A LOAD DEPENDENCY MODEL

FOR

SOFTWARE RELIABILITY

Tor Stålhane
Department of Mathematical Sciences
Group for Statistics
Norwegian Institute of Technology
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. INTRODUCTION</td>
<td>6</td>
</tr>
<tr>
<td>2. NOTATION</td>
<td>10</td>
</tr>
<tr>
<td>2.1 Software Terms</td>
<td>10</td>
</tr>
<tr>
<td>2.2 Mathematical Conventions</td>
<td>17</td>
</tr>
<tr>
<td>2.3 Other Notations</td>
<td>19</td>
</tr>
<tr>
<td>3. Requirements for a Software Reliability Model</td>
<td>20</td>
</tr>
<tr>
<td>4. A Rationale for the Load Dependency Model</td>
<td>24</td>
</tr>
<tr>
<td>4.1 Basic Assumptions</td>
<td>25</td>
</tr>
<tr>
<td>4.2 Basic Clarifications</td>
<td>36</td>
</tr>
<tr>
<td>4.3 Paths and Path Selection</td>
<td>48</td>
</tr>
<tr>
<td>4.4 Calendar Time vs Number of Executions</td>
<td>51</td>
</tr>
<tr>
<td>4.5 The Results of Different Goals</td>
<td>55</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
</tr>
<tr>
<td>------------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>5. A Formulation of the Load-Dependency Model</td>
<td>57</td>
</tr>
<tr>
<td>5.1 General Considerations</td>
<td>58</td>
</tr>
<tr>
<td>5.2 Two Theses on Software Reliability Models</td>
<td>71</td>
</tr>
<tr>
<td>5.3 The Estimation of the Total Number of Faults</td>
<td>75</td>
</tr>
<tr>
<td>5.3.1 The Number of Faults at Risk at Site i</td>
<td>75</td>
</tr>
<tr>
<td>5.3.2 The Estimation of Number of Errors at Site i</td>
<td>80</td>
</tr>
<tr>
<td>5.3.3 Environmental Effects on the Failure Intensity</td>
<td>82</td>
</tr>
<tr>
<td>6. Some Failure Time Distributions</td>
<td>92</td>
</tr>
<tr>
<td>6.1 Testing by the project team</td>
<td>94</td>
</tr>
<tr>
<td>6.1.1 Constant Testing Effort</td>
<td>94</td>
</tr>
<tr>
<td>6.1.2 Increasing Testing Effort</td>
<td>95</td>
</tr>
<tr>
<td>6.1.3 Combination of Constant and Increasing Testing Effort</td>
<td>97</td>
</tr>
<tr>
<td>6.2 Use at New Site</td>
<td>99</td>
</tr>
<tr>
<td>6.2.1 General Considerations</td>
<td>99</td>
</tr>
<tr>
<td>6.3 A Suggestion for W(t)</td>
<td>103</td>
</tr>
<tr>
<td>6.4 Two Suggestions for the New Path Probability</td>
<td>105</td>
</tr>
<tr>
<td>6.5 Assumptions Leading to Known Failure Time Distributions</td>
<td>116</td>
</tr>
<tr>
<td>6.5.1 General considerations</td>
<td>116</td>
</tr>
<tr>
<td>6.5.2 Assumptions leading to a Gamma Distribution</td>
<td>119</td>
</tr>
<tr>
<td>6.5.3 Assumptions leading to a Weibull Distribution</td>
<td>122</td>
</tr>
<tr>
<td>6.5.4 Assumptions leading to a Log Normal Distribution</td>
<td>123</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
</tr>
<tr>
<td>-----------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>6.6 Testing by an Independent Test Team</td>
<td>126</td>
</tr>
<tr>
<td>7. Grouped Observations</td>
<td></td>
</tr>
<tr>
<td>7.1 The Multinomial Formulation</td>
<td>134</td>
</tr>
<tr>
<td>7.2 Linearization of the Multinomial Distribution</td>
<td>135</td>
</tr>
<tr>
<td>7.3 An Approximation to $D(ln n)$</td>
<td>137</td>
</tr>
<tr>
<td>7.4 Estimation for the Linearized Model</td>
<td>140</td>
</tr>
<tr>
<td>7.5 Linearization of some Realistic Distributions for $T$</td>
<td></td>
</tr>
<tr>
<td>7.5.1 Gamma Distributions</td>
<td>147</td>
</tr>
<tr>
<td>7.5.2 Log Normal Distribution</td>
<td>149</td>
</tr>
<tr>
<td>7.5.3 Log Gamma Distribution</td>
<td>152</td>
</tr>
<tr>
<td>7.5.4 Other Failure Time Distributions</td>
<td>155</td>
</tr>
<tr>
<td>7.6 Maximum Likelihood Estimation for the Multinomial Model</td>
<td>157</td>
</tr>
<tr>
<td>7.6.1 Estimation of the Parameters</td>
<td>160</td>
</tr>
<tr>
<td>7.6.2 Asymptotic Distribution of the Parameters</td>
<td>164</td>
</tr>
<tr>
<td>7.7 Conditional MLE Applied to some Realistic Distributions</td>
<td></td>
</tr>
<tr>
<td>7.7.1 Gamma Distributions</td>
<td>167</td>
</tr>
<tr>
<td>7.7.2 Log Normal Distributions</td>
<td>167</td>
</tr>
<tr>
<td>7.7.3 Log Gamma Distributions</td>
<td>170</td>
</tr>
<tr>
<td>7.8 Estimation of Time to Deliver</td>
<td>174</td>
</tr>
<tr>
<td>8. Case Studies</td>
<td>176</td>
</tr>
</tbody>
</table>


<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.1 Data from One or Several Life Cycle Phases</td>
<td>180</td>
</tr>
<tr>
<td>8.2 Noise Removal</td>
<td>185</td>
</tr>
<tr>
<td>8.3 Completeness of the Data Sets</td>
<td>191</td>
</tr>
<tr>
<td>8.4 Comparison of Models</td>
<td>194</td>
</tr>
<tr>
<td>8.5 Comparison of Estimation Methods</td>
<td>196</td>
</tr>
<tr>
<td>8.6 Discussion of Some Real Data Sets</td>
<td>202</td>
</tr>
<tr>
<td>9.1 Linearization vs Maximum Likelihood</td>
<td>209</td>
</tr>
<tr>
<td>9.1.1 Gamma Distributed Data</td>
<td>210</td>
</tr>
<tr>
<td>9.1.2 Log Normal Distributed Data</td>
<td>214</td>
</tr>
<tr>
<td>9.1.3 Log Gamma Distributed Data</td>
<td>216</td>
</tr>
<tr>
<td>9.2 Weibull Distribution</td>
<td>220</td>
</tr>
<tr>
<td>10. Bayesian L-D Models</td>
<td>222</td>
</tr>
<tr>
<td>10.1 Bayesian Treatment of the Gamma Distribution</td>
<td>223</td>
</tr>
<tr>
<td>10.2 Bayesian Formulation of the Log Normal Distribution</td>
<td>229</td>
</tr>
<tr>
<td>10.3 Bayesian Estimation of the Total Number of Errors</td>
<td>233</td>
</tr>
<tr>
<td>10.3.1 A Prior Distribution for the Total Number of Errors</td>
<td>233</td>
</tr>
<tr>
<td>10.3.2 Parameter Values for the Gamma Distribution</td>
<td>243</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
</tr>
<tr>
<td>--------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>11. A Complete Bayesian Model</td>
<td>249</td>
</tr>
<tr>
<td>11.1 Some Mathematical Problems</td>
<td>251</td>
</tr>
<tr>
<td>11.1.1 Expansions into Sums</td>
<td>252</td>
</tr>
<tr>
<td>11.1.2 Lindleys Approximation</td>
<td>253</td>
</tr>
<tr>
<td>11.1.3 The Mode Approximation</td>
<td>254</td>
</tr>
<tr>
<td>11.2 The Jelinski-Moranda Model</td>
<td>256</td>
</tr>
<tr>
<td>11.2.1 The Complete Likelihood</td>
<td>256</td>
</tr>
<tr>
<td>11.2.2 The Conditional Likelihood</td>
<td>262</td>
</tr>
<tr>
<td>11.3 The Log Normal Distribution</td>
<td>268</td>
</tr>
<tr>
<td>11.3.1 The Complete Likelihood</td>
<td>268</td>
</tr>
<tr>
<td>11.3.2 The Conditional Likelihood</td>
<td>272</td>
</tr>
<tr>
<td>11.4 The Weibull Distribution</td>
<td>280</td>
</tr>
<tr>
<td>11.4.1 The Conditional Likelihood</td>
<td>280</td>
</tr>
<tr>
<td>11.5 The Gamma Distribution</td>
<td>288</td>
</tr>
<tr>
<td>12. Evaluation of the Bayesian Estimators</td>
<td>289</td>
</tr>
<tr>
<td>12.1 Simulated Data Sets</td>
<td>291</td>
</tr>
<tr>
<td>12.1.1 Log Normal Distribution</td>
<td>291</td>
</tr>
<tr>
<td>12.1.2 Weibull Distribution</td>
<td>295</td>
</tr>
<tr>
<td>12.1.3 The Gamma Distribution</td>
<td>301</td>
</tr>
<tr>
<td>12.2 Real World Data</td>
<td>306</td>
</tr>
<tr>
<td>12.2.1 The Log Normal Distribution</td>
<td>306</td>
</tr>
<tr>
<td>12.2.2 The Weibull Distribution</td>
<td>312</td>
</tr>
<tr>
<td>12.2.3 The Gamma Distribution</td>
<td>317</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>12.3 The Proof of the Pudding</td>
<td>323</td>
</tr>
<tr>
<td>12.4 Evaluation of Bayes Estimates</td>
<td>328</td>
</tr>
<tr>
<td>13. Further Work</td>
<td>330</td>
</tr>
<tr>
<td>13.1 Validation of some Important Assumptions</td>
<td>331</td>
</tr>
<tr>
<td>13.2 The Generalized Gamma Distribution</td>
<td>332</td>
</tr>
<tr>
<td>13.3 A Decision Theoretic Approach</td>
<td>333</td>
</tr>
<tr>
<td>13.4 Bayesian Estimates</td>
<td>334</td>
</tr>
<tr>
<td>13.5 The Effect of Various Software Metrics</td>
<td>336</td>
</tr>
<tr>
<td>13.6 Parameter Variation</td>
<td>337</td>
</tr>
<tr>
<td>APPENDIX A</td>
<td>338</td>
</tr>
<tr>
<td>APPENDIX B</td>
<td>341</td>
</tr>
</tbody>
</table>
Acknowledgement

There is a multitude of people who share the praise (and blame) for this work. Some of them, however, deserve special thanks.

First and foremost I should thank RUNIT and NTH for funding me through all this time (two and a half year). Without them I would probably have done something quite different.

Next in line are my scientific advisers; Bo Henry Lindquist who always has a good idea ready when it is needed and Øystein Skogstad who has fought a long (and often loosing) battle to keep the work, on a level where it can be of use for the practical software engineer.

The work, as it stands, is the sum of Bo H. Lindquists long statistical and academic experience and Øystein Skogstads and my own long experience with software projects. It is my sincere hope that the sum is larger than its parts.

Eli Hagen has transfered the manuscript from handwriting to computer readable form, a long and tedious task, hampered by the fact that she had to use the NOTIS editor and formatting system. She is absolved from absolutly all blame for the way this report is presented, typographically.

Finally, a big Thank You to everybody at the group for Mathematical Statistics and at RUNIT for a lot of encouragement and fruitful discussion. Without them I would have finished this half a year earlier and delivered a lousy product.
Conclusion

This report discusses the problems of

- developing a software reliability model that is
  - complete enough to be useful, both as a basis for estimation and as a vehicle for discussing test strategies and other matters related to testing and validation
  - simple enough to be mathematically tractable and logically transparent

- once the model is available, how can we use it to estimate such key figures as
  - total number of errors
  - expected time to the first period where predicted number of errors is less than a predefined number.

After an introduction and a chapter which contains some basic definitions, both mathematical and software related, the report sets out to discuss:

- how different aspects of user behaviour influence the observed reliability of a software product (chapter 4)

- a software reliability model which enables us to handle the issues raised in chapter 4 (chapter 5)

- how a set of reasonable assumptions on user behaviour leads us to assume that the generalized gamma distribution is a suitable distribution for software reliability (chapter 6)
two different ways to estimate the distribution parameters, called linearization and conditional likelihood respectively (chapters 7 and 9). The two estimation methods are evaluated both with simulated and real data. The real data are derived from a set of case studies, referred in chapter 8.

The two estimation methods can also be formulated within a Bayesian framework. This is discussed in chapters 10, 11 and 12.

The following are the major findings

1) Software reliability is usage-dependent. Thus, in the general case, it is not possible to predict the reliability in one environment from experience in another.

2) The complete software failure process, which includes both a usage process and a general software failure mechanism, can in a very natural way be described by a Markov process. The result is a non-homogeneous Markov chain with transition intensity proportional to the product of the user intensity and the expected probability of selecting input data that activates a new path when the number of executions till now is given.

3) Since the data data sets usually are right censored and the total number of errors is unknown, the usual MLE can not be used. An alternative is to use conditional MLE. Simulation experiments have, however, shown that estimation by linearization is superior to the conditional MLE when the total number of errors is unknown.

4) When the number of observed failures is large (greater than 10, say), mode estimation from the joint posterior distribution is just as good as the standard Bayes estimate
under a square loss functions.

5) Analysis of real world data indicates that the variations in the failure time distribution parameters are small when compared to the variation in errors per (1000) line(s).
1. INTRODUCTION

This chapter sets the scene for the problem of Software Reliability Estimation.

The main problems with software reliability models, as seen from a software engineers point of view, are the following:

- software can, per se, not fail in a stochastic manner as can, for instance, a light bulb

- the reliability of a piece of software depends on how you use it

Even though both IEEE and ISO define software reliability in terms of error content, input and conditions of use (environment), the two last terms are mostly left out in known software reliability models.

When it comes to credibility, these are probably the largest obstacles to a wider acceptance and use of software reliability estimation.

From 1945 to 1986 more than 50 software reliability models have been published [28]. In spite of the fact that software notorious is unreliable when delivered and that software failures have been shown to cause large economic losses, the interest in software reliability models has been low, both with software engineers and software users.

The only viable answer to this puzzle is that "nobody" believe that
the software reliability people can deliver anything useful.

A few examples from the fan mail file of Bev Littlewood [1] will put the problem into its right perspective

(A) "Software reliability models are statistical. Programs are deterministic. If certain input conditions cause a malfunction today, then the same conditions are certain to cause a malfunction if they occur tomorrow. Where is the randomness?"

(B) "I am paid to write reliable programs. I use the best programming methodology to achieve this. Software reliability estimation procedures would not help me to improve the reliability of my programs."

(C) "We verify our software. When it leaves us it is correct."

(D) "I ran your software reliability measurement program on some data from a current project of ours. It said there was an infinite number of bugs left in the program. Who are you trying to kid?"

(E) (same manager as in D, but one week later) "We corrected a couple of bugs and ran the reliability measurement program again. This time it said that there were 200 bugs left. Infinity minus two equals two hundred? Is this the new math?"

(F) "We put a lot of effort into testing. The selection of test data is a systematic process designed to seek out bugs. Reliability estimation based on such test data would be no guide to the performance of the program in a use environment."

(G) "We are writing an air traffic control program. Total system crash would be catastrophic. Other failures range from serious to trivial. Reliability models do not distinguish between failures of differing severity."
Most persons who have tried to sell the idea of software reliability to software engineers or managers have probably heard these remarks over and over again. This does not imply that they should be ignored or rejected out of hand. In fact, if we can not provide convincing answers to those seven objections (and maybe some more), software reliability has but a slight hope of becoming accepted by the software engineering community.

Another important reason why the practitioners ignore software reliability measures is that it is seldom used in a software contract as an acceptance criterion. It seems, however, that reliability measures are on their way into software delivery contracts and that they will be of greater importance in the future for software engineers and software contractors alike. E.g. ESA (European Space Agency) will, in the second half of 88, contract out a research project which shall look into software reliability specification and estimation for all of ESA's software.

This does not imply that various techniques for building reliable software should be discarded. Software reliability measures do not per se make software reliable. What it can do, however, is to ensure the contractor and the customer that they get the product reliability they were aiming at. Cfr. B in the Littlewood fan mail file.

Those engaged in prediction of software reliability will need to attack the problem on three fronts.

- Build a model which mirrors software reliability in such a way that it will gain credibility with the software practitioners. It is especially important to make clear under what conditions the model holds. Cfr. A and G in the Littlewood fan mail file.

- Clarify what an estimate really is. Cfr. D and E in the Littlewood fan mail file.
- Show how the model can cater for the difference between testing and usage. Cfr. F in the Littlewood fan mail file.

The report which follows is supposed to show how these three areas can be covered in a way that conforms to data and explains such anomalies as

- the large variety of forms of the distribution of failures vs time

- the large differences in assessing the reliability of a piece of software at different sites

- the differences between testing and usage and how they affect software reliability estimates.
2. NOTATION

This chapter defines some general terms which will be used later in the report, mostly without further explanation.

Since the report is intended for software engineers and statisticians alike, both parties will find definitions of terms which they consider to be obvious and thus unnecessary.

For the time being, I can only ask the two parties to show a little forebearance

2.1 Software Terms

In this report, terms relating to software and software reliability will be used in conformance with the IEEE Standard Glossary of Software Engineering Terminology [2] and the Draft IEEE Standard on Software Reliability Measurement [3]. For the readers' convenience, the technical terms used in this report are defined on the following pages.

Definitions which are not found in [2] or [3] are marked with an asterisk. These definitions are "unstandard", but will be needed to avoid confusions.

The following organizations have contributed to the definitions:

IEEE: The Institute of Electrical and Electronics Engineers
ISO: International Standards Organization

ANSI: American National Standards Institute

- **computer data.** Data available for communication between or within computer equipment. Such data can be external (in computer-readable form) or resident within the computer equipment and can be in the form of analog or digital signals.

- **corrective maintenance.** Maintenance performed specifically to overcome existing faults. (ISO) See also software maintenance.

- **error.** (1) A discrepancy between a computed, observed, or measured value or condition and the true, specified, or theoretically correct value or condition. (ANSI)

Examples include omission or misinterpretation of user requirements in a software specification, incorrect translation or omission of a requirement in the design specification. This is not a preferred usage. See also failure, fault.

- **execution.** The process of carrying out an instruction or the instructions of a computer program by a computer. (ISO).

- **failure.** (1) The termination of the ability of a functional unit to perform its required function. (ISO) (2) The inability of a system or system component to perform a required function within specified limits. A failure may be produced when a fault is encountered. (3) A departure of program operation from program requirements.

- **failure rate.** (1) The ratio of the number of failures to a given unit of measure; for example, failures per unit of time, failures per number of transactions, failures per number of computer runs. (2) In reliability modeling, the ratio of the number of failures of a given category of severity to a given
period of time; for example, failures per second of execution
time, failures per month. Synonymous with failure ratio.

* fault. (1) An accidental condition that causes a functional
unit to fail to perform its required function. (ISO) (2) A
manifestation of an error in software. A fault, if encountered,
may cause a failure. Synonymous with bug.

* illegal-input message. A message displayed from the system
when the input does not conform to the input definition.

The term is synonymous with the term "error message". This term
will, however, be avoided in order to avoid confusion between
system errors and input errors.

* input. All information to the system from its environment. Cfr.
computer data.

* regression testing. Selective retesting to detect faults
introduced during modification of a system or system component,
to verify that modifications have not caused unintended adverse
effects, or to verify that a modified system or system
component still meets its specified requirements.

* reliability. (1) The ability of an item to perform a required
function under stated conditions for a stated period of time.

* software life cycle. The period of time that starts when a
software product is conceived and ends when the product is no
longer available for use. The software life cycle typically
includes a requirements phase, design phase, implementation
phase, test phase, installation and checkout phase, operation
and maintenance phase, and sometimes, retirement phase.

* It is sometimes convenient to split the operation and mainten-
ance phase into an introduction phase and a work phase.

. **software maintenance.** (1) Modification of a software product after delivery to correct faults. (2) Modification of a software product after delivery to correct faults, to improve performance or other attributes, or to adapt the product to a changed environment. See also corrective maintenance.

. **software reliability.** (1) The probability that software will not cause the failure of a system for a specified time under specified conditions. The probability is a function of the inputs to and use of the system as well as a function of the existence of faults in the software. The inputs to the system determine whether existing faults, if any, are encountered. The ability of a program to perform a required function under stated conditions for a stated period of time.

. **specification.** (1) A document that prescribes, in a complete, precise, verifiable manner, the requirements, design, behavior, or other characteristics of a system or system component. (2) The process of developing a specification. (3) A concise statement of a set of requirements to be satisfied by a product, a material or process indicating, whenever appropriate, the procedure by means of which it may be determined whether the requirements given are satisfied. (ANSI N45.2.10-1973)

. **TAF, TAAF.** Test, Analyse And Fix. This acronym describes the three work phases in debugging. One tests, evaluate the results and then, if the test reveals an error, a fix (corrective action) is performed.

. **test phase.** The period of time in the software life cycle during which the components of a software product are evaluated and integrated, and the software product is evaluated to
determine whether or not requirements have been satisfied.

- **test plan.** A document prescribing the approach to be taken for intended testing activities. The plan typically identifies the items to be tested, the testing to be performed, test schedules, personnel requirements, reporting requirements, evaluation criteria, and any risks requiring contingency planning.

- **testing.** The process of exercising or evaluating a system or system component by manual or automated means to verify that it satisfies specified requirements or to identify differences between expected and actual results.

- **validation.** The process of evaluating software at the end of the software development process to ensure compliance with software requirements. See also verification.

- **verification.** (1) The process of determining whether or not the products of a given phase of the software development cycle fulfill the requirements established during the previous phase. See also validation.

(2) Formal proof of program correctness. (3) The act of reviewing, inspecting, testing, checking, auditing, or otherwise establishing and documenting whether or not items, processes, services, or documents conform to specified requirements. (ANSI/ASQC A3-1978).

### Input Domains

- **D_A**: the domain of all possible input
- **D_L**: the domain of all legal input
- **D_I**: the domain of all illegal input
Note that $D_A = D_L \cup D_I$

$D_F$ : the domain of all input that activates a faulty path through the system

$D_{F,j}$ : the domain of all input that activated a faulty path after execution of the input sequence $X_1, X_2, \ldots, X_j$

Note that $D_F \subseteq D_A$. In addition either or both of the following relations may be true, given that $D_F, D_L$ and $D_I$**.

$D_F \cap D_I^{**}, D_I^{**}$

$X$ : any input to the system. $X \in D_A$

$D_u$ : the domain of the input which has actually been used with the system.

The domains can have a parameter, say $j$, which binds the domain to a specific site or user community.

Eg.

$D_u(j)$ is all possible input which will be applied to the system at site $j$.

In some cases, it is necessary to distinguish not only between sites, but also between phases at each site. Examples of phases are test phase, integration phase, introduction phase and work phase.

Whenever we want to make this distinction, a second index is added.

Eg.
\(D_u(i,1)\) is the input domain in phase 1 at site \(i\). The total input domain at this site, however, is written \(D_u(i)\) and thus \(D_u(i) = UD_u(i,j)\)

**Number of Errors:**

\(N_0\) : the total number of errors in a system at time 0.

\(N_0(j)\) : the total number of errors at risk at site \(j\) at time 0.

\(N(t)\) : the number of errors observed till time \(t\)

\(n(t)\) : error density at time \(t\)

Note that \(N(t) = \int_0^t n(\tau) d\tau\)


\(N(k)\) : the number of errors observed up till and including the \(k^{th}\) execution

\(n_i\) : the number of errors observed in interval number \(i\)

Formally, \(n_i = \mid\{t_j | t_j \in [a_i, b_i]\}\mid\), where \(a_i\) and \(b_i\) are lower and upper time limit respectively for interval no. \(i\).

Note that by making the interval limits sufficiently close, we can get a situation where \(n_i \in \{0, 1\}\).

Note that \(N(k) = \sum_{i=1}^{k} n_i\)

\(N(t)\) : the number of errors remaining in the system at time \(t\)

\(N(k)\) : the number of errors remaining in the system after the \(k^{th}\) execution
2.2 Mathematical Conventions

\{F_t\} : nondecreasing family of \(\sigma\)-algebras which is complete and right-continuous. More informally, \(F_t\) is the family of events whose logical value is assumed to be known at time \(t\). This is the same as the "history" up to and including time \(t\).

\(I(p)\) : the indicator function. If \(p\) is a logical statement, the value is defined as

\[
\begin{align*}
1 & \text{ if } p \text{ is true} \\
0 & \text{ otherwise}
\end{align*}
\]

\(L(.)\) : the likelihood function

\(l(.)\) : the log likelihood function (\(\ln L(.)\))

\(\Lambda(t)\) : occurrence rate at time \(t\)

\(h(t)\) : the hazard function

\(\Lambda_0\) : initial occurrence rate (\(\Lambda(0)\))

\(R(t|s)\) : the system reliability at time \(t\) given that the last observed failure was at time \(s\).

\(\theta_i\) : model parameters
The following notation is used for probability distributions

\[ \text{Exp}(\phi) \quad : \quad f(x) = \phi e^{-\phi x} \]

\[ U(a,b) \quad : \quad f(x) = \begin{cases} \frac{1}{b-a} & \text{for } x \in [a,b] \\ 0 & \text{otherwise} \end{cases} \]

\[ \text{Bin}(a,b) \quad : \quad f(x) = \binom{a}{x} b^x (1-b)^{a-x} \]

\[ G(a,b) \quad : \quad f(x) = \frac{b x^{b-1}}{\Gamma(b)} e^{-a x} \]

\[ \text{Gamma}(a,b) \quad : \quad f(x) = \frac{a^x}{x!} e^{-a} \]

If one or more of the parameters do not matter or are not currently of interest, they are replaced by a dot.

Eg.

\[ X \sim U(\ldots) \]

means that \( X \) has a uniform distribution, but the parameters do not matter at the present.
2.3 Other Notations

\( t \): time in general

\( t_i \): occurrence time for the manifestation of the \( i^{th} \) error. The errors are numbered in the order of occurrence. Note that the occurrence times are sorted so that

\[ i > j \iff t_i > t_j \]

\( T_i \): starting time for test/useage period \( i+1 \)

\( t_{\text{acc}} \): the starting time of the first period where expected number of failures reported is less than some predefined limit (time of acceptance)

\( l \): number of lines in the system

\( W(t) \): the number of executions performed up till time \( t \)

\( w(t) \): \( \frac{\delta}{\delta t} W(t) \)
3. Requirements for a Software Reliability Model

This chapter sketches the basic requirements which a Software Reliability model must fulfil in order to gain credibility in the software community.

The two main conclusions are that

- we must be able to model the different ways software are used
- the model must clearly spell out the assumptions we have made

In order to answer the question in the heading, we first have to consider a related question, namely: "Why do we need a model in the first place?"

A clear summary of the role of models is taken from [4].

1) Our interpretation of the data depends on the theory in question. We find what we are looking for, and even more important: We do not find what we are not looking for. A model will help us to avoid the "ditch of empiricism".

We are made conscious of our choices and have to spell out clearly how the theory justifies the choices we make.

2) We can uncover interesting questions. The model can help us to put forth new hypotheses and ideas for research.
3) The model can make us aware of factors which otherwise would have been forgotten or ignored.

4) By using a model we can acquire knowledge of relationships which otherwise are only available in very limited amounts. We are made aware of lacunae in our knowledge and thus stimulated to further research.

5) If the model is clear it will be a pedagogical tool which aids us explaining and communicating our ideas.

6) When the assumptions are clearly spelled out, other people have a fair chance of checking on the results.

To sum up: We need the model as a framework for making ourself and others conscious of the choices we have made in our interpretation of reality.

With the above mentioned six points in mind we may proceed to lay down the requirements for a software reliability model.

On the topmost level, we need a model which is

- easy to comprehend, so that people can understand how the testing and maintenance process, as they know it, is mirrored in the model

- mathematically tractable

Due to the credibility gap so amply demonstrated by Littlewoods fan mail file (cfr. chapter 1), the first condition is probably more pressing than the second.

Praiseworthy as the two aforementioned requirements are, they are not enough. We must thus add a third requirement, namely
- the model must be easily adaptable to the data commonly reported by a user site

This rather mundane requirement is important if the model shall be of any help to the software engineers. Attractive as it may seem, a model which takes as input the number of executions or elapsed (CPU) time between errors will be of little use outside a controlled testing environment.

Thus, if the model shall be of any use to the software community, we must add four more criteria

- the model must be able to handle input on the following forms
  - number of failures per day, week, month etc.
  - number of failures observed in time intervals of uneven lengths

- the model must be statistically robust, i.e. one or more faulty observations must not lead to "wildly incorrect" answers

- the model must be able to handle data where errors are ordered in one or several error severity categories

- the model must take care of the differences arising from
  - differences between behaviour during test and usage. In the first case, data are selected so as to provoke a failure, while in the other case, they are selected according to a specific work situation
  - different usage. I.e: different people or environments have different tasks and thus select different data

These criteria must be balanced against the need for a model which
is easy to grasp. This problem of opposing needs leads to the following consideration:

We should start with a simple model and only add complexity to it if this significantly increases the predictive power of the model. A good rule of thumb seems to be to ignore anything that changes the result by less than five percent.
4. A Rationale for the Load Dependency Model

This chapter is long, but important. It starts with a discussion of some basic assumptions which are necessary for the software reliability model which is presented in chapter 5.

Next, it discusses the role of input domains, error domains and corrections in a software system. We introduce the notion of a path and arrive at the first theorem on software reliability.

Last, but not least, the chapter contains a discussion of how the users' aim influences his behaviour when confronted with an erroneous software system.

The rationale for the Load Dependency model (L-D model) consists of four parts.

We start by stating the basic assumptions of the model. Then we go on to some basic clarifications. The points raised there are the most important groundwork for the L-D model.

Next, we discuss the role of number of executions and how to bridge the gap between the number of executions and calendar time.

In the last part we discuss how the reason for executing a piece of software will influence the input data selection and the reaction to a failure.

Throughout chapter 4 we will use a simple, two-dimensional model to describe the ideas of input domains and error domains. The oldest reference to this model we have been able to track down, is [58] which is from 1975.
4.1 Basic Assumptions

For the rest of this report we will make the following assumptions.

A: There exists a stable specification which describes the desired behaviour of the system. (See chapter 2.1.) This specification will, on a high level of abstraction, consist of descriptions of

- classes of input data
- the system's reaction to input data from each class

E.g. For each set \{a,b,c\} the system will return the pair \{x1,x2\} which is the solution to the equation \(ax^2+bx+c=0\).

The specification bisects the input space as shown in fig. 4.1a.
Bisection of the input space by the specification

Fig. 4.1a)
The following short program is provided as an example:

```plaintext
real x, y, a, b;

read(x, y);

b := sqrt(x-y);

a := x/(y-12)+b;

write(a);
```

The domains used in fig. 4.1a will be as shown in fig. 4.1b
B: Errors manifest themselves through one or more failures (see chapter 2.1). As soon as a failure is reported from a site, the users there receive one of the following from the personnel responsible for corrective maintenance (cfr chapter 2.1):

- a (temporary) correction or fix
- an advice on how to circumnavigate the problem

In essence, this guarantees that any error manifestation is reported only once from any one site.

C: Regression testing (see chapter 2.1) is used throughout the
life cycle. This implies that the correction of an error

. really removes that error
. does not bring back an error which has previously been removed

There is, however, always a chance that a correction introduces new errors into the system.

D: The systems considered in this paper are memory-less in the following sense:

Let \{X_i\} be a data set for the system under consideration. Then, if the input sequence \(X_1, X_2, \ldots\)
\(X_n\) all give correct results, so will any other sequence \(X_{\pi_1}, X_{\pi_2}, \ldots, X_{\pi_n}\) for any permutation \(\pi\) of \(\{1, \ldots, n\}\)

If we consider systems on a transaction base, all systems which are reset after each transaction are memoryless. This is, however, the case only for simple batch oriented systems like stand alone compilers, linkers and editors and programs for numerical analysis. As soon as we introduce a compilation database, the compiler is no longer a memoryless system in the abovementioned simple sense.

There are three ways to treat this problem:

. we can ignore it, and thus confine ourselves to truly memory-less systems

. we can ignore it for the present and then later discuss the ramifications of this ignorance

. we can extend our notion of a data set to include all input that have brought the system to its present state.
The last point, attractive as it may be, is clearly impractical, since it could lead us to state that a system only get exposed to one single data set in its entire lifetime, i.e. the union of all data sets used. We are thus, nolens volens, forced to use one of the two first approaches.

We have found few data which can indicate how serious the memory problem is. [15] seems to indicate that for data base systems, most errors (132 out of 133 in the cited sample) are memory related.

E: We count only errors observed by the user. This implies that we count faulty paths and not the total number of errors in the system. This assumption is necessary when we consider the following:

- if a path contains several errors, the first error will cause a failure and the other errors can not be reached before the first error is removed.

- due to assumption D (regression testing), the rest of the errors on that path will be found during correction of the first error and will thus not be observed by the users.

F: The software system is stable in the sense that no new functionality is added to the system. This ensures us that we are considering the same system (except for error corrections) throughout.

Note that this implies that there must exist mechanisms which ensure that all system functions are included from the start.

G: The number of executions in any given period and the selection of input data are independent of the number of errors found earlier.
These assumptions are made so that we can model executions and failure events as two independent processes.

The non-fulfilment of the assumptions A-G will distort the model which we develop. The following is a short summary of the effects of non-fulfilment.

A: "There exists a stable specification ..." Without at least a rudimentary specification, any collection of legal program statements (including no statements) will do.

Specifications do, however, come in various levels of detail. E.g.

1) Solve the second order equation
   \[ ax^2 + bx + c = 0 \] for all \( \{a, b, c\} \)

2) Solve the second order equation
   \[ ax^2 + bx + c = 0 \] for all \( \{a, b, c\} \).
   If \( a = 0 \), then both \( x_1 \) and \( x_2 \) must be set to \(-c/b\).

The way the specification is formulated will influence the number of errors. Take the input \( \{0, 1, -1\} \). For specification 1, the result \( \{1, 0\} \) is OK, for specification 2 it is an error.

Since the specification decides the number of errors, it follows that any change in the specification or the users interpretation of the specifications, may change the number of errors in the system. Another nasty side effect of a change in the specification is that something that was not an error yesterday, may be so today. The importance of this assumption follows from these facts.

B: "The users receive a correction and/or advice on how to avoid the error when it is reported".

When this assumption is not fulfilled, we can risk that
some errors are reported several times

some errors are not reported because "nothing happens anyway"

Both of these effects will distort the estimation process and thus, the fulfilment of assumption B should be considered critical to the model.

C: "Regression testing".

If regression testing is not used, we run the risk of getting

the same error reported several times

old, earlier corrected, errors reported again

Both of these effects will distort the estimation process. Since it, at least in principle, is possible to weed out errors which have been reported earlier and thus not include them in the counts, this can not always be done, and assumption C is thus critical to the model, although not as critical as assumption B.

D: Memoryless systems.

If the system in fact has a memory, so that

\{x_1\}, \{x_2\} \neq \{x_2\}, \{x_1\}

the model developed could be in serious trouble. The problem is that we can not be sure how it will affect the model.

One way out is to resort to the solution that instead of the input data \{x_1\} and \{x_2\} we have the input sets \{x_1, x_2\}, which fails and \{x_2, x_1\} which is OK.
This also highlights another problem, concerning transaction-oriented and interactive systems. Here, at least from a theoretical point of view, an input sequence can consist of

1. one command or transaction
2. all commands needed to get a certain job done
3. all commands used during one session

The memory(less) property will have a different influence depending on how we choose our input definition. Due to the problems raised above, we have chosen not to include database systems in our model.

E: "We count only errors observed by the user".

As with assumption D, this assumption is concerned with input sequencing. A simple example will make this clear:
If the user first uses input \( \{x_1\} \) and then \( \{x_2\} \), he will experience only one error since \( \{x_1\} \) will provoke error \( b \) and the correction process will also lead to correction of \( d \) in order to get a correct result for the test data set \( \{x_1\} \). \( \{x_2\} \) will then behave correctly.

If the user first runs \( \{x_2\} \) only the error \( d \) will be corrected. Error \( b \) will then be left to make \( \{x_1\} \) fail.

As with assumption D, it is difficult to see how this will affect the failure model.

F: Stable software.

If this condition is not fulfilled, any continuous software
reliability model will get into trouble.

For a piece of software where functionality is added or removed there can exist no single reliability model. The assumption is thus critical to the model.

G: The independence between execution intensity, data selection and failure events.

As will be seen in chapter 5.1, this assumption is necessary for arriving at our theorems 2 and 3. It is thus of prime importance for our formulation of the load-dependency model.
4.2 Basic Clarifications

In order to build a model which has a fair chance of being accepted by the software practitioners, the model must mirror the following facts.

A: Software can only fail when in use. This implies that the number of executions (see chapter 2.1), not the time, must be the basic unit of the independent variable in the model.

B: The randomness in software system behaviour can only originate from randomness in input data. Any input data will activate a predeterminable path through the system. This path may or may not contain a fault. We have the following equivalence:

\[
\text{execution with input } X \text{ fails} \quad \text{(4.1)} \quad \Rightarrow \\
\text{input } X \text{ activates a path containing one or more faults}
\]

Clarification B leads to another formulation of fig. 4.1:
The mapping between input and output

Fig. 4.2

With the notation defined in chapter 2, we can reformulate equation (4.1) as

execution with input $X$ fails $\iff X \in D_F$  \hspace{1cm} (4.2)
C: The input domain actually used will vary from site to site. What one site consider to be of vital importance may be considered a minor function somewhere else. Thus:

\[
\text{The system performs without errors at site } i
\]

\[
\Rightarrow
\]

The system is free of errors.

The situation can be illustrated by yet another variation of fig. 4.1.

![Diagram showing input domains and illegal input]

Legal input

Input which activates a faulty path

**Fig. 4.3**

If we take the situation described by fig. 4.3 as our starting point, an error-free situation is obtained by changing fig. 4.3 to fig. 4.4 a or 4.4 b.
User adaption

Fig. 4.4a
The input domain of site i

Illegal input

Legal input

Input which activates a faulty path

Corrective maintenance

Fig. 4.4b

Fig. 4.4a shows error free performance by adapting the use to the system, while fig. 4.4b shows error free performance by removal of the errors which are inside the input domain of this site.

Thus

The system performs without errors at site i

\[ D_A(i) \cap D_F = \emptyset \]  

(4.4)
The situation in fig. 4.4.b corresponds to a situation where all reported failures are corrected promptly, while fig. 4.4a describes a situation where failure reports are answered only after a long time, if ever.

For most systems, the attention a failure report gets will depend on factors like

- the complexity of the fault. Small, easy to fix faults will usually get more attention than those which are difficult to fix.

- whether the system is in the acceptance phase or in regular use.

Fault reports in the acceptance phase will always get top priority, while faults reported after acceptance will need longer time. The reason for this is, among other things, that personnel must be taken away from other work to do the corrections.

For the above mentioned reasons, the situation will often develop as shown in fig. 4.5a and 4.5b.
The input domain of site i

Faults found after acceptance

Input which activates a faulty path

Faults found during acceptance

Fig. 4.5a
Let us by the term "execution no. \( j \)" mean the execution of the system with input \( X_j \epsilon\mathcal{D}_A (i) \). From what is said in chapters 4.1 and 4.2 it follows that the probability of a failure depends on

- past experience with the system
- the users intentions when selecting data for input

With this in mind, (4.2) can be formulated as
execution no. \( j \) fails at site \( i \)

\[ \iff \]

\[ X_j \in D_A(i) \bigcap D_F, j-1 \]

The systems reliability (see chapter 2.1) at site \( i \) after the execution of datasets \( X_1, X_2, \ldots, X_{n-1} \) can thus be defined as

\[ R_n = 1 - P(X_n \in D_A(i) \bigcap D_F, n-1) \]  \hspace{1cm} (4.6)

We can see from (4.6) that high reliability after \( n \) executions can be achieved in two ways:

- by making \( D_F, n-1 \) small, that is by corrective maintenance (see chapter 2.1)

- by making the probability of selecting an \( X_i \) in \( D_F \) small. This can again be achieved in several ways.

E.g.

- definition of a site-dependent standard usage subset.
  (Recommended practice)

- by word of mouth. (The do's and don'ts for system X)

In real life, both of these mechanisms are at work simultaneously. A failure and the subsequent correction will change the software from the situation shown in fig. 4.3 to one of the following situations
The error is removed from the system 4.6a

A new error is inserted in a part of the system which is not used at this site fig. 4.6b

A new error is inserted into another part of the system which is used at this site

Fig. 4.6c
When looking at fig. 4.6a, b and c, one can wonder if a system can ever be errorfree. That this will indeed happen, under rather weak assumptions, can be shown in the following way:

The correction process can be described by the expression

\[ N_r(k) = N_r(k-1) - 1 + Z_k \]  

(4.7)

where \( Z_k \) is i.i.d. random variables, describing the number of new errors introduced by correction no. \( k \). Since we after the \( k^{th} \) correction has a new error population depending on the distribution of \( Z_k \), the number of errors can be viewed as a branching process.

If we denote the mean and variance of the number of offspring by \( \mu \) and \( \nu \) respectively, we have that [9]

\[
\mu_n = \mu_0^n \]  

(4.7-1)

\[
\nu_n = \sigma^2 \nu_0^n \frac{1 - \mu_0^n}{1 - \mu_0} 
\]

It is easy to verify that \( \mu_n \) and \( \nu_n \) approaches 0 when \( n \rightarrow \infty \) if \( \mu_0 < 1 \)

\[
\mu_0 = E(Z_k) \]  

(4.7-2)

\[
\mu_0 = \sum_{i=0}^{\infty} i P_i, \quad P_i = P(Z_k = i) \]  

(4.7-3)

In order to reach an error free system in the end with probability 1, we must have that
\begin{equation}
\sum_{i=0}^{\infty} ip_i < 1 \\
(4.7-4)
\end{equation}

In addition we must impose the condition that the debugging-correction process is not stopped as long as there are errors left.
4.3 Paths and Path Selection

If we disregard the situation described in 4.6c, we have the following situation after a certain amount of corrective action:

\[ D_U(i) \quad D_A(i) \quad D_A \quad D_F \]

Fig. 4.7

The input space \( D_A \) can be partitioned according to the path that is activated.

Let \( S_k \) be a path through the system. Then we define a subdomain in \( D_A \), called \( B_k \) as

\[ B_k = \{ X | X \in D_A, \; X \text{ executes } S_k \} \]  \hspace{1cm} (4.8)
If there are a total of $K$ such subdomains, we can write

$$D = \bigcup_{k=1}^{K} A_k B_k$$  \hspace{1cm} (4.8-1)

"New path" is defined by the statement:

$$S_i \text{ is a new path } \iff S_i \in \{S_1, \ldots, S_{i-1}\}$$  \hspace{1cm} (4.8-2)

where the paths are numbered according to the sequence in which they are executed.

From the assumption of regressive testing it now follows that

$$X_i \text{ causes an error } \Rightarrow X_i \text{ activates a new path}$$  \hspace{1cm} (4.8-3)

Thus,

$$P(X_i \text{ produces a failure } | F_{i-1} ) =$$

$$P(S_i \text{ is new, } S_i \cap D_{F,i-1} \neq \emptyset | F_{i-1})$$  \hspace{1cm} (4.8-4)

$$P(X_i \text{ produces a failure } | F_{i-1} ) =$$

$$P(S_i \cap D_F \neq \emptyset | F_{i-1}, S_i \text{ is new})$$

$$P(S_i \text{ is new } | F_{i-1})$$

(4.9)

Our main starting point follows from (4.9).
Theorem 1

\[ P(X \text{ produces a failure } | F_{i-1}) = \]
\[ P(X \text{ executes a new path } S | F_{i-k}) \]
\[ P(S \text{ contains errors } | F_{i-1}) \]
4.4 Calendar Time vs Number of Executions

As stated in 4.2-A, the number of executions should be the basic independent variable in any credible software reliability model. The problem is that the number of executions is only available during inhouse testing. The customer usually reports only the number of failures per day/week/month or the date (and time) for each failure.

This implies that we need a function which describes the number of executions per day, week or month. We will denote this by the function \( w(k) \), so that

\[
w(k) = \text{number of executions in calendar period } k
\]  \hspace{1cm} (4.10)

It seems reasonable to assume that \( w(k) \) will, among other things, depend on

- the user environment
- the kind of task(s) that the user group is currently undertaking
- how long the system has been used at this particular site
- how easy it is to use the system

The following figures show, in a sketchy manner, some examples.
Software Testing

Fig. 4.8

A new, not userfriendly, system

Fig. 4.9
A new, userfriendly, system

Fig. 4.10

A system for printing pay checks

Fig. 4.11

The figures indicate that the number of errors reported as a function of calendar time will depend very strongly on how the
system is used.

Eg. The pay check system may only fail in the periods when the user is printing pay checks, for instance once every month.
4.5 The Results of Different Goals

A user and a tester approach a software system with quite different intentions. The user wants to solve a problem, while the tester wants to uncover errors. This difference will result in different ways to select input data.

This again implies that the probability mechanism for the process \( \{X_1, X_2, \ldots, X_n\} \) and thus for \( \{S_1, S_2, \ldots, S_n\} \) will be different in the two cases.

The tester's criterion:

Choose \( X_i \) in such a way that \( S_i \in \{S_1, S_2, \ldots, S_{i-1}\} \). I.e. \( S_i \) is new.

The user's criterion:

\( X_i \) is needed to solve a problem.

If the system is very large and very opaque, the tester's criterion can be difficult to achieve. The intention is, however, always present.

Another difference which is just as important as the first one, is the reaction to a failure. For the tester, a failure is an incitant to probe further along the same lines. For the average user, however, a failure acts as a warning sign and motivates him to steer clear of this type of input if possible.

Both of these differences work together so that the number of
failures reported per run will be incompatible for the two cases. This again implies that any useful model must reflect these differences. We see that theorem 1 (chapter 4.2) does this quite nicely by defining reliability as the product of the probability of selecting a certain path and the probability that this path contains an error.

The first probability will be a function of the users aims, the second is a function of the intrinsic reliability of this particular software system.
5. A Formulation of the Load-Dependency Model

In this chapter we present our main results. If you are short on time and are going to read one chapter only, this is it!

First, we present some general results which lead us to a software reliability model where the failure times are modelled as a non-homogenous Markov chain. The intensity of the process will depend on the use intensity and the way the software is used.

This result then sets the scene for the two theses on software reliability and a discourse into the problem of estimating \( N_0 \). This discussion will focus on the role of the different project phases.

After having stated the basic assumptions for a software reliability model, we will proceed to see how such a model can be formulated in statistical terms.

Having done this, we will state our theses on software reliability models and some implications of these theses.

The last two subchapters will describe some attempts to obtain a practical, working software reliability model.
5.1 General Considerations

To proceed beyond the simple formulation of theorem 1 we must make two assumptions, one concerning the probability of finding a new path $S_k$ and one concerning the probability that $S_k$ contains an error.

Let $S_1, S_2, \ldots$ be the executed paths in order of execution. We then have

$$F_{k-1} = \sigma(S_1, S_2, \ldots, S_{k-1}) \quad (5.1)$$

Further more, we will use the notations

- $N_{pk}$: number of faulty paths before execution of path $S_k$ and the following corrections if $S_k$ contains an error
- $N_p$: total number of paths through the system
- $A = 1/N_p$

If we assume that each path has the same probability of containing an error we can write

$$P(S_k \text{ contain errors}|F_{k-1}) = \frac{N_{pk}}{N_p - |S_{k-1}|} \quad (5.1-1)$$

Since the total number of paths usually will be large compared to the number of executions, it is reasonable to use the approximation
\[ N_p = N_p - |S_{k-1}| \]  \hspace{1cm} (5.2)

Further more, from the counting procedure assumed in assumption E, we have that

no. of faulty paths before executing \( S_k = N_0 - N(k-1) \)  \hspace{1cm} (5.3)

Thus

\[ P(S_k \text{ contains errors}|F_{k-1}) = A[N_0-N(k-1)] \]  \hspace{1cm} (5.4)

(5.4) assumes that all remaining faults are at risk. This is by no means an uncontroversial statement. For our model, however, it follows from assumption E. The \( r(N_r) \) function which Ohba somewhat artificially introduces to obtain a \( S \)-form on the \( n(t) \) path [?] is thus not necessary in our model.

We will introduce two independent processes (ref assumption G in chapter 4.1):

\( N(n) : \) a non-homogeneous Markov chain in discrete time, describing the number of remaining errors after \( n \) executions

\( M(t) : \) a non-homogeneous Poisson process, describing the number of executions in the interval \([0,t]\). The intensity of the process is \( m(t) \).

We can now prove that a process \( Z(t) \), defined as

\[ Z(t) = N_r[M(t)] \]  \hspace{1cm} (5.5)

is also a non-homogeneous Markov chain, but in continuous time.

Let us introduce the ordered set of time points \( \{t(1)\} \),
\[ t(2), t(n) \] and \( t > t(n) \). We then need to prove that

\[ P[Z(t) = j | Z(t_{(r)}) = i, r \in [1, n]] \] \hspace{1cm} (5.6) \]

depends only on \( t(n), i, t \) and \( j \).

We will use the general result

\[ P(A|B) = \sum_i P(A|B, C_i)P(C_i | B) \] \hspace{1cm} (5.7) \]

If \( C_i \) is independent of \( B \), we further have that

\[ P(A|B) = \sum_i P(A|B, C_i)P(C_i) \] \hspace{1cm} (5.8) \]

We can now use (5.8) to write (5.6) as

\[ P[Z(t) = j | Z(t_{(r)}) = i, r \in [1, n]] = \]

\[ \sum_{y, \{y_i\}} P[Z(t) = j | Z(t_{(r)}) = y, M(t_{(r)}) = y, W(t) = y, r \in [1, n]] \]

\[ P[M(t) = y, M(t) = y, r \in [1, n]] \] \hspace{1cm} (5.9) \]

since \( M(.) \) and \( N(.) \) are independent. Furthermore, since

\( Z(t) = N[M(t)] \) and \( M(t) = y \), we can write:
Since $N(t)$ is a Markov chain, the transition probability can only depend on $y$. Thus

\[ P(z(t) = j | z(t - 1) = i, r, \{1, n\}) = P(z(t) = j | y, M(t) = y, r, \{1, n\}) \]

(5.10)

The last probability on the right hand side of (5.11) can be written as

\[ P(M(t) = y, M(t - 1) = y, r, \{1, n\}) = \sum P(M(t) = y, r, \{1, n\}) \]

(5.12)
(5.11) can now be written as

$$ P[Z(t) = j | Z(t_{(r)}) = i, r \in [1, n]] = $$

$$ \sum_{y=0}^{y_0} \sum_{y=0}^{y_n} \sum_{y=0}^{y_{n-1}} \prod_{y=0}^{y_n} \prod_{y=0}^{y_{n-1}} P[M(t_{(r)}) - M(t_{(i-1)}) = y-y] $$

(5.13)

Note that we have set $t_{(0)} = 0$ and $y_0 = 0$. The last $n-1$ sums on the right hand side of (5.13) is the total probability of having $y_n$ events in $[0, t_{(n)}]$ since the sets $\{t_{(i)}\}$ and $\{y_i\}$ represent all possible partitions of $[0, t_{(n)}]$ and $y_n$ respectively and each product is the probability of observing a certain partition.

$$ P[Z(t) = j | Z(t_{(r)}) = i, r \in [1, n]] = $$

$$ \sum_{y=0}^{y_0} \sum_{y=0}^{y_n} \prod_{y=0}^{y_n} \prod_{y=0}^{y_{n-1}} P[M(t_{(r)}) - W(t_{(n)}) = y-y] $$

(5.14)

Since the transition probability on the left hand side now only depends on $t$, $t_{(n)}$, $y$ and $y_n$, we see that $Z(t)$ is a non-homogeneous Markov chain.
For $t$ sufficiently close to $t_{(n)}$ we have

$$t = t_{(n)} + \Delta t, \Delta t \to 0$$

$$y - y_n = \begin{cases} 0 & \text{with probability } 1 - m(t)\Delta t \\ 1 & \text{with probability } m(t)\Delta t \end{cases}$$

when we ignore $o(\Delta t)$.

The last sum on the right hand side of (5.14) has thus only two terms and we have:

$$\sum_{y = y_n} P[.]P[. | .] =$$

$$P[M(t + \Delta t) - M(t_{(n)}) = 0]. P[N_{(n)}(y) = j | N_{(n)}(y) = i] +$$

$$P[M(t + \Delta t) - M(t_{(n)}) = 1]. P[N_{(n)}(y + 1) = j | N_{(n)}(y) = i]$$

(5.15)

Since $j > i$, we have that

$$P[N_{(n)}(y) = j | N_{(n)}(y) = i] = 0$$

(5.16)
\[ P[\cdot | \cdot] \rightarrow m(t) \sum_{y=0}^{\infty} \frac{P[N_r(y+1) = j | N_r(y) = i]}{n^r} \Delta t \rightarrow 0 \]

By combining (5.14) and (5.17) we get

\[ \lim_{\Delta t \rightarrow 0} P[Z(t+\Delta t) = j | Z(t) = i] = \lambda_{ij}(t) \]

\[ \lambda_{ij}(t) = m(t) \sum_{y=0}^{\infty} P[M(t) = y] \sum_{i=0}^{\infty} \frac{P[N_r(y+1) = j | N_r(y) = i]}{n^r} \]

By combining theorem 1 and equation (5.4), we get

\[ P(X_{y+1} \text{ produces a failure} | F_y) = P(X_{y+1} \text{ executes a new path } S_k | F_y) A_{y} N_r(y) \]

Under the assumption that failures are corrected as they occur (assumptions B and C in chapter 4.1) we have that

\[ X_{y+1} \text{ produces a failure } \Leftrightarrow N_r(y+1) = N_r(y) - 1 \]

If we further introduce the notation

\[ P_{\text{new}}(y) = P(X_{y+1} \text{ executes a new path } S_k | F_y) \]

we can write

\[ P[N_r(y+1) = N_r(y) - 1 | F_y] = A_{y} N_r(y) P_{\text{new}}(y) \]

since the probability of selecting a certain input only depends on
the number of executions performed and not on the number of observed failures (cfr. assumption G, chapter 4.1). The only dependency left in \( F_y \) is thus \( y \) and \( N_r(y) \) (5.19-4) is then a Markov chain and we can write

\[
P(N_{r}(y+1) = j-1|N_{r}(y) = j) = \left\{ \begin{array}{ll}
P_{r, j-1}^{\text{new}}(y) A \\
1 - P_{r, j}^{\text{new}}(y) A
\end{array} \right.
\]

(5.20)

This follows from theorem 1 and (5.4). (5.19) then becomes:

\[
\lambda_{j, j-1}(t) = m(t) A_{j} \sum_{y=0}^{\infty} P(M(t) = y) P_{r, j}^{\text{new}}(y)
\]

(5.21)

or

\[
\lambda_{j, j-1}(t) = m(t) A_{j} E\{P_{r, j}^{\text{new}}[M(t)]\}
\]

(5.22)

\( \lambda_{j, j-1} \) is the failure rate at time \( t \).

It is easy to prove the following general result:

A death process with rate \( \eta(t) \) from state \( j \), given \( Z(0) = N_0 \) can be written as the sum of \( N_0 \) i.i.d. death processes with initial state 1 and rate \( \eta(t) \).

The proof consists of two steps. We must prove that
1) the sum of the sub-processes is a Markov process

2) both processes have the same transition rate.

The state of the new process is given as the number of sub-process that is in state 1. Since the \(N_0\) process are independent, each single transition intensity will only depend on the state of this process and not on what has happened to the \(N_0 - 1\) others. We thus have a Markov process.

For the sum of processes, we have

\[
P(\text{transition in } [t, t+\Delta t]) = \sum_{i=1}^{j} P(\text{transition for process no. } i) \cdot o(\Delta t) \tag{5.22-1}
\]

where \(j\) is the number of processes still in state 1. Thus:

\[
P(\text{transition in } [t, t+\Delta t]) = j \cdot \varrho(t) \Delta t \tag{5.22-2}
\]

which is the same transition probability as for the original process.

In our case we have

\[
\varrho(t) = A_m(t) \cdot E[P_{\text{new}}[M(t)]] \tag{5.23}
\]

For pure death processes we have the general result

\[
P(\text{be in state 1 at } t) = e^{-\int_0^t \varrho(u) du} \tag{5.23-1}
\]
where \( q(t) \) is the transition rate from 1 to 0. From (5.23-1) it follows that

\[
P(\text{transition time} > t) = e^{-\int_0^t q(u) \, du}
\]  \hspace{1cm} (5.23-2)

\[
1 - F(t) = e^{-\int_0^t q(u) \, du}
\]  \hspace{1cm} (5.23-3)

\[
f(t) = q(t) e^{-\int_0^t q(u) \, du}
\]  \hspace{1cm} (5.24)

Since the sum of \( N_0 \) independent processes is equivalent to the original process with transition rate \( j_p(t) \) from state \( j \), it follows that the sojourn times, and thus the failure times are i.i.d. with \( f(t) \) as defined in (5.24).

Thus, for our software failure model, we get

**Theorem 2**

\[
f(t) = A m(t) E\{P_{\text{new}}[M(t)]\}.
\]

\[
\exp\{ -A \int_0^t m(t) E\{P_{\text{new}}[M(t)]\} \, dt \}
\]

In order to arrive at a more "useful" expression for \( f(t) \) we will, however, make a detour to (5.20). Let \( y \) be approximated by a continuous variable \( x \). For a rationale of this transition see [9], chapter 5.
\[ P[N (X+\Delta X)=j-1 | N (X)=j] = \int_P (X) \Delta X \]  
\[ \qquad \text{new} \quad r \quad r \]  
\[ (5.25) \]

and thus

\[ Q_1(x) = jAP_{\text{new}} (x) \]
\[ (5.26) \]

If we now use the same approach as led us to (5.24) and thus to theorem 2, we get

\[ f(x) = AP_{\text{new}} (x)e^{-\int_0^x P_{\text{new}} (u)du} \]
\[ (5.27) \]

We can consider (5.26) and (5.27) as our software failure model in the number-of-executions domain. If we assume that

\[ x = \int_0^t w(u)du, \quad x = W(t) \]
\[ (5.28) \]

where \( w(.) \) and \( W(.) \) now are deterministic functions, we get
Theorem 3

\[ f(t) = A P_{\text{new}} \left[ W(t) \right] w(t) \exp\{ -A \int_0^t P_{\text{new}}(u) \, du \} \]  \hspace{1cm} (5.29)

Remember that \( W(0) = 0 \).

If we let \( n_i \) be given as

\[ n_i = N \int_{i-1}^t f(u) \, du \]  \hspace{1cm} (5.29-1)

we see that this theorem fulfills the basic requirement of a software reliability model in that \( n_i \) is zero if

- the system is not used, \( W(t) \) is constant
- no new path is found, \( P_{\text{new}}(\cdot) = 0 \)
- the system has no errors to start with, \( N_0 = 0 \)

This model also explains Spildes paradox, stated by Dag Spilde at Norsk Data as follows:

"The majority of errors are reported in our commercially most successful system."

Since (5.29) contains fewer terms than (5.27) it follows that a no. of executions domain model should do better than a calendar time domain model. That this is the case has been argued by Musa and Okumoto in [27]. The two domains used by Musa and Okumoto were calendar time and CPU time. It is, however, clear that the
correlation between CPU time and number of executions is much larger than the correlation between calendar time and number of executions, particularly since no. of executions has no connection to calendar time in the general case.

The load-dependency model has most of the models referred to [28] as special cases. In this connection, two models are special, namely P. Holagers model of 1981, which is the first to make use of the idea that only new input can fail and S. Yamadas model of 1986 which has a failure rate proportional to W(t). These two models can be obtained from (5.29) by setting P_{new}(.)=constant and P_{new}(.)=no. of new test/total no. of tests, respectively.
5.2 Two Theorems on Software Reliability Models

From what is said in the previous chapters it should be clear that the following factors will influence the number of errors reported per time unit:

- the number of remaining errors

- the use intensity, described by \( w(t) \)

- the way the system is used, described by \( G_{\text{use}}(\cdot) \), say

From (5.26) it follows that the expected error intensity can be written as a product of

- the number of faults at risk

- a discovery probability function, called \( P_{\text{new}}(\cdot) \)

If we assume that \( P_{\text{new}}(k) \) is deterministic (i.e. a function of \( k \) only), we have the multiplicative intensity model introduced by O. Aalen [25].

Thus,

\[
\Lambda(t) = Y(t) \alpha(t) \quad (5.30)
\]

\[
Y(t) = \sum_{i: T_i < t} I(t \leq T_i) \quad (5.31)
\]

\[
\alpha(t) = \Phi P_{\text{new}}[W(t)]w(t) \quad (5.32)
\]

If we measure time by number of executions, \( \alpha(t) \) reduces to
\[ \alpha(t) = \Phi P_{\text{new}}(t) \]  

(5.33)

The \( P_{\text{new}}(.) \) have mostly been ignored by hardware and software people alike. An exception is E. Demko [26] in his study of the reliability of car batteries. Instead of the traditional value \( \lambda_{\text{op}} \), which is the operating failure rate, he uses:

\[ \lambda = \lambda_{\text{op}} \left( 1 + \frac{\lambda}{\lambda_{\text{op}}} \theta_{\text{off}} + \frac{\lambda}{\lambda_{\text{op}}} \theta_{\text{c}} \right) \]  

(5.34)

where we have used

- \( \lambda_d \): failure rate when off
- \( \lambda_c \): failure rate when cycling
- \( \theta_{\text{off}} \): off time / on time
- \( \theta_{\text{c}} \): no. of cycles / on time

This could also be written

\[ \lambda = \lambda_{\text{op}} G_{\text{use}}(.) \]  

(5.35)

The user profile determines on time, off time and number of cycles. For software, which in this respect is more simple, we have that

\[ G_{\text{use}}(.) = P_{\text{new}}(.) b(i) \]  

(5.36)

where \( b(i) \) is a function which selects \( N_o(i) \) out of the total number of errors in the system \( N_o \).

If we leaf through the failure models described in [28] and look at the models which use the \( Y(t) \) described in (5.31) we see that their \( \alpha(t) \) falls into one of the following groups
\[ \alpha(t) = \phi \]

\[ \alpha(t) = \phi \frac{\delta}{\delta t} W(t) \]

\[ \alpha(t) = \phi \lambda(t) \]

\[ \alpha(t) = \phi (\tau_i - \tau_{i-1}) \]

(5.37)

They all fit readily into (5.21-3). From this it is easy to understand why

- most models fit some data, mostly the data used by those who present the model

- most models get into severe trouble when applied to error data from other systems or other types of environments.

As a consequence of this, I will state the following theses on software reliability:
1: The failure rate (cfr. chapter 2.1) can only be found when we know how the system will be used. That is:

- the usage profile, $P_{\text{new}}(\cdot)$
- the usage intensity, $w(t)$

2: Since $N_0$ is a characteristic of the system and its history and $P_{\text{new}}(0)$ a characteristic of the user environment under consideration, we have that $N_0$ is independent of $\theta_i$ for all $i$

This implies that $N_0$ can not be estimated from $\{\theta_i\}$

Some comments on the theses.

- 1) follows directly from the discussion in chapters 4.2-4.4. All research projects which have used a large variety of data sources in an attempt to find the best software reliability model have concluded that "there ain't no such thing". (See for instance [30,31]).

- 2) is just common sense

- the two theses presented here is in reality the same as the ideas that lead IBM/FSD to stress the need for realistic test data, preferably real world data taken from the organization where the system will be installed. [51].
5.3 The Estimation of the Total Number of Faults

From the discussion in chapter 4.2 it follows that no single site is likely to experience all errors. Cfr. fig. 4.3 for the typical situation. This statement holds also for the project or test team at the producers site.

The following chapters will present a model for \( N_0(i) \), that is, the total number of faults at risk at site \( i \). After this we will go on to discuss the role of the site and life cycle phase in the estimation of \( N_0 \).

5.3.1 The Number of Faults at Risk at Site \( i \)

Although it seems that corrective maintenance will be with us for a long time yet, it is definitively no hit with the users.

Primarily, they want a system without faults and if they can not get that, at least they want what ever faults that are left to pop up as seldom as possible.

A faultless system can be achieved by

- exhaustive test, that is: all paths through the system have been tested

or

- site exhaustive test, that is: all paths through the system that will be used at the new site have been tested.
If this cannot be done, a prospective buyer would at least want the test data to be drawn from the same distribution as the real data.

In most practical cases, neither of the two first requirements can be met. We are thus, nolens volens, stuck with the last one.

Software contracts which have stated any testing requirements besides the obvious

"The software shall be thoroughly tested"

have tried to cater for this by specifying that the test data shall be representative for the way the user will use the software. An attempt on a more precise algorithm for generating software test data has been suggested by B. de Neumann [40].

Let us now consider the situation after test and validation at the development site:
Let $D_F(i)$ be the error domain for site $i$. As a special case, let us use the following conventions

- the development and test site is denoted site 1.

- after test and validation the software is installed at sites 2, 3, ... in sequence

- $D_F = D_F, 0$

It thus follows that $D_u(1)$ is the set of all test data used before the first customer installation, while $D_u(i), i > 1$ are
customer data domains.

If we use the notation \( N_o(i) \) for the number of faults that are at risk at site \( i \), it follows that

\[
D_{F,0} \subseteq D_u(1) \Rightarrow N_o(i) = 0 \text{ for all } i > 1
\]  

(5.38)

This is, however, a manifest optimistic assumption. Most likely we will have that

\[
D_{F,0} \not\subseteq D_u(1)
\]  

(5.39)

and thus, after testing and validation

\[
D_{F,S_1} = D_{F,0} \setminus D_u(1)
\]  

(5.40)

For site \( i \), the number of errors will be

\[
N_o(i) = |D_u(i) \cap D_{F,S_1}|
\]  

(5.41)

One way to ensure faultless operation is to have

\[
D_u(i) \supseteq D_u(1), \quad i > 1
\]  

(5.42)

or, assuming that all reported errors are corrected for all previous sites

\[
D_u(i) \supseteq \bigcup_{j=1}^{i-1} D_u(j)
\]  

(5.43)

This corresponds to the programmer lore statement

"Never use anything that hasn't worked somewhere else"
\[ D_{F,S_i} = D_{F,0} \setminus \bigcup_{j=1}^{i-1} D_j \]  

(5.44)

and thus, in the same way as for (5.41), we have that

\[ N_o(i) = |D_u(i) \cap D_{F,S_i}| \]  

(5.45)

From (5.45) it follows that we will have more confidence in a software system the more it has been used. If we are the first site to use the system, we will have confidence in its operation only if we know the test data to be relevant for our way to use the system.

We can thus conclude that

Test data give information on how the system will perform at our site only if one of the two following conditions are met:

1. the test data are exhaustive

2. the test data mirror the way we will use the system

In all other cases, the test results are close to worthless.

This does not imply that the testing itself is worthless, since all testing and the following corrections will reduce \( D_F \) so that

\[ a > b \Rightarrow |D_{F,a}| \leq |D_{F,b}| \]  

(5.46)

That is, the fault domain tends to decrease as the number of executions grow. (Cfr. equations (4.7) to (4.7-4)).
5.3.2 The Estimation of Number of Errors at Site i

From thesis no. 2 it follows that we can not use the following approach

1) use the failure times \( \{t_i\} \) to estimate the parameters in \( f(\{t_i\}, \theta_1, \ldots, \theta_p) \)

2) estimate \( N_o \) by the function \( h(\theta_1, \ldots, \theta_p) \)

We are thus forced to use the number of failures per time interval \( (n_k) \) to estimate \( N_o(i) \).

The fig. 5.3 shows the heart of the problem. What we really want from our estimation algorithm is to be able to predict the future form of \( n_k \). Given the data indicated in figure 5.3, there seems to be too little information available to choose between the three suggested forms indicated by a, b and c. The influence on \( N_o \) will be enormous as can be readily seen from the figure 5.3, since \( N_o \) is equal to the total area under \( n_k \).
A further problem is that the amount of censoring on our data \( \{t_i\} \) is unknown. Thus our estimates of \( \theta_i \) are rather uncertain.

There seems to be only two ways out of this problem

- we can obtain additional information on the function \( G_{\text{use}}(.) \) (Cfr. (5.36)). This may be done by

  looking at other installations which are expected to behave in the same way as the current site
. studying the plans for introduction and use of the system at this site

. looking at the history of other, comparable systems

- we can wait until a clear trend in the $n_k$ is observable.

In order to make any progress, it seems to be necessary to study how the user environment affects $n_k$.

5.3.3 **Environmental Effects on the Failure Intensity**

Besides the number of errors at risk for site $i$, defined by (5.29), the $P_{new}(.)$ will also influence the failure intensity $n_k$. Thus, $n_k$ will vary

- from phase to phase in the project

- from site to site out in the field

- between user groups at a specific site

When we try to estimate the lifetime distribution $f(t)$ or $f(k)$, we are really trying to estimate $G_{use}(.)$ and $W(t)$. The assumption that these two functions can be cast into analytic expressions of reasonable complexity is already quite debatable.

Since any change in the user group or the tasks to be performed will affect both $G_{use}(.)$ and $W(t)$, we can only deal with $n_k$ on a per phase basis (cfr. the software life cycle definition in chapter 2.1). Furthermore, we need to assume that the staff does not change during a phase. For phases with long duration, this is perhaps unreasonable.
We will, however, for the present assume that \( G_{use}(.) \) and \( W(t) \) are determined by

- the personnel employed

- the tasks performed. This will to a large extent be determined by the current phase in the software life cycle.

When the user environment changes, so will \( W(t) \) and \( G_{use}(.) \). This will cause a discontinuity in the underlying distribution for \( n_k \), often in the form of a sharp rise. This development has been reported from several projects. As an example, fig. 5.4 shows \( n_k \) from the MARK II project at the JPL [29]
The arrow marks the time when the system is transferred to the user environment (week 12). We see that the underlying distribution of $n_k$ gets a sharp discontinuity at that point.
This is quite natural, since both $G_{\text{use}}(.)$ and $W(t)$ will change from test to use. Since the bipartition function in $G_{\text{use}}$ changes (cfr. (5.36)), we see that $N_o$ is split into $N_o(1)$ and $N_o(2)$, and since

$$N_o(1) = D_{F,0} \cap D_u(1)$$  \hspace{1cm} (5.47)$$

$$N_o(2) = D_u(2) \cap \{D_{F,0} \setminus D_u(1)\}$$  \hspace{1cm} (5.48)$$

we see that $N_o(1)$ and $N_o(2)$ must be estimated separately from the data shown in fig. 5.5-a and 5.5-b respectively.
The change in environment does not have to be as drastic as the transition from test to use. The transition from one project phase to another is more than enough to make $n_k$ change its form. The data shown below are taken from [39]
The shape of $n_k$ in the three project phases can be given several interpretations.

- validation phase
  
  the increase is due to increasing test effort
the increase is due to increasing familiarity with the system and thus an increasing probability of selecting data that fails

- integration phase

only faults due to the integration phase are left ("The sum is worse than its parts")

the personnel employed in the integration phase are less imaginative or have less system knowledge than the personnel employed in the validation phase

- operational phase

the "few" errors not discovered in the two earlier phases were found during the first three weeks of operation

so many errors were found during the first three weeks (about 400) that almost nobody dare use it except for already tested data.

That the development sketched in fig. 5.7 is not always followed is readily shown by other data from the same source ([39]).
The system shown in fig. 5.7 has a sharp discontinuity in the validation/acceptance transition. There is, however, no evidence of a discontinuity in the acceptance/integration transition. As in the previous case, the reasons for (lack of) discontinuities are legion.

Let us return to the system shown in fig. 5.6. The number of errors
reported in each phase seems reasonable enough. As should be expected, we have that

\[ N(\text{validation, acceptance}) > N(\text{integration}) > N(\text{operation}) \]

where \( N(\cdot) \) is the number of errors found.

The problem lies in the shape of \( n_k \) and specifically in the sharp discontinuity between validation and integration. Two different, but equally plausible, hypotheses can be put forward. Each of them will have a rather dramatic effect on our belief in the \( N_o \) estimate.
Our confidence in the system will depend on which of these hypotheses that are correct.

- **Increasing familiarity:**
  We have probably found less than half the errors which we could have found

- **Exponentially increasing effort:**
  Most errors are probably found
6. Some Failure Time Distributions

This chapter focuses on how different usage intensities \( w(.) \) and usage profiles \( P_{\text{new}}(.) \) effect the distribution of the failures times.

Through reasonable approximations to real world observations of the use of a software system, it can be shown that the failure times tend to be distributed according to the generalized gamma distribution and more specifically to the three offsprings of this distribution, i.e. Weibull, Gamma and Log Normal.

Last, it is shown how the initial usage intensity can produce different failure time distributions, thus giving a physical interpretation for different distributions.

It follows from (5.29) that the failure intensity depends on \( P_{\text{new}}(.) \) and \( W(t) \). In order to keep things simple, we will only use deterministic \( W(t) \)-functions.

When we move from the very general conclusions in Chapter 5, we must make assumptions on

- how the system is tested or used \( (P_{\text{new}}) \)

- how the testing effort or use is distributed over time \( (W) \)

The following chapters will discuss some sets of natural assumptions for the following situations
- testing by the project team

- use at a new site

- testing by an independent test team
6.1 Testing by the project team

We will assume that the project team has complete knowledge of the system in question. As a result of this we assume

\[ P_{\text{new}}(k) = 1.0 \]  \hspace{1cm} (6.1)

This reduces (5.27) to (6.2)

\[ f(k) = Ae^{-Ak} \]  \hspace{1cm} (6.2)

It is instructive to study how \( f(t) \) changes when \( W(t) \) is changed. The following chapters do this for some reasonable versions of the testing effort.

6.1.1 Constant Testing Effort

The most simple form of \( W(t) \) is

\[ W(t) = \#t \text{ for } t \in [0, T] \]  \hspace{1cm} (6.3)

which implies that the testing effort is constant throughout a testing period of length \( T \). Such a development would by typical for a software project which is on time and within budget. Everything is going smoothly and no major design errors are found.

If we use (5.29) and (6.3), we get
This is the well-known Jelinski-Moranda model. The form of $f(t)$ is shown in fig. 6.1.

6.1.2 Increasing Testing Effort

Another simple model for $W(t)$ is the "panic" model, where the testing effort increases as the time of delivery approaches. When all project member work full time on testing, extra shifts are thrown in, all in an ever increasing effort to remove as many faults as possible before shipment.

To model this, we set
When this form of $W(t)$ is applied to (5.29) with $P_{new}(.) = 1$, we get the log Gamma distribution

$$f(t) = A \alpha e^{-\alpha t} + A(1-e^{-\alpha t})e^{-\alpha t}, A < 1$$

The form of $f(t)$ is shown in fig. 6.2.
6.1.3 Combination of Constant and Increasing Testing Effort

In a real project both of the two aforementioned pure types of testing effort may be present. A typical software project test phase may develop as follows

- start phase.
  Here we have a constant testing effort. Everything seems to proceed in a smooth and controlled manner.

- the "worry" phase.
  There were more errors than initially expected. This is indicated in fig. 6.3 where the dotted line indicates the expected development of $f(t)$, while the continous line shows the real development. The project leader and every one else start to get worried. The testing effort is increased.

- the panic phase.
  The delivery date approaches and things start to look real bad. New personnel is drawn into the project and the testing effort increases exponentially.

The development will look like the graph in fig. 6.3.
Fig. 6.3

--- Expected development
--- Real development

$f(t)$

Panic time

(6.4)  (6.6)
6.2 Use at New Site

6.2.1 General Considerations

We will assume a stable user community. This implies that the following factors are constant

. the persons employed
. the work performed at the site

The use at a new site will affect the number of errors reported per time unit in two ways

- through the number of executions per time unit, \( w(t) \)

- through new ways to use the system, \( P_{\text{new}}(k) \)

The form of \( P_{\text{new}}(k) \) will depend on the job type and user environment.

When a new system is introduced at a site, the following development is typical

- introduction phase
  The input data are mostly taken from the examples in the manual, just to get "the feel of the system"

- intermediate phase
  The users start to experiment just to see what happens. Most errors are reported in this phase.

- production phase
The users will now use the system to do real work. Little experimenting outside the work context is performed. Few, albeit serious, errors are reported in this phase.

On the basis of these consideration, I will suggest the following form for $P_{\text{new}}(k)$:

![Graph of $P_{\text{new}}(k)$ showing three phases: Introduction phase, Intermediate phase, Production phase.]

Fig. 6.4

The function $w(t)$ must, ideally, reflect three phases in the system usage:

1) increasing use during the start phase and the intermediate phase

2) stable use during the work phase

3) decreasing use as the system becomes outdated and/or replaced
by new and better systems

The accumulated number of executions, \( W(t) \), is shown in fig. 6.5a. The function \( w(t) \) is the usage intensity and is shown in fig. 6.5b.

\[
\begin{array}{c}
\text{\( W(t) \)} \\
\text{\( K \)} \\
\text{\( k_0 \)} \\
\end{array}
\]

\[ 
\text{Accumulated number of executions} \\
\text{\( t \)} \\
\]

\text{Fig. 6.5a}

\( K \) is the total number of executions performed during the lifetime of the system. \( k_0 \) is the number of executions performed in the first time period.
Fig. 6.5b

- start phase
- work phase
- outdated

w(t)

t
6.3 A Suggestion for \( W(t) \)

A convenient, general form for \( W(t) \) is

\[
W(t) = K \frac{e^{\frac{\lambda t}{K-k}}}{\frac{\lambda t}{K-k} + e^{\frac{\lambda t}{k_0}}}
\]

(6.11)

This gives \( w(t) \) as follows

\[
w(t) = K \lambda \frac{e^{\frac{\lambda t}{K-k}}}{\frac{\lambda t}{K-k} + e^{\frac{\lambda t}{k_0}}}
\]

(6.12)

If we introduce \( G(k) = W^{-1}(t) \), we get

\[
G(k) = \frac{1}{\lambda} \ln \left( \frac{k}{K-k} \right)
\]

(6.13)

It is straightforward to obtain the value \( t_{\text{max}} \) which satisfy

\[
\max \{w(t)\} = w(t_{\text{max}})
\]

(6.14)

We find that

\[
t_{\text{max}} = \frac{1}{\lambda} \ln \frac{K-k}{k_0}
\]

(6.15)

\[
w(t_{\text{max}}) = \frac{1}{4} K \lambda
\]
The most important features for $w(t)$ are summed up in fig. 6.6.

Fig. 6.6

$$t_2 - t_1 \approx \frac{1.3}{\lambda} \quad (6.16)$$

From (6.16) we see that it is reasonable to say that $\lambda$ is determined by the length of the work phase for the system (cfr fig. 6.5.b).
6.4 Two Suggestions for the New Path Probability

While it is relatively easy to suggest a $W(t)$, a general $P_{\text{new}}$ function poses some problems.

With $W(t)$ we can identify such things as useful lifetime (6.16) and total number of executions ($K$). The only thing we have for $P_{\text{new}}$ is that it will always be in the range $[0,1]$, and that

$$\lim_{k \to \infty} P_{\text{new}}(k) = 0$$

In order to do some experimenting, we used the following simple model.

![Diagram](image)

Fig. 6.7
The parameters C, D and E were varied in order to see how they influenced the model.

In order to get a quick overview of the results we

1) computed the function $f(t)$ from (5.29) with $w(t)$ from (6.12) and the $P_{\text{new}}(k)$ suggested in fig. 6.2

2) used this function to compute the skewness ($\beta_1$) and curtosis ($\beta_2$)

3) plotted $\beta_1$ and $\beta_2$ as functions of C and D (As long as $E >> D$, E had no influence on the model behaviour).

The results of this process are shown in fig. 6.8 a-c.

In all cases, K is chosen to be 10000 and C is varied in the area 100 to 4000.

In order to study the effect of $k_0$ on $\beta_1$ and $\beta_2$, the curve is plotted for three different $k_0$ values.

- curve 1 : $k_0 = 1$
- curve 2 : $k_0 = 10$
- curve 3 : $k_0 = 500$

The following trends can be observed.

- for small $k_0$, the difference between C and D has no significant impact

- for medium size $k_0$, the $\beta_1$-$\beta_2$ plot covers a large part of both the Log Normal, Weibull and Gamma lines

- for large $k_0$ values, the plot tends to deviate
substantially from the Log Normal/Weibull/Gamma lines, except for small C and D values.

It is reasonable to assume that distributions which falls in the vicinity of the Log Normal/Weibull/Gamma lines, can be approximated by Log Normal, Weibull and/or Gamma distributions.

The parameters C, D and E in fig. 6.7 can be given the following interpretation. (Cfr. fig. 6.4)

\[
\begin{align*}
[0,C] & : \text{introductory phase} \\
[C,D] & : \text{intermediate/experimentation phase} \\
[D,E] & : \text{production phase}
\end{align*}
\]

The only cases which do not seem to fit in well with the Gamma-Weibull-Log Normal assumption are data with large \( k_0 \), C and D.

For the Gamma distribution, \( \beta_1 \) and \( \beta_2 \) will depend on the scale parameter only.

\[
\begin{align*}
t \sim \text{Gamma} (\alpha, \lambda) \implies \\
\beta_1 &= \frac{\lambda}{\alpha} \\
\beta_2 &= \frac{3}{\alpha} (\alpha + 2)
\end{align*}
\]  

(6.17)

If we use curve 2 in fig. 6.8.a, we see that this can be approximated by a straight line from (0,3) to (3,7). This gives us

\[
\beta_2 \approx 3 + 1.33 \beta_1
\]

(6.18)

while (6.17) gives us that

\[
\beta_2 = 3 + 1.5 \beta_1
\]

(6.19)

Similar approximation can be given for the other cases.
$P_{\text{new}}$ as defined in fig. 6.7

$D = C$

$3a : C < 700$

$3b : C > 800$

Fig. 6.8a
\( P_{\text{new}} \) as defined in fig. 6.7

\[ D = C + 100 \]

Fig. 6.8.b
\( P_{\text{new}} \) as defined in fig. 6.7

\[ D = C + 300 \]

3a: \( C < 200 \)

3b: \( C \in [400, 1000] \)

3c: \( C > 1000 \)

Fig. 6.8.c
Another way to model $P_{\text{new}}(k)$ is to adapt the normal probability function.

$$P_{\text{new}}(k) = P_0 \exp[-\ln(\frac{O}{\epsilon})(\frac{k-C}{C})^2]$$

(6.20)

has the form shown in fig. 6.9 (cfr. fig. 6.4 and 6.7)

By varying $\epsilon$ and $C$ in (6.20) and $k_o$ in (6.11) we can obtain $\beta_1$-$\beta_2$ plots in the same way as in the previous case. As in fig. 6.8 a-c, we have used three different $k_o$ values:

- curve 1 : $k_o = 1$
- curve 2 : $k_o = 10$
- curve 3 : $k_o = 500$

The parameter $C \in [50, 1000]$ and $\epsilon = 0.01$. The results are shown in fig. 6.10 and 6.11.
$P_{new}$ as defined in (6.20)

Fig. 6.10
In addition, we computed the $\beta_1 - \beta_2$ plot for $k_0 = 10$, $\varepsilon = 0.1$ and $k_0 = 10$, $\varepsilon = 0.001$. The results are shown in fig. 6.11.
$P_{\text{new}}$ as defined in (6.20)

Curve A : $\varepsilon = 0.100$
Curve B : $\varepsilon = 0.001$

Fig. 6.11
Both $c$ values used in fig. 6.11 seem rather extreme but not unreasonable. What is interesting is that fig. 6.10 and 6.11 together indicate that the suggested forms of $P_{new}(.)$ and $W(t)$ in most cases will give $\beta_1 - \beta_2$ plots fairly close to the Log Normal/Weibull/ Gamma area. Whether $f(t)$ is best approximated by a Log Normal, Weibull or Gamma distribution will to a large degree depend upon $k_o$ and $c$ and to a lesser degree upon $K$ and $P_o$.

It seems reasonable to conclude that:

For estimation purposes, a wide range of $f(t)$ can be approximated quite well by Gamma, Weibull or Log Normal distributions.
6.5 Assumptions Leading to Known Failure Time Distributions

It is possible to find $P_{new}$ functions which will yield known distributions. Before going through such an exercise it is useful to consider what kind of distributions that make sense, given our real world model(s).

6.5.1 General considerations

From (5.29) we have

$$f(t) = A P_{new}[W(t)] w(t)e^{-\int_0^t P_{new}(u)du}$$

(6.20-1)

It is now straightforward to obtain

$$F(t) = 1 - e^{-\int_0^t P_{new}(u)du}$$

(6.21)

and from this, the hazard function

$$h(t) = A P_{new}[W(t)] w(t)$$

(6.22)

As should be expected, the hazard rate is multiplicatively
dependent on the new path probability and the usage rate.

It is interesting to check the conditions which are necessary for satisfying one of the two conditions

- \( h(t) \) has a maximum
- \( h(t) \) is a decreasing function.

If we use the \( P_{\text{new}}(k) \) in (6.20) we get

\[
\frac{\delta}{\delta t} h(t) = A P_{\text{new}}[W(t)] \left\{ \frac{\delta^2}{\delta t^2} W(t) - \left[ \frac{\delta}{\delta t} W(t) \right]^2 \right\}.
\]

\[
P = \frac{21 n}{e^{C^2}} [W(t) - C].
\]  

(6.24)

In the work phase we can set \( w(t) \approx \text{const} \) (cfr. fig. 6.5b). This gives

\[
\frac{\delta^2}{\delta t^2} W(t) = 0 \text{ and thus}
\]

\[
\frac{A}{\delta t} h(t) = 0 \Rightarrow W(t) = C
\]  

(6.25)

Since \( W(0) = k_0 \), we can always write

\[
W(t) = k_0 + Y(t)
\]  

(6.26)

for some \( Y(t) > 0 \) for all \( t > 0 \). (6.25) can then be written as
\[ k_0 + Y(t) = C \]  

(6.27)

Since \( Y(t) > 0 \) for \( t > 0 \), it follows that

1) \( k_0 = C \Rightarrow \max\{h(t)\} \text{ for } t = 0 \)

2) \( k_0 > C \Rightarrow h(t) \text{ has no maximum} \)

3) \( k_0 < C \Rightarrow h(t) \text{ has a maximum for some } t \)

We can now sketch \( h(t) \) under the three different conditions

We see that \( k_0 < C \) can be described by a Log Normal distribution, while the two other cases are readily described by Gamma or Weibull distributions.
6.5.2 Assumptions leading to a Gamma Distribution

It is straightforward to show that

$$
P_{\text{new}}(k) = \frac{\beta^\alpha}{\Gamma(\delta)} \frac{G(k)}{1!} \prod_{i=0}^{\alpha-1} G(\beta i) e^{-\beta G(k)} \frac{\delta}{\delta k} G(k) \tag{6.28}
$$

inserted into (5.27) always gives

$$
f(k) = \frac{\beta^\alpha}{\Gamma(\alpha)} G(k) G(k) e^{-\beta G(k)} \frac{\delta}{\delta k} G(k) \tag{6.29}
$$

and will lead to

$$
f(t) \sim \text{Gamma}(\alpha, \beta) \tag{6.30}
$$

for any $k = w(t)$ which is non-decreasing in $t$. 
Fig. 6.13.a
The figures 6.13 a and b give reasonable $P_{\text{new}}$ shapes for use at a new site. They both lead to the conclusion stated in (6.30).
6.5.3 Assumptions leading to a Weibull Distribution

The hazard function for the Weibull distribution is

\[ h(t) = \lambda \alpha t^{\alpha-1} \]  \hspace{1cm} (6.31)

A possible, and not unreasonable, approximation to \( W(t) \), would be to use

\[ W(t) = a \ln t + b \]  \hspace{1cm} (6.31-1)

This gives us

\[ w(t) = \frac{a}{t} \]  \hspace{1cm} (6.31-2)

Let us choose \( P_{\text{new}}(u) \) as

\[ P_{\text{new}}(u) = ce^{gu} \]  \hspace{1cm} (6.31-3)

If we insert (6.31-1), (6.31-2) and (6.31-3) into (6.23) and combine this with (6.31), we get

\[ \lambda \alpha t^{\alpha-1} = \frac{a}{t} g (a\ln t + b) \]  \hspace{1cm} \text{Ace} \hspace{1cm} (6.31-4)

A little manipulation now gives us
\[ \lambda \alpha t = A \alpha e t \quad (6.31-5) \]

This can be fulfilled by using

\[ ga = \alpha , \quad \frac{Ac}{e} = \lambda \quad (6.31-6) \]

Note that (6.31-3) can only hold for a finite period of time since it approach infinity for large \( t \).

It is, however, only necessary that our usage model holds for the first two years, say. Note that the first part of \( P_{\text{new}}(k) \) sketched in fig. 6.13a also can be approximated by a \( e^{ak} \) function for small to medium \( k \) values. It is interesting to observe that (6.31-6) implicates that \( \alpha \) and \( \lambda \) should be negatively correlated because of the factor \( g \). That this is indeed the case can be seen from analysis of real world data (cfr. (12.9)), \( \varphi(\alpha, \lambda) = -0.78 \).

6.5.4 Assumptions leading to a Log Normal Distribution

From (6.23) it follows directly that if \( \varphi(.) \) and \( \Phi(.) \) are the probability density and distribution respectively of the Log Normal distribution, then the following will lead to Log Normal failure times

\[ A . P_{\text{new}}[W(t)]w(t) = \frac{\varphi(t)}{1-\Phi(t)} \quad (6.32) \]
\[ P_{\text{new}}[W(t)] = \frac{1}{w(t)} \frac{\varphi(t)}{1-\phi(t)} A \] (6.33)

If we use the convention that \( G[W(t)] = t \) (\( G = W^{-1} \)), we get that

\[ \frac{\delta G}{\delta k} \cdot \frac{\delta W}{\delta t} = 1 \] (6.34)

and thus

\[ P_{\text{new}}(k) = \frac{\varphi[G(k)]}{1-\phi[G(k)]} \frac{\delta}{\delta k} G(k) A \] (6.35)

This leads to

\[ f(k) = \varphi[G(k)] \frac{\delta}{\delta k} G(k) \] (6.36)

and, in the time domain

\[ f(t) = \varphi(t) \] (6.36-1)

The \( P_{\text{new}}(k) \) in (6.35) can describe a large variety of shapes, depending on \( k_0 \) and the \( \phi \) parameters (\( \mu, \sigma \)). Typical examples are shown in fig. 6.16 a and b.
Fig. 6.16.a

\[ p_{\text{new}}(k) \]

\[ \sigma_2 > \sigma_1 \]

Fig. 6.16.b

\[ p_{\text{new}}(k) \]

\[ \mu_2 > \mu_1 \]
5.6 Testing by an Independent Test Team

In some ways an independent test team is in the same position as a new user group. Among other things, they do not know the internal structure of the system and the chosen solutions. They will, in some cases, have knowledge of the algorithms and the problem reports from the implementation team. In most cases they will have access to all review documents, if any are produced. In some cases, the test team has developed a test suite from the functional specification.

The main difference between a user group and a test team lies in their purpose. This difference will influence the way the two groups react to a correct result and a failure. The reactions can be summed up in the following way:

A correct result:

. user group:
  will use the same or an similar approach the next time the same or an analogous situation arises

. test team
  will register that the function is correctly implemented, run the other data which concern this function and then leave it alone

A failure:

. user group
  may or may not report the failure to the personnel responsible for maintenance. After this, they will tend to
avoid the mechanism which caused the failure or mechanisms akin to this, even after the fault has been corrected.

- test team
  will always (with malicious glee) report the failure to the implementation team. After this, they will search for other tests akin to the failing one, to see whether they found a "bad area", or just a "bad spot"

When they get an updated version of the system, they will repeat the whole process to check if

- the fault in fact has been corrected
- the correction has introduced new faults.

Since $f(t)$ will depend on $P_{new}(k)$ and $w(t)$, it is necessary to see how these functions will be in the independent test team case.

The $P_{new}(k)$ will vary through the three phases

- acquaintance phase
- the Test, Analyse And Fix phase (TAAF)
- final validation phase
The form is basically the same as the one shown for use in fig. 6.4. The only difference is that the validation phase in fig. 6.18 is shorter than the work phase in fig. 6.4 and the TAAF phase in fig. 6.18 is more flat than the intermediate phase in fig. 6.4. The latter is a result of the different reaction to failures in the two groups.

The form of w(t) will, however, look quite different from the form shown in fig. 6.5.b. A reasonable approximation is shown in fig. 6.19.
This is the function

\[
 w(t) = \begin{cases} 
 w_0 & \text{when } t < t_{\text{delivery}} \\
 0 & \text{otherwise} 
\end{cases} \tag{6.37}
\]

and thus

\[
 W(t) = \begin{cases} 
 w_0 t + D & \text{when } t < t_{\text{delivery}} \\
 D + w_0 t_{\text{delivery}} & \text{otherwise} 
\end{cases} \tag{6.38}
\]
7. Grouped Observations

We will now consider the problem of estimating our model parameters when we have only observed the number of failures in each period and not the individual failure times.

First we show that under the proposed model (chapter 5.1), the number of failures in disjoint time intervals are multinomially distributed.

Next, two estimation methods are considered, the maximum likelihood and linearization. These two methods are then used in the estimation of the parameters in the Gamma, Weibull, Log Normal and Log Gamma distributions.

At last, we discuss the problem of estimating the period when the expected number of errors falls below a certain limit, e.g. 0.5.

We have observed the number of failures \((n_{\frac{1}{i}})\) which occurred in intervals numbered from 1 to \(k-1\). In principle, the intervals can have any finite length. In practice, however, they will usually be equal. The first interval is \([0, \tau_1]\) and the last where we have observations is \([\tau_{k-2}, \tau_{k-1}]\). We have also a last interval where we have not yet made any observations \([\tau_{k-1}, \rightarrow]\). The following relation holds:

\[
N = \sum_{i=1}^{k-1} n_{\frac{1}{i}} + n_{\frac{k}{k}}
\]

Note that \(n_{\frac{k}{k}}\), and thus \(N_{\frac{1}{0}}\), are unknown.
7.1 The Multinomial Formulation

We will partition the time axis into a set of time intervals so that interval number \( i \) starts at \( \tau_{i-1} \) and ends at \( \tau_i \).

In order to include the whole positive time axis, we define \( \tau_0 = 0 \) and \( \tau_k = \infty \) when the axis is partitioned into \( k \) intervals. Then, for any failure time \( t \), we define the event \( A_i \) as

\[
A_i \equiv t \in (\tau_{i-1}, \tau_i]
\]  \hspace{1cm} (7.1)

We define the probability vector \( P \), so that

\[
P_i = P(A_i)
\]  \hspace{1cm} (7.2)

From chapter 5.1 it follows that the \( N_0 \) failure times are i.i.d. We can thus consider the \( N_0 \) failures as \( N_0 \) independent trials, categorized by the time slots in which they occur. The following conditions are thus met:

. \( N_0 \) independent trials

. each trial results in one of the events \( \{A_i\} \)

. \( U_{A_i} = S, A_i \cap A_j = A_i \delta_{ij} \)
Since \( P \) is the same for all \( N_0 \) trials (cfr. chapter 5.1), we can conclude that

\[
n \sim \text{Mult} (N_0, P) \tag{7.3}
\]

From (7.2) it then follows that

\[
P_i = F(\tau_i) - F(\tau_{i-1}) \tag{7.4}
\]

A consequence of the model developed in chapter 5.1 is thus that the number of failures reported in each interval is binomially distributed. We can now obtain estimates for the parameters in several ways. The two methods we will discuss here are

- maximum likelihood estimation (MLE)
- linearization.

When \( N_0 \) is known, it is straightforward to find the \( P_i \) estimates. In our case, where \( N_0 \) is unknown, the expressions for the estimates get quite unwieldy. Fortunately, the problem can be solved with a reasonable effort at least asymptotically, by introducing conditional likelihoods for \( N_0 \) and \( P \). Cfr. chapter 7.6.

We will, however first look at the method of linearization.
7.2 Linearization of the Multinomial Distribution

If $\Delta_i = \tau_i - \tau_{i-1}$ is small, it is reasonable to set

$$P_i \approx \Delta_i f(t_i) \quad (7.5)$$

where we have assigned the failure time $t_i$ to all failures in interval $i$. From (7.3) it follows that

$$E(n_i) = N \sum_i P_i \quad (7.6)$$

so that is reasonable to use

$$n_i = N \sum_i \Delta_i f(t_i) e^{\epsilon_i} \quad (7.7)$$

where $\epsilon_i$ is an error term. (7.7) can be rewritten as

$$\ln(n_i) = \ln N + \ln \Delta_i + \ln f(t_i) + \epsilon_i \quad (7.8)$$

If we assume that $f(.)$ is a member of the exponential family, we can write [23]:

$$f(t|\theta) = a(\theta) b(t) e^{\sum_i g_i(\theta) h_i(t)} \quad (7.9)$$

If we use $\Delta_i = 1$, we can write (7.8) as
\[
\ln(n_i) = \ln N_i + \ln a(\theta) + \ln b(t) + \sum_{j=1}^{r} g_j(\theta) h_j(t) \quad (7.10)
\]

Let us now introduce a matrix \( X \) and a vector \( A \). These are defined as follows.

\[
A = \begin{bmatrix}
\ln[N a(\theta)] \\
\vdots \\
1 \\
g_1(\theta) \\
\vdots \\
g_r(\theta)
\end{bmatrix} \quad (7.11)
\]

\[
X = \begin{bmatrix}
1 \ln b(t_1) h_1(t_1) \ldots h_r(t_1) \\
1 \ln b(t_2) h_1(t_2) \ldots h_r(t_2) \\
\vdots \\
1 \ln b(t_k) h_1(t_k) \ldots h_r(t_k)
\end{bmatrix} \quad (7.12)
\]

The problem of estimating \( N_0 a(\theta) \) and \( \{g_j(\theta)\} \) is thus reduced to the well known problem of estimating \( A \) in the equation
\[
\ln n = X A + \epsilon \tag{7.13}
\]

This problem has the general least squares solution (cfr. for instance [10])

\[
A = (X \Sigma^{-1} X)^{-1} T^{-1} \ln n \tag{7.14}
\]

where we have

\[
\Sigma = D(\epsilon) \frac{1}{\sigma^2} \tag{7.15}
\]

The dispersion matrix for \(A\) is

\[
D(A) = \sigma^2 (X \Sigma^{-1} X)^{-1} \tag{7.16}
\]

We thus have estimators for \(g_i(\theta)\) and \(\ln[N a(\theta)]\) and their variations. At least in principle, it is thus possible to find estimates for \(\theta\), \(N\), and \(V(\theta), V(N)\).

From (7.8) it follows that

\[
D(\epsilon) = D(\ln n) \tag{7.17}
\]

Our first problem is thus to find \(D(\ln n)\).
7.3 An Approximation to $D(ln \, n)$

\[
D(ln \, n) = \{\text{Cov}[ln(n_i), \, ln(n_j)]\}_{i \neq j} \quad (7.18)
\]

We will show that the covariances in (7.18) can be approximated by the following expression.

\[
\text{Cov}(ln \, n_i, \, ln \, n_j) \approx \frac{1}{N} + \frac{1}{2N} \left( \frac{1}{p_i} + \frac{1}{p_j} \right) - 3 - \frac{1}{4N} \frac{p_i p_j}{p_i + p_j} \quad (7.18-1)
\]

We will use Taylor expansions to find the covariates in (7.18). In order to get a compact notation, we will, for a random function $f$, use the following notation for the $i$th term:

\[
\frac{1}{i! \, \delta n_j} \frac{\partial}{\partial n_j} f(n_j) \bigg|_{E(n_j)} = a_i \quad (7.20)
\]

\[
n_j - E(n_j) = Y_j \quad (7.21)
\]

The Taylor expansion of $f(n_j)$ in an interval around $E(n_j)$ can then be written as
\[ f(n) = \sum_{i=0}^{\infty} a_i Y^i j \sum_{i=0}^{\infty} a_i Y^i j \]

(7.22)

\[ \text{Cov}[f(n), f(n)] = \sum_{j} a a \text{Cov}(Y^j p, Y^j q) \]

(7.23)

In order to obtain \( E(Y^i) \), \( V(Y^i) \) and \( \text{Cov}(Y^j p, Y^j q) \) in an orderly manner, we will use moment generating functions.

If we pick two of the variables in (7.3), \( n_1 \) and \( n_2 \) say, the moment generating function \( M_{n_1, n_2}(s_1, s_2) \) is defined as

\[ M_{n_1, n_2}(s_1, s_2) = E(e^{s_1 n_1 + s_2 n_2}) \]  

(7.23-1)

Thus

\[ M_{n_1, n_2}(s_1, s_2) = \sum_{n_1=0}^{n} \sum_{n_2=0}^{n} \left\{ \binom{n}{n_1} \binom{n-n_1}{n_2} p_1^{n_1} p_2^{n_2} \right\} (1-p_1-p_2)^{n-n_1-n_2} e^{s_1 n_1 + s_2 n_2} \]

(7.24)

Some straightforward manipulations give us

\[ M_{n_1, n_2}(s_1, s_2) = (1-p_1-p_2+p_2 e^{s_2}+p_1 e^{s_1})^n \]  

(7.25)
From (7.21) we get

\[ Y = n - np \quad \text{for} \quad j \quad \text{j} \quad \text{j} \]  

(7.26)

\[ M_{Y_1, Y_2}(s_1, s_2) = e^{-n(s_1 p_1 + s_2 p_2)} M_{n_1, n_2}(s_1, s_2) \]  

(7.27)

For convenience, we shall drop the subscripts \( Y_1, Y_2 \) from now on. Thus, we have that

\[ M(s_1, s_2) = [e^{-p_1 s_1 - p_2 s_2 (1-p_1 - p_2 + p_1 e^{s_1} + p_2 e^{s_2})}]^n \]  

(7.28)

It is, however, more convenient to operate on the logarithm of the moment generating functions.

\[ \ln M(s_1, s_2) = n[-p_1 s_1 - p_2 s_2 + \ln(1-p_1 - p_2 + p_1 e^{s_1} + p_2 e^{s_2})] \]  

(7.31)

Note first that

\[ M(0), M(0,0) = 1 \]

\[ \frac{\delta}{\delta s} M(0), \frac{\delta}{\delta s} M(0,0) = 0 \]

(7.32)
It is now straightforward to obtain the following results:

\[
\frac{\delta}{\delta s} \ln M(0) = E(Y)
\]

\[
\frac{\delta}{\delta s}^2 \ln M(0) = E(Y^2)
\]

\[
\frac{\delta}{\delta s}^3 \ln M(0) = E(Y^3)
\]

\[
\frac{\delta}{\delta s}^4 \ln M(0) = E(Y^4) - 3E^2(Y^2)
\]

(7.33)

The moment generating function of a binomially distributed variable is well known, cfr. for instance [32]. If we differentiate \( \ln M(s) \) and insert into (7.33), we get

\[
E(Y) = 0
\]

\[
E(Y^2) = N_0 p(1-p)
\]

\[
E(Y^3) = N_0 p(1-p)(1-2p)
\]

\[
E(Y^4) = N_0 p(1-p)[1-6p(1-p)+3N_0 p(1-p)]
\]

(7.34)

From (7.31) we get
\[
\frac{\delta^2}{\delta s_1 \delta s_2} \ln M(0,0) = \text{Cov}(Y_1, Y_2)
\]

\[
\frac{\delta^3}{\delta s_1 ^2 \delta s_2} \ln M(0,0) = \text{Cov}(Y_1^2, Y_2)
\]

(7.35)

\[
\frac{\delta^3}{\delta s_1 ^2 \delta s_2} \ln M(0,0) = \text{Cov}(Y_1, Y_2^2)
\]

\[
\frac{\delta^4}{\delta s_1 ^2 \delta s_2 ^2} \ln M(0,0) = \text{Cov}(Y_1^2, Y_2^2) - 2 \text{Cov}^2(Y_1, Y_2)
\]

By differentiating the right hand side of (7.31) we now get

\[
\text{Cov}(Y_1, Y_2) = -N_0 p_1 p_2
\]

\[
\text{Cov}(Y_1^2, Y_2) = -N_0 p_1 p_2 (1 - 2p_1)
\]

(7.36)

\[
\text{Cov}(Y_1, Y_2^2) = -N_0 p_1 p_2 (1 - 2p_2)
\]

\[
\text{Cov}(Y_1^2, Y_2^2) = N_0 p_1 p_2 (2N_0 p_1 p_2 - 1 + 2p_1 + 2p_2 - 6p_1 p_2)
\]
We will, in the following computations, use a Taylor expansion with three terms:

\[
\ln n \approx \ln(Np) + \frac{n-Np}{Np} - \frac{(n-Np)^2}{2Np^2} \tag{7.37}
\]

or, to use (7.21)

\[
\ln n \approx \ln(Np) + \frac{Y}{Np} - \frac{Y^2}{2Np^2} \tag{7.38}
\]

By using the results in (7.34) we get

\[
E(\ln n) \approx \ln(Np) - \frac{1-p}{2Np} \tag{7.39}
\]

By using the results in (7.36) with \(p_1 = p\) and \(p_2 = 1-p\), we get

\[
V(\ln n) \approx \frac{1-p}{Np} + \frac{(1-p)[1+2p(1-p)(Np-3)]}{4Np^3} \tag{7.40}
\]

From (7.23) we get

\[
\text{Cov}(\ln n, \ln n) \approx \text{Cov}(\frac{1}{Np_1} - \frac{Y^2}{2Np_1^2}, \frac{1}{Np_2} - \frac{Y^2}{2Np_2^2}) \tag{7.41}
\]

By using the results in (7.36) we get the result in (7.18-1) We
have now enough information to compute the covariance matrix in (7.17).
7.4 Estimation for the Linearized Model

Since we have the covariance matrix $\Sigma$, it should now be possible to find $A$ in equation (7.14) for any $f(.)$ in the exponential family.

Only one problem remains. Since $N_o$ and $\theta$ must be known in order to compute $\Sigma$, we are, so to speak, missing the key for setting our mechanism in motion. We will attempt to solve this problem in the following manner:

1) By using only the first term in the approximations for $V(\ln n_i)$ and $\text{Cov}(\ln n_i, \ln n_j)$, we get

$$
\Sigma \approx \frac{1}{N_o} \begin{bmatrix}
\frac{1-p_1}{p_1} & -1 & \ldots & -1 \\
-1 & \frac{1-p_2}{p_2} & \ldots & -1 \\
\vdots & \vdots & \ddots & \vdots \\
-1 & -1 & \ldots & \frac{1-p_n}{p_n}
\end{bmatrix}
$$

(7.43)

2) If we assume that $p_i \ll 1$, we can simplify (7.43) to
\[ \Sigma = \frac{1}{N} \text{diag} \left( \frac{1}{p_i} \right) \]  \hspace{2cm} (7.44)

The approximation \( p_i \approx n_i / N \) will now enable us to write

\[ \Sigma^{-1} \approx \text{diag} \left( n_i \right) \]  \hspace{2cm} (7.45)

3) This can now be used to estimate starting values for \( \theta \) from \( A \). In the second iteration we will use the full expressions in (7.40) and (7.41) to compute the elements in \( \Sigma \). This covariance matrix can now be inserted into (7.15) in order to obtain the final estimates.
7.5 Linearization of some Realistic Distributions for \( T \)

In chapter 6 we found that it would be reasonable to assume

\[
t_i \sim \text{Gamma}(\alpha, \lambda)
\]

\[
t_i \sim \text{Log Normal}(\mu, \sigma^2)
\]

(7.46)

during stable use at a new site.

In addition, experience indicates that the failure time distributions tend towards a log gamma distribution in certain debugging situations (cfr. chapter 6.1.2). For this reason, we will also consider the Log Gamma distribution, which has the standard extreme value distribution as a special case.

We will apply the results in chapter 7.3 with these three distributions. They all belong to the exponential family and the functions \( a(.) \), \( b(t) \) \{\( g_i(.) \)\} and \{\( h_i(t) \)\} are easily identifiable.

7.5.1 Gamma Distributions

We have

\[
f(t) = \frac{\lambda^\alpha}{\Gamma(\alpha)} t^{\alpha-1} e^{-\lambda t}
\]

(7.47)
Referring back to (7.9) we have

\[ a(\lambda, \alpha) = \frac{\lambda^\alpha}{\Gamma(\alpha)} \]

\[ b(t) = 1 \]

\[ g_1(\alpha, \lambda) = \alpha - 1 \] \hspace{1cm} (7.48)

\[ g_2(\alpha, \lambda) = -\lambda \]

\[ h_1(t) = \ln t \]

\[ h_2(t) = t \]

This gives us the following \( A \) and \( X \)

\[
A = \begin{bmatrix}
\alpha \\
\ln[\text{No} \ \frac{\lambda}{\Gamma(\alpha)}] \\
1 \\
\alpha - 1 \\
-\lambda
\end{bmatrix}
\] \hspace{1cm} (7.49)
\[
X = \begin{bmatrix}
1 & 0 & \text{ln} t_1 & t_1 \\
\vdots & \vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots & \vdots \\
1 & 0 & \text{ln} t_k & t_k \\
\end{bmatrix}
\]

(7.50)

If we eliminate the 0 from \(X\), we get

\[
A = \begin{bmatrix}
\alpha & \lambda \\
\ln [\text{No} \frac{\lambda}{\Gamma(\alpha)}] & -1 \\
-\lambda & \alpha - 1 \\
\end{bmatrix}
\]

(7.51)

\[
\begin{bmatrix}
1 & \text{ln} t_1 & t_1 \\
\vdots & \vdots & \vdots \\
\vdots & \vdots & \vdots \\
1 & \text{ln} t_k & t_k \\
\end{bmatrix}
\]

(7.52)

By using (7.14) and (7.16) it is now straightforward to find the estimates \(\hat{a}_1, \hat{a}_2, \hat{a}_3\) and their variances and covariances. From (7.51) we have that
\[
\begin{align*}
\hat{\lambda} &= -\hat{a}_3 \\
\hat{\alpha} &= \hat{a}_2 + 1 \\
\frac{\Gamma(\hat{\alpha})}{\hat{\alpha}} &= e^{1 - \frac{1}{\hat{\lambda}}} \\
N_o &= \frac{\hat{\alpha}}{\hat{\lambda}}
\end{align*}
\] (7.53)

7.5.2 Log Normal Distribution

We have that

\[
f(t) = \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{(\ln t - \mu)^2}{2\sigma^2}}
\] (7.54)

From (7.9) we now have

\[
a(\mu, \sigma) = \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{\mu^2}{2\sigma^2}}
\]

\[
b(t) = \frac{1}{t}
\]

\[
g_1(\mu, \sigma) = -\frac{1}{2\sigma^2}
\] (7.55)
\[ g_2(\mu, \sigma) = \frac{\mu}{\sigma^2} \]

\[ h_1(t) = (\ln t)^2 \]

\[ h_2(t) = \ln t \]

This gives us the following \( A \) and \( X \):

\[
A = \begin{bmatrix}
-\frac{\mu^2}{2\sigma^2} \\
\ln[N_0 \frac{e^{-\mu^2/2\sigma^2}}{\sigma^{\sqrt{2\pi}}}] \\
1 \\
-\frac{1}{2\sigma^2} \\
\frac{\mu}{2\sigma^2}
\end{bmatrix}
\]

(7.56)

\[
X = \begin{bmatrix}
1 & -\ln t_1 & (\ln t_1)^2 & \ln t_1 \\
\vdots & \vdots & \vdots & \vdots \\
1 & -\ln t_k & (\ln t_k)^2 & \ln t_k
\end{bmatrix}
\]

(7.57)

Since the second and fourth column are not independent, this must be changed into
\[
A = \begin{bmatrix}
\ln[N_0 e^{-\mu^2/2\sigma^2} / \sigma \sqrt{2\pi}]
& -\frac{1}{2\sigma^2} \\
\frac{\mu}{2\sigma^2} - 1
\end{bmatrix}
\] 

(7.58)

\[
X = \begin{bmatrix}
1 & (\ln t_1)^2 & \ln t_1 \\
\vdots & \vdots & \vdots \\
1 & (\ln t_k)^2 & \ln t_k
\end{bmatrix}
\] 

(7.59)

By using (7.14) and 7.16 we can now find estimates for \(\{a_1, a_2, a_3\}\) and their variances and covariances. Thus:

\[
\hat{\sigma} = (\frac{1}{\hat{a}_2^{1/2}})
\]

\[
\hat{\mu} = -\frac{1 + \hat{a}}{\hat{a}_2^{3}}
\]

(7.60)

\[
\hat{N}_0 = \hat{\sigma} \sqrt{2\pi} \exp(\hat{a}_3 + \hat{\mu}^2 / 2\hat{\sigma}^2)
\]
7.5.3 Log Gamma Distribution

We have that

\[ f(t) = \frac{\lambda^\alpha}{\Gamma(\alpha)} e^{-\lambda t} t^{\alpha-1} \]  
\[ (7.60-1) \]

From (7.9) we thus have that

\[ a(\lambda, \alpha) = \frac{\lambda^\alpha}{\Gamma(\alpha)} \]

\[ b(t) = 1 \]

\[ g_1(\lambda, \alpha) = \alpha \]
\[ (7.60-2) \]

\[ g_2(\lambda, \alpha) = \lambda \]

\[ h_1(t) = t \]

\[ h_2(t) = -e^t \]
This gives us the following $A$ and $X$:

$$A = \begin{bmatrix} \alpha & \lambda \\ \ln[N \frac{\lambda}{\alpha \Gamma(\alpha)}] & 1 \\ 1 & \alpha \\ \lambda & \end{bmatrix}$$  \hspace{1cm} (7.60-3)$$

$$X = \begin{bmatrix} 1 & 0 & t & -e_1 \\ 1 & 0 & t_2 & -e_2 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & 0 & t_k & -e_k \end{bmatrix}$$  \hspace{1cm} (7.60-4)$$

The $A$ and $X$ can be simplified to

$$A = \begin{bmatrix} \alpha & \lambda \\ \ln[N \frac{\lambda}{\alpha \Gamma(\alpha)}] & 1 \\ \alpha & \lambda \\ \end{bmatrix}$$  \hspace{1cm} (7.60-5)$$
\[
X = \begin{bmatrix}
1 & t_1 & -e_1 \\
1 & t_2 & -e_2 \\
\vdots & \vdots & \vdots \\
1 & t_k & -e_k
\end{bmatrix}
\]

(7.60-6)

By using (7.14) and (7.16) we can now find estimates for \(a_1\), \(a_2\) and \(a_3\), together with their variances and covariances.

\[
\hat{\lambda} = \hat{a}_3
\]

\[
\hat{\alpha} = \hat{a}_2
\]

\[
N = e^{1 - \frac{\Gamma(\hat{a}_2)}{\hat{a}_3}}
\]

(7.60-7)

7.5.4 Other Failure Time Distributions

The results developed in chapters 7.1 to 7.4 are quite general since the exponential family is large. In particular, the exponential distribution, yields the Jelinski-Moranda model and all other models which are defined by

- constant hazard function
- all remaining errors are at risk

These models include, among others, the models of Musa, Shooman and Goel-Okumoto (cfr. table in [28], chapter 7). Here we have

\[ f(t) = \lambda e^{-\lambda t} \quad (7.61) \]

and thus

\[ a(\lambda) = \lambda \]
\[ b(t) = 1 \]
\[ g_1(\lambda) = -\lambda \quad (7.62) \]
\[ h_1(t) = t \]

\[
A = \begin{bmatrix}
\ln(N_0 \lambda) \\
1 \\
-\lambda
\end{bmatrix} \quad (7.63)
\]

\[
X = \begin{bmatrix}
1 & 0 & t_1 \\
\vdots & \vdots & \vdots \\
1 & 0 & t_k
\end{bmatrix} \quad (7.64)
\]

By eliminating the 0-vector, this can be changed into
\[ A = \begin{bmatrix} \ln(N \lambda) \\ 0 \\ -\lambda \end{bmatrix} \]  
(7.65)

\[ X = \begin{bmatrix} 1 & t_1 \\ \vdots & \vdots \\ i & t_k \end{bmatrix} \]  
(7.66)

This approach will give the following estimate as a first approximation

\[ \hat{\lambda} = \sum_{j} \ln n_j \ln t_j - N \sum_{j} \ln n_j \]  
(7.67)

\[ \ln N = \sum_{j} \ln n_j \ln t_j^2 - N \sum_{j} \ln n_j - \ln \hat{\lambda} \]  
(7.68)

The results, corrected for a matrix which is not diagonal are rather unwieldy, but it is straightforward to obtain the numerical results.
7.6 Maximum Likelihood Estimation for the Multinomial Model

The formulas used here are to a large extent adapted from L. Sanathanan's work on truncated samples. For a quick overview, see [41]. For a more complete treatment, including proofs, see [42].

We will first obtain the maximum likelihood estimates for \( \{\theta_i\} \) and \( N_0 \). It is difficult to obtain the distributions of the estimates. Fortunately, L. Sanathanan has proved some asymptotic results and these are referred in the next subchapter.

7.6.1 Estimation of the Parameters

For the multinomial distribution defined in chapter 7.1, we can form the following likelihood:

\[
L(P_{1:k}, N | n) = \frac{N!}{k+1 \prod_{i=1}^{k} p_i^{n_i}} \quad \text{(7.69)}
\]

Since \( P = F(\tau_i) - F(\tau_{i-1}) \), (cfr. (7.4)) \( P \) is a function of the parameters of \( F \). We can thus also use the notation \( P(\theta) \). Using the fact that \( \sum_{i=1}^{k+1} n_i = N_0 \) and introducing the notation...
we can write the likelihood in (7.69) as

\[
L(P_{k+1}, N_0 | n) = \frac{N!}{\prod_{i=1}^{k+1} n_i !} \left( \frac{N}{\prod_{i=1}^{k+1} n_i !} \right)^{N-N_k} P_{0}^N \prod_{i=1}^{k+1} P_{i}^{n_i} \quad (7.71)
\]

or

\[
L(P_{n+1}, N_0 | n) = \binom{N}{n} P_{0}^{N-k-1} (1-P_{k+1})^n \cdot \frac{N!}{\prod_{i=1}^{k+1} n_i !} \left( \frac{P}{\prod_{i=1}^{k+1} (1-P_{k+1})} \right)^{n_i} \quad (7.72)
\]

It is possible to find the ML estimators directly from (7.71) but the computations tend to be complicated and cumbersome. The only case that is reasonably well behaved, is the exponentially distributed failure times. The general likelihood in (7.72) can, however, be split into two likelihoods, denoted \( L_1 \) and \( L_2 \) respectively, where \( L_1 \) is a binomial likelihood, and \( L_2 \) is a multinomial likelihood. We can then first find the failure time distribution parameters from \( L_2 \) and then \( N_0 \) from \( L_1 \). Santhanam has proved that this two-stage approach is asymptotically equivalent to the MLE's from (7.71).
\[ L(P, N | n) = L_1(N_0 | N, P_{k+1}) L_2(P | N, P_{k+1}, n) \]  \hspace{1cm} (7.72) \\

\[ L_1(N_0 | N, P_{k+1}) = \binom{N}{N_0} P^{N_0} (1-P)^{N-N_0} \]  \hspace{1cm} (7.73) \\

\[ L_2(P | N, P_{k+1}, n) = C \prod_{i=1}^{k} \left( \frac{P}{1-P} \right)^i \]  \hspace{1cm} (7.74) \\

Refering back to the notation introduced in chapter 7.1, the quotient

\[ q_i = \frac{P_i}{1-P_{k+1}} \]  \hspace{1cm} (7.75) \\

is the probability of the event

\[ \{ A | t < \tau_i \} \]  \hspace{1cm} (7.76) \\

From what is said in (7.4) and (7.5), we have

\[ q_i(\theta) = \frac{F(\tau_i, \theta) - F(\tau_{i-1}, \theta)}{F(\tau_k, \theta)} \]  \hspace{1cm} (7.77)
In order to simplify the computations we will now replace all failure times in the interval \([\tau_{i-1}, \tau_i]\) with the interval midpoint. This point in time we will denote \(t_i\). We will use this time as the common occurrence time for all failures in the interval. In addition we will replace \(\tau_k\) by \(t_k\). Thus

\[
F(t_i, \theta) - F(t_{i-1}, \theta) \approx \Delta f(t_i)
\]

We can now use:

\[
f(t_i, \theta) = \frac{\Delta}{\Delta F(t_i, \theta)}
\]

(7.78)

It is now possible to find estimates for \(\theta\) and \(N_0\) by

1. finding \(\hat{\theta}\) \(\Rightarrow L_2(\hat{\theta}) = \max_{\theta} \{L_2(\theta)\}\)

2. finding \(\hat{N}_0\) \(\Rightarrow L_1[N_0 | \hat{\theta}] = \max_{N_0} \{L_1(N_0 | \hat{\theta})\}\)

This gives us that

\[
\hat{N}_0 = \frac{N}{F(t_i, \hat{\theta})}
\]

(7.79)

The likelihood in (7.74) can be written as
\[ L_2 = \frac{\prod_{i=1}^{N} f(t_i, \theta)^{n_i}}{C_1 \prod_{i=1}^{k} F(t_i, \theta)^{n_i}} \]  

(7.79-1)

If we use constant time steps \( \Delta t \) we have that

\[ L_2 \approx C_2 \prod_{i=1}^{k} n_i \ln f(t_i, \theta) - N \ln F(t, \theta) \]  

(7.79-2)

and \( \theta \) can be computed from

\[ \frac{\delta L_2}{\delta \theta} = 0, \quad \theta^\cdot = \frac{N}{F(t, \theta)} \]  

(7.79-3)

7.6.2 Assymptotic Distribution of the Parameters

Let us introduce the covariance matrix \( \Sigma \), defined by

\[
\Sigma^{-1} = \begin{bmatrix}
\{a\} & \{a\}^T \\
\{a\} & a_{mm} \\
\{a\} & a_{oo}
\end{bmatrix}
\]

(7.80)
\[ a_{jm} = \sum_{i=1}^{k+1} \frac{\frac{\delta}{\delta \theta} P(\theta) \frac{\delta}{\delta \theta} q(\theta)}{q(\theta)} \]  

(7.81)

\[ a_{om} = \frac{\frac{\delta}{\delta \theta} F(t, \theta)}{F(t, \theta)} \]  

(7.82)

\[ a_{oo} = \frac{F(t, \theta)}{F(t, \theta)} \]  

(7.83)

Sanathanan [42] has then proved that

\[
\begin{bmatrix}
\sqrt{N_o} \\
\frac{\hat{N}_o}{N_o} - N_o \\
\frac{1}{\sqrt{N_o}}
\end{bmatrix}
\begin{bmatrix}
(\theta - \theta) \\
D \\
N(0, \xi)
\end{bmatrix}
\sim N(0, \xi)
\]  

(7.84)

when \( N_o \to \infty \)

By introducing the matrix \( B \), defined as

\[
B = \begin{bmatrix}
1 \\
\frac{1}{\sqrt{N_o}} \\
0 \\
0 \\
\sqrt{N_o}
\end{bmatrix}
\]  

(7.85)
we can write Sanathanan's result in (7.84) as

\[
\begin{bmatrix}
\Theta \\
\Lambda \\
N_0
\end{bmatrix} = N\left(\begin{bmatrix}
\Theta \\
\Lambda \\
N_0
\end{bmatrix}, \Sigma B B\right) \tag{7.86}
\]

since \( B = B \).
7.7 Conditional MLE Applied to some Realistic Distributions

We will now use the gamma, log gamma and log normal distributions as \( F(\cdot) \) in the results found in the previous chapter.

7.7.1 Gamma Distributions

Let us use the notation

\[
\int_0^\infty \frac{\lambda^\alpha}{\Gamma(\alpha)} u^{\alpha-1} e^{-u\lambda} \, du = \frac{\gamma(\alpha, \lambda t)}{\Gamma(\alpha)}
\]  

(7.90)

(7.79-2) can now be written

\[
1_2 = \sum_{j=1}^k \left\{ n_1 \ln \lambda - \ln \Gamma(\alpha) + (\alpha-1) \ln t - \ln \lambda \right\} + \sum_{j=1}^k \left\{ \ln \gamma(\alpha, \lambda t) \right\} - \ln \Gamma(\alpha)
\]

(7.91)

(7.91) now enables us to get

\[
\sum_{j=1}^k \left\{ n_1 \ln \lambda - \psi(\lambda) + \ln t \right\} = N \left[ \frac{\delta \gamma(\lambda, \lambda t)}{\delta \alpha} - \psi(\lambda) \right]
\]

(7.92)
\[ \sum_{j=1}^{k} n \frac{\delta}{\gamma(\hat{\alpha}, \hat{\lambda}t)} = N \frac{\delta \lambda}{\gamma(\hat{\alpha}, \hat{\lambda}t)} \]  

These two equations can be rewritten as

\[ N \ln \lambda + \sum_{j=1}^{k} \ln(t_j) = N \frac{\delta \alpha}{\gamma(\hat{\alpha}, \hat{\lambda}t)} \]  

\[ N \frac{\delta}{\gamma(\hat{\alpha}, \hat{\lambda}t)} - \sum_{j=1}^{k} \ln(t_j) = N \frac{\delta \lambda}{\gamma(\hat{\alpha}, \hat{\lambda}t)} \]  

Let us use the following notation

\[ \bar{T} = \left[ \prod_{j} t_j \right]^{1/N} \]  

\[ \tilde{T} = \frac{1}{N} \sum_{j} t_j \]  

(geometric mean)  

(7.96)

The two equations (7.94) and (7.95) can then be simplified to
\[
\ln \hat{\lambda} + \ln \hat{\tau} = \frac{\delta}{\delta \alpha} \frac{\gamma(\alpha, \lambda t)}{\gamma(\alpha, \lambda t)}
\]

(7.97)

\[
\hat{\alpha} - \frac{\lambda}{\hat{\lambda}} = \frac{\delta}{\delta \lambda} \frac{\gamma(\alpha, \lambda t)}{\gamma(\alpha, \lambda t)}
\]

(7.98)

It is easy to see that [12]

\[
\frac{\delta}{\delta \lambda} \gamma(\alpha, \lambda t) = (\lambda t)^{\alpha-1} e^{-\lambda t}
\]

(7.99)

but \(\frac{\delta}{\delta \alpha} \gamma(\alpha, \lambda t)\) does not exist in closed form other than the obvious

\[
\frac{\delta}{\delta \alpha} \gamma(\alpha, \lambda t) = \int_0^\infty u^{\alpha-1} e^{-u} \ln u \, du
\]

(7.100)

Thus, it is difficult to obtain MLE's from (7.93) and (7.94). A more straightforward and simpler method would be to use a computer program to maximize (7.91) directly.

Note that approximation like [12]:

\[
\gamma(\alpha, \lambda t) \approx \Gamma(\alpha) - (\lambda t)^{\alpha-1} e^{-\lambda t} \left(1 + \frac{\alpha-1}{\lambda t}\right)
\]

(7.101)

will give (7.97) and (7.98) rather unwieldy solutions, which will
need a computer program to be solved anyway.

7.7.2 Log Normal Distributions

We will use the notation

\[ \phi(t) = \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi}} e^{-\frac{u^2}{2}} \, du \]  

(7.102)

For the log normal distribution, we then get

\[ F(t) = \phi \left( \frac{\ln t - \mu}{\sigma} \right) \]  

(7.103)

The log likelihood \( l_2 \), is thus

\[ l_2 \approx \sum_{j=1}^{k} n \left\{ \ln(\sigma \sqrt{2\pi}) - \ln t_j - \frac{1}{2\sigma^2} (\ln t_j - \mu)^2 \right\} \]

\[ - N \ln \phi \left( \frac{\ln k - \mu}{\sigma} \right) \]  

(7.104)
Differentiation gives us that

\[ -\frac{N}{\sigma} + \frac{1}{\sigma} \sum_{j=1}^{k} n \left( \text{Int} - \hat{\mu} \right)^2 - \frac{N}{\phi(\hat{\mu})} \frac{\delta \phi}{\delta \hat{\mu}} = 0 \quad (7.104-1) \]

\[ \frac{1}{\sigma} \sum_{j=1}^{k} n \left( \text{Int} - \hat{\mu} \right) - \frac{N}{\phi(\hat{\mu})} \frac{\delta \phi}{\delta \hat{\mu}} = 0 \quad (7.104-2) \]

where we have used the notation

\[ u = \frac{k}{\sigma} \quad (7.104-3) \]

in order to simplify (7.104-1) and (7.104-2).

\[ \frac{\delta \phi}{\delta u} = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2} u^2} \quad (7.104-4) \]

\[ 0 = -\frac{N}{\sigma} + \frac{1}{\sigma} \sum_{j=1}^{k} n \left( \text{Int} - \hat{\mu} \right)^2 - \frac{N}{\phi(\hat{\mu})} \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2} \left( -\frac{\hat{\mu}}{\sigma} \right)^2} \quad (7.105) \]
These two equations can be rewritten as

\[
0 = \frac{1}{\sigma^2} \sum_{j=1}^{k} n_j (\ln t - \mu_j) - \frac{N}{\phi(\mu)} \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2} \left(-\frac{\mu_j^2}{\sigma^2}\right)} \tag{7.106}
\]

\[
N \sigma^2 = \sum_{j=1}^{k} n_j (\ln t - \mu_j)^2 + \frac{N\sigma^2}{\phi(\mu)} (\ln t - \mu_j) \tag{7.107}
\]

\[
\mu = \ln T + \frac{\sigma^2}{\phi(\mu)} \tag{7.108}
\]

The same conclusions as we reached for the Gamma case, hold here. That is

- no closed form solution exists
- standard approximations for \( \phi \) will still need a computer program to get numerical results.

It is, however, possible to arrive at a more easily solvable set of equations in the following way:

From (7.108), we get that
\[ \frac{\Delta \psi(\hat{\mu})}{\hat{\phi}^2(\hat{u})} = V - \ln T \]  

(7.109)

We can then rewrite (7.107) as

\[ N \hat{\sigma}^2 = \sum_{j=1}^{k} n_{j} (\ln t_{j})^2 - 2\hat{\mu} N \ln T + N \hat{\sigma}^2 + N(\ln t_{k} - \hat{\mu})(\hat{\mu} - \ln T) \]  

(7.110)

\[ \hat{\sigma}^2 = \left[ \frac{1}{N} \sum_{j=1}^{k} n_{j} (\ln t_{j})^2 - \ln T \ln T \right] + \hat{\mu} \ln T \]  

(7.111)

\[ \hat{\sigma}^2 = a + \hat{\mu}b \]  

(7.112)

where \( a \) and \( b \) depend on the observed data only.

We can now rewrite (7.109) as

\[ \Delta \frac{\psi(\hat{\mu})}{\hat{\phi}(\hat{u})} - \frac{\hat{\sigma}^2 - a}{b} + \ln T = 0 \]  

(7.113)

\[ \Delta u = \frac{\ln t - \frac{\hat{\sigma}^2 - a}{b}}{\hat{\sigma}} \]  

(7.114)
The problem is now reduced to the problem of finding the zeros in one non-linear equation, namely (7.113). This gives us a \( \sigma \)-estimate. A \( \mu \)-estimate can then be obtained from (7.112).

### 7.7.3 Log Gamma Distributions

If we use the notation introduced in (7.90), it is straightforward to show that

\[
\int_{-\infty}^{t} \lambda^\alpha e^{\alpha u - \lambda e^u} \frac{u^t}{\Gamma(\alpha)} du = \frac{\gamma(\alpha, \lambda e^t)}{\Gamma(\alpha)} \tag{7.115}
\]

(7.79-2) can then be written

\[
\ell_2 = \sum_{j=1}^{k} \ln \left\{ \alpha \ln \lambda - \ln \Gamma(\alpha) + \alpha t^j - \lambda e^j \right\} 
\]

\[
- N \left[ \ln \gamma(\alpha, \lambda e^t) - \ln \Gamma(\alpha) \right] \tag{7.116}
\]

The two maximum likelihood equations are thus
\[ \ln \lambda + T = \frac{\delta}{\delta \alpha} \frac{\alpha^\wedge \times t}{\gamma(\alpha, \lambda e^k)} \]  

(7.117)

\[ \frac{\lambda^\wedge}{\lambda} - \frac{1}{N} \sum_{n=1}^{N} \sum_{j=1}^{t} \frac{\delta}{\delta \lambda} \frac{\alpha^\wedge \times t}{\gamma(\alpha, \lambda e^k)} \]  

(7.118)

where \( T \) is defined in (7.96).

As for the gamma distribution, we get into trouble due to the right hand side of (7.117). Cfr. chapter 7.6.1 for a discussion.
7.8 Estimation of Time to Deliver

As long as the failure time distribution does not change (stable environment), it is possible to estimate the period number where expected number of failures fall below a certain level \( c \).

This point in time can be estimated by

\[
\hat{t} = \min\{ \tau \mid N \left[ F(\tau, \hat{\theta}) - F(\tau, \hat{\theta})_i \right] < c \} \quad (7.119)
\]

A useful approximation is

\[
\hat{t} \approx \min\{ \tau \mid N f(\tau - 0.5, \hat{\theta})_i < c \} \quad (7.120)
\]

In a practical case, our choice of \( c \) will depend on the risk we are willing to take. For high quality software it might be 0.01 while it is other cases may be 0.5 or even 1.0.

A reasonable choice of \( c \) when \( t_{acc} \) is the point in time when we can deliver, would be to use

\[
\text{MTTF} = \frac{\text{period length}}{\text{no. of failures}}
\]  

(7.121)

When MTTF and the period length are specified, the number of failures in the last testing period \( (c) \) can be found directly.

E.g.
The contract specifies MTTF = 1 year. If our testing periods are equal to one month, we get from (7.121)

\[ 12 = \frac{1}{C} \Rightarrow C = \frac{1}{12} \]

This approach should, however, be used with care since \( V(n) \) is unknown.

Another way to use \( t_{acc} \) is to predict when our current test strategy will be exhausted, i.e. not discover new errors. If the current \( N_0 \) estimate is much lower than the \( N_0 \) predicted from the organisations faults per (1000) lines, we should probably search for a new test strategy.
8. Case Studies

This chapter starts with an attempt to categorize \( n_i \) data sets by the shape of the \( n_i \) plot. Three categories of shapes are used: Skewed to the left, skewed to the right and bimodal.

We then go on to discuss how to remove some of the "noise" in the data, due to holidays, sickness leave, overtime etc.

Another important point when we want to use historical failure data for estimating \( N_0 \) is whether the data sets are complete. This chapter presents a heuristic algorithm which attempts to solve this problem.

At last, we suggest ways to compare distributions and estimation methods.

It is important to see how well real life data conform to one of the suggested distributions, i.e. log normal, Weibull and gamma. Before we do this however, we need to check the data we are going to use. Three problems are important:

- Are the data sets complete? That is: Have we observed all failures which will occur with the present strategy for selecting test data
- How can we remove noise in the data which stems from
  - vacations, holidays (Christmas, Easter etc.)
  - variation in working hours due to other commitments inside
    the user organisation etc.

- Do the data stem from one or more life cycle phases

In addition we must decide upon one or more methods by which we can
evaluate the predictions computed from the models and estimation
methods in question.
8.1 Data from One or Several Life Cycle Phases

From the discussion in chapter 5.3 it follows that only data from one single life cycle phase can be expected to have a single error life time distribution. This follows from the fact that the life time distribution depends on how the input are selected and the usage intensity (i.e. $P_{new}(.)$ and $w(t)$).

When we checked the sources for the data sets gathered in [11] we found that

. all cited papers stated from which life cycle phases the data were gathered

. 22 out of the 50 data sets contained data from more than one life cycle phase.

From the 22 multi phase data sets we were able to extract 27 uni phase data set. In some cases it was possible to single out data from up to three different phases from one data set. This "weeding" process thus gave us a total of 55 uni-phase data sets. When the data sets are categorized according to lifecycle phase and application type, we get the following matrix
<table>
<thead>
<tr>
<th>System type</th>
<th>Debugging Testing</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Real time</td>
<td>24</td>
<td>11</td>
</tr>
<tr>
<td>Batch systems</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>No info available</td>
<td>11</td>
<td>3</td>
</tr>
</tbody>
</table>

Table 8.1

Another important way of categorizing the data, is according to the shape of n_i. Plotting of the datasets in [11], reveals that the data can be categorized into one of the three broad categories shown in fig. 81 a, b and c.
Fig. 8.1

Note that all these shapes can be arrived at by reasonable assumptions on $P_{\text{new}}(.)$ and $w(t)$. 
A. This is the Gamma/Weibull/Log Normal-like distribution which can be derived from a variety of assumptions. Cfr. for instance chapter 6.4.

B. This shape is derived from assumptions mostly pertaining to testing and debugging. Cfr. chapter 6.1.2.

C. This shape is also characteristic for testing and debugging, but only in a poorly controlled project. Cfr. the panic-model in chapter 6.1.3.

When the matrix in table 8.1 is further subdivided according to category, we get the results shown in table 8.2.
From the discussions of $P_{\text{new}(.)}$ and $W(t)$ in chapter 6, we should expect that

- $n_i$ in the operational phase should be log Normal, Weibull or gamma-like

- $n_i$ in the debugging and test phases should show a greater variety of shapes

When we look at table 8.2, we see that this is indeed the case. The data gathered thus support the theory very well.
8.2 Noise Removal

When we plot the number of failures reported per time unit, the emerging picture is far from smooth. In some cases the term "noisy" is probably an understatement. This should not come as a surprise when we consider that the number of failures reported among other things must depend strongly on the hours worked. This implies that such things as

- weekends
- holidays
- the state of other projects competing for the same personnel
- sickness

will have strong influence on the number of tests executed and thus on the number of failures reported.

If we want to do something useful in the area of software reliability prediction, we must come to grips with these factors one way or another. The idea of smoothing the data seems to be a simple and effective solution.

To illustrate the problem, fig. 8.2 shows the number of failures reported as a function of calendar time. The data are taken from [11].
The need to remove noise from data is well known in various statistical application areas, such as estimation of priors in Bayesian analysis and mortality tables in actuarial work. For the latter, see [44].

The basic ideas of smoothing are quite simple. I have used the ideas and results from Silverman, as published in [13]. Let us use the following notation

\(^\wedge n(x)\) is the smoothed number of failures reported per period, estimated from the data \(\{n_i\}\).

\(X\) is the period number in the smoothed function

\(X_i\) is the period number in the observed function
\( K(.) \) is the kernel (weighting) function

\( h \) is the "window" width.

The basic smoothening formula is

\[
\hat{A}(x) = \frac{1}{N} \sum_{i=1}^{N} \frac{x-x_i}{h} K\left(\frac{x-x_i}{h}\right)
\]  

(8.1)

In our case \( X \) and \( X_i \) are period sequence numbers and we have \( n_i \) observations of \( X_i \). (8.1) can thus be rewritten as

\[
\hat{n}(m) = \frac{1}{h} \sum_{j=1}^{k} \frac{n_j}{n} K\left(\frac{m-j}{h}\right)
\]

(8.1-1)

Here, as usual, \( n_j \) is the number of failures observed in period \( j \). One of the conditions for \( K(.) \) is that

\[
\int_{-\infty}^{\infty} K(x) dx = 1
\]

(8.1-2)

Thus, we can formally write that

\[
\int_{0}^{\infty} \frac{\hat{n}(m)}{h} dm = \frac{1}{h} \sum_{j=1}^{k} \frac{n_j}{n} j
\]

(8.1-3)
This will also hold if our data set \( \{n_j\} \) is not complete. The smoothed data set \( \{\hat{n}_j\} \) can formally be written as

\[
\hat{n}_{j+1} = \int \hat{n}(m) \, dm
\]

(8.1-4)

The two main considerations when one wants to do data smoothing, are the selections of \( h \) and \( K(\cdot) \).

In the general case we can minimize the mean integrated square error (MISE), defined as

\[
\text{MISE}(\hat{f}) = E\{\int [\hat{f}(x) - f(x)]^2 \, dx\}
\]

\( \Omega_x \)

(8.2)

There exists a best kernel function \( K \), known as Epanechnikovs kernel. Other, similar kernels are, however, only marginally worse. For practical reasons, I have used a Gaussian kernel, that is

\[
K(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}
\]

(8.3)

When it comes to the window width, there exists several methods ranging from subjective estimates to the minimization of a score function. I have chosen a simple method which is robust and almost
as good as the optimal width, according to Silverman. This window width, $h_{opt}$ is defined as

$$\frac{1}{5} = 0.79R N_{opt}$$ (8.4)

where $R$ is the interquartile range of the data. This value of $h$ is close to the optimal choice as long as the underlying distribution is unimodal. In this paper, we will only consider unimodal data sets.

The resulting smoothing algorithm was applied to each data set before parameter estimation. As should be expected, the results with smoothed data were better than with unsmoothed ones in all cases.

As an example, fig. 8.3 shows the results of smoothing the data from fig. 8.2.
When we compute $\beta_1$ and $\beta_2$ (skewness and curtosis) for the smoothed and unsmoothed data, we get

unsmoothed: $\beta_1 = 1.70$, $\beta_2 = 3.90$

smoothed: $\beta_1 = 1.31$, $\beta_2 = 3.47$

This represents a considerable displacement in the $\beta_1$-$\beta_2$ diagram, but since this displacement is diagonally from right to left, it does not change the character of the distribution.
8.3 Completeness of the Data Sets

Since one of our main aims is to estimate $N_0$ (number of faults inside $D_0$, cfr. chapter 4.2) it seems natural to rank estimation methods according to how well they predict $N_0$ from the observations. In order to do this, we must know the true value of $N_0$ for the current site.

One of our main problems, however, is that we do not know the true value of $N_0$. All we have is the sequence $\{n_i\}$. If no further errors are found at the site of interest, then $N_0 = \sum n_i$. The problem of finding the true $N_0$ value is thus transformed into the problem of deciding whether any new errors will be found at the site under the current usage profile ($P_{\text{new}}(.)$). If not, the data set is said to be complete.

We suggest the following rules:

A data set is complete if at least one of the following conditions are met:

A: $\{n_i\}$ have an unbroken trend towards zero for at least the last three periods and the last $n_i < 2$.

B: None of the last three periods have more than one failure reported and the total number of failures in these three periods is less than three. This is, however, only a reasonable criteria when the number of failures already observed is greater than 50, say.

C: the two last elements in $\{n_i\}$ are zero.
D: the author(s) of the paper where the data are shown concludes that "no further failures occurred" or some statement to that effect.

Some comments:

The two last conditions (C and D) are the strongest. Conditions A and B must, however, be included to take care of data sets which do not include a series of zeros at the end because

- the users or testers have stopped reporting when there was nothing to report
- the person who wrote the paper presenting the data felt it unnecessary to report a string of zeros at the end of the data set.

Of the 35 data sets which are skewed to the left and thus can be considered to be gamma, Weibull or log normal distributed, only 18 could be defined as complete. Concerning the remaining 17 data sets, the following information was available:

7 data sets: there are several errors left

5 data sets: the number of errors not included in the published data set was found in the papers

5 data sets: no information could be obtained

For the complete data sets and the data sets where the total number of errors are known, we can check the predicted $N_0$ against the real value.
For the seven cases where we can only establish that there are several errors left, the only thing we can use as a quality metric is that $N_0$ should be greater than $\ln 1$. Since this seems a rather dubious criterion, these data sets are left out, together with the five data sets where no information could be obtained.

All in all, these considerations leave us with 21 data sets which can be used to evaluate estimation models for $N_0$ and $n(t)$. 
8.4 Comparison of Models

Once we have a set of data that are

\[ \sum_{i=1}^{k} n_i = N \]

...complete; that is...

gathered under one life cycle phase only

it is possible to test which distributions fit the observed data.
I have chosen to use the Kolmogorov-Smirnov test [24] for this
purpose.

We want to test the following hypothesis

H0: the lifetimes for the faults are distributed according to
F(.)

From our discussion in chapter 5.1 (see especially (5.15) and
(5.16)) we will reject Ho if

\[ \max_{i \in [1,k]} \left| \frac{1}{N} \sum_{j=1}^{N} \frac{n_j}{n} - F(i) \right| = D > d_{k,k,\alpha} \]  \hspace{1cm} (8.5)

Here, \( \hat{F}(i) \) is the distribution function \( F(.) \) with its parameters
estimated from the data set \( \{n_j\} \) and \( d_{k,\alpha} \) is the test
limit with confidence level $\alpha$ and $k$ observations. The $d_{k,\alpha}$ values are tabulated in many textbooks on nonparametric analysis. I have used the tables in [21].
8.5 Comparison of Estimation Methods

Since we focus on estimation of $N_0$, it seems reasonable to rank the methods according to how well they estimate $N_0$.

There are at least three strategies which can be chosen. The methods can be ranked according to how well they do

1) early in the process (few data available)

2) late in the process (all or most data are available)

3) throughout the process

All these strategies can be included in the formula

$$r = \frac{\sum_{i=1}^{k} (N - \hat{N}_{0i})^2 W_i}{\sum_{i=1}^{k} W_i}$$  \hspace{1cm} (8.5-1)

where we have used the notation $\hat{N}_{0i}$ for the estimate of $N_0$ after $i$ periods

$k$ : number of periods

$W_i$ : the weight (importance) of the estimate after $i$ periods

The fig. 8.4 suggests choices for $W_i$ for the three strategies
It is easy to find situations in a software project where strategy 1 or 2 would be a reasonable choice. Eg.:

- for planning purposes in the early phase of the project one would stress the model's capability to give good predictions with few data. Typical questions one would like to ask are:

  . how many errors are we going to find with our present testing strategy and man power load

  . when can testing personnel be transferred to other projects

- for testing purposes, late in the testing phase one would stress the importance of being able to say that all errors which are going to be found by the present strategy really are found when $\hat{N}_o = N$.

Important as these considerations are, we will still use equal weighting for all estimates $\hat{N}_{oi}$. The reason for this choice are
the main purpose of this work is to find the over all best estimation method.

a method which gives good results for strategy 3 will also give reasonable results for the whole period and thus give results which can be used for planning purposes at an early stage.

Another way to compare different models is to compute the relative error at each stage $i$.

$$
\epsilon_i = \frac{N_i - N^A_i}{N^0_i}
$$

This measure is connected to the measure proposed in (8.5) since

$$
r = \frac{\sum w \epsilon_i^2}{N^0 \sum w_i}
$$

By plotting $\epsilon_i$ for an estimation method, we can get a good overview of how the method behaves for different amounts of data. A typical example of an $\epsilon_i$ plot is shown in fig. 8.4a. The results where obtained when using the log normal distribution on a data set from a telecom system and estimating the parameters by linearization. (Cfr. table 3.3e in [11]).
This plot can also be used in another way. It is important to get an $N_0$-estimate as close as possible to the real value as early as possible. Thus, the last passage through the $\epsilon=0.1$ line from above is a reasonable quality criteria. If this crossover point is called $a$, we will say that model 1 is better than model 2 if $a_1 < a_2$ when all failure times are observed. From (8.6) it follows that $\epsilon$ is a function of $N_0$. Since $N_0$ is a function of the random variables $\{t_i\}$, it follows that $N_0$ and thus $\epsilon$ are random variables.

Formally, the 10% crossover point is defined as
\[ C_{0.1}(\text{method}) = j, \varepsilon(\text{method}|t_j) = 0.10 \]
\[ , \quad \varepsilon(\text{method}|t_j) < 0.10 \]
\[ \text{for all } i > j \]  

(8.8)

Here, \( t = \{t_1, t_2, \ldots, t_j \} \). From (8.8) it now follows that \( C_{0.1}(.) \) is a random variable. Furthermore, since \( C_{0.1}(.) \) is a deterministic function of \( N \), there is no reason, a priori, why \( C_{0.1}(\text{method 1}) \) should be different from \( C_{0.1}(\text{method 2}) \) when we compare the two for many datasets. We thus have that:

Number of cases where \( C_{0.1}(m1) > C_{0.1}(m2) \) should be binomially distributed with \( p=0.5 \).

We can thus use the sign test to test whether \( m1 \) is better than \( m2 \) for the same \( t_j \). A realistic example is shown in fig. 8.5.
Note that the two criteria shown in (8.5-1) and (8.8) need not agree. The data shown in fig. 8.5 is a case in point. Here (8.5) will rank model 2 before model 1 while (8.8) will get the opposite result.

Which criterion to choose is a problem akin to selection between strategies 1, 2 and 3 in the beginning of this chapter.
8.6 Discussion of Some Real Data Sets

A total of 21 data sets taken from [11] are judged to be skewed to the left and complete (cfr. chapter 8.1 and 8.3).

If we use the Kolmogorov-Smirnov test (cfr. chapter 8.4) and take into consideration \( r \) (8.4) and \( \varepsilon_k \) (8.6) it is possible to categorize these data into one of the three categories gamma, log normal or Weibull.

Since the ML estimator for \( N_0 \) is usually bad for heavy right censoring, I have used \( r \) and \( \varepsilon \) from the linearized models for log normal and gamma. For the Weibull distribution I have used the MLE, since this distribution can not be linearized.

In order to make decisions I have used the following rule

\[
\{t_i\} \text{ i.i.d. We reject the statement} \\
\{t_i\} \sim F(.)
\]

if one or more of the following statements are true:
1) \[ \max_{i \in [1, k]} \frac{1}{n} \sum_{j=1}^{n} \hat{A} \left| F(i) \right| > d \quad \alpha = 0.10 \]

2) \( \epsilon_{\text{final}} > 0.10 \)

3) \( r > 10^4 \), when all \( W_i = 1 \).

Some comments:

- rule 1 is an absolute minimum requirement since it measure the conformance between data and distribution when all failures have been observed.

- rule 2 is necessary if we want our results to be of practical value

- rule 3 concerns the overall fit of the model and estimation method. Since \( r \sim N_0^2 \) and \( N_0 \in [100, 500] \) for most systems after delivery, the limit \( 10^4 \) seems reasonable.

Note that acceptance of the hypothesis \( \{ t_i \} \sim F(.) \) does not imply that \( F(.) \) is the distribution of the life times of the errors. This is born out very clearly by the observation that in 12 cases several distributions give a no-reject conclusion for the data. The results for the 21 afore mentioned data sets are shown in table 8.3.
<table>
<thead>
<tr>
<th>Title of data set [11]</th>
<th>Log Normal</th>
<th>Gamma</th>
<th>Weibull</th>
</tr>
</thead>
<tbody>
<tr>
<td>DS1</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>DS2</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Space Shuttle test</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Real Time Command II</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Commercial Subsys</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Military System II</td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Operating System 40</td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Fujitsu I</td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Fujitsu II</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>System A, site 2</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>System A, site 1</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Subsystem AA, site 1</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>Subsystem AB, site 1</td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>APZ</td>
<td>X</td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>APT</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MARY Compiler</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NUALG Compiler (tot)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NUALG Compiler (record)</td>
<td></td>
<td></td>
<td>X</td>
</tr>
<tr>
<td>SS1B Operating System</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>HRP (true lifetimes)</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>VTT (true lifetimes)</td>
<td></td>
<td></td>
<td>X</td>
</tr>
</tbody>
</table>

Table 8.3

The findings in table 8.3 can be summarized as follows:
- four data sets lead to rejection for all three distributions (Space Shuttle test, MARY compiler, NUALG Compiler (tot) and SS18 Operating System). These data sets will from now on be removed from consideration, thus bringing the number of data sets down to 17.

- all three distributions are accepted for three data sets

- two distributions are accepted for nine data sets

- one distribution only is accepted for five data sets

All data that conform to one or more of the three distributions will be used in the Bayesian analysis of the model in question.

Note that if we use only the Weibull distribution, we catch 76% of all cases. If we in addition include the gamma distribution we catch all cases observed.

Fig. 8.6 and 8.7 show plots of \( \hat{N}_0 \) vs \( t \) for one data set that fits all three distributions and one set that fits two of them. Except for very heavy right censoring, the differences are not large.
Fig. 8.6 Data for HRP (true lifetimes)
Fig. 8.7 Data for System A, site 1
9. **Evaluation of some Estimation Methods for the L-D Model**

Chapter nine applies the results from chapter eight to compare the two estimation methods, the MLE from Sanathanans conditional likelihood (chapter 7.6) and the linearization method (chapter 7.4). The evaluation is done by using both methods on twenty simulated data sets from each distribution.

The unanimous conclusion is that linearization is always better than, or as good as, the conditional Maximum Likelihood.

The purpose of this evaluation is twofold:

- how does linearization compare with Sanathanans solution for data-sets with known distribution?

- how does the two above mentioned methods work out for real world data?

For comparison we will use the criterions discussed in chapters 8.4 and 8.5.

- \( r \) in (8.5-1) with all \( W_i = 1 \).

- \( \varepsilon \) when all data are used (8.6)

- the 10% crossover point, as discussed at the end of the chapter 8.5 (\( C_{0.1} \))

- \( D_k \) as defined in (8.5)
9.1 **Linearization vs Maximum Likelihood**

Of the four distributions used in the previous chapters, only three can be linearized. These three are:

- \( \text{Gamma}(\alpha, \beta) \)
- \( \text{Log Normal}(\mu, \tau) \)
- \( \text{Log Gamma}(\alpha, \beta) \)

The Weibull distribution can not be linearized in this manner and is thus left out in this discussion.

For the three abovementioned distributions we have registered \( C_{0.1} \) and \( r \) from 20 simulated datasets of grouped data, each consisting of 200 failures. For a description on how the simulated data sets are generated, see appendix B.

If we observe both

\[
C_{0.1}^{\text{(method 1)}} < C_{0.1}^{\text{(method 2)}}
\]

\[
r^{\text{(method 1)}} < r^{\text{(method 2)}}
\]

(9.1)

it seems reasonable to claim that method 1 is better than method 2.

The results for \( r \) are presented as follows:

\[
r^{(\text{est.method})} \approx \text{lower bound} - \text{upper bound}
\]
We have rounded the values of both upper and lower bound in order not to clutter the presentation. For this reason we have used \( \approx \) instead of \( = \).

For \( C_{0.1} \), we use a table where we list the number of data sets where \( C_{0.1}(\text{method 1}) \) is greater than \( C_{0.1}(\text{method 2}) \) and the same for less and equal. In addition, the mean and standard deviation are computed for both \( C_{0.1} \) sets.

9.1.1 Gamma Distributed Data

When we compare the two estimation methods on 20 simulated data sets from the Gamma distribution we see that

\[
\begin{align*}
    r(\text{linear}) & \approx 3 \times 10^3 - 3 \times 10^4 \\
    r(\text{MLE}) & \approx 10^7 - 10^8
\end{align*}
\]

(9.2)

There seems to be no question about the conclusion

\[ r(\text{linear}) < r(\text{MLE}) \]

For the 10% crossover, the situation is slightly less obvious. We find:

\[ C_{0.1}(\text{linear}) < C_{0.1}(\text{MLE}) \text{ in 16 cases} \]

\[ C_{0.1}(\text{linear}) > C_{0.1}(\text{MLE}) \text{ in 2 cases} \]
\( C_{0.1} \text{ (linear)} = C_{0.1} \text{ (MLE) in 2 cases} \)

The medians for the two data sets are not, however, dramatically different. We get the following medians:

\[
\begin{align*}
C_{0.1} \text{ (MLE)} & : 51.65 \\
C_{0.1} \text{ (linear)} & : 41.95
\end{align*}
\]

(9.3)

A much more serious problem with the MLE is that for all the 20 data sets \( N_0 \) is grossly overestimated when we only use observations from the first five to 20 periods. We have that:

\[
\begin{align*}
\varepsilon_{\text{max}} \text{ (linear)} & \approx 0.7 - 0.9 \\
\varepsilon_{\text{max}} \text{ (MLE)} & \approx 100 - 400
\end{align*}
\]

(9.4)

The final \( \varepsilon \), denoted \( \varepsilon_f \) is small for both estimation methods. We get

\[
\begin{align*}
\varepsilon_f \text{ (linear)} & = 0.01\% \\
\varepsilon_f \text{ (MLE)} & = 0.11\%
\end{align*}
\]

(9.5)

There is no reason to reject the null-hypothesis

\[
H_0 : \varepsilon_f \text{ (linear)} = \varepsilon_f \text{ (MLE)}
\]

(9.5-1)

on the 5% level.
Thus:

Linearization seems to do better than the Maximum Likelihood method for Gamma distributed data when the all-over performance is considered.

A typical plot for $\epsilon$ for the two methods applied to the same dataset is shown in fig. 9.1.
Gamma distributed data

Fig. 9.1
9.1.2 Log Normal Distributed Data

As for the gamma distribution, we find that \( r(\text{MLE}) \gg r(\text{linear}) \) in all cases

\[
\begin{align*}
  r(\text{linear}) & = 2.10^2 - 6.10^3 \\
  r(\text{MLE}) & = 1.10^6 - 7.10^9
\end{align*}
\]  

There seems to be no question about the conclusion

\[ r(\text{linear}) < r(\text{MLE}) \]

Contrary to the gamma distribution case, the 10% crossover points can not be said to be different.

\[ C_{0.1}(\text{linear}) < C_{0.1}(\text{MLE}) \text{ in 13 cases} \]
\[ C_{0.1}(\text{linear}) > C_{0.1}(\text{MLE}) \text{ in 7 cases} \]
\[ C_{0.1}(\text{linear}) = C_{0.1}(\text{MLE}) \text{ in 0 cases} \]

The two medians are

\[
\begin{align*}
  C_{0.1}(\text{MLE}) & : 21.3 \\
  C_{0.1}(\text{linear}) & : 24.2
\end{align*}
\]  

As for the gamma distribution case we have a large overestimation for heavily right censored datasets. We find

\[ \varepsilon_{\text{max}}(\text{linear}) \approx 0.2 - 2.4 \]
\[ \varepsilon_{max}(\text{MLE}) \approx 17 - 840 \] (9.8)

For the log normal distribution we get

\[ \varepsilon_f(\text{linear}) = 0.00\% \] (9.9)

\[ \varepsilon_f(\text{MLE}) = 0.00\% \]

The conclusion for the log normal distribution is

Linearization seems to do better than the Maximum Likelihood method for the Log Normal distribution when the all-over performance is considered.

A typical plot of \( \varepsilon \) for the two methods applied to the same data set is shown in fig. 9.2.
9.1.3 Log Gamma Distributed Data

As for the two previous distributions, we find that $r(\text{MLE}) \gg r(\text{linear})$ in all cases. The differences are, however, not as spectacular as for the gamma and log normal distributions.
\[ r(\text{linear}) \approx 5 \times 10^3 - 10^4 \]
\[ r(\text{MLE}) \approx 7 \times 10^4 - 1.5 \times 10^5 \]  
(9.10)

As before, the conclusion

\[ r(\text{linear}) < r(\text{MLE}) \]

seems obvious. For the 10% crossover point, the conclusion is difficult, since we, for some cases fail to reach the 10% limit at all. This is the case for three data sets in the MLE case and eight data sets in the linearized case. We get the following results

\[ C_{0.1}(\text{linear}) < C_{0.1}(\text{MLE}) \text{ in 11 cases} \]

\[ C_{0.1}(\text{linear}) > C_{0.1}(\text{MLE}) \text{ in 8 cases} \]

\[ C_{0.1}(\text{linear}) = C_{0.1}(\text{MLE}) \text{ in 1 cases} \]

The two medians are

\[ C_{0.1}(\text{MLE}) : 100.0 \]  
(9.11)

\[ C_{0.1}(\text{linear}) : 105.0 \]

I have, rather arbitrary, choosen to use 110.0 for 100+ in all cases.

Another choice of value for 100+ will alter the two medians, but will not change the conclusion as long as we select a value less than say 200.0.
The difference between $\varepsilon_{\text{max}}$ for the two estimation methods is small, at least compared to the differences we have observed for the two previous distributions. We find

\[
\varepsilon_{\text{max}} \text{ (linear)} \approx 0.4 - 0.9
\]

\[
\varepsilon_{\text{max}} \text{ (MLE)} \approx 1.2 - 1.9
\]  

(9.12)

For the log gamma data we get

\[
\varepsilon_f \text{ (linear)} = 11.47\%
\]

\[
\varepsilon_f \text{ (MLE)} = 8.18\%
\]  

(9.13)

There is no reason to reject the null-hypothesis

\[
H_0 : \varepsilon_f \text{ (linear)} = \varepsilon_f \text{ (MLE)}
\]  

(9.13-1)

at the 5% level.

The conclusion for the log gamma distribution is

Linearization seems to be as good as or better than Maximum Likelihood for the Log Gamma distribution when the all-over performance is considered.

A typical plot of $\varepsilon$ for the two estimation methods is shown in fig. 9.3.
Fig. 9.3
9.2 Weibull Distribution

Since the Weibull function cannot be linearized, this chapter summarizes the result of Sanathanans MLE applied to 20 datasets of Weibull distributed failure times.

We obtained:

\[ r(\text{MLE}) \approx 4.10^4 - 2.5.10^5 \]  \hspace{1cm} (9.14)

which is in the same ballpark as \( r(\text{MLE}) \) for the log gamma distribution. It is, however, considerably better that \( r(\text{MLE}) \) for the log normal and gamma data.

\[ C_{0.1}(\text{MLE}) \approx 54.0 \]  \hspace{1cm} (9.15)

\[ \varepsilon_f(\text{MLE}) = 0.00 \% \]  \hspace{1cm} (9.16)

\[ \varepsilon_{\text{max}}(\text{MLE}) \approx 2.5 - 4.9 \]

These results are also in the same area as the corresponding quantities for the other distributions which are discussed in this chapter.

A typical example of the \( \varepsilon(\text{MLE}) \) development is shown in fig. 9.4.
Weibull distributed data

Fig. 9.4
10. **Bayesian L-D Models**

We will now discuss Bayesian estimation for the model parameters in some of the distributions.

First we discuss the Bayesian solution for the Gamma and (Log) Normal distribution parameters. This is just a short summary of the standard Bayesian results for these distributions. They are cited here just to serve as a frame of reference for the Bayesian analysis of the corresponding conditional distributions. These are shown in chapter 11.

Lastly, the Bayesian estimation of $N_0$ is considered. We treat the case where the failure time distribution parameters are known, while $N_0$ has a Poisson prior. This prior follows from a discussion on how errors are inserted into a software system.

Since many organizations have, or are about to gather failure histories from their projects, it should be reasonable to include this experience into the L-D model. A natural way to do this is by using a Bayesian model. Those not familiar with Bayesian statistics should confer a book like [21].
10.1 Bayesian Treatment of the Gamma Distribution

When we allow both $\alpha$ and $\beta$ to have prior distributions, there exists at least two approaches, put forward by Damsleth [18] and Miller [19] respectively. Their main difference lies in the amount of computational work necessary. We will use the approach suggested by Miller.

Let the failure times $t_i \sim \text{Gamma}(\alpha, \beta)$. Miller suggests the following prior joint distribution for $\alpha$, $\beta$:

$$f(\alpha, \beta) = C \frac{\beta^{\alpha - 1} \cdot \alpha - s\beta}{\Gamma(\alpha)} \cdot p^\alpha \cdot e^{-s\beta} \tag{10.15}$$

Except for the $e^{-s\beta}$ term, this distribution is akin to Damsleth’s GamconI distribution (cfr. [18], equation 3). When we have observed all the $k$ failure times in our sample, the joint distribution is

$$t, \alpha, \beta \sim C \frac{\beta^{\alpha - 1} \cdot \alpha - s\beta}{\Gamma(\alpha)} \cdot p^\alpha \cdot \frac{\beta^k \cdot (-k(\alpha - 1))}{\Gamma(\alpha)} \cdot e^{-\beta tk} \tag{10.16}$$

Let $\alpha^*$ be a fixed $\alpha$-value. We then have that
\[ f(\beta|t, \alpha^*) = \frac{\kappa(t, \alpha^*, \beta)}{m(t, \alpha^*)} \quad (10.17) \]

\[ m(t, \alpha) = g(\alpha|t) \kappa(t) \quad (10.18) \]

For a given \( F_t \), \( \kappa(t) \) is a constant. We can thus combine (10.17) and (10.18) into

\[ f(\beta|t, \alpha^*)g(\alpha^*|t) = \text{Gamma}(\alpha^*(v+k), s+kt). \]

\[ \frac{1}{\binom{v+k}{p} \alpha^*(v+k)} \frac{k}{\Gamma[\alpha^*(v+k)]} \frac{1}{s+kt} \frac{\Gamma[\alpha^*(v+k)]}{\Gamma(\alpha^*)} \frac{n+k}{n+k} \quad (10.20) \]

Thus:

\[ f(\beta|t, \alpha) = \text{Gamma}[\alpha(v+k), s+kt] \quad (10.21) \]

\[ g(\alpha|t) = c \frac{1}{\binom{v+a}{p} \alpha(v+k)} \frac{k}{\Gamma[\alpha(v+k)]} \frac{1}{s+kt} \frac{\Gamma[\alpha(v+k)]}{\Gamma(\alpha)} \frac{n+k}{n+k} \quad (10.22) \]

From (10.21) and (10.22) we see that \( v, s \) and \( p \) can be given the following interpretations

\[ v : \text{initial number of data} \]
s : \sum_{i=1}^{V} t_i \quad \text{where} \{t_i\} \text{are the initial datasets}

p : \prod_{i=1}^{V} t_i

The non-informative prior can thus be obtained by setting

\( \nu = 0, \ s = 0, \ p = 1 \quad (10.23) \)

From (10.15), (10.21) and (10.22) we then get

\[
f(\alpha, \beta) = C \frac{1}{\beta \Gamma(\alpha)^n} \quad (10.24)
\]

\[
f(\beta|t, \alpha) \sim \text{Gamma} (\alpha k, k t) \quad (10.25)
\]

\[
g(\alpha|t) = C \frac{\Gamma(\alpha k)^{\alpha k}}{\Gamma(\alpha)^n \Gamma(k)^{kt}} t^{\alpha k} \quad (10.26)
\]

The GammaII distribution of Damsleth is identical to \( g(\alpha|t) \) to the factor \( \frac{1}{\alpha k} \) when we let

\( \mu = 1, \ \delta = 0, \ n = 0 \quad (10.27) \)

In order to simplify the amount of numerical work, Miller now proceeds as follows.
From (10.22) we can compute the central moments for $\alpha$, $\mu_k$, as follows

$$
\mu_i(\alpha|t) = C \int_0^\infty \frac{(\alpha - \mu)}{\alpha} \frac{\Gamma[\alpha(v+k)]}{\Gamma(\alpha)} \frac{1}{v+k} \frac{k}{\alpha(v+k)} \left( \frac{v+k}{v+k} \left( \frac{P}{t} \right) \right)^{n+k} \left( \frac{s+kt}{s+kt} \right) \, d\alpha \quad (10.28)
$$

where $\mu_0$ and $C$ are determined by the following equations:

$$
C = \int_0^\infty \frac{\Gamma[\alpha(v+k)]}{\Gamma(\alpha)} \frac{1}{v+k} \frac{k}{\alpha(v+k)} \left( \frac{v+k}{v+k} \left( \frac{P}{t} \right) \right)^{n+k} \left( \frac{s+kt}{s+kt} \right) \, d\alpha = 1 \quad (10.29)
$$

$$
\mu_1 = C \int_0^\infty \frac{\Gamma[\alpha(v+k)]}{\Gamma(\alpha)} \frac{1}{v+k} \frac{k}{\alpha(v+k)} \left( \frac{v+k}{v+k} \left( \frac{P}{t} \right) \right)^{n+k} \left( \frac{s+kt}{s+kt} \right) \, d\alpha \quad (10.30)
$$

In both cases, numerical integration are needed.

Once we have the values for $\mu_i(\alpha|t)$, we can compute ([14]) $\beta_1(\alpha)$ and $\beta_2(\alpha)$ (skewness and curtosis). Miller then proves that

$$
k \gg 1 \Rightarrow \beta_1(\beta) \approx \beta_1(\alpha) \quad (10.31)$$
From the usual Bayes estimation we get (10.30)

\[ \hat{\alpha} | t = \mu_\alpha \]  
\[ (10.32) \]

\[ E(\beta | t) = E \{ E(\beta | t, \alpha) \} \]  
\[ (10.33) \]

From (10.21) we get

\[ E(\beta | t, \alpha) = \frac{1}{(s + kt)} \frac{\Gamma[\alpha(v + k) + u]}{\Gamma[\alpha(v + k)]} \]  
\[ (10.34) \]

\[ E(\beta | t) = \frac{1}{(s + kt)} E \left\{ \frac{\Gamma[\alpha(v + k) + u]}{\Gamma[\alpha(v + k)]} \alpha \right\} \]  
\[ (10.35) \]

If we set \( u = 1 \), we get

\[ \hat{\beta} | t = \frac{(v + k)}{s + kt} \mu_\alpha \]  
\[ (10.36) \]

By use of the recursion formula \( \Gamma(x + 1) = x \Gamma(x) \) and \( \mu_1(\alpha | t), \mu_2(\alpha | t) \) from (10.28) it is straightforward to obtain expressions for \( \hat{\alpha} \) and \( \hat{\beta} \).

Given parameters for the prior distributions (C, v, p and s) we can thus obtain the posterior estimates of \( \alpha \) and \( \beta \) and their variances.

In order to find confidence intervals for \( \alpha \) and \( \beta \), Miller proceeds
in the following manner.

- calculate $\sqrt{\beta_1(\alpha)}$ and $\beta_2(\alpha)$ from (10.28) to (10.30)

- use a table for percentage points of Pearson curves to obtain the numbers

\[
\text{tab}_{\text{lower}} (\beta_1, \beta_2), \text{tab}_{\text{upper}} (\beta_1, \beta_2)
\]

- we can then calculate $\alpha_{\text{lower}}$ and $\alpha_{\text{upper}}$ from

\[
\alpha_{\text{lower}} = \mu - \frac{\alpha}{2} t_{\text{lower}} (\beta_1, \beta_2) \tag{10.37}
\]

\[
\alpha_{\text{upper}} = \mu + \frac{\alpha}{2} t_{\text{upper}} (\beta_1, \beta_2) \tag{10.38}
\]

- the values of $\beta_{\text{lower}}$ and $\beta_{\text{upper}}$ can be obtained in the same way

There are several tables available for this purpose. I have used the one in [20].
10.2 Bayesian Formulation of the Log Normal Distribution

The problem of prior and posterior distributions for the Log Normal distribution poses few problems. The reader is referred to [23] for a complete treatment of the problem. Here we will only show the arguments which lead to the necessary posterior distributions.

\[ X \sim N(M, R^{-1}) \]  \hspace{1cm} (10.39)

We will use the following priors

\[ M | R \sim N(\mu, \frac{1}{\tau \tau}) \]  \hspace{1cm} (10.40)

\[ R \sim Gamma(\alpha, \beta) \]  \hspace{1cm} (10.41)

and thus

\[ M, R \sim N(\mu, \frac{1}{\tau \tau}) Gamma(\alpha, \beta) \]  \hspace{1cm} (10.42)

Let us use the notation

\[ A = \frac{\sum x_i \mu}{n + \tau} \]  \hspace{1cm} (10.43)

\[ B = \sum_{i=1}^{n} \frac{(X - x_i)^2}{\tau + n} + \frac{\tau n}{\tau + n} (X - \mu)^2 \]  \hspace{1cm} (10.44)

It is then straightforward to obtain
\( M | X, R \sim N(\mu, \frac{1}{r(n+\tau)}) \) \hspace{1cm} (10.45)

\( R | X \sim \text{Gamma}(\alpha + \frac{1}{\beta}, \beta) \) \hspace{1cm} (10.46)

In order to estimate the parameters in the prior distributions, we need the marginal distribution of \( M \). From (10.42) we get, by integration:

\[
 f(m) = C \left[ 1 + \frac{\tau \alpha}{\beta} \cdot \frac{1}{2\alpha} (m-\mu)^2 \right]^{-\frac{2\alpha+1}{2}} \hspace{1cm} (10.47)
\]

Thus, we get the non-central \( t \)-distribution

\( M \sim T(\mu, \frac{\tau \alpha}{\beta}, \frac{2\alpha+1}{2}) \) \hspace{1cm} (10.48)

From this it also follows that

\[ E(M) = \mu \] \hspace{1cm} (10.49)

\[ V(M) = \frac{\beta}{r(\alpha-1)} \]

(See also [23])

Thus, we get
\[ r|X = \frac{\alpha + \frac{n}{2}}{\beta + \frac{8}{2}} \quad (10.50) \]

\[ m|X,r = A \quad (10.51) \]

From (10.45) and (10.50) we see that \( r \) and \( \alpha \) can be interpreted as the number of observations our \( r \) and \( m \) priors are built on. A non-informative prior can thus be obtained by using

\[ r \to 0, \alpha \to 0 \quad (10.51-1) \]

\[ \beta \to 0 \quad \exists \frac{\alpha}{\beta} = \text{constant} \]

This will give us the estimates:

\[ m|X,r = x \]

\[ r|X = \frac{n}{n-1} \frac{s^2}{x} \quad (10.51-2) \]

which is the usual ML estimates.

In the log normal case, (10.39) must be replaced by

\[ X = \ln(M,R^{-1}) \quad (10.52) \]

This can be done by the following substitutions
\[ x \rightarrow \ln x \quad \frac{i}{i} \]

\[ x \rightarrow \frac{1}{\sum_{n} \ln x} \quad \frac{i}{i} \]

(10.53)
10.3 Bayesian Estimation of the Total Number of Errors

In the preceding chapters we have looked at prior distributions for the parameters in the fault lifetime distributions. In this chapter, however, we will consider a prior distribution for \( N_0 \) while \( F() \) is assumed to be known. This is not, by itself, a realistic assumption. We do it, however, in order to use a two-step approach:

1) estimate the parameters in the failure time distribution (Bayes or otherwise)

2) estimate \( N_0 \) from its prior plus the parameters we estimated in 1).

In chapter 10.3.1 we will show that there exists a natural choice for the prior distribution of \( N_0 \) and arrive at a general result for the posterior distribution.

In chapter 10.3.2 we will use historical data to arrive at values for the parameters in the posterior distribution.

10.3.1 A Prior Distribution for the Total Number of Errors

We may look at the construction of a piece of software as a series of stochastic experiments. Given the programmers skill and the complexity of the task, there is a certain probability, say \( p \), of making an error each time he writes a program statement. If we in
addition assumes that the two events \{make an error in line \(x\}\), \{make an error in line \(Y\)\} are independent, then the number of faults in the finished software product is binomially distributed.

If we assume that the number of lines \(l\) is large and that \(p\) is small \((p \ll 1)\), we have that

\[
\text{no. of errors is approximately } \text{Poisson}(lp) \tag{10.54}
\]


This approximation holds a fortiori when a project team is writing the code. In this case we can apply the general result that when

- we have several independent processes with \(P(\text{event in } [t, t+\Delta t]) \ll 1\)

- none of the processes dominate over the others

then the process we get when we add these processes together, will be a Poisson process [33]. In our case, a point in the process is the event that an error is inserted into the program.

For a given software system we will thus, apriori, assume that

\[
N_0 \sim \text{Poisson } (\theta) \tag{10.55}
\]

In addition, we will assume that

\[
\theta \sim \text{Gamma}(\alpha, \beta) \tag{10.56}
\]

Note that although \(\theta\) can be assumed to be constant for a given project team operating in stable environments, \(\theta\) will vary for...
- different program sizes
- different teams
- different projects
- different organisations

When we have no knowledge of who produced the software and under what circumstances, Θ should be handled as a random variable.

When we have a total of $N_0$ faults in a system and $N$ failures are observed at the times $t_1, t_2, \ldots, t_N$, we have the likelihood

$$f(t | N) = \frac{N!}{(N-N)!} \prod_{i=1}^{N} f(t_i) F(t_i)^{N-N}$$  \hspace{1cm} (10.57)

According to thesis 2 (cfr. chapter 5.2), we have that Θ is independent of $\{t_i\}$ and all parameters in $f(.)$

Thus:

$$f(t, N | Θ) = f(t | N_0) g(N | Θ)$$  \hspace{1cm} (10.58)

$$f(t, N | Θ) = \frac{Θ^{N_0 - Θ}}{(N-N)!} \prod_{i=1}^{N} f(t_i) F(t_i)^{N-N}$$  \hspace{1cm} (10.59)

$$f(t, N) = \int_{Θ} f(t, N | Θ) h(Θ) dΘ$$  \hspace{1cm} (10.60)
where \( h(\theta) \) is defined by (10.56).

\[
f(t,N) = \prod_{i=1}^{N} \frac{f(t_i) - N - N}{F(t_i) - N} \cdot \frac{\alpha}{\beta \Gamma(N + \alpha)}
\]

From (10.59) we obtain \( f(t|\theta) \) as follows

\[
f(t|\theta) = \sum_{N=0}^{\infty} \left\{ \prod_{i=1}^{N} \frac{f(t_i) - N - N}{F(t_i) - N} \cdot \frac{\theta^N}{\theta e} \right\}\frac{\theta^N}{(N-N)!}
\]

\[
f(t|\theta) = \prod_{i=1}^{N} \frac{N - \theta}{\theta e} F(t_i)
\]

By using (10.63) it is now straightforward to obtain

\[
f(t) = \int_{\theta=0}^{\infty} f(t|\theta) h(\theta) d\theta
\]

\[
f(t) = \prod_{i=1}^{N} \frac{\beta^\alpha}{[\beta + F(t_i)]^{\alpha+N}} \frac{\Gamma(\alpha+N)}{\Gamma(\alpha)}
\]

From (10.61) and (10.65) we can now at last find the distribution
of \( N_0 \) given \( t \).

\[
g(N \mid t) = \frac{f(t, N)}{f(t)} \tag{10.65}
\]

\[
g(N \mid t) = \frac{-N^{-N}}{(N-N)!} \frac{[\beta + F(t)]^N}{\Gamma(\alpha + N)} \frac{\alpha + N}{(1+\beta)^N} \frac{\Gamma(\alpha + N)}{\Gamma(\alpha)} \tag{10.67}
\]

In order to find the posterior distribution for \( \theta \), we use the following approach

\[
h(\theta \mid t)f(t) = f(t \mid \theta)h(\theta) \tag{10.68}
\]

\[
h(\theta \mid t) = \frac{f(t \mid \theta)h(\theta)}{f(t)} \tag{10.69}
\]

By using (10.56), (10.63) and (10.65), we get

\[
\theta \mid t \sim \text{Gamma}[\alpha + N, \beta + F(t)] \tag{10.70}
\]

The Bayesian estimate for \( N_0 \) under a quadratic loss function is then
\[ N^* = \sum_{N=N_0}^{\infty} N \cdot g(N \mid t) \quad \text{(10.71)} \]

\[ N^* = \sum_{N=N_0}^{\infty} \frac{F(t)}{o} \left( \frac{N - N}{(N - N)!} \right) \frac{\beta + F(t)}{\Gamma(\alpha + N)} \quad \text{(10.72)} \]

By substitution of \( i = N - N \), we get

\[ N^* = \sum_{i=0}^{\infty} \frac{(-1)^i F(t)}{i!} \left( \frac{\beta + F(t)}{\Gamma(\alpha + N)} \right) \quad \text{(10.73)} \]

\[ N^* = \left[ \frac{\beta + F(t)}{\Gamma(\alpha + N)} \right] \left( \frac{1}{(1 + \beta)} \right) \quad \text{In order to find } N^*, \text{ we need the general results for} \]

\[ S(k, p) = \sum_{i=0}^{\infty} p \frac{\Gamma(k+i)}{\Gamma(i+1)} , \quad |p| < 1 \quad \text{(10.75)} \]

\[ S(k, p) = \sum_{i=0}^{\infty} i p \frac{\Gamma(k+i)}{\Gamma(i+1)} , \quad |p| < 1 \quad \text{(10.76)} \]
It can be shown that

$$S_{\text{o}}(k,p) = \frac{\Gamma(k)}{(1-p)^k}$$  \hspace{1cm} (10.77)$$

$$S_1(k,p) = \frac{\Gamma(k)}{(1-p)^k} \cdot \frac{pk}{1-p}$$  \hspace{1cm} (10.78)$$

(Cfr. Appendix A.)

We can now use (10.77) and (10.78) to rewrite (10.74) as

$$N^* = \left[ \frac{\beta + F(t)}{\alpha + N} \right] \frac{1}{\Gamma(\alpha + N)} {\left\{ S_{\text{o}}(\alpha + N, \frac{N}{1+\beta}) - F(t) \right\}} +$$

$$\frac{F(t)}{\Gamma(\alpha + N)} \{ N_{\text{o}}(\alpha + N, \frac{N}{1+\beta}) \}$$  \hspace{1cm} (10.79)$$

$$= \left[ \frac{\beta + F(t)}{\alpha + N} \right] \frac{1}{\Gamma(\alpha + N)} \frac{\Gamma(\alpha + N)}{N} F(t) \frac{\alpha + N}{\alpha + N} +$$

$$\frac{F(t)}{\Gamma(\alpha + N)} \left[ 1 - \frac{\alpha + N}{1+\beta} \right]$$
\[
\frac{F(t)}{N} \left( \frac{(\alpha + N)}{1 + \beta} \right)^{\frac{\Gamma(\alpha + N)}{\alpha + N + 1}} \text{ (10.83)}
\]

\[
N^* = N + \frac{\alpha + N}{\beta + F(t)} F(t) \text{ (10.81)}
\]

This result can also be found by another approach. By using the general results for conditional probability, plus (10.59) and (10.63), we get

\[
g(N | t, \theta) = \frac{f(t, N_0 | \theta)}{f(t | \theta)} \text{ (10.82)}
\]

\[
g(N | t, \theta) = \frac{[\theta F(t)]^{N_0 - N}}{N!} \cdot e^{-\theta F(t)} \text{ (10.83)}
\]

\[
N - N | t, \theta \sim \text{Poisson } (\theta F(t)) \text{ (10.84)}
\]

\[
E[N - N | t] = \theta F(t) \text{ (10.84-1)}
\]

\[
\theta^o
\]
\[ E(N \mid t) = N + F(t)E(\Theta \mid t) \] (10.85)

From (10.70) we get

\[ E(N \mid t) = N + \frac{\alpha + N}{\beta + F(t)} \] (10.86)

which is the same result as in (10.81).

This result can also be obtained in an intuitive manner. Consider first the result in (10.84). From (10.55) we have that \( E(N_0) = \Theta \).

(10.84–1) can now be rewritten as

\[ E(N \mid t) = N + \Theta F(t) \] (10.87)

Since \( N_0 \sim \text{Poisson}(\Theta) \) (cf. (10.44)), we have \( \Theta = E(N_0) \) and thus

\[ E_\Theta(N_0 \mid t) = N + E(N_0) F(t) \] (10.88)

For our choice of prior distribution of \( N_0 \) (10.81) follows directly from (10.88).

Another, intuitive way to approach this problem, is as follows:

Consider the situation shown in fig. 10.1. We have observed the number of errors in each period up till \( t_N \). In the period \([0, t_N]\) we have observed a total of \( N \) out of \( N_0 \) errors.
It seems reasonable to assume that a $F(t_N)$ fraction of $N_0$ is left unobserved at time $t_N$. Thus, our naive $N_0$ estimate should be

$$N^* = N + N_0 F(t_N)$$  \hspace{1cm} (10.88-1)

If $N_0$ is a random variable, our naive predictor should be

$$N^* = N + E(N) F(t_N)$$  \hspace{1cm} (10.89)
10.3.2 Parameter Values for the Gamma Distribution

It has been common practice to estimate the number of errors in a system by using a constant value for number of errors per 1000 lines of code.

Given an upper and a lower limit on no. of errors per 1000 lines, we can use the algorithm proposed in [21], pg 239 to find initial values for $\alpha$ and $\beta$.

[34] gives a median value of five errors per 1000 lines, computed from six programs.

In [35], a rate of 21 errors per 1000 lines was reported.

The best result reported is 0.04 errors per 1000 lines [36], but this result is probably exceptional.

The Jet Propulsion Laboratory achieved two errors per 1000 lines in 1983 [29].

The Computer Science Corporation has reported error rates in the range of 0.07 to 1.5 per 1000 lines in 1983 to 1986 [37].

A project done in England some years ago reported an average of 100 errors per 1000 lines [38]. This was, however, the total number of failures observed in the project, while the other numbers quoted concern failures experienced by the users only. In most cases 90% of all faults are removed before delivery. This indicates 10 errors per 1000 lines in this case, a number which is in the same ballpark as the numbers cited earlier.

From these, rather scant data, I will suggest the values
Upper Limit : 20 errors per 1000 lines

Lower Limit : 1 error per 1000 lines

If we use a 95\% confidence limit, the algorithm in [21] gives us

\[
\alpha_0 \approx 2.1 \quad (10.90)
\]

\[
\beta_0 \approx 0.24
\]

From (10.86) we then get

\[
N^* = N + \frac{21 + N}{\frac{F(t)}{N} + 0.24} \quad (10.91)
\]

The variance for \(N^*\) can be found as

\[
V(N^*|t) = \frac{\sum_{N=N^*}^{N=N^*} g(N|t)^2}{\sum_{N=N^*}^{N=N^*} g(N|t)} - [\frac{\sum_{N=N^*}^{N=N^*} g(N|t)}{N=N^*}]^2 \quad (10.94)
\]

The first term on the right hand side can be rewritten as
\[ E(N^2 | t) = \left[ \frac{\beta + F(t)}{\alpha + N} \right] \frac{1}{\Gamma(\alpha + N)} \cdot \frac{\Gamma(\alpha + N + i)}{1 + \beta} \frac{\Gamma(i + 1)}{1 + \beta} \]  

\[ \sum_{i=0}^{\infty} \frac{F(t)}{\left( N^2 + 2iN + i^2 \right) \left( \frac{1}{1 + \beta} \right) i \frac{\Gamma(\alpha + N + i)}{\Gamma(i + 1)}} \]  

(10.95)

If we introduce

\[ S_2(k, p) = \sum_{i=0}^{\infty} i^2 p \frac{\Gamma(k+i)}{\Gamma(i+1)} \]  

(10.96)

we can now write \( E(N^2 | t) \) as

\[ E(N^2 | t) = (1 - p) \left( \frac{1}{\Gamma(K)} \right) \left\{ N^2 S(K, p) + 2NS_1(K, p) + S_2(K, p) \right\} \]  

(10.97)

where we have used \( K = \alpha + N \), \( P = \frac{N}{1 + \beta} \). Inserting for \( S_0 \), \( S_1 \) and \( S_2 \) from appendix A, we get

\[ E(N^2 | t) = (N + \frac{PK}{1 - P})^2 + \frac{PK}{(1 - P)^2} \]  

(10.98)

By using the same substitutions as we used to obtain (10.98), we can transform (10.91) to

\[ E(N | t) = N + \frac{PK}{1 - P} \]  

(10.99)
\[ V(N^*|t) = (N + \frac{PK}{1-P})^2 + \frac{PK}{(1-P)^2} - (N + \frac{PK}{1-P})^2 \quad (10.100) \]

\[ V(N^*|t) = \frac{PK}{(1-P)^2} \quad (10.101) \]

Substituting for \( P \) and \( K \) we set

\[ V(N^*|t) = \frac{(\alpha+N)(1+\beta)}{[\beta + F(t)]^2} \quad (10.102) \]

\[ F(t) \]

By using the general result

\[ V(X) = E[V(X|Y)] + V[E(X|Y)] \quad (10.103) \]

we can also obtain (10.102) in a more straightforward manner. From (10.84) we get

\[ V(N^*-N|t,\theta) = \theta F(t) \quad (10.104) \]

\[ V(N^*|t,\theta) = \theta F(t) \quad (10.105) \]

\[ V(N^*|t) = E(\theta|t) F(t) + V(\theta|t) F(t)^2 \quad (10.106) \]

From (10.70) we get
\[ V(N_*|t) = \frac{\alpha+N}{\beta+F(t)} - \frac{\alpha+N}{F(t)} + \frac{\alpha+N}{[\beta+F(t)]^2} \]  

(10.107) can be restructured to yield (10.102).

It is comforting to note that

\[
\lim_{N \rightarrow N_0} F(t) = 1
\]

\[
\lim_{N \rightarrow N_0} F(t) = 0
\]

Thus

\[
\lim_{N \rightarrow N_0} V(N_*|t) = \frac{\alpha+N}{1+\beta} \lim_{N \rightarrow N_0} F(t) = 0
\]

(10.109)

If we insert our prior values \( \alpha_0 \) and \( \beta_0 \) from (10.90) we get

\[
V(N_*|t) = \frac{1.24(21+N)}{[F(t)+0.24]^2}
\]

(10.110)

For the non-informative prior we will use \( \alpha, \beta = 0 \), and thus
\[ N^\wedge = \frac{N}{\text{F(t)}} \]  

(10.111)

\[ V(N|t) = \frac{N}{\text{F(t)}} \text{F}(t) \]  

(10.112)

(10.111) is the same as the MLE estimate.
11. A Complete Bayesian Model

This chapter treats the full model, that is, a Bayesian model with a conditional distribution for the failure times and prior distributions for the failure time distribution parameters (θ) and N₀.

We show that even for the Jelinski-Moranda model, this cannot be solved in an analytic manner when N₀ ~ Poisson(.) and λ~Gamma (., .).

There are two possible ways to obtain approximate analytic solutions:

- by expansions into infinite sums
- by using approximations to F(.)

These two methods are used to solve the complete Bayesian model for exponential, log Normal and Weibull distributions.

A third possibility is to use mode estimation. This gives us the advantage of being free to choose priors that fit our knowledge of the parameters, without having to consider the problem of getting integratable functions.

The price we have to pay is that we get an approximate solution, which often can be found only by numerical maximisation. This is done for Weibull and gamma distributed failure times. Judged by their performance, these approximations are quite good.
Our two starting equations are

\[
L(t|N, \theta) = \frac{N!}{(N-N)!} \prod_{i=1}^{k} f(t_i | \theta) F(t_i | \theta) \tag{11.1}
\]

\[
L_i(t|t_k, \theta) = \frac{\prod_{i=1}^{k} f(t_i | \theta)}{F(t_k | \theta)^N} \tag{11.2}
\]

The equation (11.1) is the standard joint probability of \( \{t\} \) when the set is ordered and we have observed the failure times \( t_1, t_2, \ldots, t_N \) and know that there are \( N-N \) items left at \( t_k \), which have not yet failed. Equation (11.2) is the likelihood for \( \{t\} \) given that \( t_i < t_k \) for all \( i \). This is the conditional likelihood used by Sanathanan. Cfr. chapter 7.5.

We will first discuss some mathematical problems, and continue with a solution for the Jelinski Moranda model before trying to attack more realistic models, especially the Log Normal version of the Load Dependency model.
11.1 Some Mathematical Problems

For all the models used in this chapter, we will soon discover that there exist no simple solution. Even for the simple Jelinski-Moranda model we get into integrals for which we have found no closed form solution.

The likelihoods have the following forms

1) Total likelihood plus a Poisson prior for \( N_0 \)

\[
\begin{align*}
    t | \theta & \sim \theta e^{-\theta F(t)} \\
    \prod_{i=1}^{N} f(t_i)
\end{align*}
\]  

(11.2-1)

2) The conditional likelihood

\[
\begin{align*}
    t | \theta & \sim F(t) e^{-N_0} \\
    \prod_{i=1}^{N} f(t_i)
\end{align*}
\]  

(11.2-2)

It is especially the two terms

\[
    F(t, \theta)^{-N_0} e^{-\theta F(t)}
\]

which cause trouble. This leaves us with three ways out.

- Abandon analytic computation all together and resort to pure numeric methods.

- Write the two troublesome terms as sums
- find some more or less reasonable approximations

11.1.1 **Expansions into Sums**

This technique is especially attractive for the J-M model, where the trouble term can be rewritten as follows

\[ F(t, \lambda) = (1 - e^{-\lambda t})^{-N} \]  \hspace{1cm} (11.2-3)

\[ F(t, \lambda) = \left\{ \sum_{i=0}^{N} e^{-\lambda t_i} \right\} \]  \hspace{1cm} (11.2-4)

\[ F(t, \lambda) = \sum_{i=0}^{N} C_i e^{-i\lambda t_k} \]  \hspace{1cm} (11.2-5)

\( C_i \) are defined by the recursive formula

\[ C_0 = 1 \]  \hspace{1cm} (11.2-6)

\[ C_i = \frac{1}{i} \sum_{k=1}^{i} (kN-i+k)C_{i-k}, \ i > 0 \]

(cfr. [43], 0.314)
11.1.2 Lindleys Approximation

Lindley has proved the following general result [47]:

Use the following notation

1) $U$ is any function of the parameters $\theta$, and can be differentiated in all its parameters.

2) $g(.)$ is the joint prior distribution of the parameters $\theta$.

3) $t$ is our observations, used to compute the likelihood for a given $\theta$.

4) $\hat{\theta}$ is the MLE for $\theta$.

Furthermore:

$$U_{ij} = \frac{\delta^2}{\delta \theta_i \delta \theta_j} U(\theta) \bigg|_{\theta=\hat{\theta}}$$

$$Q_i = \frac{\delta}{\delta \theta_j} \ln g(\theta) \bigg|_{\theta=\hat{\theta}} \quad (11.2-7)$$

$$\{\sigma_{ij}\} = \left\{-\frac{\delta^2}{\delta \theta_i \delta \theta_j} l(\theta)\right\}^{-1} \bigg|_{\theta=\hat{\theta}}$$

Then, for $|\theta| = 2$, 

\[ E[U(\Theta|t)] \approx U(\hat{\Theta}) + \frac{1}{2} \sum_{i=1}^{2} \sum_{j=1}^{2} (U_{ij} + 2U_{ij} \sigma) \]  

(11.2-8)

If we want to find \( \hat{\Theta}_k | t \), we have that

\[ U(\hat{\Theta}_k) = \hat{\Theta}_k, \quad \frac{\delta}{\delta \Theta_k} U(\Theta) = \delta_k \]  

(11.2-9)

\[ \frac{\delta^2}{\delta \Theta_j \delta \Theta_k} U(\Theta) = 0 \]

and thus:

\[ \hat{\Theta}_k | t \approx \hat{\Theta}_k + \sum_{j=1}^{2} \Theta_j \sigma_k \]  

(11.2-10)

An example of the use of this approximation is shown in connection with the Jelinski-Moranda model.

11.1.3 The Mode Approximation

The Bayes estimator under a square loss function is \( E(\Theta|t) \). The main problem with this estimator is that one must perform an integration. Unfortunately, only a small portion of all possible functions can be integrated in an analytic way, and even if we consider definite integrals, the number of known results is meager compared to all known functions.
For functions in one variable, numerical integration is a possible solution, but even this gets problematic when one deals with multi-parameter functions, like the incomplete gamma function raised to some power larger than 1.

For these reasons, the following approximation is suggested:

\[ t, \theta \sim L(t|\theta)g(\theta) \quad \tag{11.2-11} \]

\[ l_1(\theta, t) = \ln g(t, \theta) + l(t|\theta) \quad \tag{11.2-12} \]

\[ \theta^* = \left\{ \theta \mid \frac{\delta}{\delta \theta} l_1(\theta, t) = 0 \right\} \quad \tag{11.2-13} \]

We see directly from (11.2-12) and (11.2-13) that

\[ \theta^* = \hat{\theta} + \text{correction term} \quad \tag{11.2-14} \]

where the correction term is determined by the prior distribution(s).
11.2 The Jelinski-Moranda Model

11.2.1 The Complete Likelihood

This problem has been treated by Singpurwalla and Langberg in [45]. The authors did not, however, make any attempts to find the posterior distributions of neither \( \lambda \) nor \( N_0 \).

If we replace their prior distribution for \( N_0 \) (\( P(N_0 = q) \)) with the Poisson distribution, and let \( \lambda \) be apriori gamma\( (\alpha, \beta) \) distributed, we get the joint distribution in (11.3). In order to get a more compact notation we have used

\[
NT = \sum_{i=1}^{N} t_i
\]

(11.2-15)

Note that \( t_k = t \) where \( t \) is the censoring time.

\[
t, \lambda, N \sim \frac{\Theta}{(N - N)!} N_0^{\Theta - \Theta N + \alpha - 1} \frac{-\lambda^{N+\alpha-1}}{\beta + NT + t(N - N)} e^{\lambda \beta} e^{-\lambda} k \quad \text{for } \Theta = 0
\]

(11.3)

It is now straightforward to show that

\[
t, \lambda \sim \frac{\Theta}{\Gamma(\alpha)} e^{\beta} \frac{\Theta - \alpha}{\alpha} \frac{N^{\alpha - 1}}{N^{\alpha - 1}} \exp[-\lambda(NT + \beta) + \Theta e^{-\lambda t}]
\]

(11.4)
Neither of these two equations can be integrated to yield a closed form expression for the distribution of \( t \). In order to get the show back on the road, we will use the following asymptotic approach.

\[
\begin{align*}
\frac{\Gamma(\alpha+N)}{\Gamma(\alpha) [NT+\beta+t(N-N)]^{N+\alpha}} & = \frac{\beta^{N+\alpha}}{k_0^{N+\alpha}} \\
\frac{\beta + t(N-N)}{N+\alpha} & = \frac{(NT)}{1 + \frac{\beta + t(N-N)}{k_0}}
\end{align*}
\]

For large \( N \) (\( N \gg 1 \)) we can use the following approximation

\[
\frac{\Gamma(\alpha+N)}{\Gamma(\alpha) [NT+\beta+t(N-N)]^{N+\alpha}} \approx \frac{\beta^{N+\alpha}}{k_0^{N+\alpha}} \exp\left(\frac{1}{T} \frac{\beta + t(N-N)}{k_0}\right)
\]

We will introduce the notation

\[ A \equiv \text{approximately distributed} \]

When we insert (11.7) into (11.5) we get, by a simple summation

\[
A \frac{\Gamma(N+\alpha)}{\Gamma(\alpha)} \exp\left(\frac{-\beta}{T} + \frac{t k_0}{T}\right) \]

From (11.4) and (11.5) we can now obtain the posterior distributions of \( \lambda \) and \( N_0 \).
\[ N \sim \text{Poisson}(\Theta e^k \, t) \quad (11.9) \]

\[ \lambda \sim \Gamma(N+\alpha, TN) h(\lambda) \quad (11.10) \]

where \( h \) is a correction term, as defined by (11.11).

\[ h(\lambda) = \exp\{\beta(\lambda) - \lambda\} \frac{\lambda^t}{t} \exp\left[ -\lambda \left( 1 - e^{-k \frac{t}{T}} \right) \right] \quad (11.11) \]

Note that we have

\[ \lim_{T \to \infty} \frac{1}{T} = \lambda \Rightarrow \lim_{k \to \infty} h = 1 \quad (11.12) \]

Another way to obtain an approximate result is to expand the last term in (11.4) so that we get

\[ t, \lambda = \frac{N - \Theta \alpha}{\Gamma(\alpha)} \frac{\theta e^{-\beta}}{\lambda} \sum_{i=0}^{\infty} \frac{(-\lambda (\text{NT} + \beta) t)^i}{i! \Gamma(\alpha) \lambda^{i+\alpha-1}} \quad (11.15) \]

\[ t = \frac{N - \Theta \alpha}{\Gamma(\alpha)} \sum_{i=0}^{\infty} \frac{(-\lambda (\text{NT} + \beta + it))^i}{i! \lambda^{i+\alpha-1}} \quad (11.17) \]
\[ t = \frac{\sum_{\theta e^\beta}^{N-\theta \alpha} \Gamma(\alpha+N)^i}{\Gamma(\alpha) \prod_{i=0}^{N+\alpha} (NT+\beta+it)^k} \quad (11.18) \]

It is reasonable to assume that when \( N > 10 \) say, we have the relation

\[ NT + \beta >> t \quad (11.19) \]

Note that this is a weaker assumption than the one used in (11.7).

By combining (11.18), (11.19) and the approximation

\[ \frac{1}{N^{1+\epsilon}} \approx e^{-N\epsilon} \quad (11.20) \]

we get

\[ t = \frac{\sum_{\theta e^\beta}^{N-\theta \alpha} \Gamma(\alpha+N)^i}{\Gamma(\alpha) \prod_{i=0}^{N+\alpha} (NT+\beta)^k} \exp(i\frac{i\epsilon}{NT+\beta}) \quad (11.21) \]

\[ t = \frac{\sum_{\theta e^\beta}^{N-\theta \alpha} \Gamma(\alpha+N)^i}{\Gamma(\alpha) \prod_{i=0}^{N+\alpha} (NT+\beta)^k} \exp(-t\frac{N+\alpha}{NT+\beta}) \quad (11.22) \]
It is now straightforward to obtain the posterior distributions for $\lambda$ and $N_0$.

$$
\lambda | t \sim \text{Gamma}(N+\alpha, NT+\beta) \quad g_1(\lambda, \theta) \tag{11.23}
$$

$$
N_0 - N | t \sim \text{Poisson}(\theta \exp[-t \frac{N+\alpha}{k NT+\beta}]) \tag{11.24}
$$

In order to obtain (11.24) we have used (11.20) on the left hand term in (11.6). The correction term in (11.23) is

$$
g_1(\lambda, \theta) = \exp[\theta(e^{-\lambda t} - e^{-\frac{N+\alpha}{k NT+\beta}} t)] \tag{11.25}
$$

When we assume $NT \gg \alpha, \beta$, (11.23) and (11.24) will give the same results as (11.9) and (11.10).

Note that

$$
A \lambda = \frac{1}{T} \implies \begin{cases} 
  g_1(\lambda, \theta) = 1, \\
  N_0 - N | t \sim \text{Poisson}(\theta e^{-t k / T})
\end{cases} \tag{11.26}
$$
The Bayes estimates for \( \lambda \) and \( N \) are

\[
\begin{align*}
\hat{N} | t &\approx N + \theta e^{-\frac{N+\alpha}{NT+\beta}} t^k \\
\hat{\lambda} | t &\approx \frac{N+\alpha}{NT+\beta} \exp(2\theta e^{-\frac{N+\alpha}{NT+\beta}} t^k) 
\end{align*}
\]

(11.27)

(11.28) is obtained by setting

\[
\hat{\lambda} | t \approx \frac{(NT+\beta)^{N+\alpha}}{\Gamma(N+\alpha)} \exp(\theta e^{-\frac{N+\alpha}{NT+\beta}} t^k).
\]

\[
\begin{align*}
&\int_0^\infty \lambda^{N+\alpha-\lambda(NT+\beta)} e^{-\lambda t^k} \exp(\theta e^{-\lambda t^k}) d\lambda \\
&= \hat{\lambda} | t \\
&= A \sum_{i=0}^{\infty} \frac{\theta^i}{i!} \int_0^\infty \lambda^{N+\alpha-\lambda(NT+\beta+it^k)} e^{-\lambda t^k} d\lambda
\end{align*}
\]

where \( A \) is used to denote all of the right hand side of (11.29) outside the integral.
\[
\lambda | t = \frac{N+\alpha}{NT+\beta} \exp(\theta e^{t/k}) \sum_{i=0}^{\theta} \frac{i^{i\xi(N+\alpha)}}{i!} \tag{11.30}
\]

\[
\varepsilon = \frac{t}{k} \tag{11.30-1}
\]

where we have used the approximation (11.20). From (11.30) we can obtain (11.28) directly.

11.2.2 The Conditional Likelihood

The likelihood in (11.2) for the Jelinski-Moranda model is

\[
L_1(t|t,\lambda) = \frac{N^{N-\lambda NT}}{\lambda^{\lambda} e^{-\lambda t/NT} (1-e^{-\lambda t/k})^N} \tag{11.31}
\]

The joint density of $\lambda$ and $t$ is
\[
\lambda, t = \frac{\alpha}{\Gamma(\alpha)} \frac{\beta^N \alpha - 1}{(1 - e^{-\lambda t})^N} e^{\lambda(\beta + NT)} \frac{1}{k}
\]  

(11.32)

The integral of (11.32) with respect to \( \lambda \) does, however, only exist in closed form for \( N < 3 \). It is thus necessary to use an approximation for the last term in (11.32). If we use the approximation suggested in (11.2-5), we get

\[
\lambda, t = \frac{\alpha}{\Gamma(\alpha)} \frac{\beta^N \alpha - 1}{\sum_{i=0}^{\infty} e^{\lambda(\beta + NT + it)}}
\]  

(11.33)

(11.33) can be written as a sum of gamma distributions, each multiplied by a weight

\[
\lambda, t = \frac{\Gamma(\alpha + N)}{\Gamma(\alpha)} \beta^\alpha \frac{C_i}{\gamma} \frac{f(\gamma, \alpha + N, B_i)}{B_i}
\]  

(11.33-1)

where we have used

\[
B_i = \beta + NT + it
\]  

(11.33-2)

It is now straightforward to obtain the distributions for \( t \)
and $\lambda | t$.

$$t = \beta \frac{\alpha \Gamma (N+\alpha)}{\Gamma (\alpha)} \sum_{i=0}^{C} \frac{i^{N+\alpha}}{(\beta+NT+it)^{N+\alpha}}$$  \hspace{1cm} (11.34)

$$\lambda | t = f \gamma (\alpha+N, \beta+NT)g(\lambda, t)$$  \hspace{1cm} (11.35)

where $g(\lambda, t)$ is a correction term.

$$g(\lambda, t) = \frac{C}{\Sigma \text{ i } e^{-i\lambda t}} k$$  \hspace{1cm} (11.36)

The Bayes estimate can now be obtained directly from (11.35), (11.36)

$$\Lambda | t = \frac{N+\alpha \Sigma \text{i } C (1+i\kappa)^{-(N+\alpha+1)}}{\beta+NT \Sigma \text{i } C (1+i\kappa)^{-(N+\alpha)}}$$  \hspace{1cm} (11.37)

where we have used the notation
Both (11.35) and (11.37) are easily evaluated on a computer. We see that \( \hat{\lambda}|t \) usually will be very close to the "traditional" Bayes estimator.

The first term in the sum is 1, so that (11.37) can be written as

\[
\hat{\lambda}|t = \frac{N+\alpha}{\beta+NT} \left\{ 1 - \frac{Nk}{(1+k)(N+\alpha)} \right\} + \text{[correction terms]} \quad (11.39)
\]

It is also possible to use Lindely's approximation (11.2-8). Since we now only have one parameter, we have

\[
\hat{\lambda}|t \approx \hat{\lambda} - \frac{\delta}{\delta \lambda} \ln \nu(\hat{\lambda}) \frac{1}{\frac{\delta^2}{\delta \lambda^2} l(\hat{\lambda}, t)} \quad (11.40)
\]

\( \nu(.) \) is the prior distribution of \( \lambda \).

\[
\hat{\lambda}|t \approx \hat{\lambda} + \left( \frac{\alpha-1}{\lambda} - \beta \right) \frac{1}{N \hat{\lambda}^2 - \frac{t^2}{k} \frac{Ne^{-\lambda t k}}{k (1-e^{-\lambda t})^2}} \quad (11.41)
\]
From \( \frac{\delta}{\delta \lambda} l(t|\lambda) = 0 \), we can obtain

\[
\frac{t^k}{-\lambda t} = \left( \frac{1}{\lambda} - T \right)e^{\lambda t} k \\
1-e^{-\lambda t}
\] (11.41-1)

If we substitute the right side of (11.41-1) into (11.41), we get

\[
\lambda | t \approx \lambda + \frac{1}{N} \left( \frac{\alpha - 1}{\lambda} \right) \frac{\lambda^2}{1 - e^{-\lambda t}} k (1 - \lambda T) \] (11.41-2)

Let us use the following approximation

\[
e^{\lambda t k} \approx 1 + \lambda t + \frac{1}{2} \lambda^2 t^2 k
\] (11.41-3)

In this case, (11.41-1) will give us

\[
\lambda \approx \frac{1}{T} - \frac{2}{t} k
\] (11.41.4)

This estimate can be inserted into (11.41-2).

If we use the mode estimate for \( \lambda \) by maximizing the logarithm of (11.32), we get the following equation:

\[
\frac{N+\alpha-1}{\lambda^*} = \beta + NT + \frac{e^{-\lambda^* t}}{1 - e^{-\lambda^* t}} k
\] (11.41-5)

This can easily be solved numerically to give us a \( \lambda^* \) estimate. If
we assume $\lambda^* t_k \ll 1$ we get

$$\lambda^* \approx \frac{\alpha - 1}{NT + \beta} \quad (11.41-6)$$

The $N_0$ estimate follows from (7.79) so that

$$\hat{N}_0 | t = \frac{N}{1 - e^{-\lambda^* t_k}} \quad (11.42)$$
11.3 The Log Normal Distribution

In order not to clutter the presentation more than necessary, I have worked with the Normal distribution throughout this chapter. The only change needed for the Log Normal distribution is to replace all $t$ and $t_k$ with $\ln t$ and $\ln t_k$, respectively.

11.3.1 The Complete Likelihood

If we use the same assumptions as we used in chapter 10.2, (10.39)-(10.44), we can write

$$\begin{align*}
\ln t, \ln m, \ln r, N_0 & \sim (2\pi)^{-\frac{N+1}{2}} \frac{\beta^\alpha}{\Gamma(\alpha)} \frac{N_0}{(N-N)!} e^{-\theta} \sqrt{r} . \\
& \quad \alpha + \frac{N+1}{2} - 1
\end{align*}$$

$$\ln r \quad \exp\left(-\frac{r}{2} [B+2\beta+(m-A)^2(k+\tau)]\right)$$

$$Q[(t-m)\sqrt{r}]^N \quad \ln t_k$$

(11.44)

It is straightforward to obtain the distribution for $(t,m,r)$ as
\[ t, m, r = (2\pi)^{-\frac{1}{2}} e^{-\frac{\theta}{\gamma}} \frac{\beta}{\Gamma(\alpha)} N \exp(-r) r^{-\frac{\alpha+1}{2}}. \]

\[ Q[(t - m)/\gamma] e^{-k} \quad (11.45) \]

From here on, however, the mathematics gets rather hairy and even a gross approximation like

\[ Q(X) \approx \frac{\Phi(X)}{X} \quad (11.46) \]

will only lead to unwieldy results of dubious value. Fortunately, an approximation cited in [46] is both simple and gives good results.

\[ \Phi(X) \approx \frac{1}{1 + e^{-\gamma X}}, \quad \gamma = \frac{\theta}{\pi}^{1/2} \quad (11.47) \]

The error in this approximation does never exceed 2%.

Let us compress (11.45) by the introduction of
\[ C = (2\pi)^{-\frac{N+1}{2}} e^{-\Theta} \frac{\beta^\alpha}{\Gamma(\alpha)} \]

\[ \nu = \alpha + \frac{N+1}{2} \]

\[ E = \frac{1}{2} B + \beta \]

\[ H = \frac{1}{2} (N+\tau) \]

(11.48)

We can thus rewrite (11.45) as

\[ t, m, \tau \sim C e^{-r[E+(m-A)^2 H]} r^{\nu-1} e^{\Theta Q[\sqrt{r}(t-m)]} \]

(11.49)

By using the approximation (11.47), the last term can be written as

\[ \exp[\Theta \frac{e^{-\gamma \sqrt{r}(t_k-m)}}{1+e^{-\gamma \sqrt{r}(t_k-m)}}] \]

(11.50)

Unfortunately, it is still not possible to integrate (11.49) with respect to \( r \) or \( m \), except by numerical methods. We are thus forced to try to take yet another step.

In order to avoid an integral of the form
\[
I = \int_0^\infty e^{-r(.)} r^{-1} e^{-\sqrt{r}} dr
\]

we are forced towards extremely large \( \sqrt{r(t_k - m)} \) values. Typical values from the data sets included in the case studies (chapter 8) will give

\[
\theta \frac{e^{-\gamma \sqrt{r(t_k - m)}}}{1 + e^{-\gamma \sqrt{r(t_k - m)}}} < 0.1 \text{ for } t > 150
\]

One of the problems is the fact that \( \theta \) must be large, since \( E(N_o) = \theta \) indicates that \( \theta \) must typical be larger than 100 say.

For small \( t_k \) we have \( Q(\sqrt{r(t_n - m)}) \approx 1 \) and we are back to the problem treated in chapter 10.2.
11.3.2 The Conditional Likelihood

If we use the approximation (11.47), the joint distribution can be written

\[
- \left( \frac{N+1}{2} \right)^{\alpha} \beta^\alpha \left( \frac{r}{\Gamma(\alpha)} \right)^{\alpha+\frac{N+1}{2}} \cdot t, m, r = (2\pi)^{-\frac{N+1}{2}} \sqrt{\pi} \cdot \gamma^\frac{N+1}{2} - 1
\]

\[
- \frac{r}{2} [B+2\beta+(m-A)^2(k+\tau)] e^{-\gamma\sqrt{\pi}(t-m)N} [1+e^{-\gamma\sqrt{\pi}(t-m)N}] (11.53)
\]

We will again compress the expression by using the symbols in (11.48), except that

\[
C = (2\pi)^{-\frac{N+1}{2}} \beta \left( \frac{\alpha}{\Gamma(\alpha)} \right)^{\alpha+\frac{N+1}{2}} (11.54)
\]

Thus

\[
t, m, r = C \cdot r^{-\frac{N+1}{2}} \cdot \gamma^\frac{N+1}{2} \cdot [1+e^{-\gamma\sqrt{\pi}(t-m)N}] (11.55)
\]

This can be rearranged to yield.
We will now substitute \( u = \sqrt{r} \), and thus arrive at

\[
2^{1/2} \sum_{j=0}^{N} N \frac{\gamma^2 j^2}{4H} = 2^{v-2} - \gamma j (t - A) u \\
\]

This can be solved by using the standard integral.

\[
\int_{0}^{\infty} e^{-\beta x^2 - kx} \, dx = (2\beta)^{-v/2} \Gamma(v) e^{k^2/8\beta} D \left( \frac{k}{\sqrt{2\beta}} \right) \quad (11.59)
\]

which is found in [43], 3.462-1. \( D (.) \) is the parabolic cylinder function. A more convenient notation for \( D (.) \) is introduced in [12], which defines

\[
U(a,x) = D \left( \frac{x}{a} \right) \quad (11.60)
\]
By using (11.59) and (11.60) on (11.58) we get

$$
t = C \left( \frac{1}{H} \right)^{1/2} \frac{\Gamma(2\nu-1)}{\nu} \sum_{j=0}^{N-1} 2(\nu)e^{\frac{1}{4H} \left( \psi \sqrt{\frac{t-A}{2}} \right)} + \frac{k}{8E} \left( \frac{t-A}{k} \right)^{3/2} \psi \sqrt{\frac{t-A}{2}} \right)
$$

(11.61)

Let us introduce the two following notations

$$
X = \gamma \left( \frac{1}{4H} + \frac{k}{8E} \right) \psi \sqrt{\frac{t-A}{2}}
$$

(11.62)

$$
Y = \gamma \frac{k}{\sqrt{2E}} \psi \sqrt{\frac{t-A}{2}}
$$

We then have

$$
t = \frac{1}{H} \frac{\Gamma(2\nu-1)}{\nu} \sum_{j=0}^{N-1} 2(\nu)e^{\frac{1}{4H} \left( \psi \sqrt{\frac{t-A}{2}} \right)} \psi \sqrt{\frac{t-A}{2}} \right)
$$

(11.63)
\[ C = \left( \begin{array}{c} N_j^{2/2} \end{array} \right) \]  

\[ t = C \left( \frac{\Gamma(2v-1)}{(2E)^{\nu-1/2}} \right) \sum_{j=0}^{N} C_j U(2v-\frac{3j}{2}, j) \]  

\[ (11.63-1) \]

\[ (11.64) \]

We will now use the two identities:

\[ \Gamma(2v-1) = \frac{\Gamma(2v)}{2v-1} \]  

\[ (11.65) \]

\[ \Gamma(2v) = \frac{2^{2v-1}}{\sqrt{\pi}} \frac{\Gamma(v) \Gamma(v+1)}{2} \]

(cfr. [43], 8.335-1)

From (11.65), (11.64) and (11.57) it is now straightforward to obtain

\[ r |t = \frac{\Gamma(\nu-1/2)}{E \Gamma(\nu-\frac{1}{2})} e^{\frac{1}{rE \sqrt{2\pi}}} \frac{1}{\Gamma(v-\frac{1}{2})} \sum_{j=0}^{N} C_j U(\ldots) \]

\[ (11.66) \]
This can also be written on the form

\[ r|t = f \frac{1}{\text{gamma}} (v^{-\gamma}, \mu, \sigma^2_k) \cdot h(t_k, r) \]  \hspace{1cm} (11.67)

This the same result as in chapter 10.2, except for the correction function \( h(t_k, r) \).

Within a Bayesian framework, however, we are mainly interested in \( E(r^k | t) \).

Thus

\[ E(r^k | t) = \frac{\Gamma(v^{-\gamma}) \Gamma(v)}{2 \sqrt{\pi} E^2} \cdot \frac{1}{\Sigma C U(\ldots)} \]

\[ \sum_{j=0}^{N} (\gamma^2/4H)^j/\Gamma(v^{-\gamma}/2+k)^{-1} \cdot -rE^{-\gamma}j(t-A)^{\gamma} \int_0^\infty e^{-k \sqrt{r}} \, dr \]  \hspace{1cm} (11.68)

By substituting \( u^2 \) for \( r \) and using the result cited in (11.59)-(11.60), we get

\[ E(r^k | t) = \frac{1}{(2E)^k} \frac{\Gamma(2v+2k)}{\Gamma(2v+2k-1)} \cdot \frac{\Sigma C U(2v+2k-1, jY)}{2} \]

\[ \cdot \frac{\Gamma(2v)^3}{\Sigma C U(2v-1, jY)} \]  \hspace{1cm} (11.69)
It then follows directly that

$$\hat{r}|t = E(r|t)$$

(11.70)

$$V(r|t) = E(r^2|t) - E^2(r|t)$$

These two values can be easily found once we have a routine to compute $U(a,x)$. Furthermore, the distribution of $m|t,r$ can now be found from (11.56) and (11.57).

Let us introduce the function $G(j,r)$, defined as

$$G(j,r) = \left( \begin{array}{c}
N \\
1
\end{array} \right) e^{-\gamma j\sqrt{r}(t-A)+\frac{\gamma^2 j^2}{4H}}$$

(11.71)

Furthermore, define $p(j,r)$ as

$$p(j,r) = A + \frac{\gamma j}{2H/r}$$

(11.72)

The conditional distribution of $m|t,r$ is then

$$m|t,r \sim \left( \begin{array}{c}
\frac{1}{\pi} \\
\pi
\end{array} \right) \frac{1}{\Sigma G(j,r)} e^{-rH[m-p(j,r)]^2}$$

(11.73)

This gives us directly that
\[ E(m \mid t, r) = \frac{rH^{1/2}}{\pi} \frac{1}{\Sigma G(j, r)} \Sigma G(j, r) e^{-rH^p(j, r)^2}. \]

\[ \int_{-\infty}^{\infty} k - rHm^2 + 2rH^p(j, r)m \, dm \]

(11.74)

We now need the standard integral 3.462-2 in [43].

\[ \int_{-\infty}^{\infty} e^{-\beta x^2 + 2\kappa x} \, dx = \frac{1}{\sqrt{2\pi}} \left( \frac{\delta}{\beta} \right)^{\frac{1}{2}} (\kappa e^{\kappa^2/\beta})^{\frac{\delta}{\beta}} \]

(11.75)

which holds for \( \nu \in \mathbb{N}_1 \). (11.74) now gives us that

\[ E(m \mid t, r) = \frac{\Sigma G(j, r) e^{-rH^p(j, r)^2} (\delta^{k-1}) (\kappa e^{\kappa^2/\beta})}{2^{-k-1} rH \Sigma G(j, r)} \]

(11.76)

We are mainly interested in (11.76) for \( k = 1, 2 \).

Thus

\[ E(m \mid t, r) = \frac{\Sigma G(j, r) \rho(j, r)}{\Sigma G(j, r)} \]

(11.77)

When we insert \( \rho(j, r) \) from (11.72) we get
\[ E(m|t,r) = A + \frac{\gamma}{2H \sqrt{r}} \frac{\sum_{j} G(j,r)}{\sum_{j} G(j,r)} \]  

(11.77-1)

Note that A is the Bayesian estimate for \( m|t,r \) when we assume a fixed \( N \) and complete (nonconditional) likelihood.

\[ E(m^2|t,r) = \frac{\sum_{j} G(j,r)[1+2rHp(j,r)^2]}{\sum_{j} G(j,r)} \]  

(11.78)

or

\[ E(m^2|t,r) = 1 + 2rH \frac{\sum_{j} G(j,r)p(j,r)^2}{\sum_{j} G(j,r)} \]  

(11.79)

When we again insert \( p \) from (11.77), we get

\[ E(m^2|t,r) = 1 + 2A^2rH \frac{\gamma^2 \sum_{j} G(j,r)^2}{2H \sum_{j} G(j,r)} \]

\[ + \gamma A \sqrt{r} \frac{\sum_{j} G(j,r)}{\sum_{j} G(j,r)} \]  

(11.79-1)

The Bayes estimate for \( m|t,r \) and its variation can be obtained from (11.77-1), (11.79-1). As for the \( r \) estimate, a computer program will be necessary.
11.4 The Weibull Distribution

The complete likelihood is just as unwieldy in this case as it is for the log normal distribution. We will thus only treat the conditional likelihood.

11.4.1 The Conditional Likelihood

We will follow Martz and Waller [21] and assume that

\[ \lambda \sim \text{Gamma}(u,v) \]  
\[ \alpha \sim U(a,b) \]  

(11.98)

Note that Soland has proved that no continoue conjugate joint prior exists for \((\lambda, \alpha)\) [49].

The joint distribution for \((t, \lambda, \alpha)\) is

\[
t, \lambda, \alpha \sim \frac{N^u -N(\alpha-1)}{(b-a)\Gamma(u)} \frac{e^{-\lambda t}}{\lambda^{\alpha}} \frac{1}{N+u-1 -\lambda \alpha} \frac{1}{\Gamma(\alpha)} \text{ for } \alpha \in [a,b] \]  

(11.99)
By using the expression (11.2-5) and replacing $t$ by $t^\alpha_k$, we get

$$t, \lambda, \alpha \sim \frac{N-u}{v} \frac{N(\alpha-1)}{(b-a)\Gamma(u)} \lambda^{N+u-1} e^{-\lambda t^\alpha_k} + v + j t^\alpha_i \quad (11.100)$$

It is now easy to show that:

$$t, \alpha \sim \frac{u}{v} \frac{N-\alpha}{\Gamma(N+u)} \frac{\Gamma(N+u)}{(b-a)\Gamma(u)} \sum_{j=0}^{\infty} \frac{C_j}{t^\alpha_k} \frac{N+u}{v+\lambda t^\alpha_i} \quad (11.101)$$

Further more

$$\lambda | t, \alpha = \text{Gamma}(u+N, v+\lambda t^\alpha_i) g(\lambda, t^\alpha_k) \quad (11.102)$$

$$g(\lambda, t^\alpha_k) = \frac{\sum_{j=0}^{\infty} C_j e^{-\lambda t^\alpha_k}}{(1+j \frac{t^\alpha_k}{v+\lambda t^\alpha_i} - (N+u)) \quad (11.103)}$$

Let us introduce the variable $\kappa$, defined as
\[ \kappa(\alpha) = \frac{\alpha^k}{\nu + \Sigma t_i^\alpha} \]  

(11.104)

We can then write

\[ E(\lambda^k | t, \alpha) = \frac{\Gamma(N+u+k)}{(\nu + \Sigma t_i^\alpha)^k \Gamma(N+u)} \frac{\Gamma_{[1+j\kappa(\alpha)]}^{-(N+u-k)}}{\Gamma_{[1+j\kappa(\alpha)]}^{-(N+u)}} \]  

(11.105)

It is now straightforward to obtain the Bayes estimate and variance for \( \lambda \) given \( t, \alpha \).

\[ \lambda^k | t, \alpha = \frac{\kappa(\alpha)}{(N+u)} \frac{\Gamma_{[1+j\kappa(\alpha)]}^{-(N+u-k)}}{\Gamma_{[1+j\kappa(\alpha)]}^{-(N+u)}} \]  

(11.105-1)

Note that for \( \alpha = 1 \), we are back to the Jelinski-Moranda model and the result shown in (11.37).

From (11.101), we can get the distribution of \( t \).

\[ t = \frac{u \Gamma(N+u)}{(b-a) \Gamma(u)} \sum_{j=0}^{\alpha} \frac{b^\alpha T_a^{(\alpha-1)N}}{(v + \Sigma t_i^\alpha)^{\alpha N + u_i + j t_i^\alpha}} da \]  

(11.106)
This integral is, however, not analytically solvable. If we replace
the integral by \( J_0(t, j) \) we have

\[
\begin{align*}
t & = \frac{v}{b - a} \cdot \frac{\Gamma(N + u)}{\Gamma(u)} \sum_{j=0}^{\infty} c \sum_{k=0}^{m} J(t, j) \\
\end{align*}
\]

From (11.101) we now get

\[
\alpha |t| = \frac{N - \frac{N(\alpha - 1)}{\alpha T}}{\frac{\alpha N + u}{(v + \Sigma i)}} \sum_{j=0}^{\infty} \frac{C}{j \sum_{k=0}^{m} J(t, j)}^{-(N + u)} \sum_{j=0}^{\infty} \frac{1 + jk(\alpha)}{j \sum_{k=0}^{m} J(t, j)}^{-(N + u)}
\]

If we introduce the integral \( J(t, j) \), defined as

\[
J(t, j) = \int_{a}^{b} \frac{\alpha N + u}{(v + \Sigma i)} [1 + jk(\alpha)]^{-(N + u)} \, d\alpha
\]

we can get

\[
E(\alpha |t) = \frac{\sum_{j=0}^{\infty} \sum_{k=0}^{m} J(t, j)}{\sum_{j=0}^{\infty} \sum_{k=0}^{m} J(t, j)}
\]

The posterior estimate for \( \alpha \) is thus
\[ \alpha(t) = \frac{\sum_{j \in \mathcal{I}} N(t, j)}{\sum_{j \in \mathcal{I}} (t, j)} \tag{11.111} \]

As in the previous cases, a computer program is needed in order to obtain numerical values for the estimates of \(\alpha\) and \(\lambda\).

If we use the conditional likelihood and the priors in (11.98) and then use mode estimates instead of expectation, we get the following equations:

\[ N + N \ln T - \sum_{i=1}^{N} t_{i} \ln t_{i} - \ln N = 0 \tag{11.112} \]

\[ \frac{N+u-1}{\alpha} - \sum_{i=1}^{N} \ln t_{i} - u - \frac{N}{\alpha} = 0 \tag{11.113} \]

This set of equations is easily solved by the use of standard numerical methods, found for instance in the NAG library [50].

Note that this solution escapes the problem of infinite sums and is thus cheaper in terms of computational work. A modified Newton-Raphson method will converge after few iterations.

The priors suggested in (11.98) contain, however, very little information about \(\alpha\). If we are going to leave the realm of (approximate) analytic solutions, as we do in (11.112), (11.113), there is no reason why we should not use a stronger prior for \(\alpha\).
A straightforward choice would be

$$\lambda \sim \text{Gamma}(u, v) \quad (11.114)$$

$$\alpha \sim \text{Gamma}(q, r)$$

This gives us:

$$t, \alpha, \lambda \sim \frac{q^u \alpha^{N+q-1} \lambda^{N+u-1}}{\Gamma(q) \Gamma(u)} \frac{\alpha \lambda^N}{\Gamma(\alpha-1)} \exp\left(-\lambda \left(v + t \frac{\alpha}{k}\right)\right) \quad (11.115)$$

In order to avoid the costly summations of (11.105-1) and (11.111), we will use mode estimates for $\alpha$ and $\lambda$. This is equivalent to finding the maximum of the right hand side of (11.115).

A non-informative prior can be obtained by using the improper distribution

$$\lambda, \alpha \sim \frac{C}{\lambda \alpha} \quad (11.116)$$

This is equivalent to letting

$$u, q \to 0,$$

$$v, r \to 0 \quad \exists \frac{u}{v}, \frac{q}{r} = \text{constant} \quad (11.117)$$
In [59] Moore and Bilikam suggest the improper \( \lambda \) prior

\[
\lambda | \alpha \sim \frac{C}{\alpha + 1} \quad (11.118)
\]

A natural choice for a non-informative joint prior could then be

\[
\lambda, \alpha \sim \frac{C}{\alpha} \quad (11.119)
\]

Note that this distribution can be made proper by defining appropriate bounds for \( \lambda \) and \( \alpha \).

\[
\alpha \in [0, \infty), \lambda \in [0, a], \ a < 1 \quad (11.120)
\]

\[
\int_0^\infty \int_0^\infty \frac{1}{\lambda} \, d\alpha d\lambda = 1 \quad (11.121)
\]

\[
C = \frac{1}{\text{li}(a)} \quad (\text{cfr. [43], 4.211.1}) \quad (11.122)
\]

\[
E(\alpha | \lambda) = \frac{1}{\text{li}(a)} \int_0^\infty \frac{ae^{-\alpha \ln \lambda}}{\text{li}(a)} \, d\alpha \quad (11.123)
\]

\[
E(\alpha | \lambda) = \frac{1}{\text{li}(a)} \frac{1}{(\ln \lambda)^2} \quad (11.124)
\]

Note that (11.124) implies that \( \alpha \) and \( \lambda \) are negatively correlated.
From (11.119) and (11.122) we get the joint distribution

\[
\begin{align*}
t, \lambda, \alpha & = \frac{1}{T} \frac{\alpha}{\lambda} \frac{N - \alpha N - \alpha t}{\alpha e} \frac{1}{i} \frac{N(\alpha - 1)}{\ln(a)(1 - e^{-\lambda t})^\alpha N}
\end{align*}
\]  

(11.125)

From (11.125) it is again straightforward to find mode estimates for \( \lambda \) and \( \alpha \).
11.5 The Gamma Distribution

What is said about unwieldiness for the Weibull distribution, holds a fortiori for the gamma distribution. Note that the accumulated probability function does not even exist in closed form. The mode estimator seems to be the only way forward in the Bayes case.

We will use the joint prior for of $\alpha, \beta$ suggested by Miller [19], cfr. (10.15). The joint distribution of $t, \alpha$ and $\beta$ is then as shown in (10.16) except that $\Gamma(\alpha)^K$ should be replaced by $\gamma(\alpha, \lambda t, ^K)$ when we use a conditional distribution. Straight forward derivation gives the two equations

\[(v+N)\ln\hat{\beta} + \ln(pT) - n\psi(\hat{\alpha}) - \frac{N}{\gamma(\hat{\alpha}, \hat{\beta})} \frac{\delta}{\delta \alpha} \gamma(\hat{\alpha}, \hat{\beta}) = 0 \quad (11.126)\]

\[\frac{\hat{\alpha}(v+N)-1}{\hat{\beta}} - (NT+s) - \frac{Nt}{\gamma(\hat{\alpha}, \hat{\beta})} \frac{\delta}{\delta \beta} \gamma(\hat{\alpha}, \hat{\beta})^k = 0 \quad (11.127)\]

In the Weibull case we were able to obtain Bayes estimates both by taking the expectation of the posterior distribution and by finding the mode of the posterior distribution.

In the gamma case, however, the mode estimate is the only way to find the Bayes estimate with a reasonable amount of computer power.
12. **Evaluation of the Bayesian Estimators**

In this chapter we test the Bayesian models suggested in chapters 10 and 11.

The discussion is split into two parts

- Bayesian estimation with simulated data
- Bayesian estimation with real world data

The important conclusion is that the failure time distribution parameters are much more stable than number of failures per (1000) lines. As a result of this, the failure time distribution parameters are much better suited for Bayes estimation than $N_o$.

In all cases, priors will be estimated on the basis of the available data. As will be seen from the tables at the start of each chapter dealing with real world data, the failure time distribution parameters are remarkably stable.

For the simulated data sets, we will only estimate the distribution parameters in a Bayesian manner. $N_o$ will be estimated from (7.79).

For the real world data, we will discuss estimation of $N_o$ both from (7.79) and via a prior for $N_o$, by means of a two stage
approach (cfr. chapter 10.3.1).

The simulated data sets are the same as we used in chapter 9. A short description of how the results are presented can also be found there. A description on how the simulated data are generated can be found in appendix B.
12.1 Simulated Data Sets

We have compared linearization (cfr. chapter 7.1), Sananthanan's conditional likelihood (cfr. chapter 7.5) and the Bayesian estimates (cfr. chapters 10 and 11) for the Log Normal distribution, the Weibull distribution and the Gamma distribution.

12.1.1 Log Normal Distribution

We have already established that linearization is better than the conditional MLE for Log Normally distributed data (cfr. chapter 9.1.2). Thus, we will only compare the linearization method with the Bayesian estimates. The Bayesian estimate is built on the conditional likelihood of Sananthanan. Cfr. chapter 10.2.

As we can see from equations (10.40) and (10.41) the prior is defined by the parameters \( \mu, \tau, \alpha \) and \( \beta \). For the simulated data sets we have used the following values:

\[
\begin{align*}
\alpha &= 165.77, \quad \beta = 160.97 \\
\mu &= 2.20, \quad \tau = 120.59
\end{align*}
\]

(12.1)

These values were computed from the parameters estimated from linearization of all the simulated data sets.

The following results emerged from the comparison
\[ r(\text{linear}) = 2.10^2 - 6.10^3 \]
\[ r(\text{Bayes}) = 70 - 260 \]
\[ (12.1-1) \]

\[ C_{0.1}(\text{linear}) \approx 22.3\% \]
\[ (\text{Bayes}) \approx 6.2\% \]
\[ (12.2) \]

\[ C_{0.1}(\text{Bayes}) \approx 13.3\% \]
\[ (\text{Bayes}) \approx 3.3\% \]

For ten simulated data sets, we found the following:

\[ C_{0.1}(\text{Bayes}) < C_{0.1}(\text{linear}) \text{ in 9 cases} \]
\[ C_{0.1}(\text{Bayes}) > C_{0.1}(\text{linear}) \text{ in 0 cases} \]
\[ C_{0.1}(\text{Bayes}) = C_{0.1}(\text{linear}) \text{ in 1 cases} \]

A typical \( \epsilon(t) \) plot for the two estimation methods is shown in fig. 12.1.
Note that linearization does better than the Bayes method for data sets which contains number of failures for the first few periods only. This should be expected since the conditional likelihood estimates are best only in the asymptotic case.

The parameter estimates for the Bayes estimation are always better than the ones obtained by linearization. The plot shown in fig. 12.1.a below is typical.
If we use the non-informative priors suggested in (10.51) we find that

\( r(\text{non-info Bayes}) \) is always smaller than \( r(\text{MLE}) \). This is due to the damping effect of the Bayes estimators.
C_{0.1}^{(MLE)} < C_{0.1}^{(non-info Bayes)} in 7 cases

C_{0.1}^{(MLE)} > C_{0.1}^{(non-info Bayes)} in 3 cases

C_{0.1}^{(MLE)} = C_{0.1}^{(non-info Bayes)} in 0 cases

The Bayesian estimates with an empirical prior is better than the Bayes estimates with a non-informative prior. This is as expected. What is startling, however, is that the difference is rather small.

The r(non-info) is larger than r( empirical) in only six out of ten cases

C_{0.1}^{(non-info)} < C_{0.1}^{(empirical)} in 3 cases

C_{0.1}^{(non-info)} > C_{0.1}^{(empirical)} in 7 cases

C_{0.1}^{(non-info)} = C_{0.1}^{(empirical)} in 0 cases

A possible conclusion could be that the empirical prior is not particularly good.

12.1.2 Weibull Distribution

The Weibull distribution can not be linearized. We will thus use the conditional MLE as a yardstick.

During evaluation of the Bayesian estimates it turned out that the method suggested in the first part of chapter 11.4.1 had the following handicaps
- the sums converge rather slowly

- the uniform prior for \( \alpha \) contains very little information.

If we wanted a better \( \alpha \) prior, we had to turn to mode estimation instead. In this case, we used a gamma prior for \( \alpha \).

The results shown in this chapter use the priors described in equations (11.114). The parameter values used are

\[
\begin{align*}
\mu &= 43.56, \quad \nu = 1320.00 \\
\sigma &= 603.76, \quad r = 351.00
\end{align*}
\]

(12.2-1)

These values were computed from the parameters estimated by using the conditional MLE on the simulated data sets.

When we compare the conditional MLE's with the Bayes estimates, we get the following results:

\[
\begin{align*}
\text{r(MLE)} &\approx 4.10^4 - 3.10^5 \\
\text{r(Bayes)} &\approx 30 - 80
\end{align*}
\]

(12.3)

\[
\begin{align*}
\overline{C_{0.1}}(\text{MLE}) &\approx 53.86\% \\
\overline{C_{0.1}}(\text{MLE}) &\approx 12.63\%
\end{align*}
\]

(12.4)

\[
\begin{align*}
\overline{C_{0.1}}(\text{Bayes}) &\approx 55.32\% \\
\overline{C_{0.1}}(\text{Bayes}) &\approx 14.29\%
\end{align*}
\]
When we compared $C_{0.1}$ for the two estimates, we found the following:

$C_{0.1} \text{ (Bayes)} < C_{0.1} \text{ (cond. MLE)}$ in 3 cases

$C_{0.1} \text{ (Bayes)} > C_{0.1} \text{ (cond. MLE)}$ in 5 cases

$C_{0.1} \text{ (Bayes)} = C_{0.1} \text{ (cond. MLE)}$ in 12 cases

From (12.3) and (12.4) we see that although the $C_{0.1}$ values are very much the same, the reduction in $r$ makes the Bayes estimates much better.

The plot of $\epsilon$ for one of the simulated data sets is shown in fig. 12.2.
It may be a little surprising that the Bayes estimates do not lower the $C_{0.1}$ considerably. The reason is that in both cases so much data is needed to reach the 0.1-point, that the prior is "drowned" anyway.

As a non-informative prior we used both $\frac{1}{\lambda \alpha}$ (11.116) and $\lambda^{-\alpha}$ (11.119)
It turned out that

- the two priors gave roughly the same $C_{0.1}$ values
- the $\lambda^{-\alpha}$ prior gave $r$-values which were roughly 50% lower.

For these reasons, we will only consider the prior $\lambda^{-\alpha}$ in the following discussion.

$$r(\text{MLE}) \approx 4 \times 10^4 - 3 \times 10^5$$

$$r(\text{non-info Bayes}) \approx 250 - 9 \times 10^5$$

$$\frac{C_{0.1}}{\text{MLE}} = 53.86\%$$

$$\frac{\sigma_{C_{0.1}}}{\text{MLE}} = 12.63\%$$

$$\frac{C_{0.1}}{\text{non-info Bayes}} = 47.98\%$$

$$\frac{\sigma_{C_{0.1}}}{\text{non-info Bayes}} = 11.00\%$$

If we do a comparison of $C_{0.1}$ for all cases, we find that

$C_{0.1}$ (non-info Bayes) $<$ $C_{0.1}$ (MLE) in 11 cases

$C_{0.1}$ (non-info Bayes) $>$ $C_{0.1}$ (MLE) in 4 cases

$C_{0.1}$ (non-info Bayes) $=$ $C_{0.1}$ (MLE) in 5 cases
If we compare the non-informative prior with the empirical Bayes estimates, we find, rather surprisingly, that

\[ C_{0.1} \text{ (non-info Bayes)} < C_{0.1} \text{ (Bayes)} \text{ in 13 cases} \]

\[ C_{0.1} \text{ (non-info Bayes)} > C_{0.1} \text{ (Bayes)} \text{ in 4 cases} \]

\[ C_{0.1} \text{ (non-info Bayes)} = C_{0.1} \text{ (Bayes)} \text{ in 3 cases} \]

The r-values for empirical Bayes are, however, far lower, as be seen from (12.3) and (12.4-1).

When we study the estimates, we see that it is especially the first \( N_0 \) estimate (after five periods) which causes \( r \) to reach so high a value.

If this, first \( N_0 \) estimate is kept out, the non-informative Bayes estimate is as good as, or better than the MLE. This is illustrated by plotting \( \epsilon(t) \) for one of the simulated data sets, as shown in fig. 12.2-1 below.

Another thing which is brought out quite clearly is that the specified lower limit for the search space of \( \lambda \) is critical. For few observations, both the MLE and the non-informative Bayes estimates always return this value. A good estimate for the lower bound for \( \lambda \) will thus greatly improve the \( N_0 \) estimates.
12.1.3 The Gamma Distribution

Since the linearized model gives better $C_{0.1}$ and $r$ for all the 20 simulated data sets, we will compare the Bayesian estimates with the results obtained from the linearization method.
Since the Bayesian estimates for a conditional likelihood plus Millers prior gives rise to integrals that are not solvable in an analytic way, and multiple integration is rather involving when done numerically, I have used mode estimates throughout.

For the cases where the Bayes estimate and the model estimate can be compared, the error done by using the mode is always less than 1% for the gamma distribution.

When we compare the linearized model to the Bayesian, we get the following results:

\[
\begin{align*}
    r(\text{linearized}) & \approx 3.10^3 - 3.10^4 \\
    r(\text{Bayes}) & \approx 20 - 800
\end{align*}
\]

\[
\begin{align*}
    \bar{C}_{0.1}(\text{linearized}) & = 42.89\% \\
    \sigma_{\bar{C}_{0.1}}(\text{linearized}) & = 10.52\% \\
    \bar{C}_{0.1}(\text{Bayes}) & = 21.97\% \\
    \sigma_{\bar{C}_{0.1}}(\text{Bayes}) & = 14.50\%
\end{align*}
\]

When we compared the two estimation methods case by case, we got the following result:

\[
\begin{align*}
    \bar{C}_{0.1}(\text{Bayes}) < \bar{C}_{0.1}(\text{linear}) \text{ in 18 cases}
    \\
    \bar{C}_{0.1}(\text{Bayes}) > \bar{C}_{0.1}(\text{linear}) \text{ in 2 cases}
\end{align*}
\]
Note that $C_{0.1}^{\text{Bayes}}$ in (12.6) is an upper limit, since in some cases we found $\epsilon < 10\%$ after five periods, which is where we start estimation. A typical example of $\epsilon(t)$ is shown in fig. 12.3.

In some cases, $\epsilon(t)$ shows an increase before going down to a very
low level. An example is shown in fig. 12.4. This is as should be expected when we have a good prior, which is overtaken by the "not so good" data.

![Graph](image.png)

Fig. 12.4

A reasonable non-informative prior can be obtained by using \( p=1, s=0, v=0 \) and \( n=1 \). Cfr. (10.23)-(10.24). This prior is close to the one obtained by Jeffreys rule [19], which is \( \frac{1}{\beta \alpha} \).

The results obtained by using the non-informative prior are close to the results obtained by using the conditional MLE. The \( r(\text{MLE}) \) is
never more than 10\% from \( r(\text{non-info Bayes}) \). The \( C_{0.1} \) values are slightly approved:

\[
\begin{align*}
C_{0.1}^{\text{(MLE)}} & = 53.36\% \\
\sigma_{c0.1}^{\text{(MLE)}} & = 8.53\%
\end{align*}
\]

\[
\begin{align*}
C_{0.1}^{\text{(non-info Bayes)}} & = 51.94\% \\
\sigma_{c0.1}^{\text{(non-info Bayes)}} & = 8.66\%
\end{align*}
\]

If we compare \( C_{0.1} \) for MLE and non-informative Bayes we get

\( C_{0.1}^{\text{(MLE)}} > C_{0.1}^{\text{(non-info Bayes)}} \) in 8 cases

\( C_{0.1}^{\text{(MLE)}} < C_{0.1}^{\text{(non-info Bayes)}} \) in 1 case

\( C_{0.1}^{\text{(MLE)}} = C_{0.1}^{\text{(non-info Bayes)}} \) in 11 cases

As should be expected, non-informative Bayes gives the same results as the conditional MLE.
12.2 Real World Data

The real world data are gathered from a variety of sources (Cfr. the references in [11]). It was not possible to deduct from the papers such things as

- number of executions per time unit

- number of person hours per error

etc.

It is thus not possible to know what the real underlying distribution is. For this reason, we can not expect these data to give as good results as the simulated data sets.

In the sub-chapters that follows, we will use data which have been scrutinized in chapter 8. Cfr. specially table 8.3.

12.2.1 The Log Normal Distribution

Of the data discussed in chapter 8.6, eight data sets could be used in the log normal distribution Bayes model. For these eight data sets the linearization method gave us the following \( \mu \) and \( \sigma \) values:
\[
\begin{array}{|c|c|c|}
\hline
\mu & \sigma & N_0 \\
\hline
2.06 & 0.49 & 124 \\
1.82 & 0.48 & 58 \\
1.34 & 0.74 & 74 \\
2.24 & 1.15 & 101 \\
1.80 & 0.88 & 156 \\
2.45 & 1.54 & 16 \\
1.03 & 1.21 & 39 \\
1.57 & 0.84 & 191 \\
\hline
\end{array}
\]

\[\bar{\mu} = 1.79, \quad S_\mu = 0.47\]
\[\bar{\sigma} = 0.92, \quad S_\sigma = 0.37\]
\[\bar{N}_0 = 95, \quad S_{N0} = 60\]

\textbf{Table 12.1}

In order to check the correlation between the parameters, the Spearman correlation coefficient was computed for the three parameter pairs. This gave us

\[\hat{\rho}(\mu, \sigma) = 0.14\]
\[\hat{\rho}(\mu, N_0) = -0.14\]
\[\hat{\rho}(\sigma, N_0) = -0.38\]  

(12.7)

We see that the two parameters \(\mu\) and \(\sigma\) are weakly correlated. According to the prior used (Cfr. (10.40) and (10.41)) the two parameters are uncorrelated. The prior is thus well adapted to our data.

In order to use the apparatus developed in chapter 10.2, we also need \(r\) and \(\hat{\rho}_{r}\), where \(r = 1/\sigma^2\). The data in table 12.1 gives
\[ \bar{r} = 1.86, \ S_r = 1.54 \]

We can now find \( \alpha, \beta, \tau \) and \( \mu \) from (10.49) and (10.41). Some straightforward computation gives us the following parameters for the prior:

\[
\begin{align*}
\alpha &= 2.25, \ \beta = 1.21 \\
\tau &= 4.38, \ \mu = 1.79
\end{align*}
\]

In all cases we can estimate \( \mu \) and \( \sigma \) in a Bayesian manner. In five cases, the number of lines are available, so that we can use the apparatus developed in chapter 10.3 as an alternate way to estimate \( N_0 \).

The Bayes method that we use is built on the conditional likelihood (cfr. chapter 11.3.2). Its main effect is to tone down the erratic variation in the estimates. A typical example of this effect is shown in fig. 12.5.
Data set Fujitsu II

Fig. 12.5

The same effect is seen in the $N_o$ estimates, as shown in fig. 12.6.
If we now add the prior for $N_0$, found in chapter 10.3, we get results of a rather uneven quality. This is as should be expected, since the $N_0$ prior stems from a set of $N_0$ to no-of-lines ratios which is some kind of industrial average. One system in particular, which consists of 250,000 lines and contains only 0.6 errors per 1000 lines, gives rather bad results. It should be noted that the data from this system are concerned with the operational phase while all the others concern systems test or debugging phase.
The data set that performed best came from a 3400 lines system during system test. The results are shown in fig. 12.7.

For the non-informative prior, as specified in (10.51-1), we find that when we compare the MLE to the Bayes estimates with non-informative prior, we find that the non-informative Bayes produces a lower r-value than the MLE in seven out of eight cases.
For the $C_{0.1}$, the situation is less clear:

- $C_{0.1}$ (non-info Bayes) < $C_{0.1}$ (MLE) in 2 cases
- $C_{0.1}$ (non-info Bayes) > $C_{0.1}$ (MLE) in 2 cases
- $C_{0.1}$ (non-info Bayes) = $C_{0.1}$ (MLE) in 4 cases

Overall, we can say that the non-informative Bayes estimates are as good as or better than the MLE.

When we compare the non-informative Bayes with the empirical prior, we find that the r-score is lower for the empirical Bayes in all cases but one. The same holds for $C_{0.1}$.

12.2.2 The Weibull Distribution

Of the data discussed in chapter 8.6, thirteen data sets could be used in the Weibull distribution Bayes model. For these thirteen data sets we get the following $(\alpha, \lambda)$ data:
Table 12.2

The Spearman correlation coefficient was computed for all three parameter pairs. This gave the following results:

\[ \hat{\rho}(\alpha, \lambda) = -0.78 \]  \quad \quad \quad \quad \quad (12.9)
\[ \hat{\rho}(\alpha, N_0) = 0.13 \]  
\[ \hat{\rho}(\lambda, N_0) = -0.40 \]  

From the prior distributions suggested in (11.98), we get

\[ u = 2.50 \quad a = 1.10 \]  \quad \quad \quad \quad \quad (12.10)
\[ v = 41.6 \quad b = 2.50 \]  

When we experimented with the simulated data, however, we found
that there was but little difference between the results obtained by computing the expectation from the posterior distribution and computing the mode of the joint distribution. Since the latter method allows much more sophisticated priors, we have used a gamma prior for \( \alpha \) instead of the uniform prior suggested in (11.98). Cfr. (11.114) and (11.115).

As should be expected, the Bayes estimates were better than the simple, conditional MLE in all cases. A typical example is shown in fig. 12.8.
The dampening effect, which causes the Bayesian estimate to follow the non-Bayesian one, but in a more restrained way, is clearly visible.

When we introduce a prior for $N_0$, the results vary wildly. In order to show how bad it can get, the $N_0$ estimates with and without a prior distribution for $N_0$ in Military System [11], are shown in fig. 12.9.
Military System II

Fig. 12.9

On the other hand, when the prior for $N_0$ is in the right ballpark, the $N_0$ estimates are greatly improved. This implies
that the selection of a $N_0$ prior is critical to the performance of the estimates and must be selected with care. It might be possible to obtain better estimates by incorporating other information about the system in addition to the number of lines of source code. Cfr. chapters 13.4 and 13.5 (Further Work).

When we apply the non-informative prior (11.119) and compare the results to the results obtained by the MLE, we find that $r(\text{non-info Bayes})$ is less than $r(\text{MLE})$ in all but three cases. For the $C_{0.1}$ we find that non-informative Bayes is better or equal to the $C_{0.1}$ obtained by the MLE in ten out of thirteen cases.

Since the non-informative prior incorporates the negative correlation between $\lambda$ and $\alpha$ which is missing in the empirical Bayes model, it is interesting to compare the results for these two estimators.

We find that $r(\text{Bayes}) < r(\text{non-info Bayes})$ in ten cases. The $C_{0.1}(\text{Bayes})$, however, is better that $C_{0.1}(\text{non-info Bayes})$ in only four cases. In the other nine cases $C_{0.1}(\text{non-info Bayes})$ is better (two cases) or just as good (seven cases).

As for the simulated data sets, the main problem is that the minimising algorithm get stuck at the lower bound for $\lambda$.

12.2.3 The Gamma Distribution

Of the data discussed in chapter 8.6, eleven datasets could be used in the gamma distribution Bayes model. For these eleven data sets we get the following $(\alpha, \beta)$ data:
\[
\begin{array}{|c|c|c|}
\hline
\alpha & \beta & N_o \\
\hline
5.42 & 0.66 & 124 \\
5.46 & 0.86 & 58 \\
2.35 & 0.56 & 74 \\
1.57 & 0.19 & 42 \\
1.66 & 0.07 & 102 \\
2.20 & 0.16 & 74 \\
1.23 & 0.12 & 101 \\
1.58 & 0.22 & 156 \\
1.80 & 0.20 & 41 \\
2.16 & 0.22 & 155 \\
1.55 & 0.25 & 191 \\
\hline
\end{array}
\]

Table 12.3

In order to see how/if any of the parameter pairs are correlated, the Spearman correlation coefficients were computed. This gave us:

\[
\hat{\rho}(\alpha, \beta) = 0.61
\]

\[
\hat{\rho}(\alpha, N_o) = -0.26
\]

\[
\hat{\rho}(\beta, N_o) = 0.13
\]

From (10.15), it is straightforward to see that

\[
\beta | \alpha \sim \text{Gamma}(\alpha, s)
\]

Thus
\[ E(\beta|\alpha) = \frac{\alpha^n}{\Gamma(n)} \]  

(12.11-2)

From this it follows that \(\alpha\) and \(\beta\) are positively correlated in Millers prior (10.15) and the prior should thus be a good model of the \(\alpha, \beta\) relationship.

In order to use Millers' prior, we must compute the following variables:

\[ s = nT, \quad p = \frac{\gamma}{n}, \quad n = v \]

(cfr. chapter 10.1.2.2).

In order to find the equivalent \(s\), \(p\) and \(v\) we will use the following well known results:

\[ \alpha = \gamma, \quad \psi(\alpha) - \ln \gamma = \ln T \]

\[ \gamma = \frac{1}{n \psi'(\alpha) - 1} \]

This gives us the following values:

\[ T = 7.66, \quad T = 6.17, \quad n = 4.65 \]

\[ s = 35.62, \quad p = 4729.62, \quad v = 4.65 \]

(12.13)

The posterior estimates of \(\alpha\) and \(\beta\) follows directly from (11.126) and (11.127). The computer programming necessary is straightforward. We have used the NAG routine D01AMF for integration [50]. The \(N_0\) estimate can be obtained from (7.79).
A typical result is shown in fig. 12.10. For the sake of illustration, both linearization, MLE and Bayes estimates are shown in the same diagram.

![Graph](image)

Fujitsu I Data Set

**Fig. 12.10**

Of the eleven data sets which are considered to stem from a gamma distribution, nine have the number of lines in the system available and for these data sets we are able to compare $N_0$ estimates with and without a $N_0$ prior.

If we use a prior for $N_0$, we observe the same effect as we did for the log normal distributed failure times, that is: All data
sets which concern the operational phase give worse results with Bayes that they do without.

A typical result from a system test dataset is shown in fig. 12.11 below

When we compare the results of the conditional MLE and Bayes with a non-informative prior we find that
- \( C_{0.1} \) is the same for both methods for all data sets

- \( r \) is slightly better for the Bayes estimates, but only in one case is it more than 10% smaller.

If we compare the Bayes estimates with non-informative and empirical priors we see that

- \( C_{0.1} \) is the same for both methods for all data sets

- \( r \) is smaller for the Bayes estimates, but only in four out of eleven cases are the improvements more than 10%.
12.3 The Proof of the Pudding

The engineering approach:

"The proof of the pudding is in the eating."

The mathematical approach:

"The proof of the pudding is in the proofreading of the recipe."

After having been proofreading recipes for quite some time, the time has now come for "some eating".

The scene is as follows:

A data set concerning a systems test of a software product from Raytheon was published in [55]. The data is of the form number of failures reported per month for a total of 20 months. The size of the system is 28K lines of source code. Our task is to predict, as early as possible.

- total number of errors

- when will number of failures reported per month fall below 1.
<table>
<thead>
<tr>
<th>Month</th>
<th>No. of errors</th>
<th>Month</th>
<th>No. of errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>11</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>21</td>
<td>12</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>15</td>
<td>13</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>17</td>
<td>14</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>16</td>
<td>15</td>
<td>2</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>16</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>9</td>
<td>17</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>18</td>
<td>1</td>
</tr>
<tr>
<td>9</td>
<td>6</td>
<td>19</td>
<td>1</td>
</tr>
<tr>
<td>10</td>
<td>3</td>
<td>20</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 12.4

After the five first months, it seems that the data set is skewed to the left and that a Gamma, Weibull or log Normal distribution will be a reasonable approximation for the underlying failure time distribution.

We will now use the following approach:

For each month:

1) Compute the Bayes estimates for the distribution parameters $\hat{\theta}_k$ and $\hat{\theta}_0$ for the Gamma, Weibull and log Normal distributions $(\hat{\theta}_k)$ with the data available

2) plot $n(k)$ and $\hat{\theta}_0[F(i, \hat{\theta}_k) - F(i-1, \hat{\theta}_k)]$ for all the three distributions
3) Select the distribution class where observation and predictions are most close and use its predictions of \( N_0 \) and \( t_{acc} \) as your forecast.

When we use the \( \chi^2 \)-statistics as a measure in 3), we get the following results:

<table>
<thead>
<tr>
<th>Month</th>
<th>( \hat{N}_0 )</th>
<th>( \hat{t}_{acc} )</th>
<th>( \chi^2 )</th>
<th>Dist</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>121</td>
<td>11</td>
<td>37.13</td>
<td>W</td>
</tr>
<tr>
<td>7</td>
<td>95</td>
<td>12</td>
<td>9.03</td>
<td>W</td>
</tr>
<tr>
<td>9</td>
<td>109</td>
<td>15</td>
<td>10.80</td>
<td>W</td>
</tr>
<tr>
<td>11</td>
<td>110</td>
<td>17</td>
<td>5.71</td>
<td>W</td>
</tr>
<tr>
<td>13</td>
<td>109</td>
<td>16</td>
<td>2.60</td>
<td>W</td>
</tr>
<tr>
<td>15</td>
<td>115</td>
<td>20</td>
<td>4.48</td>
<td>W</td>
</tr>
<tr>
<td>17</td>
<td>117</td>
<td>21</td>
<td>3.98</td>
<td>W</td>
</tr>
<tr>
<td>19</td>
<td>119</td>
<td>24</td>
<td>4.38</td>
<td>G</td>
</tr>
<tr>
<td>21</td>
<td>120</td>
<td>24</td>
<td>4.67</td>
<td>G</td>
</tr>
</tbody>
</table>

Table 12.5

In [55], it is reported that a total of 128 failures were observed. It is, however, unclear when the last eight errors surfaced.

As seen in Table 12.5, our selection of distribution changed from Weibull to Gamma for the last two periods. This switch is, however, of small consequences, since we, with the Weibull distribution would get the same \( N_0 \) estimates and \( t_{acc} \) equal to 21 and 22 months respectively.

The development of our \( N_0 \) and \( t_{acc} \) estimates are shown in
fig. 12.12 and 12.13 respectively.

Fig. 12.12
We see from fig. 12.12 that after nine months, our $N_0$ estimate is never more than 10% off ($C_{0.1} = 45\%$).
12.4 Evaluation of Bayes Estimates

Three conclusions seem obvious from our discussions in chapter 12.1 and 12.2.

- A non-informative prior gives estimates that are as good as, or better than the MLEs.

- A bad prior causes the Bayes estimates to be worse than the MLEs.

- A good prior should incorporate parameter correlation if the model or earlier experience suggest that the parameters are in fact correlated.

These three conclusions must have effect on the choice of priors for software reliability estimates. Two important rules can be suggested:

- If we have no information on the operating environment, we are usually better off with a non-informative prior than with a prior building on irrelevant or wrong information.

- If we have reasons to believe that the problem domain is the same as at an other site, while \( w(t) \) and \( P_{new}(.) \) probably are different, we should use a prior for \( N_0 \) and non-informative priors for the distribution parameters, and vice versa.

In addition, it seems that the failure time distribution parameters vary less than number of errors per (1000) lines. The reason for this may be that \( w(t) \) and \( P_{new}(.) \) at least partly is a result
of common human characteristica and thus fairly stable. From this it follows that unless we have good reasons to believe that the number of errors to number of lines ratio is as in some other, earlier case, we should use a non-informative prior for $N_0$ and empirical priors for the distribution parameters.
13. Further Work

As the work on these thesis draws to an end, it seems clear to me that now is really the time when the real work should start. Since both my employer (RUNIT) and my family seem to crave for my presence, all I can do, however, is to point out where one should dig and then hope that I or someone else will be able to get time and funding to dig for it sometime in the future.

There are six areas which deserve further attention. Each one is treated shortly in a subchapter.
13.1 Validation of some Important Assumptions

In order to arrive at the presented solution we have made quite general assumptions on how a software system is used. Although the assumptions are founded on a long experience, they have never been validated.

The assumptions concerning $W(t)$ can be tested by logging number of executions for systems during all life cycle phases. It should then be straightforward to find a class of functions which is suitable for describing $W(t)$.

The $P_{\text{new}}(.)$ is another problem. In principle at least, it can be solved by logging a path description of each execution, but this is hardly practical since it will make it necessary to instrument the system to an extreme degree. Such instrumentation as IFTABLE and LLOPTABLE in NUALGOL [53] will do the job and may be used in initial experiments. Similar methods and instrumentation might also be available in other programming languages. In addition, experiments in the field of psychology will probably add to our knowledge of $P_{\text{new}}(.)$.

Once we have functions $P_{\text{new}}(.)$ and $W(t)$, we can attack the problem of finding the failure time distribution $f(t)$. It is straightforward to show that there exists simple and reasonable assumptions about $P_{\text{new}}(.)$ and $W(t)$ which leads to the log Normal distribution and the Weibull distributions respectively, cfr. chapters 6.5.3 and 6.5.4. These results should, however, be enhanced and generalized.
13.2 The Generalized Gamma Distribution

In chapter 8.6 we found that most of the case history data that we have been able to lay our hands on can be adequately described by assuming that the failure times are Log Normal, Gamma or Weibull distributed. These three distributions are special cases of the generalized gamma distribution [52]. It is tempting to assume that it should be possible to find general functions describing $P_{\text{new}}(\cdot)$ and $W(t)$ which, when inserted into theorem 3, chapter 5, will give the generalized gamma distribution as a failure time distribution.

Although the density function for the generalized gamma distribution is straightforward, the accumulated probability function is not, since it contains the confluent hypergeometric series ($_{1}F_{1}(\cdot:\cdot:\cdot)$). Thus, the conditional likelihood will get rather cumbersome to work with. There has, however, been published quite a few papers on estimation in the generalized gamma distribution, the latest in [54]. These papers can at last serve as a starting point.
13.3 A Decision Theoretic Approach

A possible alternative to estimation in the generalized gamma distribution, is to consider the following decision problem:

Given our experience and the data available up till now, which distribution should we assume for estimation purposes?

Our experience (prior) could be the portion of systems which had failure times that were Log Normal distributed, Weibull distributed and so on.

By using at set of scores like the r defined in chapter 8.5, it should be possible to arrive at a "best" choice of distribution to use. It should be easy to test such a decision set by using simulated failure times.

By assigning cost function to such events as

- predicting too few/too many errors
- too early / too late release
- too long / too short time till the failure rate reaches a certain level

it could be possible to define what we mean by an optimal decision in this case. The model could be further refined by the introduction of utility functions for the software producer and (end) user.
13.4 Bayesian Estimates

Our experience, as discussed in chapter 12, shows that the failure time distribution parameters tend to be more stable than number of failures per (1000) lines of source code.

The crux of the matter probably is that

- the failure time distribution parameters are functions of the way the system is used and how often it is used, as defined by $P_{\text{new}}(\cdot)$ and $W(t)$. These factors can, at least partly, be seen as a result of common human characteristica and thus fairly stable.

- the number of failures experienced is a function of problem domain, system history and life cycle phase (cfr. the discussion in chapter 5).

Based on these consideration, it would be important to measure such things as

- number of errors found per 1000 lines in different life cycle phases

- the purifying effect of a series of installation - maintenance sequences. Note that the original number of errors $N_0(0)$ is unknown. We can only observe the number of errors which has occurred at each site $(N_0(i))$. The problem is how to estimate $N_0(j)$, when $N_0(i)$ is known for all $i < j$

- the difference between sites and/or problem domains as pertaining to the number of failures that will be experienced.
Even a partial solution to these questions will help us to select a more realistic prior for $N_0$ and thus greatly improve the $N_0$ estimates.
13.5 The Effect of Various Software Metrics

In order to make our model more refined and thus more precise, we should include more knowledge. Sources of knowledge could be

- descriptions of the development process
- design review reports, test reports and trouble reports
- descriptions of sites where the system have already been installed and of how the system has performed there
- a description of the present site and its problem domains

Some of the knowledge will be of the yes/no (present/not present) type while some of it can be measured in a more precise way, such as number of issues raised during review of the integration plan.

Our total knowledge can now be encoded into a vector $X$. It should be possible to find the impact of each factor on the prior of $N_0$ and the failure time distribution parameters. The impact could be described by a coefficient vector $\beta$ and the whole model can be described by the Cox model, so that for instance

\[ N_0 = N_0^* e^{\beta^T X} \]  

(13.3)

where $N_0^*$ is some kind of "base" value for $N_0$.

This technique is already well known in life time estimates in medicine and reliability analysis. Cfr. for instance [22] and the software package BMDP [56] which can be used for this purpose.
13.6 Parameter Variation

From what is said in chapter 5 it follows that the distribution parameters are dependent on the shapes of $W(t)$ and $P_{\text{new}}(.)$. Since the estimation methods suggested in this paper deal with one distribution at a time, it is important to detect changes in usage intensity or testing strategy as early as possible.

Two aspects are important:

- change in distribution. E.g. due to new information of the operating environment, there has been a change in testing strategy. This will change $P_{\text{new}}(.)$ and thus $f(t)$.

- change in one or more distribution parameter. E.g. more (or less) people uses the system now than before.

A possible way to model such changes could be by the use of Kalman filters where each parameter can be modelled as a base value, a trend term and a noise term. Cfr. for instance [60].
APPENDIX A

Some Sums used in Chapter 10.3

\[ S_0 = \sum_{i=0}^{\infty} p \frac{\Gamma(k+i)}{\Gamma(i+1)}, \quad |p| < 1 \quad \text{(A.1)} \]

In order to find \( S_0 \), we will use the fact that (10.67) is the point probability for \( N_0 \). Let us introduce the substitutions

\[
\begin{align*}
N_0 - N &= i \\
F(t) &= i \\
\frac{N}{1+\beta} &= p
\end{align*}
\quad \text{(A.2)}
\]

From the fact that the sum of all point probabilities is equal to one, we get

\[ \frac{(1-p)^{\alpha+N}}{\Gamma(\alpha+N)} \sum_{i=0}^{\infty} p \frac{\Gamma(\alpha+N+i)}{i!} = 1 \quad \text{(A.3)} \]

Furthermore, insert \( k = \alpha+N \). The result then follows:

\[ S_0(k,p) = \frac{\Gamma(k)}{(1-p)^k} \quad \text{(A.4)} \]
The sum $S_1(k, p)$, as defined in (A.5) can now be obtained by a simple rewrite via the derivative with respect to $p$.

Similarly, the sum $S_2(k, p)$, as defined in (A.9), can be obtained by noting the facts that

- summation from 0 and 1 will give the same result

- substitution of $j = i-1$ will split $S_2$ into two sums, one of type $S_0$ and one of type $S_1$ (cfr. (A.10)).

\[
S_1(k, p) = \sum_{i=0}^{\infty} i \frac{\Gamma(k+i)}{\Gamma(i+1)} \cdot |p| < 1 \quad (A.5)
\]

\[
= p \frac{\delta}{\delta p} \sum_{i=0}^{\infty} \frac{i \Gamma(k+1)}{\Gamma(i+1)} \quad (A.6)
\]

\[
= p \frac{\delta}{\delta p} \{ S_0(k, p) - \Gamma(k) \} \quad (A.7)
\]

\[
S_1(k, p) = \frac{p^k}{(1-p)^{k+1}} \Gamma(k) \quad (A.8)
\]

\[
S_2(k, p) = \sum_{i=0}^{\infty} i \frac{\Gamma(k+i)}{\Gamma(i+1)} \cdot |p| < 1 \quad (A.9)
\]
If we substitute $j = i - 1$, we get

$$S_2(k,p) = p[S_1(k+1,p) + S_0(k+1,p)]$$

(A.10)

$$S_2(k,p) = \frac{pk(pk+1)}{k+2} \frac{\Gamma(k)}{(1-p)}$$

(A.11)
APPENDIX B

SIMULATED DATA

The simulated data are generated on a UNIVAC 1106 computer. The procedure DRAW generates pseudo-random numbers which are $U(0,1)$. From these numbers it is straightforward to obtain random numbers with a Weibull, log Normal, Gamma or log Gamma distribution.

The following algorithms are used:

Gamma (integer $k$)

Generate $k$ numbers $U_1, \ldots, U_k$ from DRAW

$$x_i := -\frac{\ln(U_i)}{\lambda}$$

$$t := \frac{\sum_{i=1}^{k} x_i}{k} \quad \text{is Gamma}(k, \lambda) \text{ distributed}$$

Weibull:

$$t := \left[-\frac{\ln(u)}{\lambda}\right]^{1/\alpha} \quad \text{is Weibull } (\alpha, \lambda)$$

Normal:

$$X := \left[-2\ln\left(\frac{u}{2}\right)\right]^{1/2}$$

$$y := x + \frac{C_0 + C_1 x + C_2 x^2}{1 + d_1 x + d_2 x^2 + d_3 x^3}$$
where $C_i, d_i$ are the coefficients in Hastings formula (Cfr. 26.2.23 in [12]).

$$t := \mu + \sigma y \text{ is } N(\mu, \sigma^2)$$

**Gamma (real } k < 1):**

$$y_1 := \frac{u_1}{\alpha}$$

$$y_2 := \frac{u_2}{1-\alpha}$$

$$w := \frac{y_1}{y_1 + y_2}$$

$v$ is drawn from the distribution $\text{Exp}(1.0)$

$$t := \frac{vw}{\beta} \text{ is Gamma}(\alpha, \beta)$$

(cfr. [46])

The basic DRAW algorithm is as follows [57]:

"A basic drawing will replace the value of a specified integer variable say, $U$, by a new value according to the following algorithm."
\[ U_i = \text{remainder } ( ( U \times 5^{2p+1} ) / 2^n )_{i+1} \]

where \( U_i \) is the \( i \)'th value of \( U \).

It can be proved that, if \( U_0 \) is a positive odd integer, the same is true for all \( U_i \), and the sequence \( U_0, U_1, U_2, \ldots \) is cyclic with the period \( 2^{n-2} \). (The last two bits of \( U \) remain constant, while the other \( n-2 \) take on all possible combinations). In UNIVAC 1107/1108/1106 we have \( n = 35 \). \( p \) is chosen equal to 6.

The real numbers \( u_i = U_i \times 2^{-n} \) are fractions in the range \( <0,1> \). The sequence \( u_1, u_2, \ldots \) is called a stream of pseudo-random numbers, and \( u_i \) \((i = 1, 2, \ldots)\) is the result of the \( i \)'th basic drawing in the stream \( U \). A stream is completely determined by the initial value \( U_0 \) of the corresponding integer variable. Nevertheless it is a "good approximation" to a sequence of truly random drawings.

By reversing the sign of the initial value \( U_0 \) of a stream variable the antithetic drawings \( 1 - u_1, 1 - u_2, \ldots \) are obtained. In certain situations it can be proved that means obtained from samples based on antithetic drawings have a smaller variance than those obtained from uncorrelated streams. This can be used to reduce the sample size required to obtain reliable estimates."

We have used the following start seed values:

Log Normal : \( U_0 = 67894559 \)

Gamma : \( U_0 = 23467719 \)
Log Gamma : \( U_0 = 23467719 \)

Weibull : \( U_0 = 78544397 \)

With these seeds we generated 20 sets in each of the following cases:

- 200 failure time with distribution Gamma(3.0, 0.2).

- 200 failure times with distribution Weibull(1.66, 0.04).

- 200 failure times with Log Normal distribution, so that \( \ln t \) was Normal(2.0, 1.0).

- 400 failure times with Log Gamma distribution, so that \( \exp(t) \) was Gamma(0.1, 5.04 \times 10^{-8})

Two data sets from each distribution are showed below. The presentation format is:

<title>

<no. of periods> '0'

<no. of failures in period no. i>*

Fig. B.1

The number of failures in each period are written line by line.
**Gamma distributed data:**

**SIMULATION 1**

<table>
<thead>
<tr>
<th></th>
<th>40</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>7</td>
<td>11</td>
</tr>
<tr>
<td>10</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td>8</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

**SIMULATION 2**

<table>
<thead>
<tr>
<th></th>
<th>49</th>
<th>9</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>11</td>
<td>10</td>
<td>13</td>
</tr>
<tr>
<td>15</td>
<td>7</td>
<td>6</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>8</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
Log Normal distributed data:

SIMULATION 2

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>11</th>
<th>16</th>
<th>12</th>
<th>17</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>18</td>
<td>5</td>
<td>8</td>
<td>6</td>
<td>5</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>2</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
SIMULATION 3

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>9</td>
<td>5</td>
<td>6</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>22</td>
<td>12</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>14</td>
<td>5</td>
<td>5</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>20</td>
<td>12</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>13</td>
<td>9</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>3</td>
<td>5</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Weibull distributed data:

SIMULATION 1

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>16</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>30</td>
<td>21</td>
<td>26</td>
<td>24</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>18</td>
<td>19</td>
<td>11</td>
<td>6</td>
<td>4</td>
<td>3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>5</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
SIMULATION 3

19  0
16 19 25 19 25 20
20 12 10  7  6  8
 2  6  2  1  0  1
 1

Log Gamma distributed data

SIMULATION 1

18  0
 8  9 12 16 26 19
16 29 14 28 26 35
40 44 34 28 15  1

SIMULATION 2

18  0
14 12 11 18 14 14
22 14 37 22 32 37
27 33 47 26 18  2
16 29 14 28 26 35
40 44 34 28 15  1
REFERENCES

[1] Bev Littlewood
What Makes a Reliable Program -
Few Bugs, or a small Failure Rate

ANSI/IEEE Std 729-1983

Software Reliability Measurement
Draft of June 1, 1985

Om å bruke modeller
Tidsskrift for Arbeiderbevegelsens Historie
nr. 2, 1981

An Input Domain Based Theory of Software Reliability and Its Applications
Ph.D. Dissertation, University of California Berkeley, 1980

A Bayesian Reliability Growth Model for Computer Software
Mathematical Department
The City University
London EC1, UK
[7] Mitsuru Ohba
Software Reliability Analysis Models
IBM Journal of Research and Development
vol. 28, no. 4, July 1984

[8] F.N. Parr
An Alternative to the Rayleigh Curve Model for Software Development Effort
IEEE Transactions on Software Engineering
vol. SE-6, no. 3, May 1980

The Theory of Stochastic Processes
ISBN 0-412-15170-7
Chapman and Hall Ltd. 1972

Linear Regression Analysis
John Wiley & Sons, Inc. 1977
ISBN 0-471-01967-4

A Collection of Software Failure Data
RUNIT, SINTEF 1987

[12] Milton Abramowitz and Irene A. Stegun
Handbook of Mathematical Functions
ISBN 486-61272-4
Density Estimation for Statistics
and Data Analysis
Chapman and Hall Ltd. 1986
ISBN 0-412-24620-1

[14] Maurice G. Kendall
The Advanced Theory of Statistics
Charles Griffin & Company Ltd.
London 1943

Why do Computers Stop and what can be done About It.
5th Symposium on Reliability in
Distributed Software and Data Base Systems. IEEE
Los Angeles, January 1986

[16] T.A. Greenwood, D. Durand
Aids for fitting the gamma distribution
by maximum likelihood
Technometrics vol. 2, Feb. 1960

[17] Lee J. Bain, Max Engelhardt and Wei-Kei Shiue
Approximate Tolerance Limits and
Confidence Limits on Reliability for the
Gamma Distribution
IEEE Transactions on Reliability
vol. RE-33, no. 2, June 1984
[18] Eivind Damsleth
Conjugate Classes for Gamma Distributions
Scandinavian Journal of Statistics
vol. 2, 1975

[19] Robert B. Miller
Bayesian Analysis of the Two-Parameter
Gamma Distribution
Technometrics vol. 22, no. 1
February 1980

Table of percentage points of Pearson curves,
for given $\sqrt{\beta_1}$ and $\beta_2$, expressed in
standard measures
Biometrika vol. 50, no. 3, 1963

[21] Harry F. Martz and Ray A. Waller
Bayesian Reliability Analysis
John Wiley & Sons 1982
ISBN 0-471-86425-0

[22] B. Lindquist, E. Molnes, M. Rausand
Analysis of SCSSV Performance Data
Reliability Engineering and Safety Systems
vol. 20, 1988

Optimal Statistical Decisions
McGraw-Hill Book Company Inc 1920
ISBN 07-016242-5
Nonparametrics: Statistical Methods
Based on Ranks
Holden-Day, Inc. 1975
ISBN 0-8162-4996-6

[25] Odd Aalen
Nonparametric Inference for a Family
of Counting Processes
The Annals of Statistics
No. 4, vol. 6, 1978

[26] Edward Demko
True Reliability Growth Measurement
Proceedings of the Annual Reliability
and Maintainability Symposium, 1986

[27] John D. Musa and Kazuhira Okumoto
A Comparison of the Time Domains
for Software Reliability Models
The Journal of Systems and Software
no. 4, 1984

[28] Tor Stålnæs
Software Reliability:
A Summary of State of the Art
RUNIT, SINTEF 1987

[29] Robert C. Tausworthe
Estimating Software Costs
Course material, Oslo, September 1983
[30] Stephan S. Yau and Terry E. MacGregor
On Software Reliability Modeling
RADC-TR-29-127, June, 1979

[31] Abdalla Ahmed Abdel-Ghaly
Analysis of Predictive Quality of
Software Reliability Models
Department of Mathematics (Statistics)
The City University, London, April 1986

[32] Arnýjot Høyland
Sannsynlighetsregning og Statistisk
Metodelære (I, II)
Tapir, Trondheim 1976
ISBN 82-519-0158-8

[33] D.R. Cox and Valerie Isham
Point Processes
Chapman and Hall, London 1980
ISBN 0-412-21910-7

[34] DACS Newsletter
Vol. II, no. 1, June 1979
RADC/ISISI, Griffiss AFB, NY.

[35] Tor Stålhane, Kari Juul Wedde
Evaluering av et større programvareprosjekt
SINTEF, Oct. 12, 1979, STF14-F79022

[36] Harlan D. Mills
Structured Programming: Retrospect and Prospect
IEEE Software 1986
[37] D.N. Card, T.L. Clark and R.A. Berg
Improving Software Quality and Productivity
Information and Software Technology
Vol. 29, no. 5, June 1987

[38] Martyn Ould
Is this data interesting?
Software Reliability and Metrics Newsletter
Issue 5, March 1987

Software Reliability Study
Interim Technical Report
contract F30602-74-C-0036
TWR Systems Group, June 1974

[40] Tor Stålhane
Report from Alvey CSR Meeting
Reading, June 2/3, 1987
SINTEF Traveler Report

[41] Lalitha Sanathanan
Estimating the Size of a Truncated Sample
Journal of the American Statistical Association
Sept. 1977, vol. 72, no. 359

[42] Lalitha Sanathanan
Estimating the Size of a Multinomial Population
The Annals of Mathematical Statistics
[43] I.S. Gradshteyn and I.M. Ryzhik
Tables of Integrals, Series and Products
Academic Press 1980
ISBN 0-12-294760-6

[44] Ørnulf Borgan
On the Theory of Moving Average Graduation
Scandinavian Actuarial Journal
1979

[45] Nozer D. Singpurwalla and Naftali Langberg
Unification of Some Software Reliability
Models via the Bayesian Approach
Georg Washington University, July 10, 1982

[46] Reuven Y. Rubinstein
Simulation and the Monte Carlo Method
ISBN 0-471-08917-6

[47] D.V. Lindley
Bayesian Approximation Methods
Trabajos Estadistica no. 31, 1980

[48] S.K. Sinha and Jeffrey Solan
Bayesian Estimation of the Parameters
and Reliability Function of a Three-
Parameter Weibull Life Distribution
Institute of Mathematical Statistics
University of Umeå 1985-5, 1985
[49] R.M. Soland
Use of the Weibull distribution in
Bayesian Decision Theory
Report no. RAC-TP-225
Research Analysis Corporation, 1966

[50] NAG FORTRAN Mini Manual, Mark 10
Numerical Algorithms Group Ltd.
Oxford OX2 7DE, UK
February 1983

[51] Tor Stålhane
Report from Alvey CSR Meeting
Bristol, September, 1987
SINTEF Traveler Report

[52] James B. McDonald, Dale O. Richards
Model Selection: Some Generalized Distributions
Communications in Statistics
vol A16, No 4, 1987

[53] Charles Linett
Specification 76121 for Modification of the
NU-ALGOL Compiler
US Department of Commerce, Bureau of the Census 1976

[54] Dallas R. Wingo
Computing Maximum-Likelihood Parameters
Estimates of the Generalized Gamma
Distribution by Numerical Root Isolation
IEEE Transactions on Reliability
vol R-36, no 5, December 1987
[55] Kenneth S. Mendis  
Quantifying Software Quality  
Quality Progress, May 1982

[56] W.J. Dixon, chief editor  
BMDP Statistical Software  
University of California Press  
Berkeley, 1983

[57] Ole Johan Dahl, Kristen Nygaard  
SIMULA  
National Computing Centre, Sept. 1967

[58] Isao Miyamoto  
Software Reliability in Online, Real Time Environment  
Sigplan Notices, vol 10, no 6, 1975

[59] A.H. Moore and J.E. Bilikam  
Bayesian Estimation of Parameters of  
Life Distribution and Reliability  
from Type II Censored Samples  
IEEE Transactions on Reliability  
vol R-27, 1978

[60] R.J. Meinhold and N.D. Singpurwalla  
Understanding the Kalman Filter  
The American Statistician, May 1983 vol 37, No 2