Software Reliability Evaluation

Radek Mařík

address: ProTyS, s.r.o., Americka 24,
120 00 Praha 2 - Vinohrady
tel.: (+420) 2 254548
fax: (+420) 2 250467
e-mail: rmarik@ra.rockwell.com

URL of lectures:

`http://labe.felk.cvut.cz /~marikr/teaching/indexSQT.html`

The Contents

- Software Reliability - definition
- Software Reliability Models
  - Static Models
    - The Rayleigh Model
  - Dynamic Models
    - The Exponential Model
    - Reliability Growth Models

Software Reliability [Kan95]

- **Reliability** is often defined as the probability that a system, vehicle, machine, device, and so on will perform its intended function under operating conditions, for a specified period of time.
- **Software reliability models** are used to estimate the reliability or the number of latent defects of the software product when it is available to the customers.
- The reasons:
  1. an objective statement of the quality of the product,
  2. resource planning for the software maintenance phase.
- **The criteria variable** under study is the number of defects (or defect rate normalized to lines of code) in specified time intervals (weeks, months, etc.), or the time between failures.

Software Reliability [Kan95]

- **A static model** uses other attributes of the project or program modules to estimate the number of defects in the software.
  - The parameters of models are estimated based on a number of previous projects.
- **A dynamic model** uses the current development defect patterns to estimate end-product reliability.
  - The parameters of the dynamic models are estimated based on multiple data points gathered to date from the product of interest
  - Categories:
    * The entire development process is modeled. The model is represented by the Rayleigh model.
    * The back-end formal testing phase is modeled. The model is represented by the exponential model and other reliability growth models.

The Weibull Distribution [Kan95]

- one of the three known extreme-value distributions,
- the tail of its probability density function approaches zero asymptotically, but never reaches it.
- its cumulative distribution function (CDF):

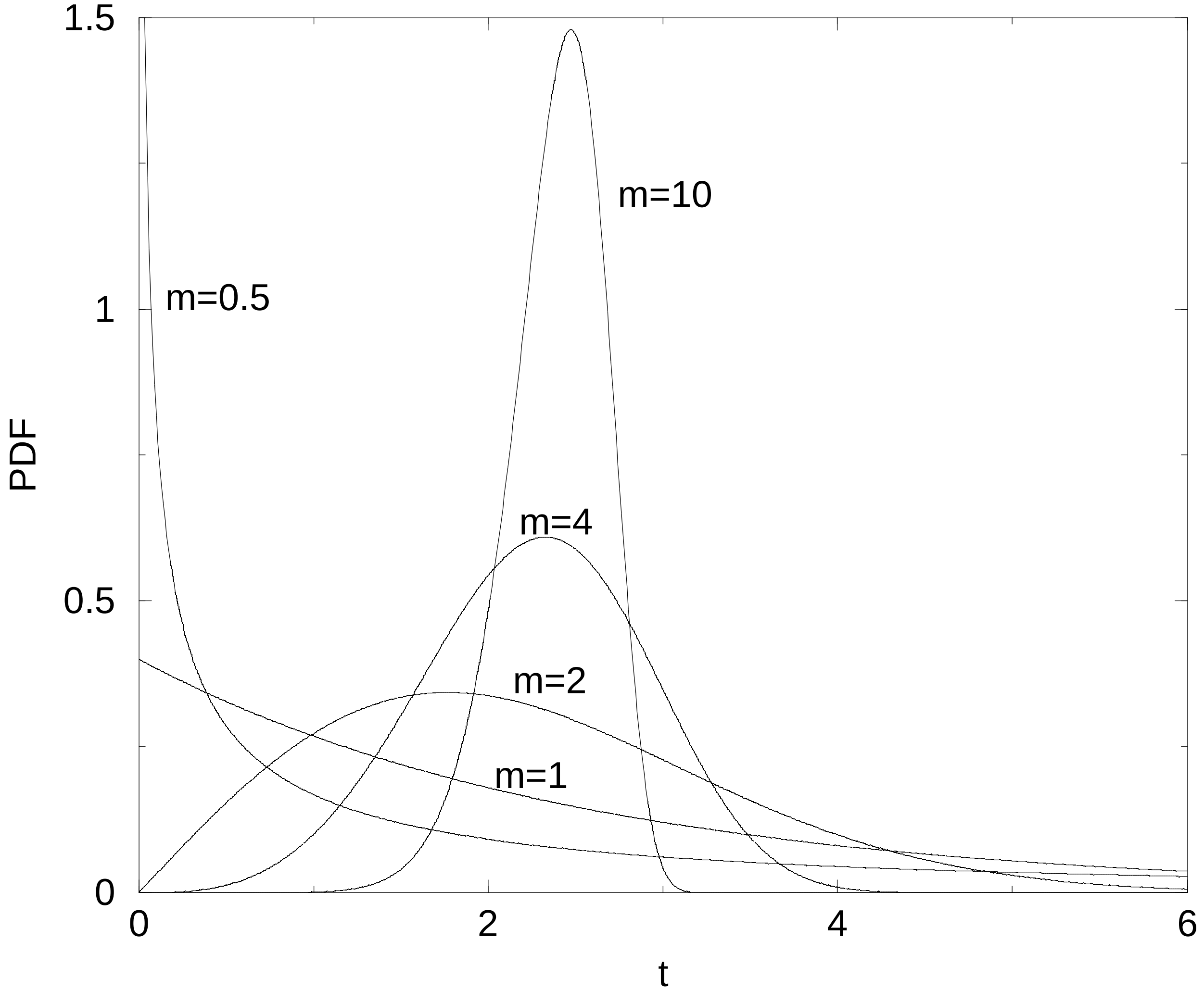
$$F(t) = 1 - e^{-(t/c)^m}$$

- its probability density function (PDF):

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

- where
  - m is the shape parameter,
  - c is the scale parameter,
  - t is time.
- When applied to software, the PDF often means the defect density (rate) over time of the defect arrival pattern (valid defects) and CDF means the cumulative defect arrival pattern.

The Weibull Distribution



The Rayleigh Model [Kan95]

- The Rayleigh model is a member of the family of the Weibull distribution.
- $m = 2$
- its cumulative distribution function (CDF):

$$F(t) = 1 - e^{-(t/c)^2}$$

- its probability density function (PDF):

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

- t_m is the time at which the curve reaches its peak.

$$t_m = \frac{c}{\sqrt{2}}$$

- After t_m is estimated, the shape of the entire curve can be determined. The area below the curve up to t_m is 39.35% of the total area.

The Rayleigh Model in Practice [Kan95]

- In actual applications, a constant K is multiplied to the formulas (K is the total number of defects or the total cumulative defect rate).

$$c = t_m\sqrt{2}$$

$$F(t) = K\left[1 - e^{-(1/2t_m^2)t^2}\right]$$

$$f(t) = K\left[\left(\frac{1}{t_m}\right)^2 te^{-(1/2t_m^2)t^2}\right]$$

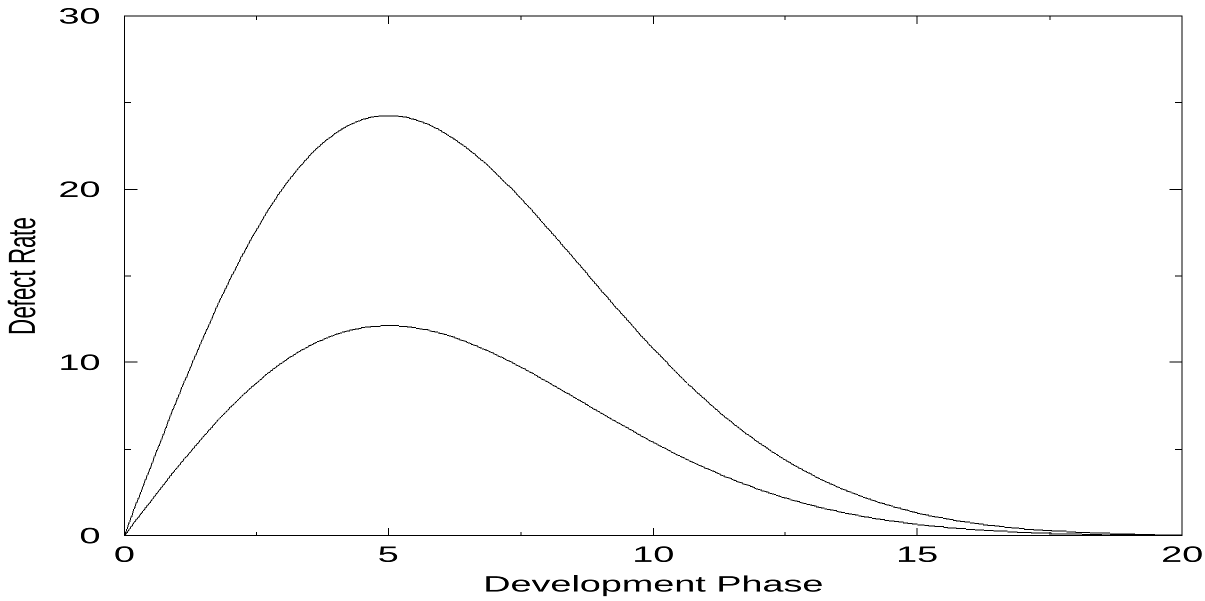
- The software projects follow a life-cycle pattern described by the Rayleigh density curve.
- The defect removal pattern of software projects also follows the Rayleigh pattern.
- The total actual defects are within 5% to 10% of the defects predicted from the model.

Basic Assumptions [Kan95]

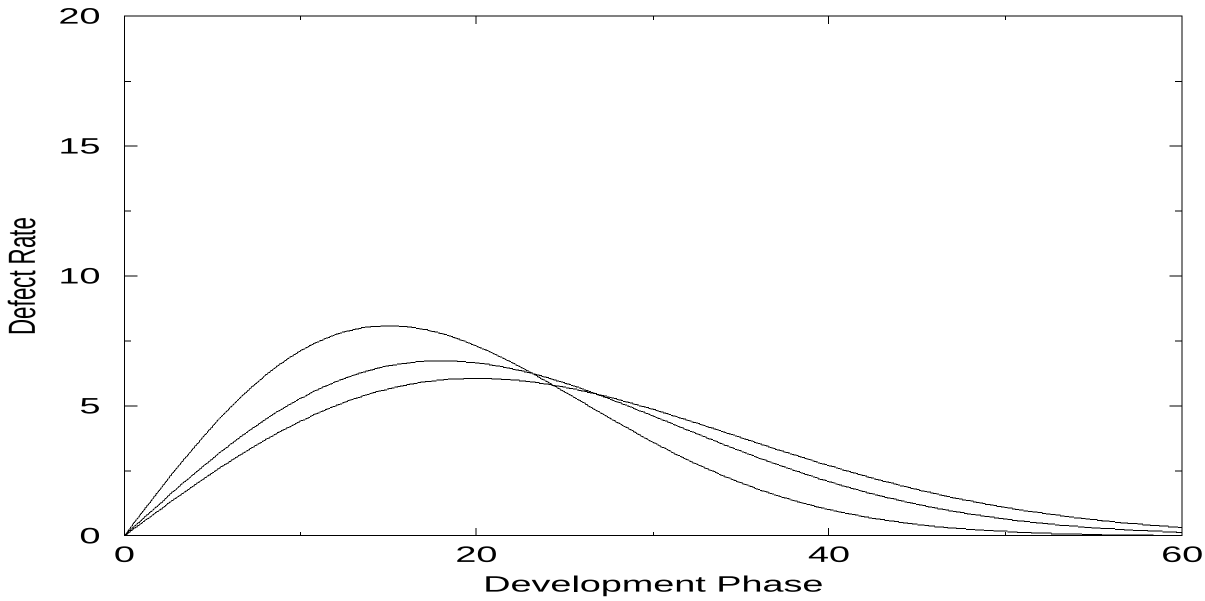
- *The defect rate observed during the development process is positively correlated with the defect rate in the field*
- *Given the same error injection rate, if more defects are discovered and removed earlier, fewer will remain in later states.*
- "Do it right the first time" principle: if each step of the development process is executed properly with minimum errors being injected, the end-product quality will be good. It also implies that if errors are injected, they should be removed as early as possible.

Basic Assumptions - graphs [Kan95]

Correlation:



Defect removal:



Reliability and Predictive Validity [Kan95]

- **Reliability** refers to the degree of change in the model output due to chance fluctuations in the input data.
- The narrower the confidence interval, the more reliable the estimate.
- Larger samples yield narrower confidence intervals.
- Use as many models as appropriate and rely on inter-model reliability to establish the reliability of the final estimates.
- The foremost thing to achieve **predictive validity** is to make sure that the input data are accurate and reliable.
- Model estimates and actual outcomes must be compared and empirical validity must be established.

Exponential Distribution and Reliability Growth Models [Kan95]

- Reliability growth models are usually based on data from the formal testing phases.
- The rationale is that defect arrival of failure patterns during such testing is a good indicator of the reliability of the product when used by customers.
- During such postdevelopment testing, when failures occur and defects are identified and fixed, the software becomes more stable, and reliability grows over time. Therefore models that address such a process are called reliability growth models.

The Exponential Model [Kan95]

- another special case of the Weibull family, $m = 1$
- its cumulative distribution function (CDF):

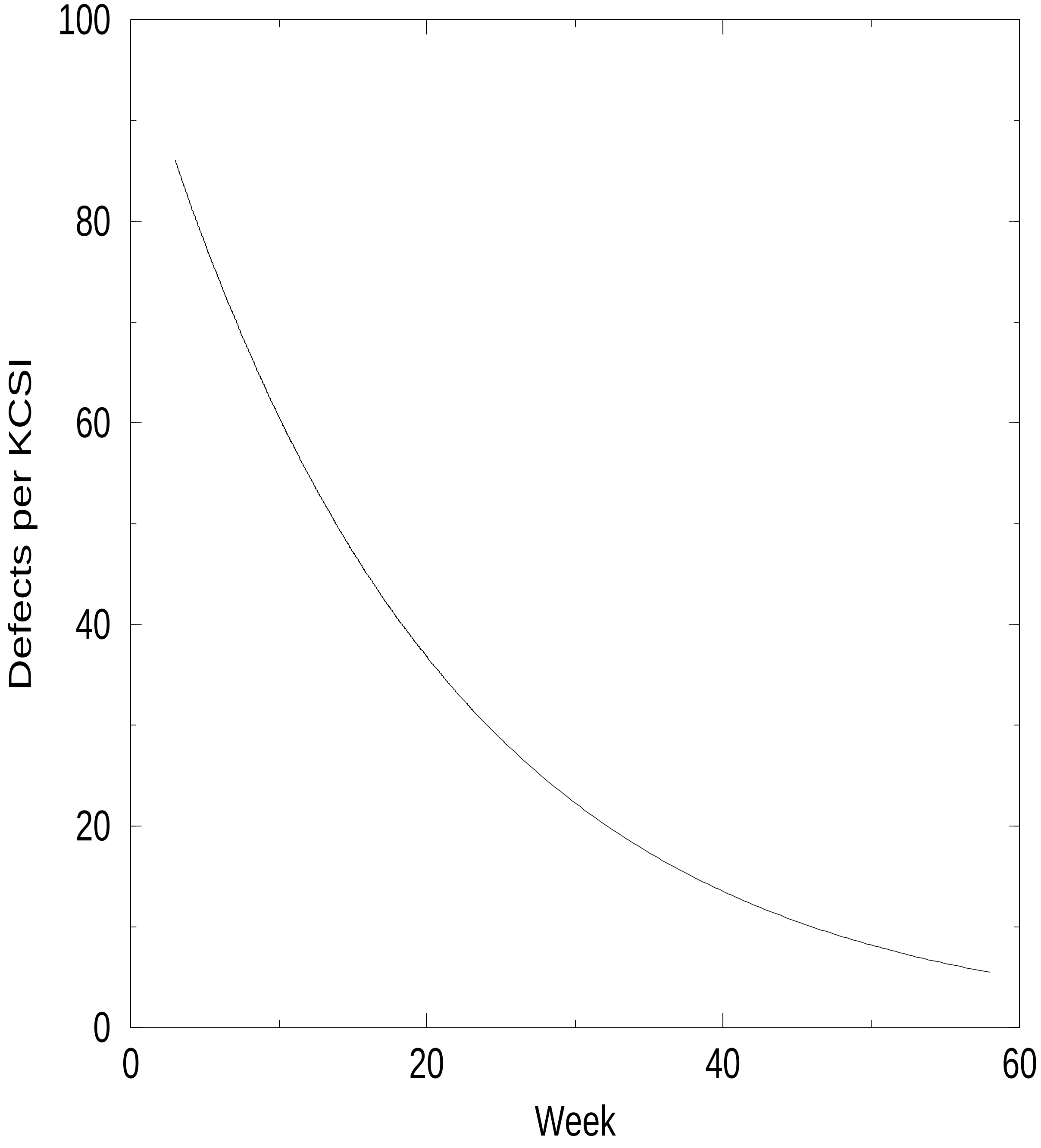
$$F(t) = 1 - e^{-(t/c)} = 1 - e^{-\lambda t}$$

- its probability density function (PDF):

$$f(t) = \frac{1}{c} e^{-(t/c)} = \lambda e^{-\lambda t}$$

- where
  - c is the scale parameter,
  - t is time,
  - $\lambda = 1/c$
- λ is referred to as the **error detection rate** or **instantaneous failure rate** (in statistical terms it is also called the *hazard rate.*
- In actual application, the total number of defects or the total cumulative defect rate K needs to be multiplied to the formulas.

The Exponential Model - Density Distribution



Reliability Growth Models [Kan95]

- Not many models have been tested in practical environments with real data.
- **The time between failures models:**
  - It is expected that the successive failure times will get longer as defects are removed from the software product.
  - A common approach of this class of model is to assume that the time between the $(i-1)$'st and the i'th failures follows a distribution whose parameters are related to the number of latent defects remaining in the product after the $(i-1)$'st failure.
- **The fault count models:**
  - the number of faults or failures (or normalized rate) in a specified time interval.
  - The time interval is fixed *a priori*.
  - The number of defects or failures observed during the interval is treated as a random variable.
  - It is expected that the observed number of failures per unit time will decrease.

Jelinski-Moranda (J-M) Model [Kan95]

- one of the earliest models in software reliability research (1972),
- a time between failures model,
- Assumptions:
  - There are N software faults at the start of testing.
  - Failures occur purely at random.
  - All faults contribute equally to cause a failure during testing.
  - The fix time is negligible.
  - The fix is perfect for each failure that occurs.
- The hazard function at time t_i, the time between the $(i-1)$'st and the i'th failures, is given by:

$$Z(t_i) = \phi[N - (i-1)]$$

- where
  - ϕ is a proportionality constant.
- The hazard function is constant between failures but decreases in steps of ϕ following the removal of each fault.
- As each additional fault is removed, the time between failures is expected to be longer.

Littlewood (LW) Models [Kan95]

- similar to the J-M model,
- It assumes that different faults have different sizes, thereby contribution unequally to failures.
- Larger sized faults tend to be detected and fixed earlier.
- The introduction of the error size concept makes the model assumption more realistic.
- In real-life software operation, the assumption of equal failure rate be all faults can hardly be met, if at all.

Goel-Okumoto (G-O) Imperfect Debugging Model [Kan95]

- The J-M assumes perfect debugging (negligible fix time, perfect fix).
- During the testing stages, the percentage of defective fixes in large commercial software development organizations may range from 1% or 2% to more than 10%.
- The assumption of an imperfect debugging model.
- The hazard function at time t_i, the time between the $(i-1)$'st and the i'th failures, is given by:

$$Z(t_i) = [N - p(i-1)]\lambda$$

- where
  - N is the number of faults at the start of testing,
  - p is the probability of imperfect debugging,
  - λ is the failure rate per fault.

Goel-Okumoto Nonhomogeneous Poisson Process Model (NHPP) [Kan95]

- modeling the number of failures observed in given testing intervals
- The assumptions:
  - The cumulative number of failures observed at time t, $N(t)$, can be modeled as a nonhomogeneous Poisson process - as a Poisson process with a time-dependent failure rate.
  - The time-dependent failure rate follows an exponential distribution.
- The model is given by

$$P\{N(t) = y\} = \frac{[m(t)]^y}{y!} e^{-m(t)},\ y = 0, 1, 2, \cdots$$

- where

$$\begin{aligned} m(t) &= a(1 - e^{-bt}) \\ \lambda(t) &\equiv m'(t) = abc^{-bt} \end{aligned}$$

The NHPP Model [Kan95]

- $m(t)$ is the expected number of failures observed by time t,
- $\lambda(t)$ is the failure density,
- a is the expected number of failures to be observed eventually,
- b is the fault detection rate per fault.
- The number of faults to be detected, a, is treated as a random variable whose observed value depends on the test and other environmental factors. This is fundamentally different from the other models that treat the number of faults to be a fixed unknown constant.
- It has been observed that there are cases in which the failure rate first increases and then decreases (the Goel generalized nonhomogeneous Poisson process model):

$$\begin{aligned} m(t) &= a(1 - e^{-bt^c}) \\ \lambda(t) &\equiv m'(t) = abc^{-bt^c}t^{c-1} \end{aligned}$$

  – b and c are constants that reflect the quality of testing.

Musa-Okumoto (M-O) Logarithmic Poisson Execution Time Model [Kan95]

- The observed number of failures by a certain time, τ, is assumed to be a nonhomogeneous Poisson process.
- The mean value function attempts to consider that later fixes have a smaller effect on the software's reliability than earlier ones.
- The logarithmic Poisson process is claimed to be superior for highly nonuniform operational user profiles, where some functions are executed much more frequently than others.
- The mean value function is given by

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

- where
  - λ_0 is the initial failure density,
  - θ is the rate of reduction in the normalized failure intensity per failure.

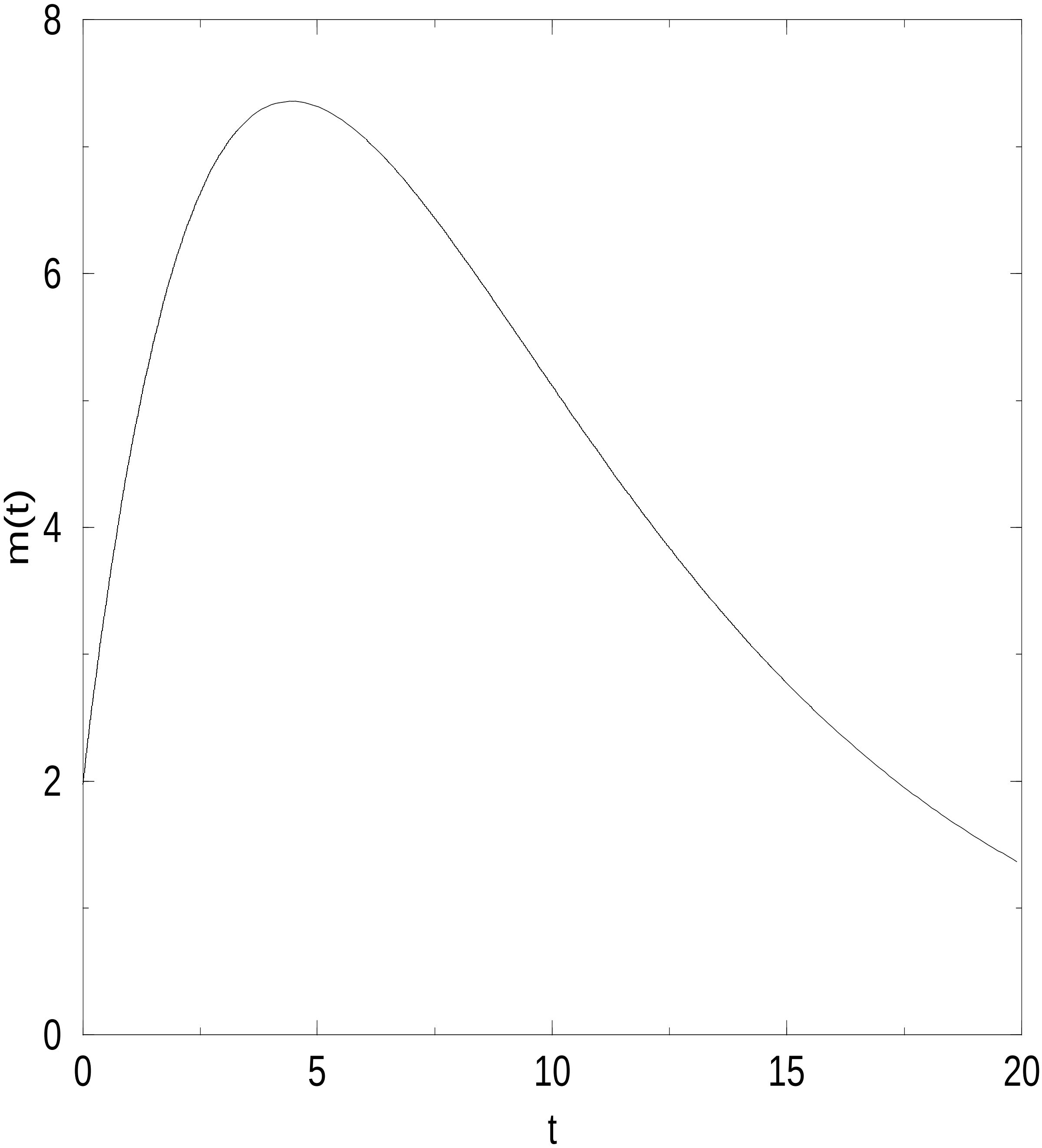
The Delayed S Model [Kan95]

- A testing process consists of not only a defect detection process, but also a defect isolation process.
- Significant delay can occur between the time of the first failure observation and the time of reporting because of the time needed for failure analysis.
- The delayed S-shaped reliability growth model based on the nonhomogeneous Poisson process with a different mean value function to reflect the delay in failure reporting:

$$m(t) = K[1 - (1 + \lambda t)e^{-\lambda t}]$$

- where
  - t is time,
  - λ is the error detection rate,
  - K is the total number of defects or total cumulative defect rate.

The Delayed S Model - the mean value function



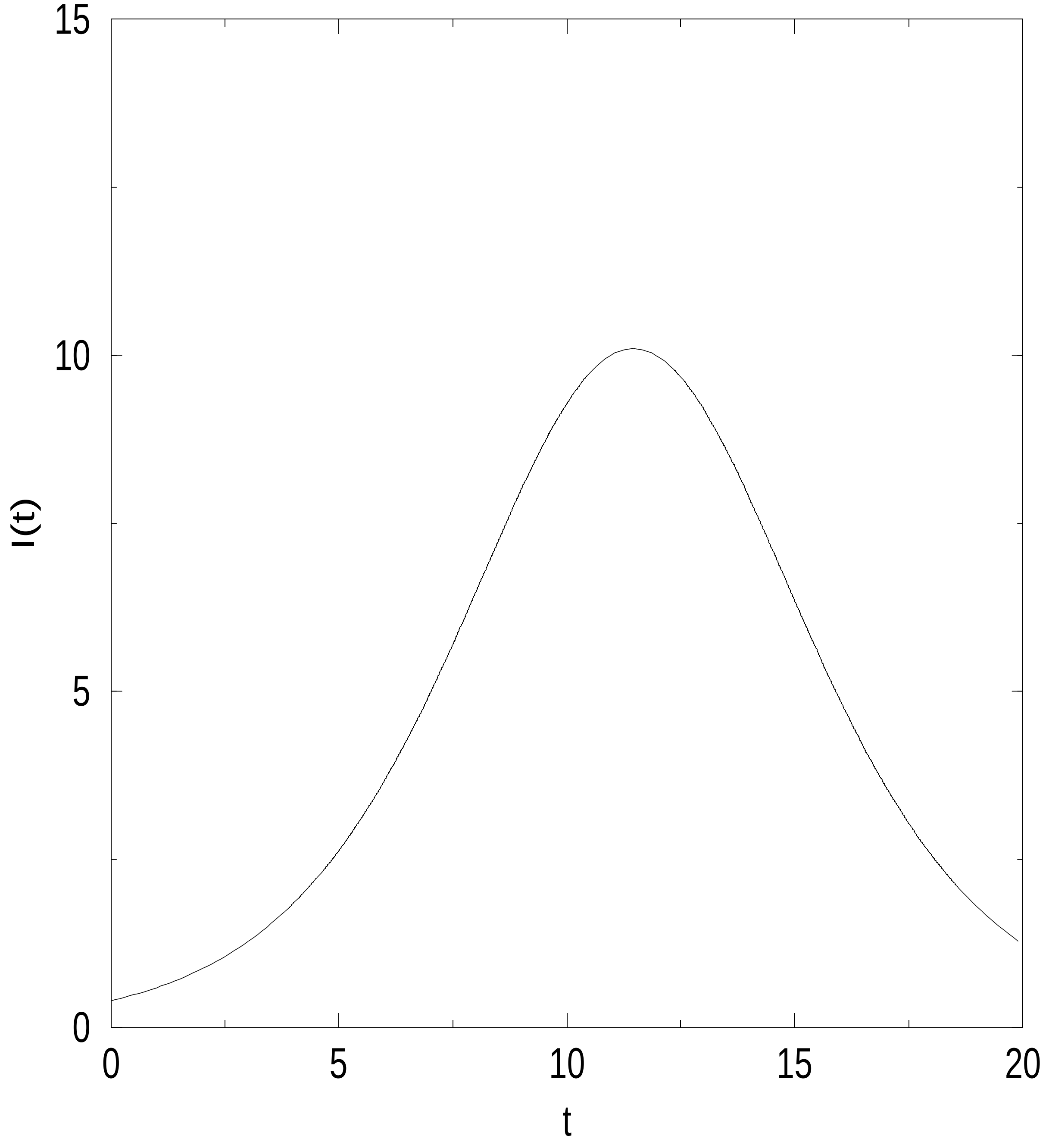
The Inflection S Model [Kan95]

- another S-shaped reliability growth model,
- The model describes a software failure detection phenomenon with a mutual dependence of detected defects.
- The more failures we detect, the more undetected failures become detectable.
- a significant improvement over the assumption of the independence of faults in a program.
- Based on the nonhomogeneous Poisson process, the model's mean value function is

$$I(t) = K\frac{1 - e^{-\lambda t}}{1 - ie^{-\lambda t}}$$

- where
  - t is time,
  - λ is the error detection rate,
  - i is the inflection factor,
  - K is the total number of defects or total cumulative defect rate.

The Inflection S Model - the mean value function



The Delayed S and Inflection S Models [Kan95]

- accounting for the learning period that testers go through as they become familiar with the software at the beginning period of testing.
- The learning period is associated with the delayed or inflection patterns as described by the mean value functions.
- Comparison:
  - The exponential model assumes that the peak of defect arrival is at the beginning of the system test phase and continues to decline thereafter.
  - The delayed S model assumes a slightly delayed peak.
  - The inflection S model assumes a later and sharper peak.

References

[Kan95] Stephen H. Kan. *Metrics and Models in Software Quality Engineering.* Addison-Wesley, 1995.