Modeling Space Shuttle Software Failures at Varying Criticality Levels

George J. Knafl Joseph A. Morgan
School of Computer Science,
Telecommunications, & Information Systems
DePaul University
243 South Wabash Avenue
Chicago, IL 60604
312-362-8715
knafl@cs.depaul.edu cphdjam@cs.depaul.edu

Abstract–Regression methods are employed in analyzing a space shuttle software failure data set, classified into three criticality levels. A family of models based on transforms of cumulative time and cumulative failures is considered for the purposes of generating long-term future predictions of the software failure process for any criticality level. This family includes several established software reliability models including the exponential, logarithmic, and power models. It also includes models based on transforms of the time per failure, the time-varying analogue to the mean time between failures. Models are compared through crossvalidation on the basis of various predictive performance measures related to the established predicted residual sum of squares (PRESS) criterion. Predictions are generated through the observed failure data for the minor and the noncritical major failure processes since ample failures occur at each of these criticality levels. Predictions for the critical failure process on the other hand are generated using observed major failures, both critical and noncritical, since the critical failures by themselves are too few in number to predict that process well. Predictions are generated using models that produce minimal prediction intervals, and the sizes of these prediction intervals are compared to those generated using established software reliability models. Nonparametric prediction intervals are proposed as alternatives to the usual ones based on the assumption of approximate normality since that assumption is likely to not apply in the context of long-term predictions, especially when predicting the critical failure process.

TABLE OF CONTENTS

1. INTRODUCTION

Several of the commonly used software reliability models, in particular, the exponential, logarithmic, and power models, are equivalent to the assumption that the log of the failure intensity is linear in an appropriate transform of cumulative time or of cumulative failures [2]. This fact provides the basis for least squares estimation of the parameters of such models [5,6], and hence for the application of regression methods to software failure data.

Knafl, Morgan, Follenweider, and Karcich [2] use regression methods to conduct a thorough assessment of a software failure data set for an industrial software development project. They propose models based on the time per failure, that is, the ratio TPF(t)=t/n(t) of cumulative time t divided by the cumulative number n(t) of failures occurring up to that point in time. They then demonstrate that models based on this time-varying analogue to the mean time between failures can provide effective predictions of current log failure intensity while outperforming well-established software reliability models like the exponential, logarithmic, and power models.

Karcich, Knafl, and Morgan [1] continue that investigation, applying regression methods using a broad family of time and failure models to analyze the major failures of the space shuttle data set reported by Misra [4]. They select a final time segment of that data set, analyze its failures, and assess the impact on their results of using those failures compared to using all the failure data. They extend the methods used in [2] to a more extensive family of models and to consideration of short-term future predictions as well as current predictions.

In this paper, we also analyze the space shuttle data, but model the software failure process for each of the three criticality levels. To predict the critical failure process, we use both types of major failures, not just the critical failures, since these are few in number. This is not necessary at noncritical levels since there are sufficient observations to generate reasonable predictions for each such level separately.

We begin by considering several forms of crossvalidation that may be used in selecting models from among those employed in [1] as alternatives to the approach used in that paper. The appropriate form of crossvalidation to use will depend on what it is that the software reliability engineer wants most to predict. In the situation considered in [1], it is desirable to predict well counts of major failures within distinct time intervals. However, the standard form of crossvalidation used in [1] is based on predictions of the transformed response variable of the regression model which, in the reliability context, is the natural logarithm of failure intensity. In order to assess the impact of this discrepancy between the model selection criterion and the desired effect, we compare the results for noncritical failure levels using several alternative forms of crossvalidation measures, in particular, measures based on predictions of log failure intensities, failure intensities, failure counts, and proportions of failures within time intervals.

Since Misra reports failure data by criticality level, it is possible to consider the issue of how the space shuttle software failure process varies by criticality level. While the Misra data set contains critical major failures, these occur infrequently, and so there is an insufficient number of such failures to conduct a thorough modeling of the critical failures by themselves. This is obviously going to be the case for most software systems and a desirable one at that. Thus there is a need to extrapolate from the failure process at the noncritical levels to the critical level. We address this issue through methods that leverage the relatively larger number of noncritical major failures into better predictions of the critical failure process than may be obtained from the critical failures by themselves. Moreover, since our goal for the critical failure process is to predict its reliability well, we use a crossvalidation procedure in that case that is based on predictions of proportions of critical failures within time intervals, rather than on predictions of the failure intensities or failure counts for those intervals.

Specialized software reliability tools are not required for our analyses. We use instead the powerful matrix language PROC IML as well as the standard regression procedures of the Statistical Analysis System (SAS®), a popular commercial statistical tool.

2. The Space Shuttle Data

Misra [4] reports failure data for space shuttle software with failures grouped into 38 weeks of activity and classified by severity into major and minor failures. Major failures are further categorized into critical and noncritical major failures. Time is measured in test hours, and the reported data cover a period of T=2456.9 total test hours. The data are collected for unspecified phases within the development of software to support selected space shuttle missions and are generated through independent verification, mission simulations, and actual missions. See [4] for further details on the process used to generate these data.

Misra analyzes major failures, critical and noncritical combined, over the whole observed time interval. Karcich et al. [1] also use both kinds of major failures combined, but analyze only those occurring within a final segment of the observed time interval. These prior analyses restrict consideration to a single, possibly combined, failure type. In this paper, we model all three failure types in order to make comparisons across criticality levels. Moreover, the critical failure data are not readily modeled by themselves due to their sparsity, and so we investigate methods for leveraging predictions for the more extensive noncritical failures into predictions for the critical failures.

Software failure data are either reported in ungrouped or grouped form. As mentioned above, the space shuttle data are reported grouped into weeks of testing activity. Ungrouped data always require grouping prior to applying regression methods, and even grouped data may require further regrouping.

Failure intensity estimates are computed for each grouped time interval as the ratio of the number of failures observed in the interval divided by the test time expended during the interval. Okumoto [6] recommends grouping consecutive sets of five failures. Karcich et al. [1] regroup the space shuttle data starting with the 20th week into consecutive intervals of at least two failures, using a smaller number of failures per group than recommended in [6] in order to have more data points to use in their analyses. They consider both current predictions and short-term future predictions. This paper considers only long-term predictions, and so its analyses are based on the space shuttle data grouped into intervals of at least 200 test hours. That particular length is chosen since it is the length of the future interval reported and addressed by Misra, and hence is a natural interval length to want to predict well.

3. Regression Methods

In Sections 4-5, we use regression methods similar to those used in [1] to analyze the space shuttle data set grouped into time intervals w. For each time interval w, the software underwent $\Delta t(w)$ hours of testing resulting in $\Delta n(w)$ major failures. As mentioned in Section 2, we regroup the initial set of intervals into intervals of length at least 200 test hours long. In particular, starting with w=1, if $\Delta t(w)<200$, merge that interval with intervals for subsequent weeks until the combined interval is of length $\Delta t_i \geq 200$ hours, then continue this adjustment starting with the next unadjusted interval. It is possible there will be an interval left over at the end of length less than 200 hours, in which case it is merged with the previous interval. This produces consecutive time intervals indexed by $i=1,2,...,I=11$ during each of which $\Delta t_i \geq 200$ test hours were expended resulting in the occurrence of Δn_i major failures and an associated failure intensity of $\lambda_i = \Delta n_i / \Delta t_i$ failures per test hour. We compute failure intensities in this way for different failure types, but suppress that fact in our notation for simplicity.

For each time interval produced by our grouping scheme, we compute the values for the basic variables of our analyses. The log failure intensities are

$$y_i = \ln(\lambda_i).$$

The cumulative test times are

$$t_i = \Delta t_1 + ... + \Delta t_i,$$

while the cumulative failure counts are

$$n_i = \Delta n_1 + ... + \Delta n_i.$$

The time per failure values are

$$TPF_i = t_i / n_i,$$

and the time by failures values are

$$TxF_i = t_i \cdot n_i.$$

The family of time and failure models proposed in [1] corresponds to the regression models with dependent observation values y_i together with independent variable observation values x_i computed as one or more transforms of the four quantities t_i, n_i, TPF_i, or TxF_i.

Regrouping may be necessary to force all intervals to have a nonzero number of failures so that the y_i will be computable. This is not necessary for the space shuttle data grouped into time intervals of at least 200 test hours since we only compute y_i for minor and noncritical major failures. Computing these for critical failures would force us to regroup the space shuttle data into only four intervals. We do, however, apply models estimated using noncritical failures to the critical failure data, and so we need to be able to compute all possible independent variables for the critical failure data. Thus, it may be necessary to regroup so that the initial interval has a nonzero number of critical failures. This was not necessary for our analyses.

It is well-known [2] that for models with an intercept, the choice of independent variable $x_i = n_i$ corresponds to the logarithmic (or Musa-Okumoto) software reliability model, the choice $x_i = t_i$ corresponds to the exponential (or Goel-Okumoto) model, and $x_i = \ln(t_i)$ corresponds to the power (or Duane) model. Thus, the family of time and failure models includes several of the more popular software reliability models considered in the literature. It also contains the time-varying analogue TPF_i of the mean time between failures and the interaction variable TxF_i .

We consider as in [1] a broad family of models based on a single independent variable computed using a variety of transforms. Specifically, we use the natural log as well as powers of -2, -1, -1/2, 1/2, 1, and 2 of the four basic variables for a total of 28 different variables. We consider all such models both with and without intercept parameters and, for completeness, the constant model as well. Altogether, we consider a total of 57 simple regression models. We do not consider multiple independent variable models in this paper as considered in [1]. While such models may very well produce better models than single variable models, we expect they would produce results of a similar nature to those we report in Sections 5-6.

Some analyses of software failure data, for example, as in [1,2], address current failure intensity, that is, they use the observations (x_i, y_i) for $i=1,2,...,I$. Many comparisons of software reliability models, for example, in [3,5,7], utilize some aspect of the future software failure process to distinguish between models. Karcich et al. [1] on the other hand consider both current and future prediction. They use a simple adjustment of the data to convert standard crossvalidation procedures based on current predictions to ones based on future prediction. In particular, they model the shifted observations (x_i, y_{i+1}) for $i=1,2,...,I-1$. However, they only consider short-term future predictions, but this is a consequence of their choice to group the data into time intervals of at least two failures. We analyze shifted data grouped into intervals of at least 200 hours in order to address long-term future predictions. Although we only consider this one specific long-term horizon, our methods are easily adapted to handle predicting future intervals of any fixed length. Small horizons correspond to short-term future predictions, large horizons to long-term future predictions.

4. CROSSVALIDATION

The family of time and failure models described in Section 3 includes many individual models, and so needs to be complemented with a method for choosing one of those models to apply to specific software failure data. Since we want to predict the reliability of the space shuttle software failure process at any criticality level over a time interval immediately subsequent to the observed time interval, we select models in such a way that prediction intervals directly related to reliability are as small as possible. In other words, we distinguish between models using a predictive performance measure that has a form related to that of the established predictive performance measure called the predicted residual sum of squares (PRESS) (see [1,2]), but one that is adjusted to address the purpose for our predictions.

Model selection through PRESS is a special case of a procedure statisticians refer to as crossvalidation. In particular, it is a leave-one-out-at-a-time crossvalidation. We define a class of predictive performance measures similar to PRESS but computed for shifted data sets as described in Section 3. We use shifted data since we are concerned with future predictions rather than with current predictions, but our performance measures are readily adjusted to handle the case of current predictions by using unshifted data.

A model's PRESS score is computed as the sum of the squares of its "deleted" prediction errors. More precisely, for each shifted observation (x_i, y_{i+1}), a model M is fit using the remaining shifted observations. The resulting parameter estimates are used to predict some aspect u_i of the data at $x = x_i$. This is a deleted prediction in the sense that the response value y_{i+1} corresponding to the aspect u_i is not included in the computation of its prediction, only the value of its explanatory variable/vector x_i is used. Next, the deleted predicted value $\hat{u}_i(M)$ for u_i is compared to the actual value u_i to produce the associated deleted prediction error, and these are squared and summed to obtain the model's PRESS score.

We average rather than sum since that quantity represents the square of the deleted standard prediction error

$$s^2(M)=\frac{\sum_{i=1}^{I-1}\left(u_i-\hat{u}_i(M)\right)^2}{I-1}.$$

This adjustment has no impact on the results of model selection, but is directly related to our goal of producing minimal prediction intervals. A model M with a smaller performance score $s^2(M)$ than another model M' provides better overall predictions than M' of the associated aspect u used in its computation.

The main purpose of our analyses is to predict the unknown value u_I of the aspect of interest for the interval of size 200 time units immediately subsequent to the observed time interval. Each model M produces a prediction $\hat{u}_I(M)$ of u_I based on the observed shifted data. The associated prediction error $u_I-\hat{u}_I(M)$ has the same form as each of the deleted prediction errors used in computing s(M), and so s(M) is a measure of the prediction error for $\hat{u}_I(M)$.

Response values in the context of grouped software failure data are the natural log transforms of the failure intensity estimates, that is, $y_{i+1}=\ln(\lambda_{i+1})$. The PRESS score is usually based on deleted predictions of these response values, that is, $u_i=y_{i+1}$. Such an approach was used in [1] where it is demonstrated that the associated 95% prediction intervals for future y values are noticeably smaller than those produced by certain established software failure models, that is, by the exponential, logarithmic, and power models.

However, it is more important to obtain minimal prediction intervals for other aspects u of future time intervals, in particular, for untransformed failure intensity over a subsequent time interval

$$u_i=\lambda_{i+1},$$

for the number of failures in a subsequent time interval

$$u_i=\Delta n_{i+1}=\lambda_{i+1}\cdot\Delta t_{i+1}$$

as in [7], or for the proportion of failures in a subsequent time interval

$$u_i=p_{i+1}=\frac{\Delta n_{i+1}}{n_I}.$$

In Section 5, we compare predictions of these three aspects for the minor and noncritical major failure processes. In Section 6, our focus is on predicting the reliability of the critical failure process, and so the analyses in that section are based on predictive performance measures computed for the last case with $u_i=p_{i+1}$.

Although a model may have the best, or nearly best, predictive performance score, a residual analysis is still required to assess whether that model is appropriate to use with the data under analysis. Residuals are usually computed as differences between actual response values y and predictions of those response values based on all the data. Since our focus is on future predictions rather than on current predictions, we use deleted predictions instead. Furthermore, we express residuals in terms of the aspect u of importance to our analysis rather than in terms of response values y. In other words, we report residuals that are the "deleted" prediction errors

$$r_i(M) = u_i - \hat{u}_i(M)$$

for the aspect u of interest.

The residuals may be combined to obtain the estimate s(M) of the standard error for the prediction $\hat{u}_I(M)$ of the aspect u_I of interest for the 200 time units subsequent to the observed time interval. Associated two-sided prediction intervals for u_I may be computed in the conventional parametric manner as

$$u_I = \hat{u}_I(M) \pm z\cdot s(M)$$

with the value of z chosen using a standard normal table to produce the desired prediction level. The use of these prediction intervals based on s(M) are predicated on the assumption of approximate normality of the predictor $\hat{u}_I(M)$. This issue as well as issues of unbiasedness, constant standard deviation, and the presence of outliers may be assessed in the usual manner with residual and normal plots.

When the analysis is based on current predictions or on short-term predictions, it is possible that the number of residuals will not be too small. On the other hand, when the analysis is based on long-term predictions, the number of residuals will necessarily be small. For example, in our analysis, there can be at most 12 disjoint time intervals of length 200 hours since the observed time interval is 2456.9 hours long. In actuality, there are 11 intervals since the data are grouped to start with, and the number of shifted observations is one less than this. Even so, there is some chance of reasonable features for the residuals for the minor and noncritical major failure processes since associated interval counts/proportions may possibly change in a reasonable manner, but this will not be the case for the critical failure process. The total number of critical failures will typically be very small, most intervals will contain zero failures, most proportions will be 0, but associated predictions $\hat{u}_i(M)$ will be positive, and so most residuals $u_i - \hat{u}_i(M)$ will be negative. Thus, the histogram for the residuals is likely to be skewed, and so methods are required that do not rely on the usual assumption of approximate normality.

The way we circumvent this problem is to base prediction intervals on percentiles of the observed residuals. In particular, if $r(\gamma,m)$ denotes the $\gamma\cdot 100$ percentile of the empirical distribution of the residuals r_i, then a $(1-\alpha)\cdot 100\%$ prediction interval for u_I is given by

$$\hat{u}_I(M)+r(\alpha/2,M)\le u_I\le\hat{u}_I(M)+r(1-\alpha/2,M)$$

with associated $(1-\alpha/2)\cdot 100\%$ upper prediction bound

$$u_I\le\hat{u}_I(M)+r(1-\alpha/2,M).$$

In our analysis of critical failures, we use 80% prediction intervals, that is, with α=0.20, and corresponding 90% upper prediction bounds, and so

$$r(\alpha/2,M)=r_{(2)} \text{ and } r(1-\alpha/2,M)=r_{(9)},$$

the 2nd and 9th largest residuals, since we have 10 residuals in total. We employ the model selection procedure that minimizes the asymmetric half-width of the prediction interval, that is,

$$AHW(M)=\max(|r(\alpha/2,M)|,|r(1-\alpha/2,M)|)$$

as a nonparametric alternative to the usual procedure that minimizes s(M). In this case, we consider only those models for which

$$r(\alpha/2,M)\leq 0\leq r(1-\alpha/2,M)$$

to avoid models with a tendency to being biased. In this way, the prediction $\hat{u}_I(M)$ falls within the $(1-\alpha)\cdot 100\%$ prediction interval for u_I.

5. Predicting Noncritical Failures

Table 1 contains standard prediction errors s(M) for a variety of models and for predicting four aspects u of the minor failure process, specifically, the proportion of failures p, the failure count Δn, the failure intensity $\lambda=\Delta n/\Delta t$, and the log failure intensity $\ln(\lambda)$. The same information is provided in Table 2 for the noncritical major failure process. Predictions $\hat{y}_{i+1}$ of $y_{i+1}=\ln(\lambda_{i+1})$ are obtained first through regression estimation applied to shifted data. These are then used to generate predictions of the other three aspects as follows:

$$\hat{\lambda}_{i+1}\equiv\exp(\hat{y}_{i+1}),\ \Delta\hat{n}_{i+1}\equiv\hat{\lambda}_{i+1}\cdot\Delta t_{i+1},\ \hat{p}_{i+1}\equiv\frac{\Delta\hat{n}_{i+1}}{n_I}.$$

Results for the first 5 models of Table 1 are computed by analyzing the minor failure data by themselves. There are only small differences between the standard errors for the first 4 standard models of Table 1 and between the best of these models and the best model for the whole family of time and failure models. The minor failure process can be predicted without too much of a penalty with any of these models. Furthermore, the relative position of a model as a predictor of one aspect is the same as for any other aspect. Also, the same model, the one with $x=(TPF)^{-2}$ and an intercept, is best for predicting all aspects.

We also report results in Table 1 for the best 1 variable model obtained by combining the minor and noncritical major failures. In this case, we model both types of failures the same way, that is, models for both failure types have the same explanatory variable x with the same slope parameter, but with different intercept parameters. We estimate the parameters of this combined model using failure data of both types. Since we seek to predict the minor failure process, we minimize the associated prediction error for the minor failures and not the prediction error for the noncritical major failures. As indicated by the results presented in Table 1, there is at most a small benefit to combining both types of failures over using just the minor failures for predicting all aspects of the minor failure process except the log failure intensity aspect. Also, the best 1 variable model changes with the aspect being predicted, in particular, x=TPF is best for predicting p and Δn, $x=(TxF)^{-1}$ is best for predicting λ, and $x=t^2$ is best for predicting $\ln(\lambda)$, all with intercept parameters.

Table 1
Comparison of Standard Prediction Errors
for Minor Space Shuttle Software Failures
over 200 Test Hours into the Future

Using Minor Failures Only

model	**aspect predicted**			
	p	**Δn**	**λ**	**ln(λ)**
exponential	0.034	5.51	82.2	0.467
logarithmic	0.034	5.46	81.6	0.460
power	0.033	5.33	81.5	0.456
time per failure	0.032	5.10	78.2	0.443
best 1 variable	0.031	4.98	76.9	0.435

Using Minor & Noncritical Major Failures

model	**aspect predicted**			
	p	**Δn**	**λ**	**ln(λ)**
best 1 variable	0.030	4.84	75.5	0.453

Results for the first 5 models of Table 2 are computed by analyzing the major failure data by themselves. There is a substantial difference between the standard errors for these models, with the best 1 variable model providing more than a 15% improvement over the other standard models, for all 4 aspects. Hence, there is a noticeable benefit to consideration of the full family of time and failure models. Furthermore, as for minor failures, the relative position of a model as a predictor of one aspect is the same as for any other aspect. Also, the same model, the one with $x=(TxF)^2$ and an intercept, is best for predicting all aspects.

As in Table 1, we report results in Table 2 for the best 1 variable model obtained by combining the minor and noncritical major failures, but with the roles of the two types of failures reversed. As before, this best model changes with the aspect being predicted, in particular, $x=t^2$ is best for predicting p, Δn, and $\ln(\lambda)$ while $x=(TxF)^2$ is best for predicting λ, all with intercepts. However, although the best model for the combined approach outperforms the first 4 standard models, it does not predict as well as the best model based on the noncritical major failures by themselves. This indicates that there may be little benefit to using the minor failure process to predict the critical failure process. On the other hand, the noncritical major failure process can be used to advantage in the prediction of the minor failure process, indicating that there is a good chance that the noncritical major failures may also be used to advantage in the prediction of the critical failure process. Thus, we use only the noncritical major failure data, and not the minor failure data, in Section 6 in predicting the critical failure process and have some confidence that this has a beneficial effect.

Table 2
Comparison of Standard Prediction Errors
for Noncritical Major Space Shuttle Software Failures
over 200 Test Hours into the Future

Using Noncritical Major Failures Only

model	**aspect predicted**			
	p	**Δn**	**λ**	**ln(λ)**
exponential	0.047	4.07	63.0	0.568
logarithmic	0.046	3.99	61.9	0.559
power	0.048	4.18	64.1	0.588
time per failure	0.049	4.23	64.4	0.606
best 1 variable	0.037	3.28	51.6	0.460

Using Minor & Noncritical Major Failures

model	**aspect predicted**			
	p	**Δn**	**λ**	**ln(λ)**
best 1 variable	0.040	3.46	54.0	0.476

6. PREDICTING CRITICAL FAILURES

Our focus in this section is on predicting the long-term reliability of the critical failure process. For this reason, we group the observed time interval (0,T] into I=11 subintervals of at least 200 test hours each. We then apply linear regression models to the I-1 shifted noncritical major failure observations with independent variables $x_{i,nc}$ computed for the ith subinterval and response variables

$$y_{i+1,nc}=\ln(\lambda_{i+1,nc})$$

computed for the subsequent subinterval, reflecting our need to predict into the future. This produces predictions

$$\hat{y}_{i+1,nc}=\hat{\alpha}_{nc}+\hat{\beta}_{nc}\cdot x_{i,nc}$$

for the noncritical major failure process. Predictions for the proportions of noncritical failures are computed as

$$\hat{p}_{i+1,nc}=\frac{\exp\left(\hat{\alpha}_{nc}+\hat{\beta}_{nc}\cdot x_{i,nc}\right)\cdot\Delta t_{i+1}}{n_{I,nc}}$$

for i=1,...,I-1. Due to the use of shifted data, the proportion $p_{1,nc}$ is not predicted, but its actual value $n_{1,nc}/n_{I,nc}$ is used instead. Similar predictions

$$\hat{y}_{i+1,c}=\hat{\alpha}_c+\hat{\beta}_{nc}\cdot x_{i,c}$$

and

$$\hat{p}_{i+1,c}=\frac{\exp\left(\hat{\alpha}_c+\hat{\beta}_{nc}\cdot x_{i,c}\right)\cdot\Delta t_{i+1}}{n_{I,c}}$$

are used for the critical failure process, but with the slope estimate β_{nc} computed from the noncritical major failures and with the intercept parameter $\hat{\alpha}_c$ computed in such a way that the predicted proportions of critical major failures sum to 1, that is, so that

$$p_{1,c}+\sum_{i=1}^{I-1}\hat{p}_{i+1,c}=1.$$

Thus,

$$\hat{\alpha}_c\equiv\ln\left(\frac{n_{I,c}-n_{1,c}}{\sum_{i=1}^{I-1}\exp\left(\hat{\beta}_{nc}\cdot x_{i,c}\right)\cdot\Delta t_{i+1}}\right).$$

Our goal is to predict $p_{I,c}$ the proportion of critical failures that occur in the time interval (T,T+200], immediately subsequent to the observed time interval, out of the total failures occurring in (0,T+200]. This quantity represents the probability of a critical failure occurring in the future time interval (T,T+200], relative to the combined time interval (0,T+200], and so is equivalent to predicting the reliability $1-p_{I,c}$ of the critical failure process over that interval.

Since we need to predict an aspect of the future time interval (T,T+200] using failures within the smaller observed time interval (0,T], we distinguish between models with a crossvalidation procedure that reflects this disparity between the time interval used for modeling and the time interval over which predictions are desired. In particular, for each i, we apply a model M as described above, but to all observations but the ith, and hence use all the failures but those in the (i+1)st time interval. This generates a prediction $\ddot{p}_{i+1,c}(M)$ of the proportion $p_{i+1,c}$ of failures occurring in the (i+1)st interval, out of all the failures occurring in the observed time interval (0,T], and a corresponding deleted residual

$$r_{i,c}(M)=p_{i+1,c}-\hat{p}_{i+1,c}(M).$$

These residuals may then be used to compute the standard error $s_c(M)$ for the prediction $\ddot{p}_{I+1,c}(M)$ of $p_{I+1,c}$, the proportion of critical failures that occur in the time interval (T,T+200], immediately subsequent to the observed time interval, out of all the failures occurring in (0,T+200].

Table 3 contains predictions of $p_{I+1,c}$ for the same 4 standard models considered in Section 5 as well as for the best 1 variable model, the one with $x=(TxF)^{-1}$ and an intercept. This model is best in the sense that it has minimal associated standard prediction error $s_c(M)$ within the family of time and failure models. The predictions reported in Table 3 are all similar, except for the time per failure model, while the standard prediction error and hence the 90% one-sided upper bound $1.28\cdot s_c(M)$ for the best 1 variable model are marginally smaller, that is, 6.8% smaller, than associated values for the best of the standard models. Lower prediction bounds are not reported in Table 3 since, in all cases, the lower 80% prediction bound is negative, and so the appropriate lower bound on $p_{I+1,c}$ is zero.

Table 3
Upper Prediction Bounds for the Probability
of a Space Shuttle Software Critical Failure
within a Future Time Interval of 200 Test Hours
Parametric Approach

model	prediction	standard error	90% upper bound
exponential	0.048	0.138	0.225
logarithmic	0.066	0.136	0.240
power	0.058	0.133	0.228
time per failure	0.000	0.281	0.360
best 1 variable	0.055	0.124	0.214

The results of Table 3 indicate that there is only a small benefit to consideration of the full family of time and failure models. However, the residuals produced by these models are significantly nonnormal (using the Shapiro-Wilk test of normality), and so the use of the parametric upper prediction bounds of Table 3 is inappropriate. Table 4 presents results for the alternative nonparametric prediction bounds discussed in Section 4 that avoid this problem.

Table 4 contains predictions of $p_{I+1,c}$ for the same 4 standard models as in Table 3 as well as for the best 1 variable model, in this case, the one with $x=n^{-1}$ and an intercept. This model is best in the sense that it has minimal associated asymmetric half-width $AHW_c(M)$, as defined in Section 4, within the family of time and failure models. The prediction for the best 1 variable model is much smaller than those for the first three well-established models and closer to that of the time per failure model which still rounds to zero. Furthermore, the asymmetric half-width for the best 1 variable model is just over 10% better than the best value for the other standard models and its 90% upper bound is substantially smaller, about 25% smaller, than the best bound for the standard models, which produce very similar bounds.

Table 4
Upper Prediction Bounds for the Probability
of a Space Shuttle Software Critical Failure
within a Future Time Interval of 200 Test Hours
Nonparametric Approach

model	prediction	asymmetric half-width	90% upper bound
exponential	0.048	0.195	0.243
logarithmic	0.066	0.186	0.252
power	0.058	0.189	0.247
time per failure	0.000	0.250	0.250
best 1 variable	0.016	0.167	0.183

These results indicate there is a noticeable benefit to consideration of the full family of time and failure models. Moreover, the nonparametric approach of Table 4 does not require a normality assumption to justify its use while generating a 90% prediction bound that is over 14% smaller than the one generated by the parametric approach. While the parametric approach generates a conservative upper bound when applied to the space shuttle critical failure process, it is not clear that this will hold in general. In any case, the nonparametric upper bound is appropriate to use whether it is smaller or larger than the parametric upper bound.

Thus, we conclude, with 90% confidence, that the probability of a critical failure of the space shuttle software within the future interval of length 200 test hours is no larger than 0.183. Equivalently, we conclude, with 90% confidence, that the reliability of that software for critical failures is at least 0.817.

7. SUMMARY

We use regression methods to analyze software failure data within the three criticality levels of minor, noncritical major, and critical major failures for the space shuttle software. We regroup these data prior to analysis into intervals of at least 200 test hours in order to address long-term prediction. We apply these regression methods to shifted data in order to address future predictions. We consider prediction of 4 different aspects of the future failure process: the log failure intensity predicted directly through the regression model as well as the failure intensity, failure counts, and proportions of failures.

We consider a broad family of regression models based on transforms of cumulative time and cumulative failures. This family includes several established software reliability models including the exponential, logarithmic, and power models. It also includes the model based on the time per failure, the time-varying analogue to the mean time between failures. We use results for these 4 standard models as benchmarks for assessing the performance of other models in the family chosen for their optimal properties.

We employ several crossvalidation schemes for distinguishing between models. In particular, we adapt the usual crossvalidation performance measure to the 4 different aspects of the future failure process. We also define two varieties of crossvalidation, a parametric approach based on minimizing the standard prediction error and a nonparametric alternative based on minimizing the maximum distance from the prediction to asymmetric lower and upper prediction bounds. The parametric approach, while used more commonly, relies on an assumption of approximate normality for the prediction errors. This is not likely to hold for the small samples obtained for long-term predictions, especially for the critical failure process with zero critical failures in most time intervals, hence the need for the nonparametric alternative.

We model the minor and the noncritical failure processes separately, that is, using failures within each of these criticality levels by themselves. We also model each of these failure processes using failures of both types to assess the benefits of combining failures of different types. We compare in Tables 1-2 standard errors for predicting the 4 different aspects of the failure process generated by the 4 standard models and by the best models using separate as well as combined data.

The minor failure process can be predicted without too much penalty with any of the models of Table 1. Furthermore, the relative position of a model as a predictor of one aspect is usually the same as for any other aspect, and so it may not matter which aspect is used in selecting a model. However, it seems prudent to use the aspect of greatest importance for the problem at hand. For example, we address the proportion of failures aspect in our analyses of the critical failure process since that aspect is directly related to our concern for predicting reliability at the critical failure level.

For the noncritical major failure process, on the other hand, there is a noticeable benefit to consideration of the full family of time and failure models over the 4 standard models. Also, as for minor failures, the relative position of a model as a predictor of one aspect is about the same as for any other aspect.

Aspects of the noncritical major failure process may be predicted better with the noncritical failures by themselves than by combining them with the minor failures. The noncritical major failure process, however, can be used to advantage in the prediction of the minor failure process, indicating that there is a good chance that it may also be used to advantage in the prediction of the critical failure process. Thus, we predict the critical failure process with the critical failures combined with only the noncritical major failures, and not with the minor failures, although that is possible, and do so with some confidence that this has a beneficial effect.

We propose a method that leverages the relatively larger number of noncritical major failures into better predictions for the critical failure process than is possible with the few failures of that type by themselves. More specifically, we fit linear regression models to the noncritical major failures, and apply these models to the critical failure process by reusing slope estimates based on the noncritical major failures and adjusting intercept estimates on the basis of observed critical failures.

We report in Tables 3-4 predictions of the probability of a critical failure within the future time interval of 200 test hours together with both parametric and nonparametric 90% upper bounds on this probability. Although the parametric bounds are conservative in this case, they are inappropriate to use since they are predicated on the assumption of approximate normality of the prediction errors and this is inconsistent with the observed prediction errors. For this reason, we base our final predictions on the best 1 variable model using the nonparametric approach which is appropriate to use whether or not the normality assumption holds.

We predict, with 90% confidence, the probability that a critical failure of the space shuttle software will occur within the future interval of length 200 test hours to be no larger than 0.183. Equivalently, we predict, with 90% confidence, the reliability of the critical failure process for the space shuttle software over that same future time interval to be at least 0.817.

REFERENCES

[1] R. M. Karcich, G. J. Knafl, J. A. Morgan, "A Regression Analysis of a Space Shuttle Software Failure Data Set," *1996 IEEE Aerospace Applications Conference Proceedings*," vol. 4, 321-334, 1996.

[2] G. J. Knafl, J. A. Morgan, R. Follenweider, & R. M. Karcich, "Software Failure Data Analysis Using the Least Squares Approach and the Time per Failure Concept," in *International Journal of Reliability, Quality, and Safety*, **2**, 161-176, June 1995.

[3] Y. Malaiya, N. Karunanithi, & P. Verma, "Predictability of Software-Reliability Models," *IEEE Transactions on Reliability*, **41**, 539-546, December 1992.

[4] P. N. Misra, "Software Reliability Analysis," *IBM Systems Journal*, **22**, 262-270, 1983.

[5] J. D. Musa, A. Iannino, & K. Okumoto, *Software Reliability: Measurement, Prediction, Application*, New York: McGraw-Hill, 1987.

[6] K. Okumoto, "A Statistical Method for Software Quality Control," *IEEE Transactions on Software Engineering*, **11**, 1424-1430, Dec. 1985.

[7] N. Schneidewind, "Optimal Selection of Failure Data for Predicting Failure Counts," *IEEE Transactions on Software Engineering*, **19**, 1095-1104, Nov. 1993.

George J. Knafl *received his Ph.D. in Mathematics, specializing in Statistics, from Northwestern University in 1978. He is currently professor in the School of Computer Science, Telecommunications, & Information Systems and director of the Software Engineering Division at DePaul University. He served on the program committee for COMPSAC, the International Conference of Computer Software and its Applications, from 1986-1993 and edited the proceedings of that conference from 1987-1991; as tutorials co-chair for ISSRE, the International Symposium on Software Reliability Engineering, in 1995; and as the chair of the Tactical Planning Group for Software Reliability Engineering Standards within the Software Reliability Engineering and Software Engineering Standards Committees of the IEEE*

Technical Council on Software Engineering from 1994-1996.

Joseph A. Morgan *received his B.S. in Mathematics from the University of the West Indies in 1980 and his M.S. in Information Systems, specializing in Software Engineering, from DePaul University in 1991. He is currently a Ph.D. candidate in the School of Computer Science, Telecommunications, & Information Systems at DePaul University. His research interests include the application of statistical methods to software engineering, software reliability engineering and data mining.*