) = probability of defect arriving at time 1 (K=1)

What do these charts mean?

Plotting the graphs (ctd.)





These are all for K = 1.

For case 1,

c=2 $\Rightarrow t_m = \frac{c}{\sqrt{2}} = \sqrt{2} \cong 1.4$

For example, the probability that the defect will arrive at time 2 is ~.39, and the probability that it has arrived by time 2 is ~.62

For case 2,

c=10 $\Rightarrow t_m = \frac{c}{\sqrt{2}} \cong 7$

The percent of defects that have appeared by t_m is:

$$100 \times \frac{F(t_m)}{K} = 100 \times (1 - e^{-0.5}) = 40\%$$

Therefore, ~40% of the defects should appear by time t_m.

Methods for Predicting Arrival Distributions using Rayleigh

- Given *n* data points, plot them
- Determine t_{max} (the time *t* at which f(t) is max)
- Then use the Rayleigh formulae to predict the later arrival of defects:

$$f(t) = K\left(\frac{2t}{c^2}\right)e^{-\left(\frac{t}{c}\right)^2}$$

$$F(t) = K\left(1 - e^{-\left(\frac{t}{c}\right)^2}\right)$$

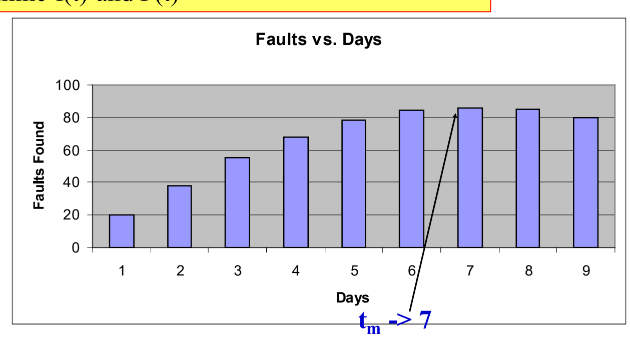
  - Where F(t) is the cumulative arrival rate; f(t) is the arrival rate for defects, and K is the total number of defects.

Method 1: Extremely Simple Method – the 40% rule

- Given that you have defect arrival data, and the curve has achieved its maximum at time t_m (e.g., the inflection point), you can calculate f(t), assuming the Rayleigh distribution.
- The simplest method is to use the pattern that ~40% of the total defects have appeared by t_m. This is a crude calculation, but is a place to start.

Simple Method – Example

Given the data shown below (429 defects by t_m) – determine f(t) and F(t)



Therefore, expected total number of faults = 429×(100/40) = 1072 or~1073

Then you can determine f(t), since you know K and t_m

Simple Method – Example (ctd.)

Since:

$$f(t) = K\left(\frac{2t}{c^2}\right)e^{-\left(\frac{t}{c}\right)^2}$$

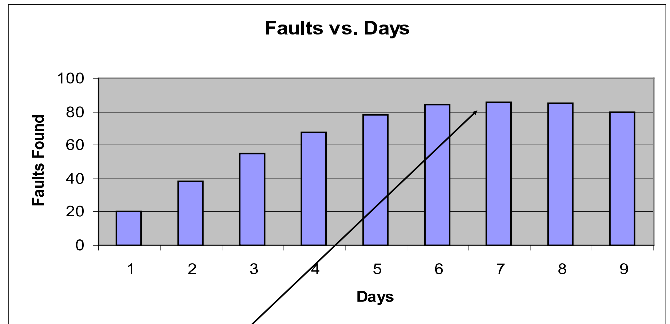
$$F(t) = K\left(1 - e^{-\left(\frac{t}{c}\right)^2}\right)$$

- Then substituting in:
  - $f(t) = 1073 \times \left(\frac{t}{49}\right) \times e^{-\left(\frac{t}{9.9}\right)^2} = 21.93 \times t \times e^{-0.01t^2}$
  - $F(t) = 1073 \times [1 - e^{-0.01t^2}]$
- Then you could plot this out on the same chart and see how well it matches the data.

Predicting Arrival Distributions: Method 2

- You can solve for f(t) by using t_{max} and one or more data points to solve for K and f(t). The simplest way is to take just one data point.

Example: *594 Faults found by day 9*



tmax -> 7

What is the arrival function f(t)?
Need t_{max} and K to determine f(t).

$$\boxed{f(t) = K\left(\frac{1}{7}\right)^2 te^{-\left(\frac{t}{7\times\sqrt{2}}\right)^2}}$$

Method 2 – Example (ctd.)

- Solve for K (use t = 1, defects = 20) =>
  $K=20\times49/e^{(-1/98)}=\sim990$
- You now have an equation:

$$f(t) = (990/49)te^{-(1/98)t^2}$$
$$= 20.2te^{-.01t^2}$$

- Then plot out the equation and use it to predict arrival rates (and also see how well it matches to data!)

Exercise 4.6

- You are now in system test for an online Theater Tickets purchasing software. You have the defect arrival data points below. Assume a Raleigh curve.
  - What do you predict as the total number of bugs in the system? - Use two methods.
  - How many bugs do you predict as being left in the system?
  - What is the equation that predicts the defects?
  - If you shipped at the end of week 6 (and assuming you removed all the defects found at that time), what would you predict as the defect removal efficiency?
- If this is a 5,000 LOC program, what would you predict as the remaining defect density after 6 Months?
- Should you ship after 6 Months? Why or why not?

Month Found	1	2	3	4	5	6
Defects	13	22	25	22	17	5