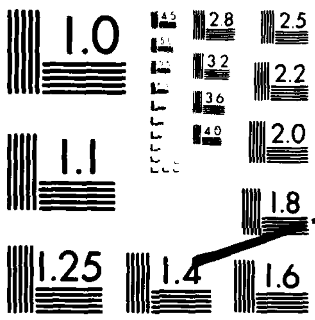
AD-A099 482 CARNEGIE-MELLON UNIV PITTSBURGH PA DEPT OF ELECTRICA--ETC F/G 9/2
WORKLOAD, PERFORMANCE AND RELIABILITY OF DIGITAL COMPUTING SYST--ETC(U)
APR 81 X CASTILLO, D P SIEWIOREK DAS060-80-C-0057

UNCLASSIFIED NL

1 of 1
AD
A099482

END
DATE
FILMED
6-81
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS 1963 A

LEVEL II

(2)

AD A099482

(6) # Workload, Performance, and Reliability of Digital Computing Systems.

(10) Xavier Castillo

Daniel P. Siewiorek

(11) 6 Apr[redacted]81

(12) 94

(9) FINAL REP[redacted]T,

Synthesis of Fault Tolerant
Distributed Computing Systems

Ballistic Missile Defense Systems Command
Contract no. DASG60-80-C-0057

(15)

Carnegie-Mellon University
Departments of Electrical Engineering
and Computer Science

* Additional support from the Fundacion I.T.P., Madrid, Spain.

[illegible]

81 4 24 077

Abstract

In this paper a new modeling methodology to characterize failure processes in Time-Sharing systems due to hardware transients and software errors is summarized. The basic assumption made is that the instantaneous failure rate of a system resource can be approximated by a deterministic function of time plus a zero-mean stationary Gaussian process, both depending on the usage of the resource considered. The probability density function of the time to failure obtained under this assumption has a decreasing hazard function, partially explaining why other decreasing hazard function densities such as the Weibull fit experimental data so well. Furthermore, by considering the Kernel of the Operating System as a system resource, this methodology sets the basis for independent methods of evaluating the contribution of software to system unreliability, and gives some non obvious hints about how system reliability could be improved. A real system has been characterized according to this methodology, and an extremely good fit between predicted and observed behavior has been found. Also, the predicted system behavior according to this methology is compared with the predictions of other models such as the exponential, Weibull, and periodic failure rate.

Table of Contents

List of Figures

List of Tables

1. Introduction

There are several trends in Distributed Data Processing (DDP) Systems that, when applied to the Ballistic Missile Defense (BMD) Task make reliability and fault tolerant requirements not only desirble but also nexessary. These trends include :

- Systems are becoming more complex. Thus, even though component reliability is improving, the total system reliability may be unacceptably low.

- BMD systems must work. Since national survival may depend on the proper functioning of a BMD system whose capabilities are unused except at the moment of crisis, the BMD system must be designed to detect and tolerate failures while in the dormat or monitoring state.

- Repair. System maintenance is often the dominant cost in the system life cycle. This trend is even more amplified by the ever shrinking cost of hardware. A related problem is the disparity between increasing system complexity and decreasing repairman skills. Fault tolerance and Built in Tets (BIT), an application of the fault detection phase of fault tolerance, can help reduce repair costs and repair skill levels.

- Transients. Data from several uniprocessor [McConnel 79] and multiprocessor [Siewiorek 78] systems indicates that transient failures are 20 to 60 times more likely than hard failures. Further, transients exhibit a strong clustering phenomenum. That is, once a transient has occured, there is a high probability that another transient will occur in a very short period of time. This clustering might overwhelm fault tolerant techniques designed for hard failure survival (i.e., a second transient might occur before that the system can recover from what it expects is a first hard failure). Transients will become more of a problem with shrinking device dimensions where small local electrical fields can cause devices to change state. Cases where cosmic rays and background radiation in packaging material caused transients have already been documented. Transients occurrences also seem related to system load, a particularely disastrous feature for a BMD system which must respond to threats with peak processing power.

Therefore, fault-tolerance cost and benefit measures are needed. To evaluate the impact of unreliability on the performance of computing systems, a knowledge of the mechanisms leading to unreliable system behavior is required. From a hardware viewpoint, transients are the dominant cause of system unreliability. However, unreliable software manifestations are almost indistinguishable from hardware transients.

A methodology capable of modeling and characterizing the impact of hardware transients and software errors on the performance of DDP systems must be developed. Unfortunately, not that many DDP systems are available for general use and/or experimentation. And of the DDP systems available, none has the necessary instrumentation tools required to validate a possible theoretical model. Furthermore, modeling methods for hardware transients and software errors is in a very primitive state. Hence, it is not possible to attack the transient/software error problem for DDP systems. Instead, the problem has to be satisfactorily solved for simpler, more accessible systems such as uniprocessor time-sharing systems.

This report develops a methodology and a model for hardware transients and software errors on time-sharing systems. Tools have been developed to gather data from several available systems. Thus, all theoretical results are compared with the behavior of real systems. Some of the results can be extended to DDP systems. In any event, the results presented in this report are a necessary step towards the characterization of the effect of transients and software errors on DDP systems.

1.1 Definitions

The following concepts need to be precisely defined :

Hardware Fault	Erroneous state of hardware due either to failures of components or to physical interference from the environment.
Hardware Error	Manifestation of a hardware fault within a program or data structure.
Permanent Hardware Fault	Hardware fault which is continuous and stable, reflecting an irreversible physical change in the hardware.
Transient Hardware Fault	Hardware fault due to temporary environmental conditions.
Software Fault	Imperfection in the design or implementation of a software module such that upon some timing or value conditions in its input data stream it fails to accomplish its designed task.
Software Error	Manifestation of a software fault within a program or data structure.
System Failure	Manifestation of software or hardware errors that force an entire computing system to suspend its operation.

Since no repair takes place after system failures due to software faults or transient hardware faults, the time of system failure is essentially equal to the system restart time. Since this report is concerned solely in modelling hardware transient faults and software faults, the words "system failure" and "system restart" will be used interchangeably to describe the same event in time.

1.2 The problem of Characterizing System Reliability

Fault-tolerance has traditionally been characterized by relatively simple functions based on strict assumptions. The Reliability function R(t) is defined as the probability of uninterrupted operation up to time t given that all hardware was correctly operating at time t = 0. R(t) may be used to characterize either permanent or transient faults. The usual assumption is made that the failure rate is constant and, for nonredundant systems the reliability function becomes $e^{-\lambda t}$, where λ is is the sum of the failure rates of all the components in the system. A very common quantitative measure is the Mean Time To Failure (MTTF)

$$\mathrm{MTTF} = \int_0^{\infty} R(t)\,dt \tag{1.1}$$

The popularity of the MTTF stems mainly from the fact that, for nonredundant systems, it is easily estimated by dividing the time a system is operational by the number of failures reported. Other reliability indices used to compare two systems A and B, are the Reliability Improvement factor (RIF) [Anderson 67]

$$\mathrm{RIF} = \frac{1-R_A(t)}{1-R_B(t)} \tag{1.2}$$

and the Mission Time Improvement Factor (MTIF) [Bouricious 69]

$$\mathrm{MTIF} = \frac{T_A}{T_B} \quad \text{when } R_A(T_A) = R_B(T_B) = R_{min} \tag{1.3}$$

which are useful only when the system under study must be available for a predetermined period of time T called "mission time".

The concept of *coverage* [Bouricious 69] is defined as the conditional probability of successful recovery, given that a fault has occurred. Although mathematically attractive, coverage has proven to be very difficult to estimate for real systems. Finally, if the Mean Time To Repair (MTTR) is also known, an estimate of the system usefulness given by the *Availability*

$$A = \frac{MTTF}{MTTF + MTTR} \tag{1.4}$$

These and other measures traditionally used to compare systems do not take into account the performance of the system whose reliability is being measured. Consider Table 1-1 which lists the results obtained from seven different experiments whose goal was explicitly to gain experience on systems reliability. Data for the first system [Yourdon 72], was obtained from a summary of failure statistics on a Borroughs 5500 over a 15 month period starting in April of 1969. Limited information about the cause of each failure is available. For instance, one of the categories includes system failures due to unexpected I/O intercepts. These failures are recorded whenever the software responds to an interrupt signifying that some I/O action has taken place, but discovers that it has no record of having initiated such action. It is thus an indication of some form of hardware or software error but the particular cause for the failure (hardware or software) remains unknown. The data for the second system was reported in [Lynch 75] and comes from the first thirteen months of operation of an operating system called Chi/OS for the Univac 1108 developed by the Chi Corporation between 1970 and 1973. No explanation is given about how such an accurate decomposition of failures due to hardware and software could be obtained. [Reynolds 75] reports data obtained from a dual IBM 370/165 at Hughes Aircraft Company over a period of three years installed to handle a mixed batch and time sharing load. The forth system is at the Stanford Linear Accelerator Center (SLAC) where the main workload is processed as multi-stream background batch. The system consists of a foreground host (IBM 370/168) and two background batch servers (IBM 370/168 and IBM 360/91). The architecture is designed to be highly available and reconfigurable. The CMU-10A is an ECL PDP-10 used in the Computer Science Department at Carnegie-Mellon University. The data for the CRAY-1 was reported in [Keller 76], and the data for the three generic UNIVAC systems was reported in [Siewiorek 80].

Table 1-1 gives, when available, a Mean Time to reStart (MTTS) value in hours (that is, the Mean Time to System Failure), a Mean Number of Instructions to Restart (MNIR) which is an estimate of the mean number of instructions executed from system start up until system failure, and the percentages of system failures that were caused by hardware faults, software faults, and whose cause could not be resolved. The information about execution rates needed to compute the MNIR value was obtained from [Phister 79].

Obviously, the figures shown in Table 1.1. do not carry much information. A MTTS figure alone

System	MTTS (hours)	MNIR	% HW	% SW	% Unknown
B 5500	14.7	$2.6\ 10^{10}$	39.3	8.1	52.6
Chi/05 (Univac 1108)	.17	$6.7\ 10^{10}$	45	55	-
dual 370/165	8.86	$2.8\ 10^{11}$	65	32	3
SLAC	20.2	$2.3\ 10^{11}$	73.3	21.6	5.1
CMU-10A	10	$4.3\ 10^{10}$	-	-	-
CRAY-1	4	$1.9\ 10^{12}$	-	-	-
UNIVAC (Large)			51	42	7
UNIVAC (Medium)			57	41	2
UNIVAC (Small)			88	9	3

Table 1-1: Reliability experience of several comercial systems. MTTS is the Mean Time to reStart. MNIR is the Mean Number of Instructions to Restart.

does not tell the impact of unreliability on system use. Compare fcr example the CRAY-1, [Russell 78], with the CMUA, [Bell 78]. Although the CRAY-1 crashes twice as often as the CMUA, it can operate continously at rates above 138 Million Instructions Per Second (MIPS), while the CMUA operates at 1.2 MIPS. Hence the CMUA executes $\sim 10^{10}$ instructions between crashes while the CRAY-1 executes $\sim 10^{12}$ instructions between crashes. Inconsistancies like this one suggest that reliability modelling and measuring should be closely related with the characterization of the performance of the system under study. Integrated performance-reliability models have already started to appear in the literature. In [Meyer 79], a performance measure called "performability" gives the probability that a system performs at different levels of "accomplishment". In [Gay 79], systems are modelled with Markov processes in order to estimate the probability of being in one of several capacity states. This is a similar approach tc the one previoulsy taken in [Beaudry 78], where the concept of "computation reliability" was introduced as a measure which takes into account the computation capacity of a system in each possible operational state. Finally, a Performance/Availability model for gracefully degrading systems with critically shared resources is given in [Chou 80].

However, most of the above models have been developed mainly for hard failures, that is, stable failures that reflect an irreversible physical change in the hardware. Unfortunately, as it has been repeatedly reported ([Fuller 78], [McConnel 79], [Morganti 78], [Siewiorek 78], [Ohm 79]), transient failures occur at least an order of magnitude more often than hard failures. A cost effective analysis should then consider transients as the main reason for system unreliability.

Simultaneoulsy with the developments described above, qualitative relationships between workload and unreliability have also been noted. The results published in [Beaudry 79] suggest a strong dependency between workload and reliability of digital computing systems. And in the paper by [Butner 80], this dependency is stated explicitly claiming that a periodic, workload-dependent failure rate is more appropriate to characterize the reliability of time-sharing systems than the classical constant failure rate model traditionally used. As reported in [Castillo 80], if such a dependency is taken into account it is possible to characterize the performance of digital computing systems considering reliability as an inherent attribute.

1.3 Software Reliability

The problem of software reliability assessment is part of the more general area of software quality assessment [Mohanly 73]. Effective machanisms for measuring software quality are required due to the high cost of software development and maintenance. By 1985 forecasts indicate that over 90% of the total computing dollars spent annually will be for software [Horowitz 75]. The development of techniques for measuring software reliability has been motivated mainly by project managers that need both ways of estimating the man-power needed to develop a software system with a given level of performance and techniques to detect when this level of performance has been reached. However, most software reliability models presented up to date are still far from satisfying these two needs in a general context.

Software reliability models can be roughly grouped in four categories. The first category would include models formulated in the time domain. These models attempt to relate software reliability (characterized, for instance, by a MTTF figure under typical workload conditions) to the number of bugs present in the software at a given time during its development. Typical of this approach are the models presented in [Shooman 73], [Musa 75], and [Jelinsky 73]. Bug removal should increase MTTF and correlation of bug removal history with the time evolution of the MTTF value may allow the prediction of when a given MTTF value will be reached. An example of the application of time domain models to the development of a real-time system is given in [Miyamoto 75]. The main disadvantages of time domain models are that bug correction can generate more bugs, and that software unreliability can be due not only to implementation errors (bugs) but also to design (specification) errors.

Another approach to software reliability modeling is based on studying the data domain. The first model of this kind is described in [Nelson 73]. In principle, if sets of all input data values upon which a computer program can operate are identified, an estimated of the reliability of the program can be obtained by running the program for a subset of input data values. A more detailed description of data domain techniques is given in [Thayer 78]. In the paper by [Schick 78] the time domain and data domain models are compared. However, different applications will tend to use different subsets of all possible input data values, "seeing" different reliability values for the same software system. This fact is formally take into account in [Cheung 80], where software reliability is estimated from a Markov model whose transition probabilities depend on a user profile. Techniques for evaluating the transition probabilities for a given profile are given in [Cheung 75].

The third category includes models in which software reliability (and software quality in general) is postulated to obey certain laws [Ferdinand 74], [Fitzsimmons 78]. Although such models have generated high amounts of interest, their general validity has never been proven and, at most, they only give a figure for the number of bugs present in a program.

Finally, there have been some attempts to characterize total system reliability (hardware and software) in [Costes 78], modelling of fault-tolerant software (through module duplication) in [Hecht 76], and warnings about how not to measure software reliability [Littlewood 79].

What all the above models have in common is that none of them characterizes system behavior accurately enough as to give to the user a figure of guaranteed level of performance under general workload conditions. They concentrate in estimating number of bugs present in a program but do not give any accurate method to characterize and measure operational system unreliability due to software. There is a wide gap between the varaiables that can be easily measured in a running system and the number of bugs in its operating system. However, a cost effective analysis should precisely allow to evaluate the impact of software unreliability from variables easily accessible in an operational system, without knowing the details of how the operating system has been written.

1.4 Measuring Reliability under typical and atypical conditions

The assumption here is that reliability is a performance attritbute in the sense that a lack of reliability increases the expected value of the system response time. If such a relationship can be derived in a general context, policies and/or design parameters could be used to optimize the ultimate system performance. In this report, the approach taken is that failure rate time variations should closely follow workload time variations. Intuitively, the dependency between workload and lack of reliability can be explained quite easily. Assume that we have a constant failure rate for the primary memory of a digital computing system operating in a stable environment under a time sharing policy. That the transient failure rate in a memory is constant is a reasonable assumption. There is justification for thinking that certain complex devices might follow an exponential failure law ([Barlow 65], pp 18-22). The physical characteristics of the memory IC's do not change with time (at least during the effective life cycle of modern digital computing systems). We have to look then for the origin of these transients either in external sources, such as radiation, the presence of noise (possibly impulsive) in the power supply or in the limitations of the manufacturing process. In fact, it has been reported in [Geilhofe 79] that MOS memory devices exhibit non recurring bit failures caused by Alpha particles emitted from small amounts of radioactive elements present in IC packaging material. The failure rate for this kind of failures is of course constant. Assume now that a transient memory failure has higher probability of leading to a system crash when the central processor is executing in Kernel mode than when it is executing in user mode. A memory failure when the CPU is executing in user mode may affect a user process but will not crash the system. The system failure rate due to transient memory failures will then depend on the ratio of the number of memory references while in Kernel mode to the total number of memory references per unit time. Since it is a well known fact that operating system overhead increases with workload, the previous ratio will also be a nondecreasing function of the system workload, increasing in turn the observed system failure rate. The result is that the observed system failure rate due to transient memory failures should be equal to the sum of a component following the operating system ovehead variations in time (or indirectly, workload variations in time), plus a constant, workload independent component (even if the system is idle, there may still be memory errors that corrupt, for instance, the clock interrupt subroutine).

Even if the fact that a computing system is not always equally sensitive to the presence of hardware errors is not considered, there are still arguments to support the idea that the apparent system failure rate should depend on the workload. The fact is that in most computing systems, a component failure will be noticed only if the component is "exercised". A time sharing system with no load, spending most of its time in a wait state and only a fraction of the time executing the clock interrupt routine may sustain several failures and still not report any errors if the minimal hardware configuration required to execute these basic functions is not affected. It is not maintained here that failures will be caused by increased utilization (although in some cases this situation is certainly possible) but that they will be detected by an increase in system utilization. This effect has also been referred to as "error latency" [Shedletsky 73].

Analogous arguments lead to the expectation that the rate of system failures due to software unreliability will depend on how much the software is exercised. System software failures are due to: a) the (static) input data to a progam module presents some peculiarities that the program is not able of handling or, b) the software is not capable of handling some time dependent (dynamic) sequence in the input data stream. In the case of a time sharing system, the only software capable of provoking a system failure is the Kernel of the Operating System. This software usually executes in a privileged processor state and a software error that corrupts some critical information in the Kernel data structures may lead to a system failure. However, since nobody has any a priori knowledge of what these errors are, it is less likely that the system finds one of these combinations in its input stream under low load (that is, small amounts of input data to process per unit time) than in a high load situation. Again, the observed system failure rate has to depend on the system load. Furthermore, upon correct system operation, a user program is restricted to access any resource for which it has not been given explicit permission by the kernel. Hence, it is not necessary to worry about the effects of user programs. Unfortunately, a mathematical characterization of these phenomena is not available. Most of the so called software reliability models attempt, at most, to give a figure for the Mean Time To Failure of a software system under some "typical" workload conditions. As will be seen in the following sections, the characterization of a "typical" workload is in itself an important problem.

One of the more important byproducts of considering a time varying failure rate in which failures can be due either to hardware transients or software design errors is that the relative contribution of software to system unreliability can be estimated directly from the history of system failures. From a software point of view, the model presented here is more in the line of the ideas exposed in [Littlewood 79] in the sense that the concepts of bug identification and elimination should be separated from reliability measurement. No one cares about how many bugs remain in a software system if the sytem operates at an acceptable level of performance. The modeling methodology presented in this report does not give any solution to the problem of improving software reliability (although it gives some non trivial hints about how that could be done) but gives a method to characterize the distribution of the time to failure due to software under general workload conditions.

The formal characterization of performance of a digital computing system may be very elusive. As described in [Ferrari 75], there are no known system-independent and workload-independent performance indices (two necessary properties to consider a measure a universal measure). But the average user is only concerned in the elapsed time since a computation is requested and the correct result is produced. This time will depend on the load of the system, the operating system overhead, on the probability that the system fails and his particular task has to be restarted, and of course, on the underlying hardware configuration. It is then an important problem to establish formal quantitative relationships between workload, performance, and reliability.

In summary, this report gives a solution to the mathematical characterization of the relationships between workload, performance, and reliability due to transient failures and software design errors. The mathematical analysis developed here can be applied not only to computing systems, but to any complex systems in which reliability is an important characteristic and for which some knowledge about workload variations is available. Since a large class of these systems operate under a quasi-periodic demand (such as public transportation systems, power distribution networks, time sharing and some real-time computing systems, etc.), the mathematical characterization has been developed first for systems in which the workload can be characterized by a cyclostationary stochastic process (a time varying stochastic processes with periodic mean and variance).

In Section 2 the formal assumptions made in the characterization of the failure process of a time-shared computer are stated in detail. Also, general expressions are derived for the Probability Distribution Function, Reliability Function, and Hazard Function of the times to hardware failure, software failure, and system failure when the system overhead is described by a cyclostationary process that can be approximated by a periodic function plus a "corrected" zero mean Gaussian process.

The results presented in Section 2 are elaborated in Section 3 where the failure process of a real system is studied in detail, and exact expressions are given that characterize the software, hardware, and system reliability for that partcular system. In Section 4 these results are compared with the available data regarding the reliability of the system under consideration and with the characterization that would result from more traditional models such as constant failure rate (time to failure expontially distributed), Weibull, and periodic failure rate. Finally, in Section 5, a list of items to be further investigated is proposed, along with some preliminary conclusions. Two mathematical derivations particularly tedious have been left to appendices so not to distract the reader with cumbersome details that are not relevant to the ideas presented in this report.

2. Mathematical characterization

This Section gives the mathematical basis of a model able of predicting and calibrating the unreliability of digital computers due to hardware transients and software errors. First, the necessary definitions are given in Section 2-1. The assumptions that the systems to be modelled are assumed to satisfy are stated in detail in Section 2-2. In Section 2-3 a mathematical skeleton is built based on these assumptions. The result is a general expression for the Probability Distribution Function (PDF) of the time to system failure. Finally, in Section 2.3. the general procedure for evaluating the maximum likelihood estimates of the model parameters is outlined.

2.1 Definitions

A stochastic process $\{x(t,\omega);\ t\epsilon T,\ \omega\epsilon\Omega\}$ is a family of random variables all defined in the same probability space Ω and indexed by a real parameter t that takes values in a parameter set T called the *index set* of the process. The indexing parameter t will represent time in all the processes presented in this report and T will always be equal to the real line $\mathbb{R}$, that is, only continuous time processes will be considered. For each fixed $t\epsilon\mathbb{R}$, $x(t,\omega)$ as a function of ω will be a real valued random variable. For each $\omega\epsilon\Omega$, $x(t,\omega)$ as a function of t will be a real valued function of time called a *realization* of the process. The set of all these time functions is called the *ensemble* of the process.

Definition 1: A *counting process* $\{N(t,w);\ t\geq t_0\}$ is a stochastic process having the set $\mathbb{N}^+ = \{0,1,2,..,\infty\}$ of nonnegative integers as its state space.

For each $\omega\epsilon\Omega$, $N(t,\omega)$ is a piecewise-constant function of t with jumps at $t_1(\omega), t_2(\omega),...,t_n(\omega)$, the values of t_1 ,...., t_n depending on the realization of the process. Counting processes are always associated with *point processes*, the value of $N(t,\omega)$ for $t_i\leq t<t_{i+1}$ being the total number of "points" generated up to t_{i+1}. All counting processes presented in this report will be associated to *failure processes* of a given system, the value of $N(t,\omega)$ for $t_i\leq t<t_{i+1}$ being the number of system failures detected up to t_{i+1}. A typical realization or sample function of a counting process is shown in fugure 2-1.

Definition 2: A *Poisson process* is a counting process $\{N(t)\ ;\ t\leq t_0\}$ with the following three properties :

1. $Pr[N(t_0)=0] = 1$

2. For $t_0\leq s<t$, the increment $N(s,t)=N(t)-N(s)$ is Poisson distributed with parameter $\Lambda(t)-\Lambda(s)$, where $\Lambda(t)$ is a nonnegative, nondecreasing function of t.

3. $\{N(t);t\geq t_0\}$ has independent increments.

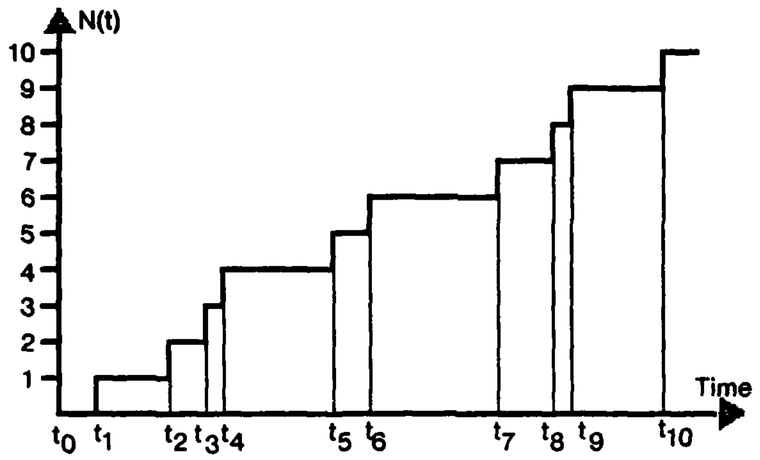


Figure 2-1: A possible sample function of a counting process.

Property 3 is the distinguishing property. It means that for a Poisson counting process, the number of points in nonoverlapping intervals are statistically independent random variables, no matter how large or small the intervals are and no matter how distant or close they may be. The function $\Lambda(t)$ in property 2 is termed the *parameter function* of the process. If $\Lambda(t)$ is an absolutely continuous function of t, it can be expressed as

$$\Lambda(t) = \int_{t_0}^{t} \lambda(\tau)\, d\tau \qquad (2.1)$$

where $\lambda(\tau)$ is a nonnegative function of t for $t \geq t_0$. The function $\lambda(\tau)$ is termed the *intensity function* of the process N(t). At any time $t \geq t_0$, the intensity function $\lambda(\tau)$ is the instantaneous average rate at which points occur. If N(t) is a failure process $\lambda(t)$ is the *failure rate* of the process.

Definition 3: A Poisson process is said to be *homogeneous* when the intensity function $\lambda(t)$ is a constant independent of time.

Definition 4: Whenever the intensity function $\lambda(t)$ is not a constant but a deterministic function of time, the corresponding Poisson process is said to be *inhomogeneous*.

Definition 5: Let x(t) be a stochastic process that is an "outside" process influencing the evolution of a counting process $\{N(t); t \geq t_0\}$. N(t) is a *doubly stochastic Poisson process* with intensity process $\{\lambda(t,x(t)); t \geq t_0\}$ if for almost every realization of the process x(t), N(t) is a Poisson process with intensity process function $\lambda(t,x(t))$.

The process x(t) carries the information about how the intensity process varies, and for this reason will be also called the *information process*.

Definition 6: A *stationary process* (in the strict sense) is a stochastic process $\{x(t), t \in T\}$ with the property that for any positive integer k and any points $t_1, \ldots, t_k$ and h in T, the joint distribution

$$\{x(t_1), \ldots, x(t_k)\}$$

is the same distribution of

$$\{x(t_1+h), \ldots, x(t_k+k)\}$$

Intuitively, a process is stationary if it has the same joint statistics regardless of where the time origin is set. Hence, if x(t) is a *stationary Gaussian process*, the joint distribution function of $\{x(t_1+h), \ldots, x(t_k+h)\}$ is a multivariate Gaussian distribution whose covariance matrix is independent of h.

Definition 7: The *Autocorrelation function* $R_{xx}(t_1,t_2)$ of a process x(t) is defined as

$$\begin{aligned} R_{xx}(t_1,t_2) &= E\{x(t_1)x(t_{t2})\} \\ &= \int p_{x(t_1),x(t_2)}(\alpha_1,\alpha_2)\, d\alpha_1\, d\alpha_2 \end{aligned}$$

where E{..} stands for expected value and $p_{x(t_1),x(t_2)}(\alpha_1,\alpha_2)$ is the joint probability density function of $x(t_1)$ and $x(t_2)$.

If x(t) is stationary and real, $R_{xx}(t_1,t_2)$ depends only on the time difference $\tau = |t_1 - t_2|$ and

$$R_{xx}(\tau) = E\{x(t+\tau)x(t)\}$$

Definition 8: A stochastic process $x(t,\omega)$ is *ergodic* in the most general sense if all its statistics can be determined from a single realization $x(t,\omega_0)$ of the process.

Loosely speaking, a process is ergodic if time averages (the only ones that can be obtained from a single realization of the process) equal ensemble averages (i.e. expected values). Obviously, ergodicity can be defined with respect to certain parameters of the process. Only ergodicity with respect to the autocorrelation function will be needed in this report, which is defined as follows :

Definition 9: A stochastic function is ergodic with respect to the autocorrelation function if

$$R_{xx}(\tau) = \lim_{T \uparrow \infty} \frac{1}{2T} \int_{-T}^{T} x(t+\tau)x(t)\, dt$$

If ergodicity of the autocorrelation function is satisfied, the autocorrelation function can be estimated by computing the above integral for a finite record of a single realization of the process x(t).

Definition 10: A real valued, continuous time stochastic process is defined to be a *cyclostationary process* with period T if and only if

1. $E\{x(t)\} = E\{x(t+T)\}$

2. $E\{x(t)x(s)\} = E\{x(t+T)x(s+T)\} \quad \forall s,t$

that is, it is a stochastic process with periodic mean and autocorrelation functions.

Definition 11: A doubly stochastic Poisson process will be said to be a *cyclostationary Poisson process* if its information process is cyclostationary.

In summary, and as a short introduction, this report summarizes the results obtained by assuming the failure processes of Time-Sharing computing systems to be characterized by cyclostationary Poisson processes.

2.2 Basic assumptions made in the characterization of failure processes

First, the behavior of failures in Time-Sharing computing systems will be characterized. Since the occurrence of failures is random, a necessary requirement to understand the process of how a lack of reliability affects the performance of a system is to find an expression for the probability density function of the time to failure.

2.2.1 Characterization of the failure process

The approach taken has been to assume that the different subsystems failure processes can be accurately modeled by cyclostationary Poisson processes. Although it is common in reliability theory to assume that failure processes are properly modeled by Poisson processes, one may well wonder why this assumption leads to good results. There are at least three reasons for characterizing failure processes with Poisson processes.

First, the conditions for a Poisson process are very likely to be valid for many physical environments. Qualitatively, these conditions can be summarized as follows:

- Two failures cannot occur simultaneously.
- At any time, there exists an instantaneous failure rate at which failures occur per unit time and such that the value of this instantaneous failure rate is independent of the past history of the system.
- the number of failures at start time is zero.

(see [Sneyder 75] for a formal proof that the above are sufficient conditions for a process to be Poisson). If this "instantaneous" failure rate is a constant, the above three conditions define a homogeneous Poisson process, for which the interarrival times are independent and exponentially distributed random variables. If the failure rate is a deterministic function of time, a nonhomogenous

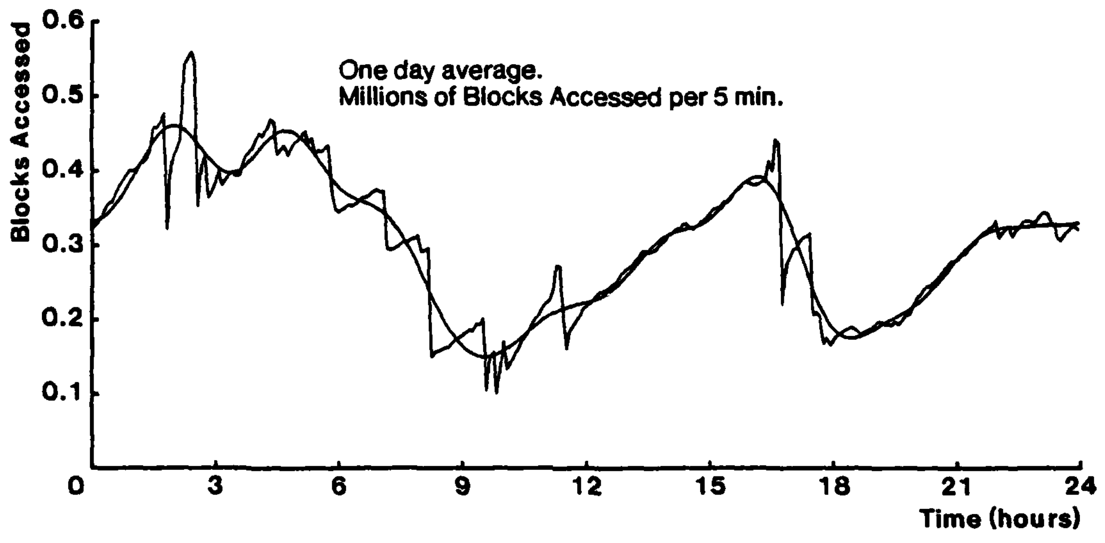


Figure 2-2: Average number of blocks accessed in the file system as a function of time of day.

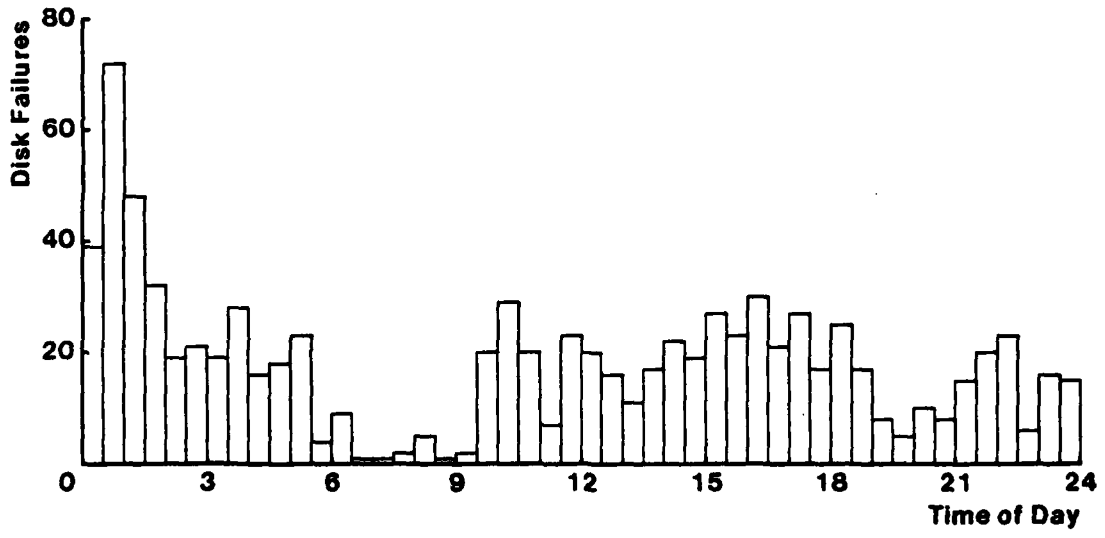


Figure 2-3: Disks failures as a function of time of day.

Poisson process is defined. Finally, if the failure rate is another stochastic process, the above three definitions define a doubly stochastic Poisson process.

The second reason for using a Poisson process is that whenever we have a point process that is the result of pooling the points of many independent point processes (whatever their characterization may be), and the component processes are sufficiently sparse, the pooled process converge to a Poisson process [Cinlar 72]. This is certainly the case of modern digital computing systems. The complexity of a minicomputer like the PDP-11/40 [Bell 78b] in a minimal configuration of 64 Kbytes of memory, clock, and a terminal interface is on the order of 10^3 IC packages. For an supercomputer like the CRAY-1 [Russell 78], the complexity is on the order of 10^5 IC packages. The average Mean Time To Failure (MTTF) per component is on the order of 10^6 hours (~ 10^3 years) for hard failures [Hodges 77]. Hence, the system failure rate due to transients is the superposition of ~10^3 failure processes, the probability of observing a failure of any of the component processes in a meaningful time interval is very small (of the order of [illegible] for a month interval). The fact that the superposition of sparse point processes converges to a Poisson process guarantees that, independently of the characterization of each of the component processes, the system failure process will be very close to a (non necessarely homogeneous) Poisson process.

Finally and most importantly, even if system characterization by means of Poisson processes is only approximate, these processes are very well understood and fairly complex mathematical tools exist.

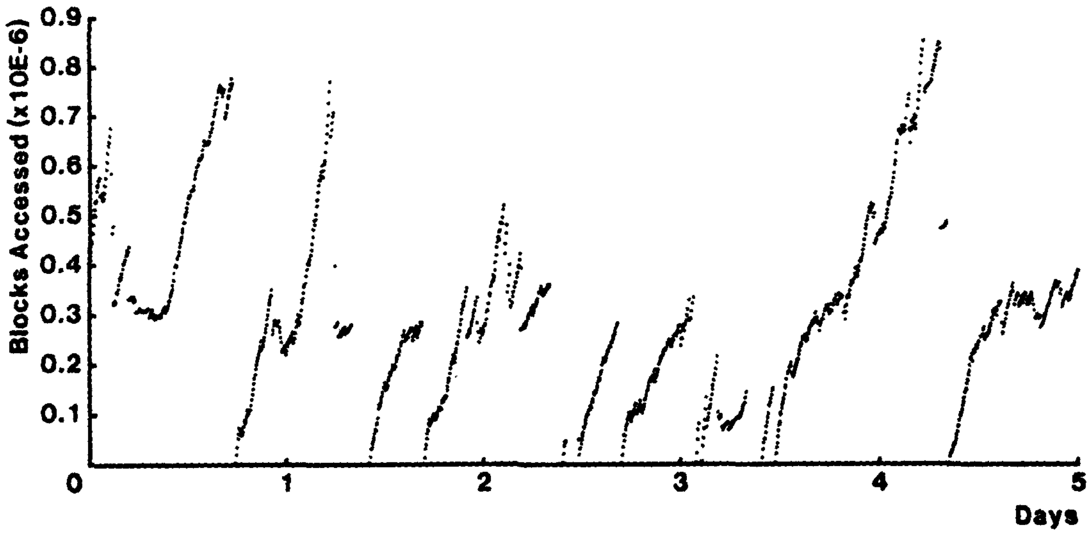


Figure 2-4: Number of blocks accessed per unit time in a file system during five consecutive weekdays (millions of blocks accessed per 5 minutes).

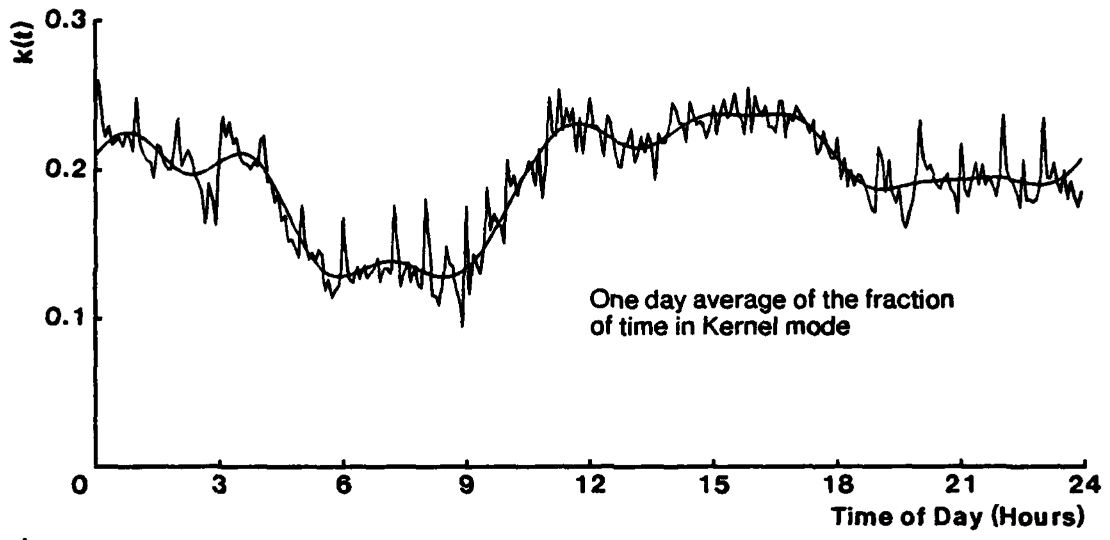


Figure 2-5: Average fraction of time in kernel mode, $\bar{k}(t)$, as a function of time of day.

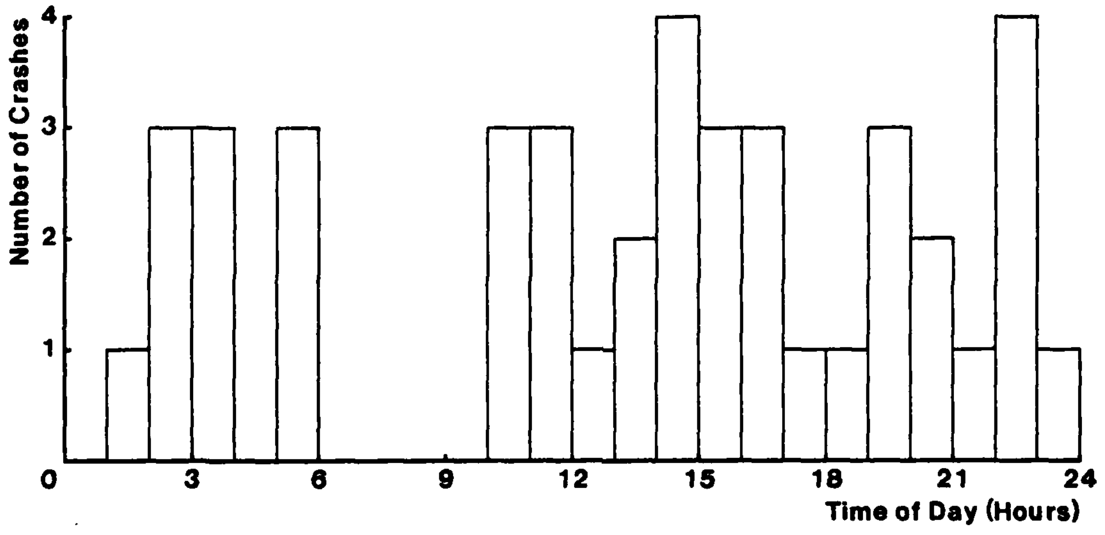


Figure 2-6: System failures (restarts) as a function of time of day.

That a doubly stochastic Poisson process should be used (that is, the failure rate being another stochastic process) is a fact suggested by the data presented in Figures 2-2 thru 2-6. Figure 2-4 shows the values of the number of blocks read and written to the file system of a time-sharing digital computing system during five consecutive weekdays. There is a clear (although nondeterministic) periodicity in this data .Note, for instance, that there is always a peak after a new day is started. This peak is due to the backup of disks to magnetic tape, which the operator does daily after midnight. If we average the data for these five days and plot the profile for the average disks use in one day we see that there is a common time varying pattern for all days (Figure 2-2). If we now examine the one day profile of disks failures detected during the same period of time in Figure 2-3 we note a remarkable similarity between the two plots. Although different, the plots in Figures 2-2 and 2-3 present the main peaks and valleys at approximately the same time of day. It seems that in the long run, after averaging over a one day period both the failure rate and the system usage variables show the same temporal behavior. If such a dependency exists instantaneoulsy, that is, if the failure rate at a given time depends on the system load at that time, it is clear that the failure rate must be characterized as a stochastic process, since the load variations presented in Figure 2-4 cannot be considered deterministic.

Figures 2-5 and 2-6 show the average fraction of time in kernel mode for a Time-Sharing system and the number of crashes detected during 29 days, both plots as a function of time of day. Again, there is some simlilarity between the two plots. The fraction of time in kernel mode for a Time Sharing system during five consecutive days, shown in Figure 2-7 suggests a cyclostationary process.

Figures 2-2 thru 2-6 should be enough evidence to justify an experiment based on the assumption that failure rate is a stochastic process. Let $\lambda(t)$ be the value of the instantaneous failure rate at time t. For a doubly stochastic Poisson process, the probability density function of the time between failures conditioned to a realization of the process $\lambda(t)$ is given by [Sneyder 75]

$$p(t \mid \lambda(\tau), 0<\tau<t) = \lambda(t)\, e^{-\int_0^t \lambda(\tau)d\tau} \qquad (2.2)$$

2.2.2 Failure rate characterization

Based on the arguments in Section 1, it will be assumed that the instantaneous value of the failure rate for a particular resource is a nondecreasing function of the "utilization" of that resource. For instance, more failures per unit time will be detected in a file system when the number of blocks read and written to the disks per unit time is near its maximum value than when it is used only occasionally. The fact that system crashes occur more often in periods of high load has been noted in [Butner 80].

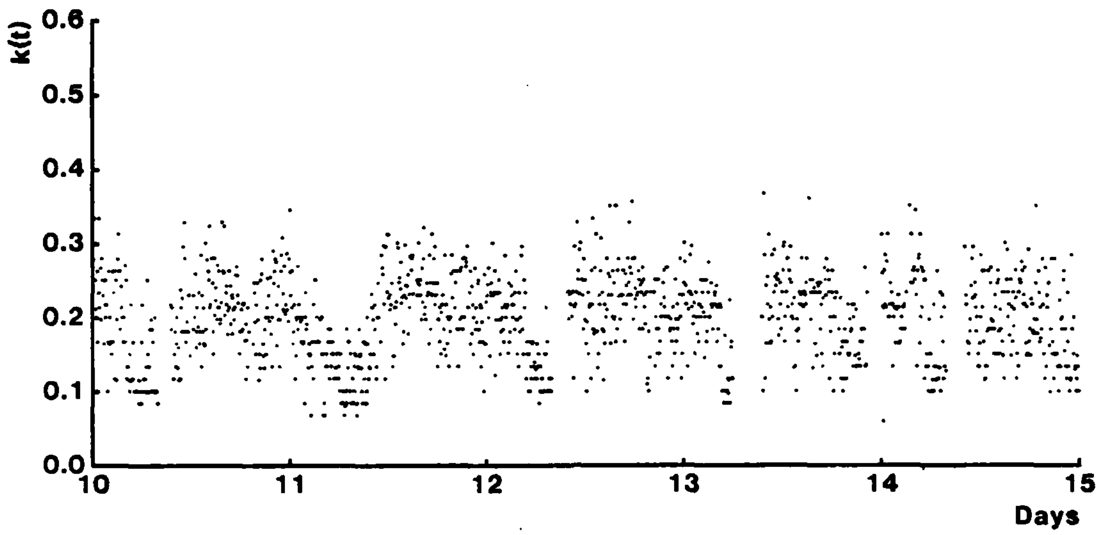


Figure 2-7: Fraction of time in kernel mode, k(t), during five consecutive weekdays.

The exact nature of the functions relating resource utilization and failure rates may be complex, different for each resource and difficult to characterize from observed data. Since no previous experience has been reported of working under these assumptions, a cautious approach will be taken and, as a first step, only linear relationships will be considered. In general then, the failure rate $\lambda(t)$ of a particular resource whose use is characterized by a function u(t) will be given by

$$\lambda(t) = a\,u(t) + b \qquad (2.3)$$

where *u(t) will be a function such as the ones shown in* Figures 2-4 or 2-7. For instance, the failure rate of a file system $\lambda_{dk}(t)$ will be given by

$$\lambda_{dk}(t) = s_{dk}\,b(t) + c_{dk} \qquad (2.4)$$

where b(t) is equal to the sum of blocks read and written to the file system per unit time as shown in Figure 2-4. s_{dk} is a sensitivity coefficient relating disks usage to failure rate and the offset term c_{dk} should take care of any possible drift in the relation between usage and failure rate.

The system failure rate, that is, the rate at which the system crashes and has to be restarted from scratch is not so obviously characterized. The protection mechanisms provided by the state of the art operating systems and computer architectures try to maintain continued system operation regardless of individual component or subsystem failures. The fact is that in most computers the CPU executes in one of several processing modes, each of the modes having different privileges respect to the overall system control. A system crash due to a hardware transient is only possible when it affects the operation of the system in the most privileged mode, the only one able of halting the entire system or entering into an infinite loop with no other entity capable of correcting the situation. This most privileged mode of operation is usually referred to as the *kernel*, and the system failure rate should be a nondecreasing function of the fraction of time that the system operates in kernel mode, that is,

$$\lambda_{sh}(t) = s_{hw} k(t) + c_{hw} \tag{2.5}$$

where $\lambda_{sh}(t)$ is the system failure rate due to hardware transients, s_{hw} is a sensitivity coefficient, k(t) is the instantaneous value of the fraction of time that the system operates in kernel mode and c_{hw} is a residual, workload independent, failure rate (even if the kernel is only slightly exercised there is the possibility that a transient in the main mamory will corrupt parts of the kernel data structures).

The system failure rate due to software errors will also depend on the fraction of time that the system operates in kernel mode because the kernel of the operating system is the only software capable of leading to a system crash. However, when the workload is very low, and the kernel executes only relatively simple operations it is to be expected that this part of the kernel will be well debugged such that the system failure rate is zero for low values of k(t). It will then be assumed that the software failure rate will be zero for values of k(t) below a threshold value k_0 and increase with k(t) above k_0. Again, the relationship between k(t) and failure rate will be assumed linear such that

$$\lambda_{ss}(t) = \begin{cases} s_{sw}\, k(t) - s_{sw} k_0 & \text{if } k(t) > k_0 \\ 0 & \text{otherwise} \end{cases} \tag{2.6}$$

where $\lambda_{ss}(t)$ is the system failure rate due to software errors, s_{sh} is its sensitivity coefficient, and k_0 is the value of k(t) below wich this failure rate is zero. $\lambda_{ss}(t)$ can be rewritten as

$$\lambda_{ss} = s_{sw}[k(t) - \bar{k}(t)] - s_{sw}k_0 \quad (2.7)$$

where

$$\bar{k}(t) = \begin{cases} k(t) & \text{if } k(t)<k_0 \\ 0 & \text{otherwise} \end{cases} \quad (2.8)$$

The following expression can then be obtained for the system failure rate, that is, the rate at which the system crashes due either to hardware transients or software errors

$$\lambda_{sy}(t) = [s_{hw} + s_{sw}]k(t) + c_{hw} - s_{sw}\bar{k}(t) - s_{sw}k_0 \quad (2.9)$$

Only these three cases (the failure process of a file system, the system failure process due to transients, and the system failure process due to software errors) will be studied in this report. Expressions for the probability density function of the time to failure and reliability function for the three cases are given in Sections 2-2 thur 2-4.

2.2.3 Workload characterization

Something more can be said about the "utilization" functions. Although being nonstationary processes, it is obvious that due to the operational policies that regulate the use of Time-Sharing systems, they will have a periodic behavior. The second hypothesis that we make is that workload, and hence system usage for time sharing systems can be modeled as a cyclostationary process [Gardner 75], [Gardner 78]. A cyclostationary process is defined as a second order process with periodic mean and autocorrelation function. The periodicity of the mean is obvious from Figure 2-4 and in fact it is possible to make the simplifying assumption that the workload causing such overhead can be described by a periodic (hence deterministic) function of time. This is the approach taken in [Butner 80], where it is expected that a periodic failure rate Poisson process will lead to a more accurate failure process characterization than a homomgeneous Poisson process model (time to failure exponentially distributed). Here, the instantaneous value of the failure rate will be considered a random variable with periodic mean, and the failure rate will be a cyclostationary process. The third hypothesis is that u(t) (the usage function of a particular system resource) can be properly modeled by adding a deterministic, period function of time m(t) plus a stationary, zero mean, Gaussian process. That is,

$$u(t) = m(t) + z(t) \tag{2.10}$$

such that in general

$$\lambda(t) = a\,m(t) + a\,z(t) + b \tag{2.11}$$

where m(t) is a periodic, deterministic function of time and z(t) is a stationary, zero mean, Gaussian process, independent of m(t). This third hypothesis, although attractive, cannot be correct. If z(t) is a purely Gaussian process, there is a non-zero probability that $\lambda(t)<0$ and the above expression cannot be used as a failure rate of a Poisson process. To avoid this problem let

$$\tilde{z}(t) = \begin{cases} m_{min} + z(t) - u_{min} & \text{if } z(t) < u_{min} - m(t) \\ 0 & \text{otherwise} \end{cases} \tag{2.12}$$

and set

$$u(t) = m(t) + z(t) - \tilde{z}(t) \tag{2.13}$$

from where we obtain

$$\lambda_s(t) = a[\, m(t) + z(t) - \tilde{z}(t) \,] + b \tag{2.14}$$

$$= a\,m(t) + b + a[z(t) - \tilde{z}(t)] \tag{2.15}$$

In summary, the three hypothesis in which this work is based are :

1. *The failure process* of a digital computing system can be [illegible]cribed by a doubly stochastic Poisson process.

2. The failure rate is a linear function of the operating system overhead (so, indirectly, depends on the system workload)

3. For computing systems with cyclostationary workload, system overhead time variations can be modeled as a periodic, deterministic function of time plus a stationary, zero mean Gaussian process, independent of the underlying periodic function and adequately corrected in order to have a positive failure rate.

Note that assumption *one* is much less restrictive than the usual assumption of considering the time between failures being exponentially distributed (i.e., the failure process is usually considered a homogeneous Poisson process). In later Sections, the insight gained in understanding system behavior from dropping this oversimplification will be discussed. Also, the implications of considering (or not considering) assumptions 2 and 3 will be discussed.

2.3 Characterization of a file system failure process

As a first application of the hypothesis described above, the failure process of a file system under cyclostationary workload will be studied in detail. The hypothesis are that the subsystem failure rate is given by

$$\lambda_{dk}(t) = s_{dk}\, b(t) + c_{dk} \tag{2.16}$$

where

$$b(t) = m_{dk}(t) + z_{dk}(t) - \tilde{z}_{dk}(t) \tag{2.17}$$

The pdf of the time to failure conditioned to a realization of the process $\lambda_{dk}(t)$ is given by

$$p_{dk}(t \mid \lambda_{dk}(\tau), t_0 \leq \tau \leq t) = \lambda_{dk}(t)\, e^{-\int_{t_0}^{t} \lambda_{dk}(\tau) d\tau} \tag{2.18}$$

The general pdf is given by

$$p_{dk}(t) = E\left\{\lambda_{dk}(t)\, e^{-\int_{t_0}^{t} \lambda_{dk}(\tau) d\tau}\right\} \tag{2.19}$$

where the expectation is taken over the ensemble realizations of the process $\lambda_{dk}(t)$. It is shown in the Appendix I that, under the assumption that $\lambda_{dk}(t)$ is given by (2.14), the above expectation is equal to,

$$p_{dk}(t) = -\frac{\partial}{\partial t}\left\{ \phi_{dk}(t)\, e^{s_{dk}^2 \frac{\sigma^2(t)}{2} \cdot [s_{dk} m_{dk} + c_{dk} \cdot \rho_{dk}(s_{dk}, b_{min})]t} \right\} \tag{2.20}$$

The meanings and values of each of the parameters on which p(t) depends are described in detail in Appendix I, and only a summary will be given here. $\phi_{dk}(t)$ is a periodic function of time, depending on the periodic component of $\lambda_{dk}(t)$. The first term in the exponent is the variance of the integral of $z_{dk}(t)$, and depends on the autocorrelation function of $z_{dk}(t)$, $R_{zz}(\tau)$. The last term depends on the mean value of the deterministic part of $\lambda_{dk}(t)$ and the correction factor $\rho_{dk}(s_{dk}, b_{min})$ takes care of the contribution of $\tilde{z}_{dk}(t)$. Finally, it should be noted that this expression is only valid when the second derivative of the autocorrelation function of $z_{dk}(t)$ at the origin is finite.

The following expression can be obtained for the Probability Distribution Function of the time between errors :

$$P_{dk}(t<\tau) = \int_0^\tau p(t)\, dt \tag{2.21}$$

$$= \phi_{dk}(0)\, e^{s_{dk}^2 \frac{\sigma^2(0)}{2}} - \phi_{dk}(\tau)\, e^{s_{dk}^2 \frac{\sigma^2(\tau)}{2} \cdot [s_{dk} m_{dk} + c_{dk} \cdot \rho_{dk}(s_{dk}, b_{min})]\tau} \tag{2.22}$$

$$= 1 - \phi_{dk}(\tau)\, e^{s_{dk}^2 \frac{\sigma^2(\tau)}{2} \cdot [s_{dk} m_{dk} \cdot \rho_{dk}(s_{dk}, b_{min})]\tau} \tag{2.23}$$

To compare our model with a real system we still need to estimate the parameters s_{dk} and c_{dk} from observed data and obtain analytical expressions for the autocorrelation function $R_{zz}(\tau)$ and the variance $\sigma^2(t)$ in equation (2.23). The general problem of parameter estimation for doubly stochastic Poisson processes is described in Section 2-5, and a numerical procedure for estimating s_{dk} is given in section .

2.4 The system failure process

The expression for the system failure rate due to hardware transients and software errors has been given in (2.9), where k(t) is the fraction of time that the system operates in kernel mode. With the hypothesis that

$$k(t) = m_{sy}(t) + z_{sy}(t) - \tilde{z}_{sy}(t) \tag{2.24}$$

(2.9) can be rewritten as

$$\lambda_{sy}(t) = [s_{sw} + s_{hw}][m_{sy}(t) + z_{sy}(t) - \tilde{z}_{sy}(t)] + c_{hw} - s_{sw}\tilde{z}'_{sy}(t) - s_{sw}k_0 \tag{2.25}$$

where

$$\tilde{z}'_{sy}(t) = \begin{cases} m_{min} + z(t) + k_0 & \text{if } z(t) < k_0 - m_{min} \\ 0 & \text{otherwise} \end{cases}$$

An additional assumption has been made here, that $k_{min} < k_0 < m_{min}$. That is, it is assumed that the value at which the failure rate due to software failures starts being nonzero (k_0) lies somewhere between the minimum value of the periodic component of k(t) (m_{min}) and the minimum value of k(t) (k_{min}). The reason for this assumption is that only in this case a closed form expression can be found for the pdf of the time to system failure. Whether this assumption holds or not in a real system is checked later in the report.

Again, the pdf of the time to system failure is given by

$$p_{sy}(t) = E\left\{ \lambda_{sy}(t)\, e^{\cdot\int_{t_0}^{t} \lambda_{sy}(\tau)\,d\tau} \right\} \tag{2.26}$$

Using the results of Appendix I, the following expression is obtained

$$p_{sy}(t) = -\frac{\partial}{\partial t}\left\{ \phi_{sy}(t)\, e^{(s_{sw}+s_{hw})^2\frac{\sigma^2(t)}{2} \cdot [(s_{sw}+s_{hw})m_{sy} + c_{hw} - \rho_{sy}(s_{sw}+s_{hw}, k_{min}) - \rho_{sy}(s_{sw}, k_0)]t} \right\} \tag{2.27}$$

and the following expression is obtained for the PDF of the time to system failure

$$P_{sy}(t<\tau) = 1 - \phi_{sy}(\tau)\, e^{(s_{sw}+s_{hw})^2 \frac{\sigma^2(\tau)}{2} [(s_{sw}+s_{hw})m_{sy} + c_{hw} \cdot \rho_{sy}(s_{sw}+s_{hw},k_{min}) \cdot \rho_{sy}(s_{sw},k_0)]\tau} \tag{2.28}$$

Again, to completely characterize the failure process of a real system, the values of s_{sw}, s_{hw}, c_{hw}, k_0 need to be estimated from the history of failures of the system.

2.5 Parameter estimation

The general problem of parameter estimation for doubly stochastic Poisson processes can be stated as follows. Let $\{N(t);t>t_0\}$ be a doubly stochastic Poisson counting process with intensity $\lambda(t,z(t),\vec{x})$, where z(t) is an stochastic process and $\vec{x} = (x_1,x_2,\ldots\ldots,x_m)$ is a vector of unknown parameters. The occurrence density function that a given realization of the process has a failure at time <u>tf</u> if it has been started at time <u>ts</u> is, given by

$$p(tf|\vec{x},z(\tau),ts<\tau<tf) = \lambda(tf,z(tf),\vec{x})\, e^{-\int_{ts}^{tf} \lambda(\tau,z(\tau),\vec{x})\,d\tau} \tag{2.29}$$

If we observe n failures at times $tf_1,\ldots\ldots,tf_n$ with associated starting times $ts_1,\ldots\ldots,ts_n$, the probability density function of observing such set of events is

$$p^{(n)}(tf_1,\ldots,tf_n|\vec{x},z(\tau),ts_i<\tau<tf_i,\forall i) = \prod_{i=1}^{n} P(ts_i)\, \lambda(tf_i,z(tf_i),\vec{x})\, e^{-\int_{ts_i}^{tf_i} \lambda(\tau,z(\tau),\vec{x})\,d\tau} \tag{2.30}$$

where $P(ts_i)$ is the a priory probability that the system is started at time ts_i. Taking the expectation with respect the statistics of z(t) we can obtain,

$$p^{(n)}(tf_1,\ldots,tf_n|\vec{x},ts_i,\ldots,ts_n) = E\left\{ \prod_{i=1}^{n} P(ts_i)\, \lambda(tf_i,z(tf_i),\vec{x})\, e^{-\int_{ts_i}^{tf_i} \lambda(\tau,z(\tau),\vec{x})\,d\tau} \right\} \tag{2.31}$$

The maximum likelihood estimate $\vec{x}' = (x'_1,x'_2,\ldots\ldots,x'_m)$ of of $\vec{x}$ in terms of a particular realization of the process is by definition the value of $\vec{x}$ that maximizes the above density function [Melsa 78]. That is, $p^{(n)}(tf_1,\ldots\ldots,tf_n|\vec{x},ts_i,i=1,\ldots\ldots,n)$ will be maximum for $\vec{x} = \vec{x}'$. In the cases presented in this report, closed form expressions have been obtained for the pdf of the time to failure. They are all of the form

$$p(t) = h(tf,\vec{x})\, e^{-H(tf,ts,\vec{x})} \qquad (2.32)$$

the function to be maximized is then

$$p^{(n)}(tf_1,\ldots\ldots,tf_n) = \prod_{i=1}^{n} P(ts_i)\, h(tf_i),\vec{x})\, e^{-H(tf_i,ts_i,\vec{x})} \qquad (2.33)$$

Note that this problem is equivalent to minimazing the function

$$l(\vec{x}) = \sum_{i=1}^{n} H(tf_i,ts_i,\vec{x}) - \sum_{i=1}^{n} \ln[h(tf_i,\vec{x})] \qquad (2.34)$$

subject to the constraints

$$h(tf_i,\vec{x})>0 \qquad i=1,\ldots\ldots,n \qquad (2.35)$$

Since closed form expressions for the components of $\vec{x}$ at the minimum are not generaly available, this is a typical nonlinear programming problem, subject to nonlinear inequality constraints. Since this problem will have to be solved every time that the failure process of a resource has to be modeled for a real system, particular care has been taken in finding an efficient procedure for the location of minimums of functions of the type (2.34). In Appendix II, this procedure is described, along with detailed descriptions of all the functions for which it has been used in the evaluation of maximum likelihood parameters.

2.6 The implications of a workload dependent model in software reliability evaluation

A general methodology for characterizing system reliability in terms of resource utilization functions has been presented in the previous sections. First, the "typical" measuring conditions have been generalized to a situation in which workload patterns are mapped into resource utilization functions, modeled by cyclostationary processes. Second, by considering the kernel of the operating system as a system resource, an integrated hardware/software reliability model has been built. The assumption is that the system failure rates due to hardware transients and software errors depend on the kernel utilization process. Third, since the functional dependencies of the failure rates due to hardware transients and software errors with respect to kernel utilization are different, it is in principle possible to evaluate the relative contribution of each failure rate to the unreliability of the total system.

Once the general functional dependency between failure rate and kernel utilization has been established, all it is needed to completely characterize a real system is to evaluate the maximum likelihood values of the function parameters. But all it is needed to evaluate the maximum likelihood values of these parameters is a history of system failures. Hence, the contribution of software to system unreliability can be evaluated just knowing the times of a set of system failures, without needing any information about how the kernel has been written, let alone how many bugs remain in it.

3. Failure process analysis of a real system

In order to verify that the assumptions stated in Section 2 lead to a better modeling of failure processes than other models, an experiment was designed. The experiment consisted in the adquisition of data concerning both the failure process and the use of a general purpose time sharing system. The system choosen was the CMU-A, a PDP-10 used by the Computer Science Department at Carnegie-Mellon University as the main general purpose computational tool. The system consists of a KL-10 processor, one megaword of memory, eight disk drives totalling 1600 megabytes of online storage and two magnetic tape drives. The system runs a slightly modified version of the standard TOPS-10 operating system [Bell 78a].

The software packages used to instrument the experiment are illustrated in Figure 3-1. Information about failures is obtained from an online error log file maintained by a system program, which records the information produced by different error formatting routines. Entries are made to this file for each hardware error detected in the system, for system reloads, for disks performance statistics, and so on [Digital 78]. The error log is later processed by SEADS, a FORTRAN package which allows to list the times of detection of errors associated with a particular resource. In order to obtain accurate information about the use of the system, a special SAIL program, SYSMON, was written that samples the values of 30 system parameters twice every five minutes, the two samples in a five minutes interval being one second apart. In this way, I/O traffic, system overhead values, etc., can be obtained averaged on a one second interval or in a five minute interval with a resolution of 5 minutes. The files generated by SYSMON are later processed by another SAIL package, READSY, which computes the periodic component and autocorrelation function of the utilization function of a particular system resource. The information generated by SEADS and READSY is then processed by an APL package (POWELL) which estimates the maximum likelihood parameters of the pdf of the time to failure of a particular resource. Finally, in a separate SAIL package, C2TST, the values predicted by the cyclostationary model and other models described in Section 4 are compared with the information stored in the error log according to a χ^2 goodness-of-fit test.

The operational policies regulating the use of this system at CMU make it a good starting point to check the validity of the ideas exposed in Section 2. Its steady state operation during weekdays can be understood from Figures 2-4 and 2-7. Recall that this figure plots the sampled values of the fraction of time considered to be operating system overhead for five consecutive weekdays. The value of the accumulated overhead time is obtained by executing a Monitor Call and includes the time spent in clock queue processing, short command processing, swapping and scheduling decisions, and software context switching [Digital 77]. This value does not include Monitor Calls execution nor I/O interrupt times. It is not exactly the time that the system is executing in kernel mode, but it is close enough for our purposes.

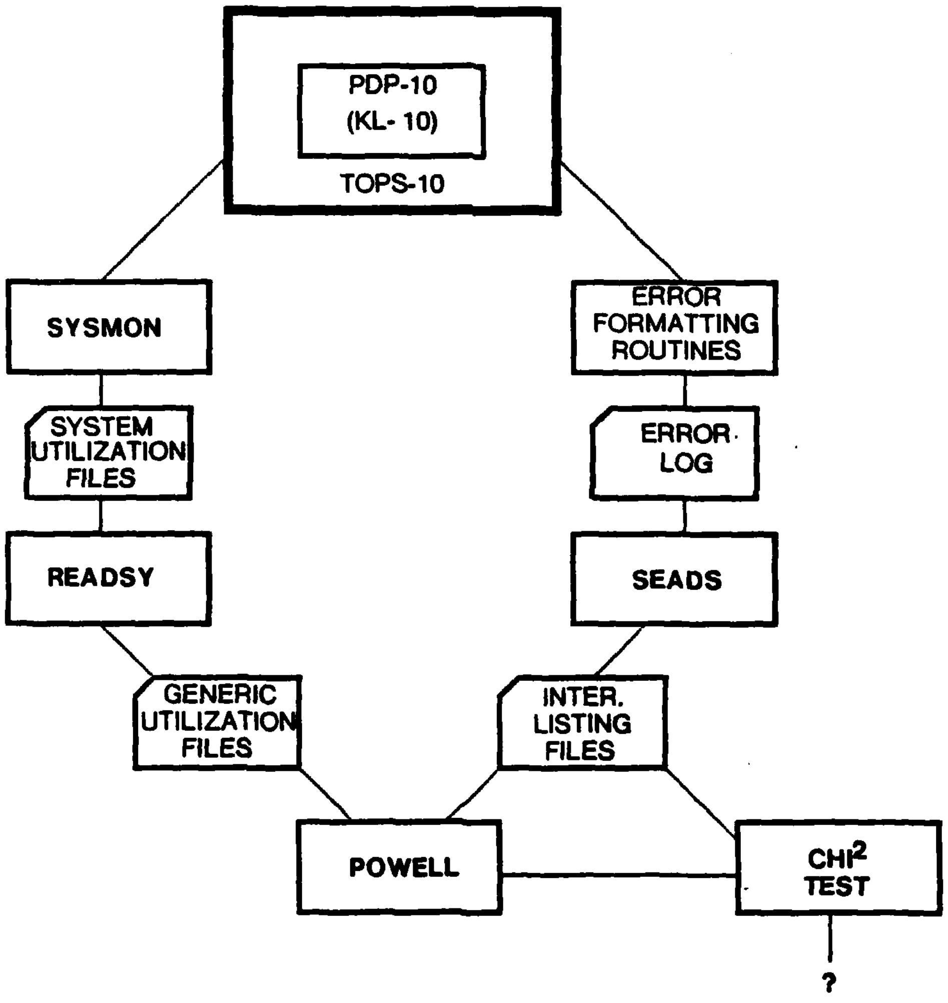


Figure 3-1: Software packages used in the validation of the cyclostationary modeling methodology.

3.1 Probability Distribution Function of the Time to Failure of a File System

Figure 2-2 shows the results of compiling five days of disk utilization samples into a single 24 hour period. Along with the estimated average, this figure shows the function $m_{dk}(t)$ obtained from a finite Fourier series expansion (see Appendix I for details). A Fourier series expansion is a least squares fit to our data and is a good way of eliminating the "noise" present in the estimated average due to the finiteness of the sample. The data in Figure 2-2 corresponds in fact to the function $m_{dk}(t)$ in Section 2.3. after sampling its values every five minutes. After substracting from b(t) the value of $m_{dk}(t)$, the sampled values of the process $z_{dk}(t)$ are avaliable for estimation of its autocorrelation function.

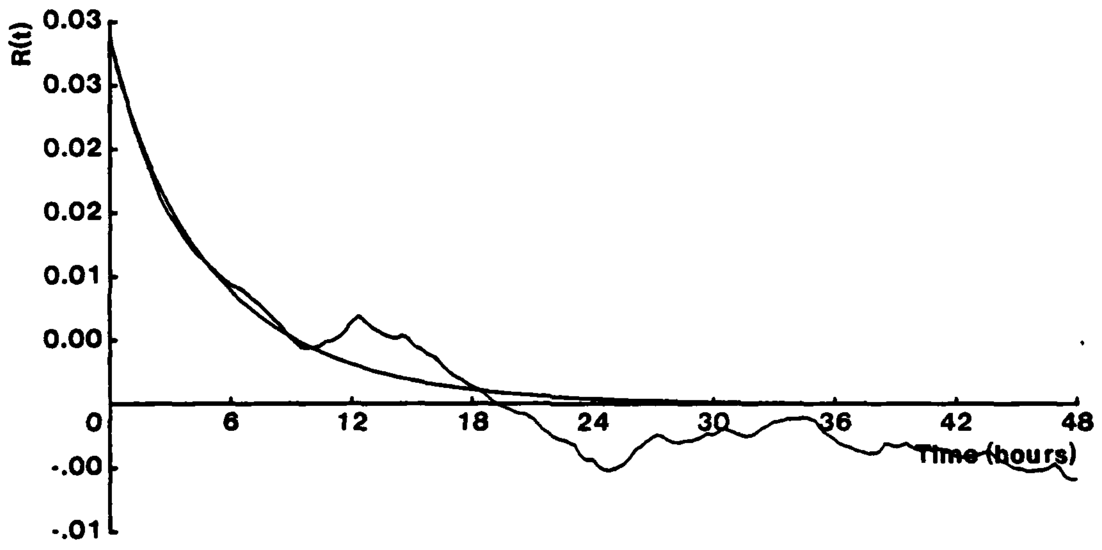


Figure 3-2: Estimated and approximated Autocorrelation functions of the file system utilization process.

Figure 3-2 shows the estimated autocorrelation function for the process $z_{dk}(t)$. From its appearance, it seems that an autocorrelation function of the form

$$R_{zz}(t) = \alpha_1 e^{-\beta_1 |t|} + \alpha_2 e^{-\beta_2 |t|} \qquad (3.1)$$

would be appropiate to approximate the real autocorrelation function. The noisy appearance of the estimated autocorrelation function is again a consequence of the finite sample size available for its computation. The main problem in the evaluation of the α_i and β_i is that they are, in principle, very sensitive to the sampling interval (in this case, 5 minutes). In the Appendix III, the exact procedure followed to evaluate them is described in detail.

With the autocorrelation function given in (3.1), the following expression is obtained for the variance $\sigma^2(t)$:

$$\sigma^2(t) = 2\alpha_1 \int_0^t (t-\tau)\, e^{-\beta_1 \tau}\, d\tau + 2\alpha_2 \int_0^t (t-\tau)\, e^{-\beta_2 \tau}\, d\tau \tag{3.2}$$

$$= 2\left[\frac{\alpha_1}{\beta_1} + \frac{\alpha_2}{\beta_2}\right]t - \frac{2\alpha_1}{\beta_1^2}\left[1 - e^{-\beta_1 t}\right] - \frac{2\alpha_2}{\beta_2^2}\left[1 - e^{-\beta_2 t}\right] \tag{3.3}$$

and substituting (3.3) in (2.23) we obtain,

$$P_{dk}(t<\tau) = 1 - \phi_{dk}(\tau)\, e^{-(\alpha_{dk}-\sigma_{dk1}-\sigma_{dk2})t - \frac{\sigma_{dk1}}{\beta_1}[1-e^{-\beta_1 t}] - \frac{\sigma_{dk2}}{\beta_2}[1-e^{-\beta_2 t}]} \tag{3.4}$$

where the following constants have been defined

$$\alpha_{dk} = s_{dk}\bar{m}_{dk} + c_{dk} - \rho_{dk}(s_{dk}, b_{min}) \tag{3.5}$$

$$\sigma_{dk1} = \frac{\alpha_1}{\beta_1} s_{dk}^2 \tag{3.6}$$

$$\sigma_{dk2} = \frac{\alpha_2}{\beta_2} s_{dk}^2 \tag{3.7}$$

The hazard function is given by

$$h_{dk}(\tau) = \alpha_{dk} - \sigma_{dk1}\left[1 - e^{-\beta_1 t}\right] - \sigma_{dk2}\left[1 - e^{-\beta_2 t}\right] - \frac{1}{\phi_{dk}(\tau)} \frac{\partial \phi_{dk}(\tau)}{\partial \tau} \tag{3.8}$$

The statistics of the time to failure for a doubly stochastic Poisson process when the intensity process is a cyclostationary process are then equivalent to the statistics of a non homogeneous Poisson process with hazard function given in (3.8). Although impressive, this hazard function reduces to a constant term plus a periodic component plus an exponentially decreasing term. Note that neglecting the periodic component, this hazard function is exponentially decreasing with the following extreme values

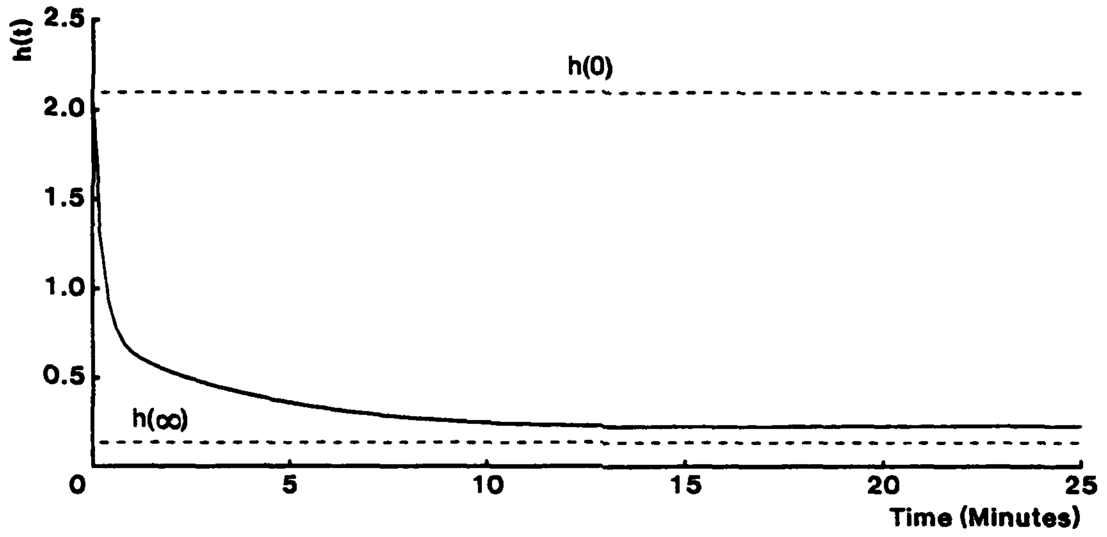


Figure 3-3: Hazard function of the equivalent non homogeneous Poisson process characterizing the statistics of the time to failure of a file system.

$$h_{dk}(0) = \alpha_{dk} \tag{3.9}$$

$$h_{dk}(\infty) = \alpha_{dk} - \sigma_{dk1} - \sigma_{dk2} \tag{3.10}$$

as shown in Figure 3-3.

3.2 The Probability Distribution Functions of the Time to System Failure

The periodic component of the kernel utilization process, $m_{sy}(t)$, has been shown in Figure 2-5. Figure 3-4 shows the autocorrelation function of the process $z_{sy}(t)$, suggesting again an approximation of the form given in (3.1). The following expression is then obtained for the PDF of the time to system failure

$$P_{sy}(t<\tau) = 1 - \phi_{sy}(\tau)\, e^{-(\alpha_{sy}-\sigma_{sy1}-\sigma_{sy2})t - \frac{\sigma_{sy1}}{\beta_1}[1-e^{-\beta_1 t}] - \frac{\sigma_{sy2}}{\beta_2}[1-e^{-\beta_2 t}]} \tag{3.11}$$

where

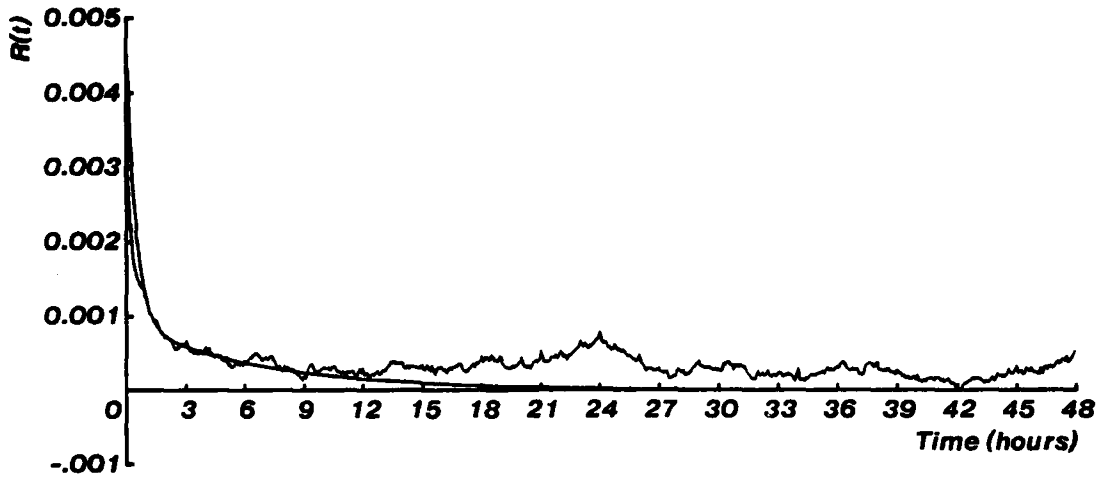


Figure 3-4: Estimated and approximated Autocorrelation functions of the kernel utilization process.

$$\alpha_{sy} = (s_{sw} + s_{hw})\dot{m}_{sy} + c_{hw} - \rho_{sy}(s_{sw} + s_{hw}, k_{min}) - \rho_{sy}(s_{sw}, k_0) \tag{3.12}$$

$$\sigma_{sy1} = \frac{\alpha_1}{\beta_1}(s_{sw} + s_{hw})^2 \tag{3.13}$$

$$\sigma_{sy2} = \frac{\alpha_2}{\beta_2}(s_{sw} + s_{hw})^2 \tag{3.14}$$

The hazard function is given by

$$h_{sy}(\tau) = \alpha_{sy} - \sigma_{sy1}\left[1 - e^{-\beta_1 t}\right] - \sigma_{sy2}\left[1 - e^{-\beta_2 t}\right] - \frac{1}{\phi_{sy}(\tau)}\frac{\partial \phi_{sy}(\tau)}{\partial \tau} \tag{3.15}$$

3.3 Simplified expressions for known starting time

All the expressions given in Sections 3.1. and 3.2. have been obtained after computing the expectation for all possible values of the starting time in a one day period. If the system starting time is known, different expressions are obtained. The only differences between the PDF of the time to failure with known starting time and the PDF averaged over a one day period is that the function $\phi(\tau)$ becomes a constant equal to one and that the α term in the exponential is slightly different. In particular, for the case of the file system failures,

$$P_{dk}(t<\tau|ts) = 1 - e^{-\alpha'_{dk} - (\alpha''_{dk} - \sigma_{dk1} - \sigma_{dk2})t - \frac{\sigma_{dk1}}{\beta_1}[1-e^{-\beta_1 t}] - \frac{\sigma_{dk2}}{\beta_2}[1-e^{-\beta_2 t}]} \tag{3.16}$$

where $P_{dk}(t<\tau|ts)$ is the probability that a failure will be detected before time $\tau + ts$ given that no failure had been detected at time ts, and

$$\alpha'_{dk} = s_{dk}[M_{dk}(\tau + ts) - M_{dk}(ts)] \tag{3.17}$$

$$\alpha''_{dk} = c_{dk} - \rho_{dk}(s_{dk}, b_{min}) \tag{3.18}$$

$$M_{dk} = \int_0^t m_{dk}(\tau)\, d\tau \tag{3.19}$$

The hazard function for known starting time is

$$h_{dk}(\tau|ts) = s_{dk}\, m_{dk}(ts + \tau) + \alpha''_{dk} - \sigma_{dk1}\left[1 - e^{-\beta_1 t}\right] - \sigma_{dk2}\left[1 - e^{-\beta_2 t}\right] \tag{3.20}$$

Similar expressions can be derived for the distribution of the time to system failure.

3.4 Distribution functions of the time to system failure due to software and of the time to system failure due to hardware transients

Once the values of s_{sw}, s_{hw}, c_{hw}, k_0 are known, it is straightforward to derive an expression for the PDF of the time to system failure due to hardware transients. Repeating the derivation described in Section 2.4. with $s_{sw} = k_0 = 0$, the following expression is obtained

$$P_{hw}(t<\tau) = 1 - \phi_{hw}(\tau)\, e^{-(\alpha_{hw} - \sigma_{hw1} - \sigma_{hw2})t - \frac{\sigma_{hw1}}{\beta_1}[1-e^{-\beta_1 t}] - \frac{\sigma_{hw2}}{\beta_2}[1-e^{-\beta_2 t}]} \tag{3.21}$$

where

$$\alpha_{hw} = s_{hw}\bar{m}_{sy} + c_{hw} - \rho_{sy}(s_{hw},k_{min}) \tag{3.22}$$

$$\sigma_{hw1} = \frac{\alpha_1}{\beta_1}s_{hw}^2 \tag{3.23}$$

$$\sigma_{hw2} = \frac{\alpha_2}{\beta_2}s_{hw}^2 \tag{3.24}$$

In the general case, to obtain the PDF of the time to system failure due to software errors, a similar equation to (3.16) would be obtained, but with the following parameters

$$\alpha_{sw} = s_{sw}\bar{m}_{sy} - \rho_{sy}(s_{sw},k_{min}) - \rho_{sy}(s_{sw},k_0) \tag{3.25}$$

$$\sigma_{sw1} = \frac{\alpha_1}{\beta_1}s_{sw}^2 \tag{3.26}$$

$$\sigma_{sw2} = \frac{\alpha_2}{\beta_2}s_{sw}^2 \tag{3.27}$$

Note that if the system failure processes due to software errors and hardware transients are considered to be nonhomogeneous Poisson processes, each with a PDF of the form (3.16), the superposition of both processes (i.e., the process obtained by adding the hazard functions of the software failure process and the hardware failure process) is *not* equal to the total system failure process, whose PDF is given by (3.11). This is because they are not statistically independent. Indeed, both failure processes have a common cause, the utilization process of the kernel of the operating system.

Table 3-1 gives the maximum likelihood values of s_{sw}, s_{hw}, c_{hw}, and k_0 for the CMU-A, along with the value of $\bar{m}_{sy}$. Note that since the value of K_0 is larger than $\bar{m}_{sy}$, expressions (3.22) thru (3.25) may not be valid. The correction term $\rho(s_{sw},k_0)$ has been computed assuming that $K_0 \lll \bar{m}_{sy}$, condition that the maximum likelihood value of K_0 does not verify. In fact, if $K_0 > \bar{m}_{sy}$ the probability density function of the time to failure due to a software error degenerates into an exponential, such that

$$P_{sw}(t<\tau) = 1 - e^{-\rho_{sy}(s_{sw},K_0)\tau} \tag{3.28}$$

the PDF of the time to system failure due to hardware transients being given in (3.21).

Figure 3-5 shows the relationship between the instantaneous value of the system failure rate and the software and hardware components. Note how the software failure rate is zero for a wide range of

Parameter	Value
s_{sw}	0.158
s_{hw}	0.086
c_{hw}	0.0079
k_0	0.225
$\bar{m}_{sy}$	0.19

Table 3-1: Maximum likelihood values of the coefficients defining the relationship between kernel utilization and system failure rate.

values of k(t), but that its slope is larger than the slope of the failure rate due to hardware errors. Figure 3-5 thus suggests that to assume a linear relationship between the system failure rate due to software errors and kernel utilization may be an oversimplification. In fact, it seems reasonable to expect the probability of observing a software error to increase with the length of time that the software is exercised *and* with a "stress" factor depending on the apparent complexity of the input data to be processed at a given time. In this case, perhaps a higher degree polynomial would better describe the relationship between the software failure rate and software utilization.

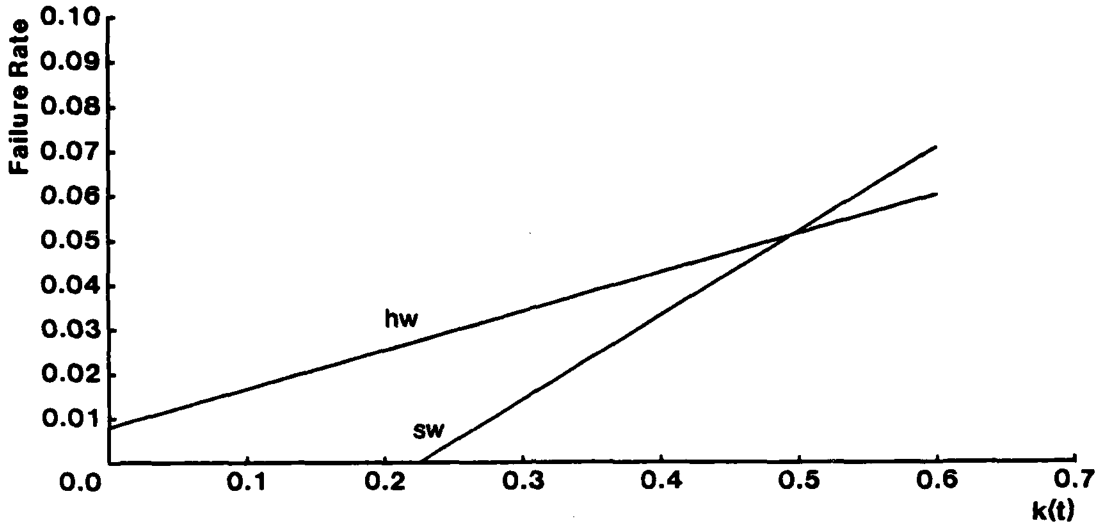


Figure 3-5: Relationship between the system failure rate and its software and hardware components.

4. Discussion

Although the assumptions made in Section 2 are less restrictve than the usual assumption of modelling the failure process with a constant failure rate, the validity of the methodology presented here can be asserted only by comparison with the behavior of a real system and contrasting the results that would be predicted by traditional models. This is the subject of Section 4.1., where the results given in Section 3 are compared with the values predicted by assuming either an exponential distribution, a Weibull distribution, and a periodic distribution for the time to failure. In section 4.2. an explanation is given for why the apparent failure rate is decreasing, and finally in Section 4.3. some preliminary conclusions are summarized.

4.1 Comparisons with other models

The more widespread model used to characterize the failure process of digital computers assumes the failure process to be a homogeneous Poisson process. The PDF of the time to failure is then given by

$$P_e(t<\tau) = 1 - e^{-\lambda_e \tau} \qquad (4.1)$$

where λ_e is the (constant) failure rate. The maximum likelihood estimate of λ_e is obtained simpy by dividing the time that the system has been operational by the number of failures reported. All functions and parameters related to this model will be noted with subindex "e" and from now on this model will be referred to as the *exponential model.*

However, empirical studies [McConnel 79a], [Wagoner 73] have shown that a Weibull distribution gives much better goodness of fit to experimental data than a simple exponential. The Weibull PDF is given by

$$P_w(t<\tau) = 1 - e^{-(\lambda_w \tau)^{\alpha_w}} \qquad (4.2)$$

The Weibull distribution is characterized by two parameters : λ_w , the scale parameter, and α_w , the shape parameter. For $\alpha_w = 1$, the Weibull distribution degenerates to the exponential. For $\alpha_w > 1$, the Weibull distribution has an increasing failure rate. A decreasing failure rate corresponds to $\alpha_w < 1$. All reports published to date claim that a decreasing failure rate Weibull distribution fits experimental data much better than a plain exponential model. Numerical procedures have been developed to find the maximum likelihood estimates of λ_w and α_w. These procedures are based on the works of [Thoman 69, Berger 74, Romano 77] and FORTRAN programs implementing them are given in [McConnel 79b].

A workload dependent model has been presented in [Butner 80]. A linear dependency between failure rate and workload is also assumed. The workload is characterized by a periodic function of time. The PDF becomes an exponential "modulated" by a periodic function

$$P_p(t<\tau) = 1 - e^{-K_p\tau} e^{-F_p U_p(\tau)} \tag{4.3}$$

where F_p is defined as the load induced failure rate and $U(\tau)$ denotes the instantaneous load value. This model will be referred to as the *periodic model*, all its parameters having the subindex "p". Using the notation developed in Section 2, this is equivalent to assume an utilization function u(t) = m(t), where only the periodic component is taken into account, and where the Gaussian process z(t) has been neglected. In this case,

$$p(t) = E\left\{\lambda_p(t)\, e^{-\int_{t_0}^{t} \lambda_p(\tau)d\tau}\right\} \tag{4.4}$$

where

$$\lambda_p(t) = s_p\, m(t) + c_p \tag{4.5}$$

and

$$P(t<\tau) = 1 - e^{-(s_p\bar{m} + c_p)\tau}\, e^{\ln \phi(\tau)} \tag{4.6}$$

Note that (4.3) and (4.6) are equivalent. In Section II.5. the equations for computing the maximum lykelihood values of s_p and c_p from a history of failures are given.

Finally, the model presented in this report will be referred as the *cyclostationary model*. An expression for its PDF is rewritten here

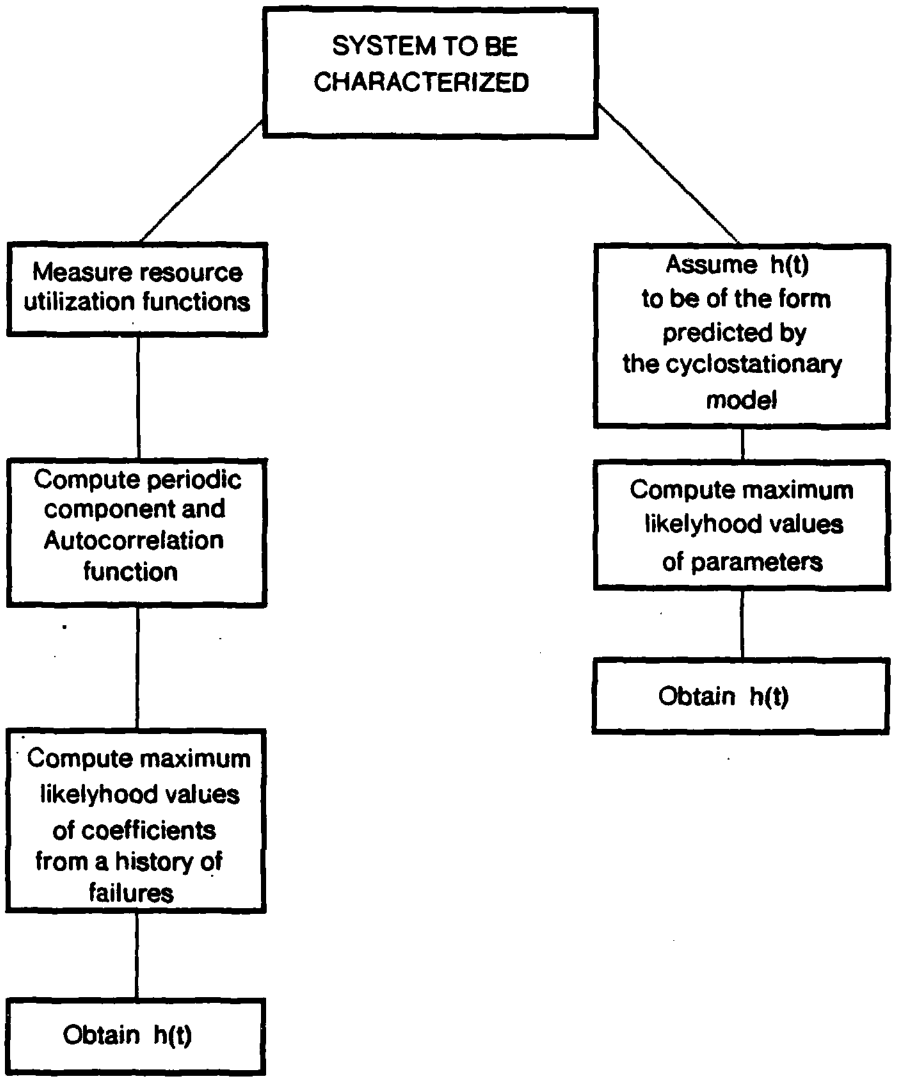

SYSTEM TO BE CHARACTERIZED
Measure resource utilization functions
Compute periodic component and Autocorrelation function
Compute maximum likelyhood values of coefficients from a history of failures
Obtain h(t)
Assume h(t) to be of the form predicted by the cyclostationary model
Compute maximum likelyhood values of parameters
Obtain h(t)

Figure 4-1: Two alternatives to characterize system reliability. The maximum likelihood values of the hazard function parameters can be evaluated from the resource utilization functions or directly from a history of failures.

$$P(t<\tau) = 1 - e^{-(\alpha_c + \sigma_{c1} + \sigma_{c2})\tau - \frac{\sigma_{c1}}{\beta_1}[1 - e^{-\beta_1 \tau}] - \frac{\sigma_{c2}}{\beta_2}[1 - e^{-\beta_2 \tau}] + \ln \phi(\tau)} \qquad (4.7)$$

Table 4.1. summarizes the densities, Reliability and Hazard functions for each of the four models. Note that the cyclostationary model has both an asymptotically decreasing failure rate and a periodic component. Qualitatively, the cyclostationary model seems to integrate the approaches of the Weibull and periodic models. Note also that, for the case of a file system failures, only two parameters need to be estimated from a history of system failures (s_{dk} and c_{dk}), the other parameters [illegible]ation (4.7) being measured from the actual system behavior (the periodic component and the autocorrelation function of the resource utilization process). Since the cyclostationary model suggests a PDF of the form shown in (4.7), it is conceivable to postulate (4.7) as the real PDF of the failure process and estimate the values of α_c, σ_{c1}, σ_{c2}, β_1, β_2 *directly* from a history of failures, therefore avoiding the measurement of the resource utilization functions. Figure 4-1 describes these two alternatives available to characterize system reliability. In Section II.3. the equations used to estimate the values of these parameters drirectly from a history of system failures are given.

The fifth distribution in Table 4.1. is a simplified version of the distribution obtained with the cyclostationary model, considering only one exponential in the hazard function, and neglecting the periodic component $\phi(\tau)$. Section II.4. gives the equations for estimating the maximum likelihood parameters of this distribution from a history of system failures.

Next, quantitative comparisons using data of a real system are in order. Table 4-2 show the results of applying a χ^2 goodness of fit test between the actual failures observed in the file system described in section 3.1. and the distributions predicted by the above four models. A χ^2 value smaller than $\chi^2_{0.05}$ (i.e., a level of confidence greater than 0.05) indicates a good fit between predicted and observed behavior and suggests the acceptance of the hypothetical distribution as the real distribution underlying the failure process.

As can be seen from Table 4-2, only the cyclostationary model (both with direct and indirect evaluation of its maximum likelihood parameters) show a good fit with experimental data. Neither the exponential nor the periodic failure rate models seem to be able to describe the failure process with significant accuracy. The simplified cyclostationary model distribution and the Weibull distibution are almost in the border line of acceptance. Further insight can be gained by direct comparison of the hazard functions of the above four models. Figure 4-2 shows the hazard function of the above four models for the case of file system failures.

Table 4-3 shows the results of applying a χ^2 goodness-of-fit test to the four models in the case of system failures. Note that for the periodic model, the maximum likelihood values of the coefficient are such that the proportionality term vanishes, and the constant term equals the λ value of the exponential model. Although all models give a level of significance larger that 0.05, the cyclostationary model is again clearly superior giving levels of significance of 0.9. The hazard functions of the four models for the case of system failures are given in Figure 4-3.

4.2 The decreasing hazard function paradox

Tha hazard function found in the cyclostationary model presents the following paradox : neglecting the periodic component, expression (3.15) means that no matter at what time we start observing a system, the statistics of the time to failure are equivalent to the statistics of a non homogeneous Poisson process with decreasing hazard function. Since the hazard function is roughly the rate at which failures will be detected, this means that no matter at which time we start observing a system the apparent rate at which failures are detected will be a decreasing funtion of time. In this section, the reason for such surprising behavior is investigated and explained.

To understand the decreasing hazard function paradox, start with the simplest possible case. Assume that the real failure rate is given by a constant plus white noise.

$$\lambda_{s1}(t) = m + x_1(t) \tag{4.18}$$

where m is the (constant) mean failure rate and $x_1(t)$ is a stationary, zero mean Gaussian process with autocorrelation function

$$R_{x_1x_1}(\tau) = \frac{W_1}{2}\delta(\tau) \tag{4.19}$$

and

$$\delta(\tau) = \lim_{h\downarrow 0}\begin{cases} 1/h & 0\leq\tau\leq h \\ 0 & \tau>h \end{cases}$$

Assume that $m>>W_1$ such that the probability of $\lambda_{s1}(t)$ being negative can be neglected. The probability density function of the time to failure is then given by

<u>Exponential</u>

$$R_e(\tau) = e^{-\lambda_e \tau} \tag{4.8}$$

$$h_e(\tau) = \lambda_e \tag{4.9}$$

<u>Weibull</u>

$$R_w(\tau) = e^{-(\lambda_w \tau)^{\alpha_w}} \tag{4.10}$$

$$h_w(\tau) = \frac{\alpha_w \lambda_w}{(\lambda_w t)^{1-\alpha_w}} \tag{4.11}$$

<u>Periodic</u>

$$R_p(\tau) = e^{-\lambda_p \tau} e^{-F_p u(\tau)} \tag{4.12}$$

$$h_p(\tau) = \left[\lambda_p + F_p \frac{\partial u(\tau)}{\partial \tau t}\right] \tag{4.13}$$

<u>Cyclostationary</u>

$$R_c(\tau) = e^{-(\lambda_c + \sigma_{c1} + \sigma_{c2})\tau - \frac{\sigma_{c1}}{\beta_1}[1 - e^{-\beta_1 \tau}] - \frac{\sigma_{c2}}{\beta_2}[1 - e^{-\beta_2 \tau}] + \ln\phi(t)} \tag{4.14}$$

$$h_c(\tau) = \lambda_c - \sigma_{c1}[1-e^{-\beta_1 \tau}] - \sigma_{c2}[1-e^{-\beta_2 \tau}] - \frac{1}{\phi(t)} \frac{\partial \phi(t)}{\partial t} \tag{4.15}$$

<u>Simplified Cyclostationary</u>

$$R_m(\tau) = e^{-(\alpha_m - \gamma_m)\tau - \frac{\gamma_m}{\beta_m}[1 - e^{-\beta_m \tau}]} \tag{4.16}$$

$$h_m(\tau) = \alpha_m - \gamma_m[1 - e^{-\beta_m \tau}] \tag{4.17}$$

Table 4-1: Reliability and Hazard functions of the five compared models.

Model	Parameter Values	Degrees of Freedom	χ^2 value	$\chi^2_{0.05}$	Level of Confidence
Exponential	$\lambda_e = 0.67$	7	130	14.067	0
Weibull	$\lambda_w = 0.91$ $\alpha_w = 0.68$	8	17.717	15.507	0.026
Periodic	$s_p = 1.25$ $c_p = 0.28$	12	1007.194	21.026	0
Cyclostat.	$s_c = 14.00$ $c_c = 2.01$	8	8.69	15.07	0.36
Cyclostat. (Direct)	$\alpha_c = 2.13$ $\sigma_{1c} = 1.42$ $\sigma_{2c} = 4.03$ $\beta_1 = 0.59$ $\beta_2 = 0.21$	6	8.642	12.592	0.19
Simplified Cyclostat.	$\alpha_c = 1.69$ $\sigma_{1c} = 1.38$ $\beta_1 = 1.38$	8	19.434	15.507	0.013

Table 4-2: Results of a χ^2 goodness-of-fit test with the Exponential, Weibull, Periodic, and Cyclostationary models for file system failures. Only the Cyclostationary model gives a level of confidence greater than 0.05. The Weibull and simplified cyclostationary models give smaller levels of confidence but close to 0.05. The hypothesis that the time to failure can be characterized with Exponential or Periodic models has to be rejected. The data used was obtained from five weekdays of system operation during which 877 (transient) failures were detected. The MTTF value is 7 minutes. The file system is composed of 8 RP06 disk drives totalling 1600 megabytes of on line storage.

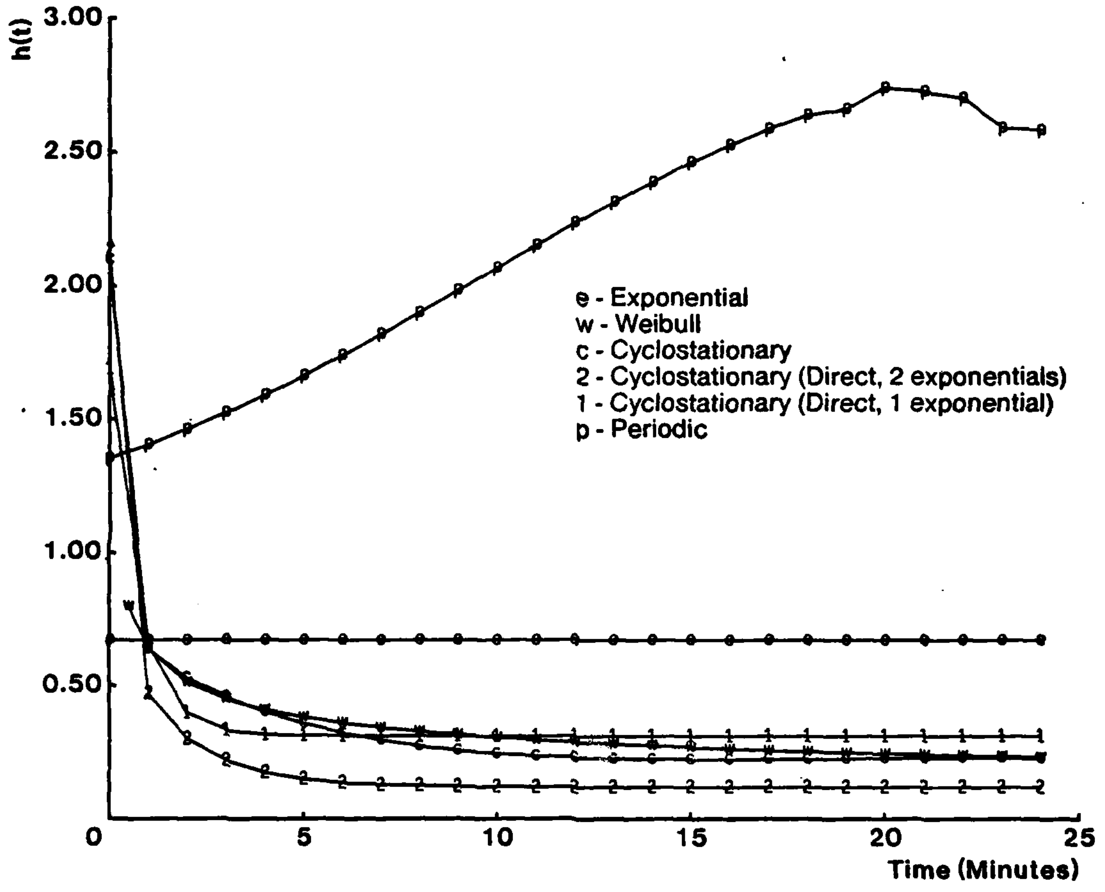


Figure 4-2: Hazard functions predicted by Exponential, Weibull, Periodic, and Cyclostationary models for file system failures.

Model	Parameter Values	Degrees of Freedom	χ^2 value	$\chi^2_{0.05}$	Level of Confidence
Exponential	$\lambda_e = 0.0073$	8	7.87	15.507	0.45
Weibull	$\lambda_w = 0.0074$ $\alpha_w = 0.98$	7	7.95	14.067	0.35
Periodic	$s_p = 0.0$ $c_p = 0.0073$	-	-	-	-
Cyclostat.	$s_{sw} = 0.158$ $s_{hw} = 0.0869$ $c_{hw} = 0.0079$ $k_0 = 0.0357$	5	1.61	11.070	0.9
Cyclostat. (Direct)	$\alpha_c = 0.013$ $\sigma_{1c} = 0.0054$ $\sigma_{2c} = 0.0080$ $\beta_1 = 0.21$ $\beta_2 = 0.0041$	5	1.66	11.070	0.9
Simplified Cyclostat.	$\alpha_c = 0.014$ $\sigma_{1c} = 0.0064$ $\beta_1 = 0.21$	6	0.75	12.592	0.9

Table 4-3: Results of a χ^2 goodness-of-fit test with the Exponential, Weibull, Periodic, and Cyclostationary models for system failures (crashes). Although all models give a level of confidence larger than 0.05, the Cyclostationary model shows a better fit to real data. Note that for the Periodic model the maximum likelihood values of the coefficients is such that the proportionality coefficient vanishes and the constant term is equal to the λ value of the Exponential model. The data used was obtained from 29 weekdays of system operation during which 60 failures were detected giving a MTTS (Mean Time To reStart) value of 11 hours.

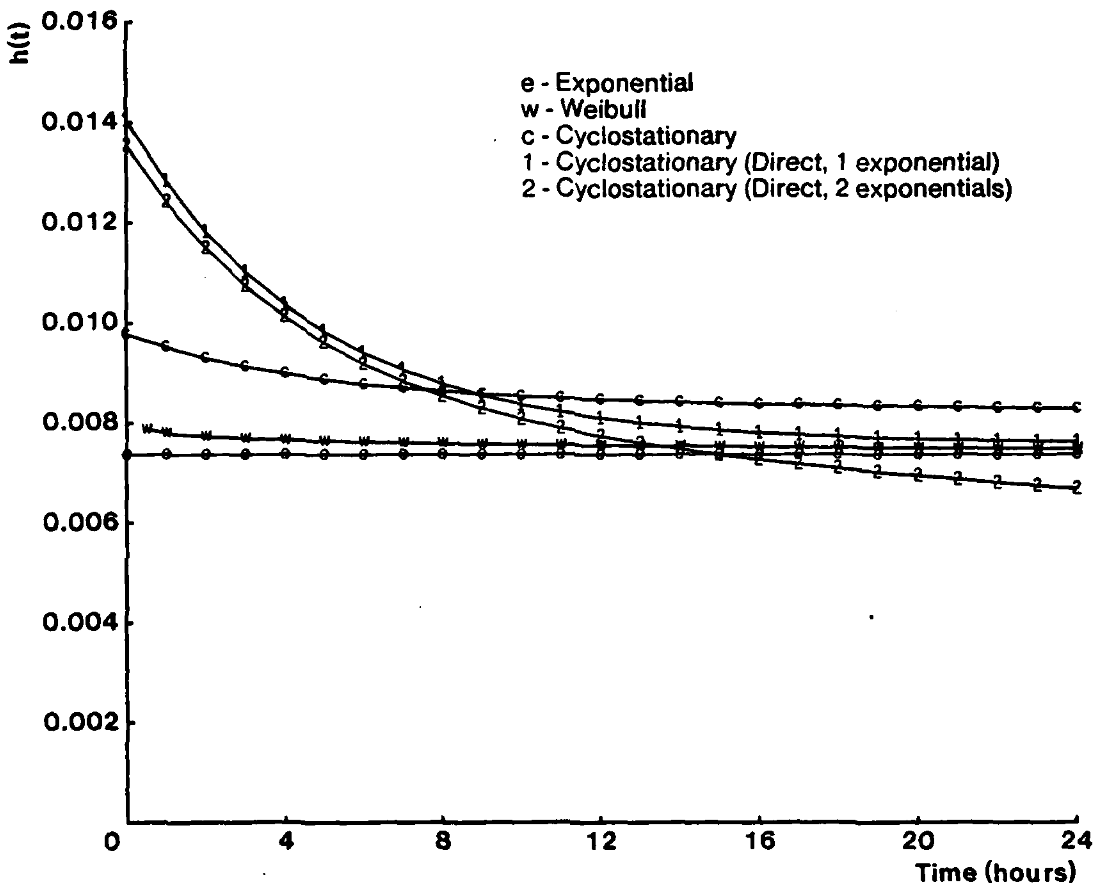


Figure 4-3: Hazard functions predicted by Exponential, Weibull, Periodic, and Cyclostationary models for system failures.

$$p_1(t) = -\frac{\partial}{\partial t}\left(e^{-mt} e^{\frac{\sigma^2(t)}{2}} \right) \tag{4.20}$$

where $\sigma^2(t)$ is the variance of the integrated process

$$\sigma^2(t) = W_1 \int_0^t (t-\tau)\, \delta(\tau)\, d\tau \tag{4.21}$$

$$= W_1 t \tag{4.22}$$

such that

$$p_1(t) = \frac{\partial}{\partial t}\left(e^{-mt} e^{W_1 t} \right) \tag{4.23}$$

$$P(t<\tau) = 1 - e^{-(m-W_1)\tau} \tag{4.24}$$

$$h_1(\tau) = m - W_1 \tag{4.25}$$

This failure process is equivalent to a homogeneous Poisson process with an apparent hazard function not equal to the mean failure rate, but equal to the difference of the mean failure rate minus the "power" of the noise, W_1. The reason for that is that the failure rate appearing in the exponent of an exponential, variations above the mean failure rate are not equally weighted with variations below the mean. In fact, the variations below the mean value are more heavily weighted, and hence the resulting smaller limiting failure rate when the expectation is taken over all possible realizations of the failure rate process.

Assume now that the real failure rate is equal to a constant plus a zero mean, stationary Gaussian process

$$\lambda_2(t) = m + x_2(t) \tag{4.26}$$

but that now the autocorrelation function of the Gaussian process is given by

$$R_{x_2x_2}(\tau) = \frac{\alpha}{2} e^{-\beta|\tau|} \tag{4.27}$$

In this case,

$$\sigma^2(t) = \frac{\alpha}{2\beta} t - \frac{\alpha}{2\beta^2}\left[1 - e^{-\beta t}\right] \tag{4.28}$$

Defining

$$W_2 = \frac{\alpha}{\beta} \tag{4.29}$$

the following expressions are obtained

$$p_2(t) = -\frac{\partial}{\partial t}\left(e^{-mt} e^{W_2 t - \frac{W_2}{\beta}[1 - e^{-\beta t}]}\right) \tag{4.30}$$

$$P_2(t<\tau) = 1 - e^{-(m-W_2)\tau - \frac{W_2}{\beta}[1 - e^{-\beta\tau}]} \tag{4.31}$$

$$h_2(\tau) = m - W_2\left[1 - e^{-\beta\tau}\right] \tag{4.32}$$

This failure process is then equivalent to a non homogeneous Poisson process with an exponentially decreasing hazard function. For $\tau = 0$, the apparent hazard function is equal to the mean real failure rate, m. For $\tau = \infty$, the apparent hazard function equals the same value that had been obtained assuming the Gausian process to be white noise. And as τ increases, the failure rate approaches this limiting value exponentially.

Finally, note that if $W_1 = W_2$, the system with white noise utilization process will be more reliable than another system having a utilization process with autocorrelation function given by (4.27). In the case of white noise, the system reaches the minimum value of its hazard function at $t = 0$, while for an autocorrelation function of the form (4.27) the minimum value is approached exponentially. Hence a non obvious way of increasing the reliability of a particular resource would be to build a system such that the utilization process of that resource approaches white noise as much as possible.

4.3 Preliminary conclusions

It has been shown how the cyclostationary model is capable of predicting the reliability of a Time-Sharing system in steady state operation. Workload dependency is stated explicitely in the model, by means of resource utilization functions. System reliability is evaluated in terms of utilization of system singularities (i.e., the Kernel of the operating system). And, in general, resource reliability is evaluated in function of the utilization of each resource.

Sofware and hardware reliability can be evaluated separately merely by observing a history of system failures and some knowledge of how the system behaves (periodic mean and autocorrelation function). A linear relationship between software failure rate and software utilization has given somewhat contradictory results, suggesting that perhaps more complex relationships need to be considered. In any case, it has been shown how establishing the relationships between software failure rate, hardware failure rate, and kernel utilization, it is in principle possible to evaluate the contribution of software and hardware to the unreliability of the total system.

Perhaps one of the more important results is that the probability density function for the simplified cyclostationary model (having a single exponential in its hazard function) has a known Laplace transform, making it suitable for Markov modelling. Neither the complete Cyclostationary, nor the Weibull, nor the Periodic models lead to probability density functions with known Laplace transforms.

From a user viewpoint, there is a reinforcement effect between workload and lack of reliability. Higher workload implies that the Kernel of the operating system has to take more decisions per unit time, increasing the probability of a system failure. Hence, not only the user receives less CPU cycles per unit time, but the probability that these cycles will become useless because the job will have to be restarted also increases.

Hence, high reliability seems to be in contradiction with other performance measures (such as the maximum number of jobs allowed to be simultaneoulsy active).

But the contradiction between reliability and other performance measures seems to be of a deeper nature. In [Spirn 77] several paging algorithms are described and modelled. Page faulting in a virtual memory system can be described as a stochastic process, and a usual optimality criteria for paging algorithms is how well they are able to perdict future system behavior given past and present system behavior. This is exactly the information given by the autocorrelation function, and in [Spirn 77] several paging algorithms are compared in terms of how well their predictions fit the autocorrelation function of a real page faulting process. But in Section 4.2. it has been shown that a way of improving reliability is to have white noise as the resource utilization process. For white noise, the future values of the process are completelly unpredictable no matter for how long has the system been observed and no matter how close in the future is the prediction desired. Hence, for an optimally reliable paging system, its utilization process should be white noise. Since future system behavior would be unpredictable, an optimum paging algorithm under these conditions would just swap out of memory pages at random, therefore lowering system performance.

5. Proposed research

5.1 On the linear dependency between overhead and failure rate

The first topic to be investigated in depth would be the real dependency between overhead and failure rate. A linear relationship has been assumed for the cyclostationary model, and any other relationship may lead to hopelessly complex mathematical problems in the evaluation of the expectation of the modified Gaussian process. However, there is always a possibility that other dependencies may be more accurate, and even if exact expressions for the distribution of the time to failure cannot be obtained for dependencies other than linear, errors due to a linear relationship assumption should be understood.

Since a failure cannot be detected if the system is not used, the only a priori assumption that seems reasonable is that the failure rate must be a non-decreasing function of the system overhead. What needs to be known is for what ranges a linear dependency is accurate, what are the confidence intervals that can be obtained, and to explore the possibility of characterizing the failure rate with relationships other than linear if necessary.

5.2 Generalization to systems showing a non-cyclostationary behavior

One of the fundamental assumptions made to develop the cyclostationary model has been that the system overhead could be approximated by adding a periodic function to a modified Gausian process. This may be a good approximation for time-sharing systems, but it certainly does not apply to many real-time and command and control systems. In fact, the highest demand for high availability systems comes from special purpose command and control systems like the ones to be installed in aircrafts, missiles, satelites, and so on. For some of these systems, the workload can be modelled by a sequence of load states. If the exact sequence is not known in advance but the possible alternatives are known, the instantaneous value of the mean workload could be modeled by a semi-markov process [Howard 71].

The cyclostationary model would then evolve to a model in which the instantaneous mean failure rate is not a periodic function of time, but a random variable whose statistics depend on the mission to accomplish. This new model would have, in addition, the ability to incorporate the effect of permanent hardware failures, transient hardware failures, and software failures. In fact, most performance-reliability models presented to date just assume that in the presence of a permanent hardware failure the system reconfigures itself and continues to operate with a possibly different computational capacity. Whether the capacity diminishes or the workload increases, the result will be a system operating in one of a set of possible states for a given period of time.

5.3 Characterizing total system performance

Present reliability evaluation tools, such as Reliability or Availability, are felt to be innadaquate due to the large gap that separates say, the Availability of a computing facility and the cost that has to be paid due to of lack of reliability. Digital computers are used to store and process information, and the sooner the desired information is available, the better the system. The occurrance of a system failure means waiting until operation is restored, bringing the machine to a consistent state, possibly restarting computations that were interrupted because of the failure and (if possible) updating the system with the information it was supposed to process while it was not operational. In short, it means a delay in obtaining the desired information and an added cost associated with the extra computations related to restoring the system to the desired state after the failure occurred.

From a single user viewpoint, a failure also means a delay. In [Castillo 80] the expected ellapsed time required to complete a program was computed under rather restrictive assumptions, but separating the "useful" time that leads to program completion from the "useless" time due to lack of reliability. Hence, a possible extension of the methodology presented in this report would be to try to caracterize the cost associated with lack of reliability from the resource utilization functions of each system resource.

5.4 System design optimization criteria derived from this model

Finally, it has been described in Section 4 how reliability seems to be in contradiction with other performance indicators. If it is assumed that the performance of a digital computer can be characterized by a vector, each component measuring a different aspect of performance (for example, throughput, execution rate, reliability, storage capacity, etc.) the arguments exposed in Section 4 seem to indicate that it is not possible to raise the value of all these components at the same time (except by enforcement of fault-intolerance in each resource). Hence, it seems in principle possible to look for the optimum performance point, as the point in which the system operates in a state in which a cost function associated with each performance measure is minimum.

Acknowledgments

I would like to thank Stephen McConnel, autor and maintainer of SEADS, for updating the software package according to the needs of the present work. Craig Everhart provided the core of SYSMON, the program with which resource utilization functions have been measured. Howard Wactlar allowed SYSMON to be running permanently on the CMU-10A for several months, even though it meant the permanent loss of one job slot in an already overloaded system. Luis Vidigal wrote the APL program implementing the Powell algorithm used to compute the maximum likelyhood values of the parameters of several functions.

I. The cyclostationary Poisson process

The problem is to obtain an expression for the pdf of the time to failure of a doubly stochastic failure process

$$p(t) = E\left\{\lambda(t)e^{-\int_0^t \lambda(\tau)\,d\tau}\right\} \tag{I.1}$$

where the time origin is assumed to be $t = 0$ and

$$\begin{aligned} \lambda(t) &= a\,u(t) + b \\ &= am(t) + b + az(t) - a\tilde{z}(t) \end{aligned} \tag{I.2}$$

m(t) being a periodic function of time, z(t) a stationary Gaussian process independent of m(t), and

$$\tilde{z}(t) = \begin{cases} z(t) & \text{if } z(t)<u_{min} \\ 0 & \text{otherwise} \end{cases} \tag{I.3}$$

Define

$$\Lambda(t) = \int_0^t \lambda(\tau)\,d\tau \tag{I.4}$$

It is shown in [Saleh 74] that since

$$\lambda(t) = \frac{\partial}{\partial t}\Lambda(t) \tag{I.5}$$

(I.1) can be rewritten as

$$p(t) = E\left\{ -\frac{\partial \Lambda(t)}{\partial t} e^{-\Lambda(t)} \right\} \tag{I.6}$$

$$= -\frac{\partial}{\partial t} E\left\{ e^{-\Lambda(t)} \right\} \tag{I.7}$$

Define

$$M(t) = \int_0^t m(t)\, dt \tag{I.8}$$

$$\mathbb{Z}(t) = \int_0^t z(t)\, dt \tag{I.9}$$

$$\tilde{\mathbb{Z}}(t) = \int_0^t \tilde{z}(\tau)\, d\tau \tag{I.10}$$

The problem reduces to evaluating the expectation

$$E\left\{ e^{-[aM(t) + a\mathbb{Z}(t) - a\tilde{\mathbb{Z}}(t) + b]} \right\} \tag{I.11}$$

or, since m(t) and z(t) are independent, the problem is equivalent to evaluate

$$E\left\{ e^{-aM(t) - bt} \right\} E\left\{ e^{-[a\mathbb{Z}(t) - a\tilde{\mathbb{Z}}(t)]} \right\} \tag{I.12}$$

I.1 The deterministic part

Let us examine now the first expectation. m(t) is a periodic function of time. However, the time origin is, in principle, unknown. The probability density function (pdf) given here is going to be compared with the estimated pdf of a real system. In an observed pdf, each system failure will be associated with a time origin corresponding to the moment at which the system was started. However, if the failure rate is a time-dependent function, it cannot be assumed that the system will be started with equal probability at any time during a one day period. In fact, the system will be started more often in the periods of time in which it fails more. Since m(t) is precisely the periodic component of the failure rate,

$$E\left\{ e^{-M(t)} \right\} = \int_0^T m_0(\upsilon)\, e^{-\int_{\upsilon}^{\upsilon+t} m(\tau)\, d\tau}\, d\upsilon \qquad (I.13)$$

whehe $m_0(t)$ is m(t) after normalizing to have area one in a one day period,

$$m_0(t) = \frac{m(t)}{\int_0^T m(\tau)\, d\tau} \qquad (I.14)$$

and T is the period of m(t).

To evaluate the expectation in (I.13), m(t) will be approximated by a finite Fourier series expansion.

$$m(t) = \bar{m} + \sum_{n=1}^{N} c_n \sin\left(n\omega t + \varphi_n\right) \qquad (I.15)$$

where the following constants have been used

$$\omega = \frac{2\pi}{T} \qquad (I.16)$$

$$\bar{m} = \frac{1}{T}\int_0^T m(t)\, dt \qquad (I.17)$$

$$c_n = (a_n^2 + b_n^2)^{1/2} \qquad (I.18)$$

$$\varphi_n = \arctan \frac{a_n}{b_n} \qquad (I.19)$$

$$a_n = \frac{2}{T}\int_0^T m(t) \cos(n\omega t)\, dt \qquad (I.20)$$

$$b_n = \frac{2}{T}\int_0^T m(t) \sin(n\omega t)\, dt \qquad (I.21)$$

Taking into account this approximation, we can obtain another representation for M(t). First, note that

$$\int_{\tau}^{\tau+t} m(\upsilon)d\upsilon = a\bar{m}\,t + \sum_{n=1}^{N} \frac{ac_n}{n\omega}\Big[\cos(n\omega\tau + \varphi_n) - \cos(n\omega(\tau+t) + \varphi_n)\Big] \qquad (I.22)$$

$$= a\bar{m}\,t + g_a(t,\tau) \qquad (I.23)$$

where $g_a(t,\tau)$ is defined as

$$g_a(t,\tau) = \sum_{n=1}^{N} \frac{ac_n}{n\omega}\Big[\cos(n\omega\tau + \varphi_n) - \cos(n\omega(\tau+t) + \varphi_n)\Big] \qquad (I.24)$$

$E\{e^{-aM(t)+bt}\}$ can now be written as

$$E\{e^{-aM(t)+bt}\} = e^{-(a\bar{m}+b)t}\int_0^T m_0(\tau)e^{-g_a(t,\tau)}d\tau \qquad (I.25)$$

$$= \phi_a(t)e^{-(a\bar{m}+b)t} \qquad (I.26)$$

where

$$\phi_a(t) = \int_0^T m_0(\tau)e^{-g_a(t,\tau)}d\tau \qquad (I.27)$$

is also a periodic function of t.

I.2 The stochastic part

The problem is now to compute

$$E\{e^{a[\tilde{\mathbb{Z}}(t)-\mathbb{Z}(t)]}\} \qquad (I.28)$$

Since z(t) is a zero mean Gaussian process, a$\mathbb{Z}$(t) will be another zero mean Gaussian process with variance (see [Papoulis 65], pp. 323-325)

$$\sigma^2(t) = 2a^2 \int_0^t \int_0^t R_{zz}(t_1,t_2)\, dt_1 dt_2 \qquad (I.29)$$

where $R_{zz}(t_1,t_2)$ is the autocorrelation function of the process z(t). If in addition z(t) is stationary,

$$\sigma^2(t) = 2a^2 \int_0^t \int_0^t (t-\tau)\, R_{zz}(\tau)\, d\tau \qquad (I.30)$$

The main problem in the evaluation of (I.28) is the evaluation of the statistics of the process $\tilde{z}(t)$ after integration, that is, the statistics of the excess area of a Gaussian process above a given level c in [0,t] (see Figures I-1 and I-2). The problem of level crossing for Gaussian processes has been extensively treated in the literature. In particular, in [Stratonovich 67] this problem is studied in detail, and expressions are given for the duration of peaks above a given level, and the excess area under such peaks. This is exactly what is required. The following is a summary of Chapter 1-3, Vol II, of [Stratonovich 67]. The derivation will only be outlined here with remarks on the assumptions used and the results obtained.

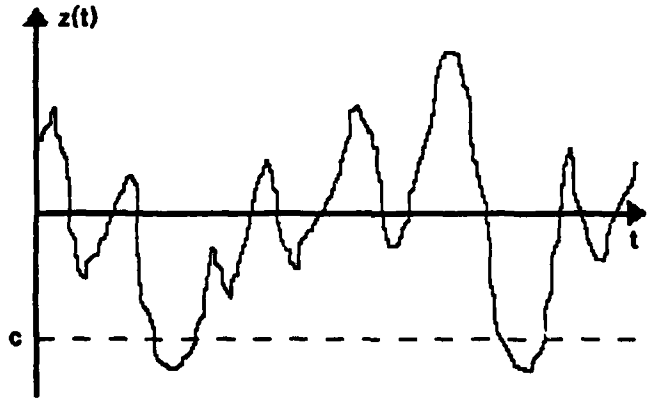


Figure I-1: A possible realization of $z(\tau)$-$\tilde{z}(\tau)$

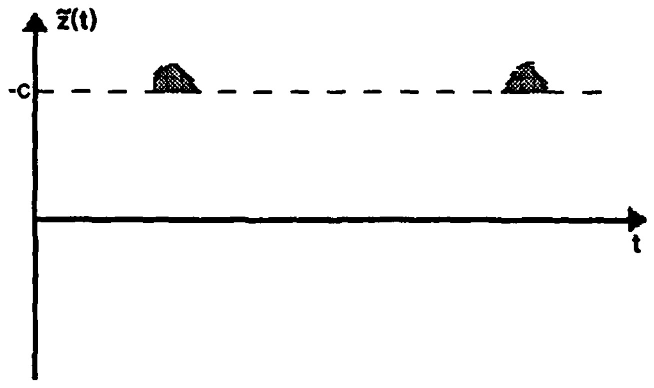


Figure I-2: A realization of $\tilde{z}(\tau)$. The shadowed area corresponds to the integral of $\tilde{z}(\tau)$ from 0 to t, $\tilde{Z}(t)$.

$\tilde{Z}(t)$ is a random variable whose exact statistics may be impossible to compute. Its value for a given realization of the process z(t) is equal to the addition of the excess areas of all peaks of z(t) above a level c. It is shown in [Stratonovich 67], p. 59, that if the duration of the peaks is much smaller than the time between peaks, the time between upcrossings (downcrossings) can be approximated by an exponentially distributed random variable. The probability of having k peaks in [0,t] is then given by

$$P(n = k) = \frac{(\eta t)^k}{k!} e^{-\eta t} \qquad (I.31)$$

where η is the mean number of peaks per unit time. In the case of a stationary Gaussian process, η is given by ([Stratonovich 67], p. 7)

$$\eta = \frac{(R_2)^{1/2}}{2} e^{-\frac{c^2}{2\sigma^2}} \qquad (I.32)$$

where

$$R_2 = -\frac{\partial^2 R_{zz}(\tau)}{\partial \tau^2}\Big]_{\tau=0} \qquad (I.33)$$

These expressions are valid only for "smooth" processes. Qualitatively, the smoothness condition means the z(t) must be differentiable and close to a straight line segment during a sufficiently small time interval. In the case of Gaussian processes, this condition means that $R_{zz}(\tau)$ must have a finite second derivative at the origin. Assuming that this condition is satisfied, it is required now to characterize the excess area under each peak.

If the duration of peaks is small and the process is smooth, the second derivative of z(t) can be considered constant over the duration of a peak, the peak can be assumed to be of parabolic shape, and the excess area depends only on the value of the first derivative of z(t) at the time of crossing the level c. Under these assumptions, the following expression can be obtained for the probability density function of the area under a peak ([Stratonovich 67], Vol II, pp. 68-72)

$$p(s) = \frac{1}{3}\left[\frac{3c^2}{2\sigma^3} (R_2)^{1/2} \right]^{2/3} s^{-1/3} e^{-\frac{1}{2}\left[\frac{3c^2}{2\sigma^3} (R_2)^{1/2} s\right]^{2/3}} \tag{I.34}$$

The value of $\tilde{\mathbb{Z}}(t)$ is given by

$$\tilde{\mathbb{Z}}(t) = s_1 + \ldots\ldots + s_k \tag{I.35}$$

where each of the s_i has density given in (I.34) and k is another random variable with density given in (I.31). An exact evaluation of (I.28) would require knowledge of the joint probability density function of $\mathbb{Z}(t)$ and $\tilde{\mathbb{Z}}(t)$. Since this impossible to obtain, the approximation will be made that

$$\tilde{\mathbb{Z}}(t) = k\,E[s] \tag{I.36}$$

where k is the number of peaks in [0,t] and E[s] is the mean peak area. In this case, $\mathbb{Z}(t)$ and $\tilde{\mathbb{Z}}(t)$ are independent random variables and

$$E\left\{ e^{a[\tilde{\mathbb{Z}}(t) - \mathbb{Z}(t)]} \right\} = E\left\{ e^{a\tilde{\mathbb{Z}}(t)} \right\} E\left\{ e^{-a\mathbb{Z}(t)} \right\} \tag{I.37}$$

Since $\mathbb{Z}(t)$ is a zero mean Gaussian variable with variance $\sigma^2(t)$ given in (I.31),

$$E\{e^{-a\mathbb{Z}(t)}\} = \frac{1}{2\pi^{1/2}\sigma(t)}\int_{-\infty}^{\infty} e^{-a\mathbb{Z}(t)}\, e^{-\frac{\mathbb{Z}^2(t)}{2\sigma^2(t)}}\, d\mathbb{Z}(t) \tag{I.38}$$

$$= e^{\frac{a^2\sigma^2(t)}{2}} \tag{I.39}$$

and

$$E\{e^{a\tilde{\mathbb{Z}}(t)}\} = \sum_{k=0}^{\infty} P(n=k)\, e^{akE[s]} \tag{I.40}$$

$$= e^{\eta(e^{aE[s]}-1)t} \tag{I.41}$$

$$= e^{\rho(a,c)t} \tag{I.42}$$

where $\rho(a,c)$ is the correction factor

$$\rho(a,c) = \eta(e^{aE[s]}-1) \tag{I.43}$$

Note that both η and E[s] depend on the value of the level c. The value of E[s] can be computed from (I.34).

$$E[s] = \frac{2^{3/2}\Gamma(5/2)}{\frac{3}{2}\left[\frac{c^2}{\sigma^3}(R_2)^{1/2}\right]} \tag{I.44}$$

$$= \frac{(2\pi)^{1/2}}{\frac{c^2}{\sigma^3}(R_2)^{1/2}} \tag{I.45}$$

If aE[s]<<<1 (that is, if the value of c is much larger than the variance σ^2), the approximation can be made that

$$e^{aE[s]} \simeq 1 + aE[s] \tag{I.46}$$

Substituting now (I.46) in (I.43)

$$\rho(a,c) \simeq a\eta E[s] \tag{I.47}$$

Substituting now (I.32) and (I.45) into (I.47) a simpler expression is obtained for $\rho(a,c)$

$$\rho(a,c) = \frac{a\,\sigma^3 (\pi/2)^{1/2}}{c^2} e^{-\frac{c^2}{2\sigma^2}} \tag{I.48}$$

where σ^2 is the variance of z(t), and c is the level below which z(t)-$\tilde{z}$(t) must vanish.

I.3 Example

If z(t) is a zero mean Gaussian process with variance σ^2 and autocorrelation function

$$R_{zz}(t) = \sigma^2 e^{-\beta|t|} \tag{I.49}$$

the expectation in (I.40) cannot be evaluated beacuse the second derivative of (I.49) at the origin does not exist. However, (I.49) can be approximated by the following equation

$$R'_{zz}(t) = \frac{\sigma^2}{1-\tau_c^2\beta^2}\left[e^{-\beta t} - \tau_c \beta e^{-\tau/\tau_c}\right] \tag{I.50}$$

provided that $\tau_c\beta \lll 1$. In this case, the variance if the integrated process becomes, according to (I.30)

$$\sigma^2(t) = 2\sigma^2 \int_0^t (t-\tau)\, e^{-\beta\tau}\, d\tau$$

$$= 2\frac{\sigma^2}{\beta}t - \frac{2\sigma^2}{\beta^2}\left[1 - e^{-\beta t}\right] \tag{I.51}$$

and the expectation (I.11) becomes

$$\phi_a(t)\, e^{-[a\bar{m} + b - \rho(a,c) - a^2\frac{\sigma^2}{\beta}]t - \frac{\sigma^2}{\beta^2}[1 - e^{-\beta t}]} \tag{I.52}$$

and $\rho(a,c)$ is given in equation (I.48). The fact that the term $(R_2)^{1/2}$ cancels in the approximated value of the correction factor $\rho(a,c)$ has an interesting physical meaning. $R'_{zz}(t)$ is the autocorrelation function of the Gaussian process that would be obtained at the output of a low-pass filter with bandiwth $1/\tau_c$ when its input is the process z(t). The fact that the value of $\rho(a,c)$ is independent of R_2 means that the area generated under the peaks of z(t) per unit time is independent of the bandwith of this filter. The area generated per peak diminishes for higher bandwith, while the number of peaks per unit time increases with the bandwith. Fortunately, these to effects cancel each other, such that the area generated under the peaks per unit time remains a constant, independent of the process bandwith.

II. Parameter Estimation

As it has been described in Section 2.5., the problem that has to be repeatedly solved in this report is that of finding the minimum of a function of the form

$$l(\vec{x}) = \sum_{i=1}^{n} H(tf_i,ts_i,\vec{x}) - \sum_{i=1}^{n} \ln[h(tf_i,\vec{x})] \tag{II.1}$$

subject to the constraints

$$h(tf_i,\vec{x})>0 \qquad i=1,\ldots\ldots,n \tag{II.2}$$

where n failures of a resource have been observed at times tf_i, after observing the system since ts_i. Take, for instance, the case of the distribution obtained from the simplified cyclostationary model presented in Section 4.1..

$$p(t) = \left[\alpha - \gamma[1 - e^{-\beta t}]\right] e^{-(\alpha-\gamma)t - \frac{\gamma}{\beta}[1 - e^{-\beta t}]} \tag{II.3}$$

Given a history of n failures represented by a set of pairs $[ts_i,tf_i]$ $i=1,\ldots\ldots,n$ the maximum likelihood values of α,γ,β are these values that maximize the function

$$p^{(n)}(tf_1-ts_1,\ldots\ldots,tf_n-ts_n) = \prod_{i=1}^{n} \left[\alpha - \gamma[1 - e^{-\beta(tf_i-ts_i)}]\right] e^{-(\alpha-\gamma)(tf_i-ts_i) - \frac{\gamma}{\beta}[1 - e^{-\beta(tf_i-ts_i)}]} \tag{II.4}$$

or, equivalently, they are the values wich minimize the function

$$l(\alpha,\gamma,\beta) = \sum_{i=1}^{n} (\alpha-\gamma)(tf_i-ts_i) + \sum_{i=1}^{n} \frac{\gamma}{\beta}[1 - e^{-\beta(tf_i-ts_i)}] - \sum_{i=1}^{n} \ln\left[\alpha - \gamma[1 - e^{-\beta(tf_i-ts_i)}]\right] \tag{II.5}$$

subject to the constraints

1) $\alpha \geq 0$

2) $\beta \geq 0$

3) $\gamma \geq 0$

4) $\alpha - \gamma > 0$ (II.6)

Although this is a typical nonlinear programming problem for which general methods are available (see, for example, [Bazaraa 79]), the problem of minimizaing functions like (II.5) present two peculiarities. First, every time that the function has to be evaluated it requires to compute the sum of n terms, n being the number of observed failures. In the case of the file system described in section 3.1., the number of observed failures in five days of system operation is 877. Hence, function evaluation (or gradient evaluation) is computationally expensive and an efficient method will be needed.

Second, although several efficient methods are known for minimization subject to nonlinear inequality constriants, these methods usually assume that the constraints are external to the mathematical statement of the problem, and that the objective function can in fact be evaluated outside the constraints. This is not the present case. The fourth constraint in (II.6) must be satisfied plainly because the objective function (II.5) does not exist unless its parameters satisfy this constraint (in the sense that the logarithm of a negative number does not exist). Indeed, the fourth constraint says that the hazard function must be positive, and a solution that does not satisfy the fourth constraint in (II.6) invalidates the existence of the objective function itself. Hence, minimization algorithms that require the evaluation of the objective function outside the constraints cannot be used.

For this reason, the first algorithm to be used to find a minimum of functions of the type (II.5) was the gradient projection method of Rosen [Rosen 60]. This algorithm follows a steepest descent direction until one or several constraints are violated, projecting then the gradient on the subspace defined by the active constraints. This method has proven to be very slow with the functions tested in this report. After some experimentation, the fastest algorithm found has been a slightly modified version of a variable metric algorithm proposed by Powell [Powell 78]. The original Powell algorithm occasionally requires the evaluation of the objective function outside the constraints and has been modified such that the maximum step size at each iteration never leads to a point outside the constraints. The modified algorithm converges more slowly that the original Powell algorithm, but for all the cases in this report, the minimum has been found in less than 30 iterations, which is a very good rate of convergence given the functions under consideration.

The algorithm has been implemented as an APL package that requires the definition of the objective function, the gradient of the objective function, the constraints and the gradients of the constraints. The following sections desfcribe each function in detail, providing a notational dictionary consistent with the programs used.

II.1 File System Failures (Cyclostationary Model)

Function to be minimized

$$l(s_{dk},c_{dk}) = \sum_{i=1}^{n} s_{dk}[M(tf_i)-M(ts_i)] + \sum_{i=1}^{n} \left[c_{dk} - \rho(s_{dk},b_{min}) - s_{dk}^2[\frac{\alpha_1}{\beta_1} - \frac{\alpha_2}{\beta_2}]\right](tf_i-ts_1)$$

$$+ \sum_{i=1}^{n} s_{dk}^2 \frac{\alpha_1}{\beta_1^2}\left[1 - e^{-\beta_1(tf_i-ts_i)}\right]$$

$$+ \sum_{i=1}^{n} s_{dk}^2 \frac{\alpha_2}{\beta_2^2}\left[1 - e^{-\beta_2(tf_i-ts_i)}\right]$$

$$- \sum_{i=1}^{n} \ln\Big[s_{dk}m(tf_i) + c_{dk} - \rho(s_{dk},b_{min})$$

$$- s_{dk}^2 \frac{\alpha_1}{\beta_1}[1 - e^{-\beta_1(tf_i-ts_i)}] - s_{dk}^2 \frac{\alpha_2}{\beta_2}[1 - e^{-\beta_2(tf_i-ts_i)}]\Big]$$

where

$$\rho_{dk}(s_{dk},b_{min}) = \frac{s_{dk}(\alpha_1+\alpha_2)^{3/2}(\pi/2)^{1/2}}{b_{min}^2} e^{-\frac{b_{min}^2}{2(\alpha_1+\alpha_2)}}$$

Definitions

$$K_{1_n} = M(tf_i)-M(ts_i) \qquad K_1 = \sum_{i=1}^{n} K_{1_i}$$

$$K_{2_i} = tf_i-ts_i \qquad K_2 = \sum_{i=1}^{n} K_{2_i}$$

$$K_{3_i} = \frac{\alpha_1}{\beta_1}\left[1 - e^{-\beta_1 K_{2_i}}\right] \qquad K_3 = \sum_{i=1}^{n} K_{3_i}$$

$$K_{4_i} = \frac{\alpha_2}{\beta_2}\left[1 - e^{-\beta_2 K_{2_i}}\right] \qquad K_4 = \sum_{i=1}^{n} K_{4_i}$$

$$K_{5_i} = m(tf_i)$$

$$K_6 = \frac{(\alpha_1+\alpha_2)^{3/2}(\pi/2)^{1/2}}{b_{min}^2} e^{-\frac{b_{min}^2}{2(\alpha_1+\alpha_2)}}$$

$x_1 = s_{dk}$

$x_2 = c_{dk}$

Objective Function

$$l(\vec{x}) = x_1K_1 + \left[x_2 - x_1K_6 - x_1^2\left[\frac{\alpha_1}{\beta_1} + \frac{\alpha_2}{\beta_2}\right]\right] K_2 + x_1^2\left[\frac{K_3}{\beta_1} + \frac{K_4}{\beta_2}\right]$$

$$- \sum_{i=1}^{n} \ln\left[x_1K_{5_i} + x_2 - x_1K_6 - x_1^2[K_{3_i} + K_{4_i}]\right]$$

Gradient

$$\frac{\partial l}{\partial x_1} = K_1 - K_6K_2 - 2x_1\left[\frac{\alpha_1}{\beta_1} + \frac{\alpha_2}{\beta_2}\right] K_2 + 2x_1\left[\frac{K_3}{\beta_1} + \frac{K_4}{\beta_2}\right]$$

$$- \sum_{i=1}^{n} \frac{K_{5_i} - K_6 - 2x_1[K_{3_i} - K_{4_i}]}{x_1K_{5_i} + x_2 - x_1K_6 - [K_{3_i} + K_{4_i}]x_1^2}$$

$$\frac{\partial l}{\partial x_2} = K_2 - \sum_{i=1}^{n} \frac{1}{x_1K_{5_i} + x_2 - x_1K_6 - [K_{3_i} + K_{4_i}]x_1^2}$$

Constraints

$C_1(\vec{x}) = x_1 > 0$

$C_2(\vec{x}) = x_2 > 0$

$C_3(\vec{x}) = x_1 \min\{K_{5_i}\} + x_2 - x_1K_6 - x_1^2[\max\{K_{3_i} + K_{4_i}\}] > 0$

Gradient of the constraints

$$\frac{\partial c_1}{\partial x_1} = 1 \qquad \frac{\partial c_1}{\partial x_2} = 0$$

$$\frac{\partial c_2}{\partial x_1} = 0 \qquad \frac{\partial c_2}{\partial x_2} = 1$$

$$\frac{\partial c_3}{\partial x_1} = \min\{K_{5_i}\} - 2x_1 \max\{K_{3_i} + K_{4_i}\} \qquad \frac{\partial c_3}{\partial x_2} = 0$$

II.2 System Failures (Cyclostationary Model)

Function to be minimized

$$l(s_{sw},s_{hw},c_{hw},k_0) = \sum_{i=1}^{n} s_{sy}[M(tf_i)-M(ts_i)]$$

$$+ \sum_{i=1}^{n} \left[c_{hw} - \rho(s_{sy},k_{min}) - \rho(s_{sw},k_0) - s_{sy}^2[\frac{\alpha_1}{\beta_1} - \frac{\alpha_2}{\beta_2}]\right](tf_i-ts_1)$$

$$+ \sum_{i=1}^{n} s_{sy}^2 \frac{\alpha_1}{\beta_1^2}\left[1 - e^{-\beta_1(tf_i-ts_i)}\right]$$

$$+ \sum_{i=1}^{n} s_{sy}^2 \frac{\alpha_2}{\beta_2^2}\left[1 - e^{-\beta_2(tf_i-ts_i)}\right]$$

$$- \sum_{i=1}^{n} \ln\left[s_{sy}m(tf_i) + c_{sy} - \rho(s_{sy},k_{min}) - \rho(s_{sw},k_0)\right.$$

$$\left. - s_{sy}^2 \frac{\alpha_1}{\beta_1}[1 - e^{-\beta_1(tf_i-ts_i)}] - s_{sy}^2 \frac{\alpha_2}{\beta_2}[1 - e^{-\beta_2(tf_i-ts_i)}]\right]$$

where

$$s_{sy} = s_{sw} + s_{hw}$$

Definitions

$$\begin{array}{ll}
K_{1_n} = M(tf_i)-M(ts_i) & K_1 = \sum_{i=1}^{n} K_{1_i} \\
K_{2_i} = tf_i-ts_i & K_2 = \sum_{i=1}^{n} K_{2_i} \\
K_{3_i} = \frac{\alpha_1}{\beta_1}\left[1 - e^{-\beta_1 K_{2_i}}\right] & K_3 = \sum_{i=1}^{n} K_{3_i} \\
K_{4_i} = \frac{\alpha_2}{\beta_2}\left[1 - e^{-\beta_2 K_{2_i}}\right] & K_4 = \sum_{i=1}^{n} K_{4_i} \\
K_{5_i} = m(tf_i) & \\
x_1 = s_{sw} &
\end{array}$$

$x_2 = s_{hw}$

$x_3 = c_{hw}$

$x_4 = k_0$

K_6 as defined in Section II.1.

$$K_7 = \frac{(\alpha_1+\alpha_2)^{3/2}(\pi/2)^{1/2}}{x_4^2} \quad e^{-\frac{x_4^2}{2(\alpha_1+\alpha_2)}}$$

Objective Function

$$I(\vec{x}) = (x_1+x_2)K_1 + \Big[x_3 - (x_1 + x_2)K_6 - x_1K_7 - (x_1+x_2)^2\Big[\frac{\alpha_1}{\beta_1} + \frac{\alpha_2}{\beta_2}\Big]\Big]K_2$$

$$+ (x_1+x_2)^2\Big[\frac{K_3}{\beta_1} + \frac{K_4}{\beta_2}\Big]$$

$$-\sum_{i=1}^{n} \ln\Big[(x_1+x_2)K_{5_i} + x_3 - (x_1+x_2)K_6 - x_1K_7 - (x_1+x_2)^2[K_{3_i} + K_{4_i}]\Big]$$

Gradient

$$\frac{\partial I}{\partial x_1} = K_1 - K_6K_2 - K_7K_2 - 2(x_1+x_2)\Big[\frac{\alpha_1}{\beta_1} + \frac{\alpha_2}{\beta_2}\Big]K_2 + 2(x_1+x_2)\Big[\frac{K_3}{\beta_1} + \frac{K_4}{\beta_2}\Big]$$

$$-\sum_{i=1}^{n} \frac{K_{5_i} - K_6 - K_7 - 2x_1[K_{3_i} + K_{4_i}]}{(x_1+x_2)K_{5_i} + x_3 - (x_1+x_2)K_6 - x_1K_7 - [K_{3_i}+K_{4_i}](x_1+x_2)^2}$$

$$\frac{\partial I}{\partial x_2} = K_1 - K_6K_2 - 2(x_1+x_2)\Big[\frac{\alpha_1}{\beta_1} + \frac{\alpha_2}{\beta_2}\Big]K_2 + 2(x_1+x_2)\Big[\frac{K_3}{\beta_1} + \frac{K_4}{\beta_2}\Big]$$

$$-\sum_{i=1}^{n} \frac{K_{5_i} - K_6 - 2x_1[K_{3_i} + K_{4_i}]}{(x_1+x_2)K_{5_i} + x_3 - (x_1+x_2)K_6 - x_1K_7 - [K_{3_i}+K_{4_i}](x_1+x_2)^2}$$

$$\frac{\partial l}{\partial x_3} = K_2 - \sum_{i=1}^{n} \frac{1}{(x_1+x_2)K_{5_i} + x_3 - (x_1+x_2)K_6 - x_1K_7 - [K_{3_i}+K_{4_i}](x_1+x_2)^2}$$

$$\frac{\partial l}{\partial x_4} = -x_1K_7[\frac{2}{x_4} + \frac{x_4}{\alpha_1+\alpha_2}] + \sum_{i=1}^{n} \frac{x_1K_7[(2/x_4)+(x_4/(\alpha_1+\alpha_2))]}{(x_1+x_2)K_{5_i} + x_3 - (x_1+x_2)K_6 - x_1K_7 - [K_{3_i}+K_{4_i}](x_1+x_2)^2}$$

Constraints

$$C_1(\vec{x}) = x_1 > 0$$

$$C_2(\vec{x}) = x_2 > 0$$

$$C_3(\vec{x}) = x_3 > 0$$

$$C_4(\vec{x}) = x_4 > 0$$

$$C_5(\vec{x}) = (x_1+x_2)\min\{K_{5_i}\} + x_2 - (x_1+x_2)K_6 - x_1K_7 - (x_1+x_2)^2[\max\{K_{3_i}+K_{4_i}\}] > 0$$

Gradient of the constraints

$$\frac{\partial c_i}{\partial x_j} = \delta_{ij} \qquad i,j = 1,\ldots,4$$

$$\frac{\partial c_5}{\partial x_1} = \min\{K_{5_i}\} - K_6 - K_7 - 2x_1 \max\{K_{3_i}+K_{4_i}\}$$

$$\frac{\partial c_5}{\partial x_2} = \min\{K_{5_i}\} - K_6 - 2x_1 \max\{K_{3_i}+K_{4_i}\}$$

$$\frac{\partial c_5}{\partial x_3} = 1$$

$$\frac{\partial c_5}{\partial x_4} = -x_1K_7[\frac{2}{x_4} + \frac{x_4}{\alpha_1+\alpha_2}]$$

II.3 Cyclostationary model - Direct Method

Function to be minimized

$$I(\alpha,\sigma_1,\beta_1,\sigma_2,\beta_2) = \sum_{i=1}^{n} (\alpha-\sigma_1-\sigma_2)(tf_i-ts_i)$$

$$+ \sum_{i=1}^{n} \frac{\sigma_1}{\beta_1}[1 - e^{-\beta_1(tf_i-ts_i)}] + \sum_{i=1}^{n} \frac{\sigma_2}{\beta_2}[1 - e^{-\beta_2(tf_i-ts_i)}]$$

$$- \sum_{i=1}^{n} \ln\left[\alpha - \sigma_1[1 - e^{-\beta_1(tf_i-ts_i)}] - \sigma_2[1 - e^{-\beta_2(tf_i-ts_i)}]\right]$$

Definitions

$x_1 = \alpha \qquad x_2 = \sigma_1 \qquad x_3 = \beta_1 \qquad x_4 = \sigma_2 \qquad x_5 = \beta_2$

K_{2_i} and K_2 as in section II.1.

Objective Function

$$I(\vec{x}) = (x_1-x_2-x_4)K_2 + \frac{x_2}{x_3}\sum_{i=1}^{n} [1 - e^{-x_3K_{2_i}}] + \frac{x_4}{x_5}\sum_{i=1}^{n} [1 - e^{-x_5K_{2_i}}]$$

$$- \sum_{i=1}^{n} \ln\left[x_1 - x_2[1 - e^{-x_3K_{2_i}}] - x_4[1 - e^{-x_5K_{2_i}}]\right]$$

Gradient

$$\frac{\partial I}{\partial x_1} = K_2 - \sum_{i=1}^{n} \frac{1}{x_1 - x_2[1 - e^{-x_3K_{2_i}}] - x_4[1 - e^{-x_5K_{2_i}}]}$$

$$\frac{\partial I}{\partial x_2} = - K_2 - \frac{1}{x_3}\sum_{i=1}^{n} [1 - e^{-x_3K_{2_i}}] + \sum_{i=1}^{n} \frac{1 - e^{-x_3K_{2_i}}}{x_1 - x_2[1 - e^{-x_3K_{2_i}}] - x_4[1 - e^{-x_5K_{2_i}}]}$$

$$\frac{\partial l}{\partial x_3} = \frac{x_2}{x_3}\sum_{i=1}^{n} K_{2_i}\, e^{-x_3 K_{2_i}} - \frac{x_2}{x_3^2}\sum_{i=1}^{n}[1 - e^{-x_3 K_{2_i}}]$$

$$+ \sum_{i=1}^{n} \frac{x_2 K_{2_i} e^{-x_3 K_{2_i}}}{x_1 - x_2[1 - e^{-x_3 K_{2_i}}] - x_4[1 - e^{-x_5 K_{2_i}}]}$$

Constraints

$C_1(\vec{x}) = x_1 > 0$

$C_2(\vec{x}) = x_1 - x_2 > 0$

$C_3(\vec{x}) = x_3 > 0$

Gradient of the constraints

$$\frac{\partial c_1}{\partial x_1} = 1 \qquad \frac{\partial c_1}{\partial x_2} = 0 \qquad \frac{\partial c_1}{\partial x_3} = 0$$

$$\frac{\partial c_2}{\partial x_1} = 1 \qquad \frac{\partial c_2}{\partial x_2} = -1 \qquad \frac{\partial c_2}{\partial x_3} = 0$$

$$\frac{\partial c_3}{\partial x_1} = 0 \qquad \frac{\partial c_3}{\partial x_2} = 0 \qquad \frac{\partial c_3}{\partial x_3} = 1$$

II.4 Simplified Cyclostationary Model

Function to be minimized

$$l(\alpha,\gamma,\beta) = \sum_{i=1}^{n} (\alpha-\gamma)(tf_i-ts_i) + \sum_{i=1}^{n} \frac{\gamma}{\beta}[1 - e^{-\beta(tf_i-ts_i)}] - \sum_{i=1}^{n} \ln\left[\alpha - \gamma[1 - e^{-\beta(tf_i-ts_i)}]\right]$$

Definitions

$x_1 = \alpha \qquad x_2 = \gamma \qquad x_3 = \beta$

K_{2_i} and K_2 as in section II.1.

Objective Function

$$l(\vec{x}) = (x_1-x_2)K_2 + \frac{x_2}{x_3}\sum_{i=1}^{n} [1 - e^{-x_3K_{2_i}}] - \sum_{i=1}^{n} \ln\left[x_1 - x_2[1 - e^{-x_3K_{2_i}}]\right]$$

Gradient

$$\frac{\partial l}{\partial x_1} = K_2 - \sum_{i=1}^{n} \frac{1}{x_1 - x_2[1 - e^{-x_3K_{2_i}}]}$$

$$\frac{\partial l}{\partial x_2} = -K_2 - \frac{1}{x_3}\sum_{i=1}^{n} [1 - e^{-x_3K_{2_i}}] + \sum_{i=1}^{n} \frac{1 - e^{-x_3K_{2_i}}}{x_1 - x_2[1 - e^{-x_3K_{2_i}}]}$$

$$\frac{\partial l}{\partial x_3} = \frac{x_2}{x_3}\sum_{i=1}^{n} K_{2_i} e^{-x_3K_{2_i}} - \frac{x_2}{x_3^2}\sum_{i=1}^{n} [1 - e^{-x_3K_{2_i}}] + \sum_{i=1}^{n} \frac{x_2K_{2_i} e^{-x_3K_{2_i}}}{x_1 - x_2[1 - e^{-x_3K_{2_i}}]}$$

Constraints

$C_1(\vec{x}) = x_1 > 0$

$C_2(\vec{x}) = x_1 - x_2 > 0$

$C_3(\vec{x}) = x_3 > 0$

Gradient of the constraints

$$\frac{\partial c_1}{\partial x_1} = 1 \qquad \frac{\partial c_1}{\partial x_2} = 0 \qquad \frac{\partial c_1}{\partial x_3} = 0$$

$$\frac{\partial c_2}{\partial x_1} = 1 \qquad \frac{\partial c_2}{\partial x_2} = -1 \qquad \frac{\partial c_2}{\partial x_3} = 0 \; \frac{\partial c_3}{\partial x_1} = 0 \qquad \frac{\partial c_3}{\partial x_2} = 0 \qquad \frac{\partial c_3}{\partial x_3} = 1$$

II.5 Periodic Failure Rate

Function to be minimized

$$I(s_p,c_p) = \sum_{i=1}^{n} s_p[M(tf_i)-M(ts_i)] + \sum_{i=1}^{n} c_p(tf_i-ts_i) - \sum_{i=1}^{n} \ln\left[s_p m(tf_i) + c_p\right]$$

Definitions

$x_1 = s_p$

$x_2 = c_p$

K_1, K_2, and K_{5_i} as defined in Section II.1

Objective Function

$$I(\vec{x}) = x_1K_1 + x_2K_2 - \sum_{i=1}^{n} \ln\left[x_1K_{5_i} + x_2\right]$$

Gradient

$$\frac{\partial I}{\partial x_1} = K_1 - \sum_{i=1}^{n} \frac{K_{5_i}}{x_1K_{5_i} + x_2}$$

$$\frac{\partial I}{\partial x_2} = K_2 - \sum_{i=1}^{n} \frac{1}{x_1K_{5_i} + x_2}$$

Constraints

$$C_1(\vec{x}) = x_1 \min\{K_{5_i}\} + x_2 > 0$$

Gradient of the Constraints

$$\frac{\partial c_1}{\partial x_1} = \min\{K_{5_i}\} \qquad \frac{\partial c_1}{\partial x_2} = 1$$

III. Autocorrelation function estimation

Given an ergodic and stationary process z(t), the problem is to estimate the function

$$R_{zz}(\tau) = \lim_{T\uparrow\infty} \frac{1}{2T} \int_{-T}^{T} z(t+\tau)z(t)\,dt \qquad \text{(III.1)}$$

For a finite record of observed values z(n), the autocorrelation function is usually estimated using the expression

$$R_{zz}(n) = \frac{1}{N} \sum_{i=1}^{N-n} z(i+n)z(i) \qquad \text{(III.2)}$$

This estimate is intuitive except for the factor 1/n. Since N-n terms are summed, it seems that 1/(N-n) would be more exact. In fact (III.2) is a biased estimator of the real autocorrelation function. However, its expected error is smaller than the expected error that would be obtained using the (unbiased) estimator with factor 1/(N-n) [Jenkins 68].

In the cases presented in this report the values of z(n) are not directly observable. In the case of sampling the values of fraction of time in Kernel mode, what was measured was the average fraction of time in Kernel mode during the last second, recording a sample every five minutes. In the case of the number of blocks accessed to the file system, the available samples are the number of blocks accessed during the last five minutes, also with a resolution of five minutes. The measured values are not the values of z(n), but the values of the process

$$z'(n) = \int_{n-\Delta}^{n} z(t)\,dt \qquad \text{(III.3)}$$

where Δ equals five minutes or one second, and the available samples of z'(n) are five minutes appart.

It has been observed that in the two cases studied in this report, the autocorrelation function suggests an approximation of the form

$$R_{zz}(t) = \alpha_1 e^{-\beta_1|t|} + \alpha_2 e^{-\beta_2|t|} \qquad \text{(III.4)}$$

The problem is then to estimate the values of the α_i, β_i from the observed values of z'(n). If

$$R_{z'z'}(t) = \alpha'_1 e^{-\beta_1|t|} + \alpha'_2 e^{-\beta_2|t|} \tag{III.5}$$

it is easy to show that

$$R_{zz}(t) = \alpha_1 e^{-\beta_1|t|} + \alpha_2 e^{-\beta_2|t|} \tag{III.6}$$

where

$$\alpha_i = \frac{\beta_i^2 \alpha_i'}{2[\cosh(\beta_i \Delta) - 1]} \tag{III.7}$$

The problem is then to estimate the values of the α'_i, β_i using (III.2) and the observed values of z'(n), and use (III.7) to obtain the values of α_i of the autocorrelation function of z(t).

Unfortunately it has not been possible to follow this procedure. The accuracy of the estimated autocorrelation function is limited basicly by two factors : the sampling frequency and the length of the available record, N. Although many techniques exist for power spectrum estimation that take into account these two factors [Oppenheim 75] (the power spectrum is the Fourier transform of the autocorrelation function), no techniques are available for correcting the estimates of the autocorrelation function itself.

If the sampling frequency is comparable to the bandwith of the power spectrum, the power spectrum estimate may be poor due to aliasing. Under these conditions, the estimate of the autocorrelation function given by (III.2) may take negative values. This is precissely what happens for the estimated autocorrelation function of the file system utilization process as shown in Figure 3-2. For a sampling frequency equal to one, the bandwith of the process would be equal to $\beta_1 = 0.59$, that is, the sampling frequency is not even twice the process bandwith.

The solution adopted has been to estimate the σ_i and β_i directly from a history of failures as described in Section 4.1., and to estimate the variance of the process σ^2_{zz}. Since

$$\sigma^2_{zz} = \alpha_1 + \alpha_2 \tag{III.8}$$

and

$$\sigma_i = s^2 \frac{\alpha_i}{\beta_i} \tag{III.9}$$

And knowing σ_i, β_i and σ_{zz}^2, the values of the α_i can be computed.

References

[Anderson 67]	J.E. Anderson and F.J. Macri. Multiple Redundancy Applications in a computer. In *Proceedings of the 1967 Annual Symposioum on Reliability*, pages 553-562. January, 1967.
[Avizienis 78]	A. Avizienis. Fault-Tolerance : The Survival Attribute of Digital Systems. *Proceedings of the IEEE* 66(10):1109-1125, Oct, 1978.
[Barlow 65]	R.E. Barlow and F. Proschan. *Mathematical Theory of Reliability.* John Wiley & Sons, 1965.
[Bazaraa 79]	M.S. Bazaraa and C.M. Shetty. *Nonlinear Programming. Theory and Algorithms.* John Wiley & Sons, 1979.
[Beaudry 78]	M.D. Beaudry. Performance Related Reliability Measures for Computing Systems. *IEEE Transactions on Ccomputers* C-27(6):540-547, June, 1978.
[Beaudry 79]	M. D. Beaudry. A Statistical Analysis of Failures in the SLAC Computing Center. In *Digest of Papers, Compcon Spring 79*, pages 49-52. IEEE Computer Society, 1979.
[Bell 78a]	C.G. Bell, A. Kotok, T.N Hastings, and R. Hill. The Evolution of the DECsystem 10. *Communications of the Association for the Computing Machinery* 21(1):44-63, January, 1978.
[Bell 78b]	C. Gordon Bell, J. Cragig Mudge. The Evolution of the PDP-11. In C. Gordon Bell, J. Craig Mudge, John E. McNamara (editor), *Computer Engineering : A DEC View of Hardware Systems Design*, . Digital Press, 1978.
[Berger 74]	R.W. Berger and K. Lawrence. Estimating Weibull Parameters by Linear and Nonlinear Regression. *Technometrics* 16(4):617-619, November, 1974.
[Bouricious 69]	W.G. Bouricious, W.C. Carter, and P.R. Schneider. Reliability Modeling Techniques for self-repairing Computer Systems. In *Proceedings of the 24th National Conference of ACM*, pages 295-383. 1969.
[Butner 80]	S.E. Butner and R.K. Iyer. *A Statistical Study of Reliability and System Load at SLAC.* Technical Report, Center for Reliable Computing, Stanford University, January, 1980.

[Castillo 80] X. Castillo, D.P. Siewiorek.
A Performance-Reliability Model For Computing Systems.
Technical Report, Carnegie-Mellon University, Computer Science Department, 1980.

[Cheung 75] R.C. Cheung and C.V. Ramamoorthy.
Optimal Measurement of Program Path Frequencies and its Applications.
In *Proc. 1975 Int. Fed. Automat. Contr. Congr..* August, 1975.

[Cheung 80] R.C. Cheung.
A User-Oriented Software Reliability Model.
IEEE Transactions on Software Engineering SE-6(6):118-125, March, 1980.

[Chou 80] T.C.K. Chou and J.A. Abraham.
Performance/Availability model of Shared Resource Multiprocessors.
IEEE Transactions on Reliability R-29(1):70-74, April, 1980.

[Cinlar 72] E. Cinlar.
Superposition of point processes.
In P.A.W. Lewis (editor), *Stochastic Point Processes Statistical Analysis, Theory, and Applications*, pages 549-606. Wiley, 1972.

[Costes 78] A. Costes, C. Landrault, and J.C. Laprie.
Reliability and Availability Model for Maintained Systems Featuring Hardware Failures and Design Faults.
IEEE Transactions on Computers C-27(6):548-560, June, 1978.

[Digital 77] *TOPS-10 Monitor Calls Manual*
Digital Equipment Corporation, 1977.

[Digital 78] *TOPS-10 and TOPS-20 SYSERR Manual*
Digital Equipment Corporation, 1978.

[Ferdinand 74] A.E. Ferdinand.
A Theory of Systems Complexity.
Int. J. Gen. Syst. 1:19-33, 1974.

[Ferrari 75] D. Ferrari.
Computer Systems Performance Evaluation.
Prentice-Hall, 1975.

[Fitzsimmons 78] A. Fitzsimmons and T. Love.
A review and evaluation of Software Science.
ACMCS 10(1):3-18, March, 1978.

[Fuller 78] S.H. Fuller and S.P. Harbison.
The C.mmp multiprocessor.
Technical Report, Carnegie-Mellon University, Computer Science Department, October,, 1978.

[Gardner 75] W. A. Gardner, L. E. Franks.
Characterization of Cyclostationary Random Signal Processes.
IEEE Transactions on Information Theory IT-21(1):4-14, January, Year = 1975.

[Gardner 78] W. A. Gardner.
Stationarizable Random Processes.
IEEE Transactions on Information Theory IT-24(1):8-22, January, 1978.

[Gay 79] F. A. Gay and M. L. Ketelsen.
Performance Evaluation for Gracefully Degrading Systems.
In *Digest of Papers, Ninth Annual International Conference on Fault-Tolerant Computing*, pages 51-58. IEEE Computer Society, 1979.

[Geilhofe 79] M. Geilhofe.
Soft errors in semiconductor memories.
In *Digest of Papers, Compcon Spring 79*, pages 210-216. IEEE Computer Society, 1979.

[Hecht 76] H. Hecht.
Fault-Tolerant Software for Real-Time Applications.
ACM Computing Surveys 8(4):391-407, December, 1976.

[Hodges 77] D.A. Hodges.
Progress in Electronic Technologies for Computers.
Technical Report, National Bureau of Standards, March, 1977.

[Horowitz 75] E. Horowitz.
Practical Strategies for Developing Large Scale Systems.
Addison-Wesley, 1975.

[Howard 71] R. A. Howard.
Dynamic Probabilistic Systems.
Wiley, 1971.

[Jelinsky 73] A. Jelinsky and P.B. Moranda.
Applications of a probability Based Method to a Code Reading Experiment.
In *Proceedings of the 1973 Symposioum on Software Reliability*, pages 78. IEEE Computer Society, 1973.

[Jenkins 68] G.M. Jenkins and D.G. Watts.
Spectral Analysis and its Applications.
Holden-Day, 1968.

[Keller 76] T.W. Keller.
CRAY-1 Evaluation Final Report.
Informal Report LA-6456-MS, Los Alamos Scientific Laboratory, December, 1976.

[Littlewood 79] B. Littlewood.
How to Measure Software Reliability and How Not To.
IEEE Transactions on Reliability R-28(2):103-110, June, 1979.

[Lynch 75] W.C. lynch, W. Wagner, and M.S. Schwartz.
Reliability Experience with Chi/OS.
IEEE Transactions on Software Engineering SE-1(2):253-257, June, 1975.

[McConnel 79a] S. R. McConnel, D. P. Siewiorek, and M. M. Tsao.
The Measurement and Analysis of Transient Errors in Digital Computing Systems.
In *Digest of Papers, Ninth Annual International Conference on Fault-Tolerant Computing*, pages 67-70. IEEE Computer Society, 1979.

[McConnel 79b]	S. R. McConnel, D. P. Siewiorek, and M. M. Tsao. *Transient Error Data Analysis.* Technical Report CMU-CS-79-121, Carnegie-Mellon University, Departments of Electrical Engineering and Computer Science, May, 1979.
[Melsa 78]	J.L. Melsa and D.L. Cohen. *Decision and Estimation Theory.* McGraw-Hill, 1978.
[Meyer 79]	J. F. Meyer, D. G. Furchtgot, and L. T. Wu. Performability evaluation of thh SIFT computer. In *Digest of Papers, Ninth Annual International Conference on Fault-Tolerant Computing*, pages 43-50. IEEE Compter Society, 1979.
[Miyamoto 75]	I. Miyamoto. Software Reliability in Online Environment. In *Digest of Papers, 1975 International Conference on Software Reliability*, pages 194-203. IEEE Computer Society, 1975.
[Mohanly 73]	S.N. Mohanly. Models and Measurements for Quality Assesment of Software. *ACMCS* 11(3):250-275, September, 1973.
[Morganti 78]	M. Morganti, G. Coppadoro, and S. Ceru. UDET 7116 - Common Control for PCM Telephone Exchange : Diagnostic Software Design and Availability Evaluthation. In *FCTS8*, pages 16-23. IEEE Computer Society, 1978.
[Musa 75]	J.D. Musa. A Theory of Software Reliability and its Applications. *IEEE Transactions on Software Engineering* SE-1(x):312-327, September, 1975.
[Nelson 73]	E.C. Nelson. *A Statistical Basis for Software Reliability Assesment.* Technical Report, TRW, March, 1973.
[Ohm 79]	V.J. Ohm. Reliability Considerations for Semiconductor Memories. In *Digest of Papers, Compcon Spring 79 ,Organization = IEEE Computer Society*, pages 207-209. 1979.
[Oppenheim 75]	A.V. Oppenheim and R.W. Schafer. *Digital Signal Processing.* Prentice-Hall, 1975.
[Papoulis 65]	A. Papoulis. *Probability, Random Variables, and Stochastic Processes.* McGraw-Hill, 1965.
[Phister 79]	M. Phister Jr. *Data Processing Technology and Economics.* Digital Press, 1979.

DATE
ILME