New Ways to Get Accurate Reliability Measures

SARAH BROCKLEHURST and BEV LITTLEWOOD
City University, London

◆ *In spite of extravagant claims, no reliability model can be trusted to be accurate. Now, statistical techniques let you determine which model gives acceptable results.*

Over the years, many software reliability models have been published, quite a few from our own Centre for Software Reliability. Unfortunately, no single model can be universally recommended. In fact, the accuracy of the reliability measures generated by the models varies dramatically: Some models sometimes give good results, some are almost universally awful, and none can be trusted to be accurate at all times. Worse, it does not seem possible to identify in advance those data sets for which a particular model is appropriate.[1]

This unsatisfactory situation has undoubtedly been the major factor in the poor adoption of reliability models. Users who have experienced poor results are once bitten, twice shy, unwilling to try new techniques.

It is with some trepidation that we claim our approach has largely eliminated these problems – our credo contains some caveats. We believe it *is* now possible *in most cases* to obtain *reasonably accurate* reliability measures for software *and to have reasonable confidence that this is the case* in a particular situation, so long as the reliability levels required are *relatively modest*. The italicized words here are important — there *are* some limits to what can be achieved — but these limits are not so restrictive that they should deter you from trying to measure and predict software reliability in an industrial context.

RELIABILITY AS A PREDICTION PROBLEM

In the form in which it has been most studied, the software-reliability problem involves dynamic assessment and prediction of reliability in the presence of that reliability growth which results from fault removal. This usually involves executing a program in an environment (test or real), observing failures, and fixing the faults

0740-7459/92/0700/0034/$03.00 © IEEE

that caused the failures. The expected failure behavior is therefore reliability growth, at least in the long term, although bad fixes that introduce new faults may cause short-term reversals.

Reliability-growth models use the data collected in this procedure, usually in the form of successive execution times between failures (or sometimes the number of failures in successive, measured time intervals), to estimate current reliability and predict future reliability growth.

What's important is that all questions of practical interest involve prediction. Even if you want to know current reliability, you are really asking about the future: In this case, about the random variable T, the time to the next failure. However you express reliability — as a rate of failure occurrence, the probability of surviving a specified mission without failure, the mean time to next failure, or any other convenient way — you are trying to predict the future.

So when you ask if a model is giving accurate reliability measures, you are really asking if it is predicting accurately. This is sometimes overlooked even in the technical literature, where authors have "validated" a model by showing that it can accurately explain past failure behavior and thereby claim that it is "accurate." The ability to capture the past accurately does not necessarily imply an ability to predict accurately. As Niels Bohr said, "Prediction is difficult, especially of the future."

NEW APPROACH

Consider the simplest prediction problem: estimating current reliability. Assume you have observed the successive interfailure times $t_1, t_2, \ldots, t_{i-1}$, and you want to predict the next time to failure, T_i. To do this, you use a model to obtain an estimate, $\hat{F}_i(t)$, of the true (but unknown) distribution function $F_i(t) \equiv P(T_i < t)$. If you knew the true distribution function, you could calculate any current reliability measure.

You start the program again and wait until it fails, which is a realization t_i of the random variable T_i. You repeat this operation for some range of i values. Informally, you say that the model gives good results if what you *observe* tends to be in close agreement with what had been *predicted*.

Our approach is based on formal ways to compare prediction with observation.

Of course, this problem would be easier if you could observe the true $F_i(t)$ and compare it with the prediction, $\hat{F}_i(t)$. But you must somehow use only t_i, which is all you have. This is not a simple problem, and it is compounded because it is nonstationary — you want to predict accurately a sequence of different distributions, from each of which you will observe only one t_i.

However, you can make some simple comparisons. Suppose you need an accurate estimate of only the median of T_i, the value of T_i that is exceeded with probability 1/2. You could count what proportion of the actual t_i exceeded their predicted medians, and if this proportion is very different from 1/2, you can conclude that the median predictions were poor.

But this analysis does not tell you very much. Even if a series of predictions passed this test, it would give you confidence in the medians only. It does not tell you if other measures are accurate. What you really need is a way to detect any difference between prediction, $\hat{F}_i(t)$, and truth, $F_i(t)$.

The ability to capture the past accurately does not necessarily imply an ability to predict accurately.

***u*-plot.** Our first technique aims to detect *systematic* differences between predicted and observed failure behavior. The u-plot, described in the box on p. 36, is a generalization of the median check.

The idea behind it is very similar to bias in statistics. In statistics, you use data to calculate an estimator of a population parameter. The estimator is unbiased if its average value is equal to the (unknown) parameter.

Of course, our case is more complex because we want to estimate a function, not just a number, at each stage, and because the problem's inherent nonstationarity means we can detect prediction errors only over a sequence of different predictions.

In the event that the prediction errors are stationary — the nature of the error is the same at every stage — there will be a constant functional relationship between $\hat{F}_i(t)$ and $F_i(t)$. In such cases, you can use the u-plot to recalibrate the model — essentially training it to learn from its mistakes — and obtain more accurate predictions.

Recalibrating models to improve prediction accuracy exploits the fact that sometimes prediction errors are indeed approximately stationary. Clearly, there is always an unknown function, G_i, that will transform the predicted distribution into the true distribution. However, only sometimes is this function approximately the same in all cases: $G_i \approx G$ for all i.

When it is, you can estimate G using the comparison of earlier predictions against their corresponding observations and, by adjusting future predictions, improve their accuracy. In fact, the u-plot based on these earlier predictions is a suitable estimator of G.[2]

You adjust, or recalibrate, a model in four steps:

1. Obtain the u-plot, G_i^*, of predictions made before stage i. (It is better if G_i^* is a smoothed version of a joined-up, step-function u-plot, so our examples use a spline-smoothed version.)
2. Obtain $\hat{F}_i(t)$ from the raw (unrecalibrated) model for prediction at stage i.
3. Calculate the recalibrated prediction $\hat{F}_i^*(t) \equiv G_i^*[\hat{F}_i(t)]$.
4. Repeat at each stage i.

This procedure is truly predictive, because it uses only the past to predict the future. You must believe neither that the recalibrated predictions will be better than the raw ones, nor that the prediction errors, G_i, are approximately stationary. You can use other techniques to compare and analyze prediction accuracy, including the prequential likelihood ratio, described next, which can show if recalibration has

ASSESSING PREDICTIVE ACCURACY: *u*-PLOT

You use a *u*-plot to determine if predictions, $\hat{F}_i(t)$, are on average close to the true distribution, $F_i(t)$. For example, if you can show that the random variable T_i truly had the distribution $\hat{F}_i(t)$ — the prediction and the truth were identical — then the random variable $U_i = \hat{F}_i(T_i)$ will be uniformly distributed on (0,1). In statistics, this is called the probability integral transform.[1]

If you observe the realization t_i and calculate $u_i = \hat{F}_i(t_i)$, then u_i will be a realization of a uniform random variable. Doing this for a sequence of predictions gives a sequence $\{u_i\}$, which should look like a random sample from a uniform distribution. Any departure from the uniformity indicates some deviation between the predictions, $\{\hat{F}_i(t)\}$, and the truth, $\{F_i(t)\}$.

To find such departures, you plot a sample distribution function of $\{u_i\}$. This is a step function, constructed by placing points $u_1, u_2, \ldots, u_n$ (each of these is a number between 0 and 1) on the interval (0,1). Then, from left to right plot an increasing step function, with each step of height $1/(n+1)$ at each u on the abscissa. The resulting monotonically increasing function has a range (0,1). This is the *u*-plot.

If the $\{u_i\}$ sequence is truly uniform, the *u*-plot should be close to the line of unit slope. Any serious departure indicates inaccurate predictions. A common way to test if departures are significant is to compare them to tables for the Kolmogorov distance, the maximum vertical deviation of the plot from the line.[1] However, formal tests to prove significant departures are often unnecessary: As with the examples in this article, it is often clear from simply looking at the plots that the predictions are poor.

More important, informal inspections of *u*-plots can reveal a lot about prediction errors. For example, when predictions are consistently too optimistic, the model underestimates the chance of the next failure occurring before t (for all t). In such a *u*-plot, the us will bunch to the left of the (0,1) interval, giving a plot that is above the line of unit slope. Similarly, a *u*-plot that is entirely below the line indicates predictions that are too pessimistic. More complex *u*-plots can sometimes be interpreted in terms of the nature of the inaccurate predictions they represent.

REFERENCES

1. M.H. DeGroot, *Probability and Statistics*, Addison-Wesley, Reading, Mass., 1986.

produced better results than the raw model.

Even when a model gives predictions for a data set that have a good *u*-plot, there is no guarantee that the model is accurate in every way. In statistics, even if you have an unbiased estimator, you might still decide to use a biased one. For example, the unbiased estimator may have a large variance, so although its expected value is equal to the unknown parameter, its value in a particular case may be far from the expected value. This is the difference between what happens on average and what happens in a particular instance. Similar arguments apply to a good *u*-plot, which describes average behavior but can also mask large inaccuracies in particular predictions.

Prequential likelihood ratio. This deficiency led us to adopt a second technique, the prequential likelihood ratio, described in the box on p. 38.

The PLR lets you compare two models' abilities to predict a particular data source so that you can select the one that has been most accurate over a sequence of predictions. Unlike the *u*-plot, which is specific to a particular type of inaccuracy, the PLR is general — the model it selects as best is objectively best in a general way.[3] For example, it can detect when the predictions are too noisy and so are individually inaccurate, even when the *u*-plot looks good and the predictions seem unbiased.

We admit that both techniques are nontrivial, and you may find them very unfamiliar at first. This is not surprising, because traditional statistical methods have neglected prediction in favor of estimation. Techniques like PLR analysis have become available only recently. However, it is really very straightforward to use these techniques, which involve nothing more than simple graphical analysis.

THREE DATA SETS

Here we illustrate how to apply these techniques, using three sets of real failure data.

SS3 data set. Our first example uses the SS3 data set of 278 interfailure times, collected by John Musa.[4] This data set is unusual because all eight models we use for comparison seem to give extremely poor results as determined by the *u*-plot, but recalibration dramatically improves all eight. The box on p. 40 lists the models we used.

Figure 1a shows the raw data plotted as cumulative number of failures against total elapsed execution time. Figure 1b shows the successive predictions of the median next time to failure. The graph shows extraordinary disagreement among the models. The LV and KL models give results that are far more pessimistic than the other six. Although the graph applies to the predicted medians only, if these are inaccurate, then other measures of reliability will also be inaccurate.

In fact, the *u*-plots in Figure 1c show that *all* predictions are extremely inaccurate; plots differ from the line of unit slope with very high statistical significance. The six models that approximately agree in Figure 1b are in fact much too optimistic — their *u*-plots are almost always above the line of unit slope. LV and KL, on the other hand, are too pessimistic — their *u*-plots are generally below the line of unit slope. These results suggest the true medians lie somewhere between the two clusters in Figure 1b.

Figure 1d shows log(PLR) plotted for each model against a reference model, DU. This analysis reveals that KL and LV are significantly superior to the other six models for this data set, even though we know them to be poor, too.

Because all the models give poor *u*-plots, we have no trustworthy predictions for this data set. Therefore, all models are candidates for recalibration. Figure 1e shows that recalibration brings much closer agreement in the predicted medians. And the *u*-plots of the recalibrated predictions in Figure 1f are an enormous improvement over the raw predictions. Now none of the deviations from the unit slope is statistically significant.

Figure 1g shows the log(PLR) plots of recalibrated versus raw predictions. The improvement in predictive accuracy is

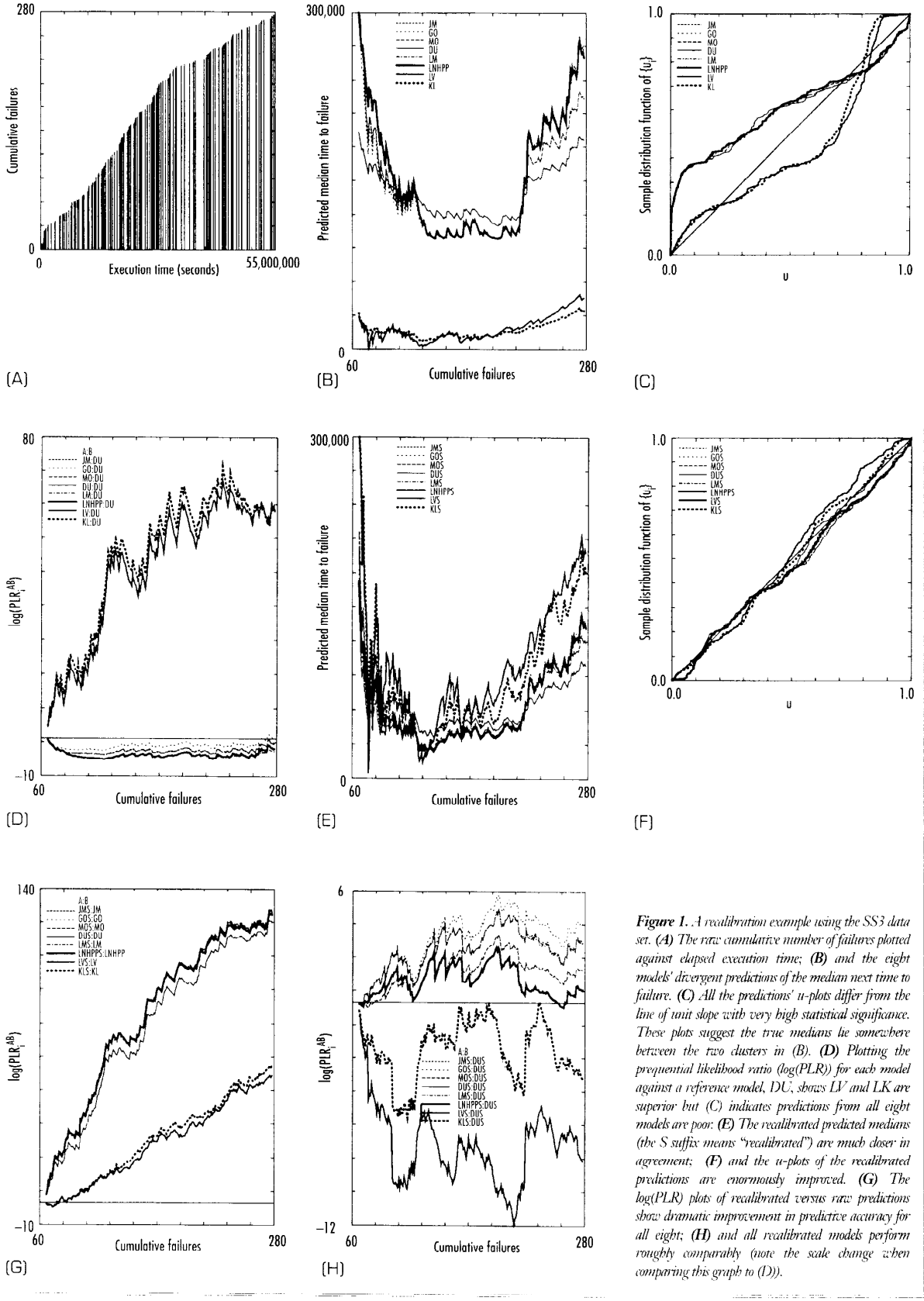


Figure 1. *A recalibration example using the SS3 data set.* ***(A)*** *The raw cumulative number of failures plotted against elapsed execution time;* ***(B)*** *and the eight models' divergent predictions of the median next time to failure.* ***(C)*** *All the predictions' u-plots differ from the line of unit slope with very high statistical significance. These plots suggest the true medians lie somewhere between the two clusters in (B).* ***(D)*** *Plotting the prequential likelihood ratio (log(PLR)) for each model against a reference model, DU, shows LV and LK are superior but (C) indicates predictions from all eight models are poor.* ***(E)*** *The recalibrated predicted medians (the S suffix means "recalibrated") are much closer in agreement;* ***(F)*** *and the u-plots of the recalibrated predictions are enormously improved.* ***(G)*** *The log(PLR) plots of recalibrated versus raw predictions show dramatic improvement in predictive accuracy for all eight;* ***(H)*** *and all recalibrated models perform roughly comparably (note the scale change when comparing this graph to (D)).*

ASSESSING PREDICTIVE ACCURACY: PLR

The prequential likelihood ratio is a way to decide which of a pair of prediction systems gives the most accurate results for a particular data source.

Figure A shows a true distribution (the probability density function) of the time to failure, T_j, and predictions of the PDF from two models, A and B. A is clearly better than B.

Eventually the failure occurs after time t_j. Obviously, you would expect t_j to lie in the main body of the true distribution (it is more likely to occur where $f_j(t)$ is larger). If you evaluate the two predictions at this value of t, you see there is a tendency for model A's prediction to be larger than model B's — model A's PDF tends to have more large values close to the large values of the true distribution than B's. This is what it means to say that A's predictions are closer to the truth than B's.

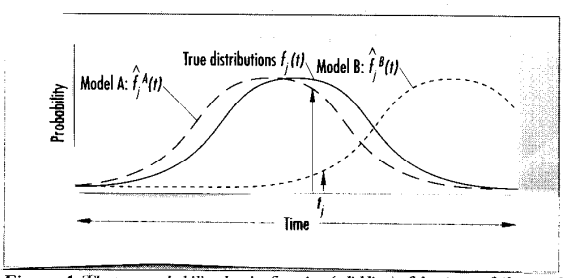


Figure A. The true probability density function (solid line) of the time to failure and the predicted PDFs from model A (long dashed line) and model B (short dashed line).

If model A's predictions are more accurate than B's, $\hat{f}_j^A(t_j)/\hat{f}_j^B(t_j)$ will tend to be larger than 1. The prequential likelihood ratio is simply a running product of such terms over many successive predictions:

$$PLR_i^{AB} = \prod_{j=k}^{j=i} \frac{\hat{f}_j^A(t_j)}{\hat{f}_j^B(t_j)}$$

and this should tend to increase with i if model A's predictions are better than B's. Conversely, B's superiority is indicated if this product shows a consistent decrease.

Of course, even if A is consistently more accurate than B, there is no guarantee that a single $\hat{f}_j^A(t_j)/\hat{f}_j^B(t_j)$ will always be greater than one. But you can expect the plot of PLR (or for convenience, its log) to exhibit overall increase with some fluctuation.

Usually, we are interested in comparing the accuracy of more than two prediction sequences. To do this, we select one arbitrarily as a reference and conduct pairwise comparisons of others against it, as above.

PLR is a completely general procedure for identifying the better of a pair of prediction sequences. Apart from the intuitive plausibility of PLR as a means of selecting between many competing prediction methods on a particular data source, support for this technique comes from a more formal asymptotic theory.[1]

REFERENCES

1. A.P. Dawid, "Statistical Theory: The Prequential Approach," *J. Royal Statistical Soc. A*, Vol. 147, 1984, pp. 278-292.

dramatic in all cases, but slightly less so for KL and LV. These two models were the best performing pair of the original eight — since they were not as bad as the other six, they had less room for improvement.

Figure 1h shows that all recalibrated models perform roughly comparably. In this graph, no single plot shows a consistent trend, compared with the corresponding plots in Figure 1d (note the scale change).

Clearly, recalibration had a dramatic effect on the accuracy of the predictions you can make about this data set. Faced with these results, a user would clearly make future predictions using one of the recalibrated versions of the models, possibly GOS (the S suffix designates a recalibrated model), although there is little difference between GOS and JMS or LMS.

As more data becomes available, of course, you must update the analysis and decide which, if any, predictions to trust. The most important advantage to this procedure is that it lets the data speak for itself and does not require the user to believe ahead of time that a model will give accurate predictions. Since such beliefs are highly questionable, this is an important new way to acquire confidence in reliability predictions.

CSR1 data set. The CSR1 data set, collected from a single-user workstation at the Centre for Software Reliability, represents some 397 user-perceived failures: genuine software failures, plus failures caused by usability problems, inadequate documentation, and so on.

Figure 2a shows the cumulative failure plot for the raw data; Figure 2b shows the median predictions from the eight models. Two things are striking: There is little evidence of reliability growth until about halfway through the data set, and again there is marked disagreement among the models when the growth does start.

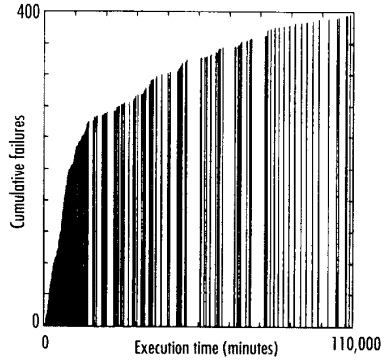
In Figure 2c, the u-plots again show that all models perform poorly — all deviations from the unit slope are significant. More to the point, there are great differences in the nature of the prediction errors: JM, GO, LM, and LNHPP are too optimistic; KL and LV are pessimistic; and MO and DU have a pronounced S-shaped u-plot, intersecting the line of unit slope at about (0.5, 0.5). This indicates MO and DU predict the median time to failure accurately, but are too optimistic in estimating the probability of small times to failure and too pessimistic in estimating large times to failure.

The PLR analysis in Figure 2d shows that KL performs best overall, with LV second. The relatively poor performance of the other models is due partly to bias, as shown by the u-plots, and in some cases partly to noise, as evidenced by the great fluctuations in the medians in Figure 2b.

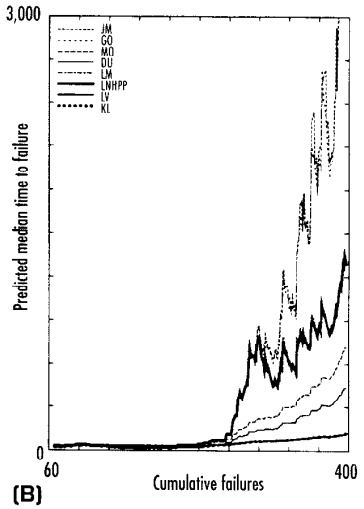
Once again, none of the raw predictions can be trusted according to the u-plot analysis, and these models are candidates for recalibration. Figure 2e shows the effect of recalibration on the median predictions. The change in medians from Figure 2b is in the right direction, according to the raw u plots. The u-plots of the recalibrated predictions in Figure 2f confirm that there has indeed been an improvement. However, only KLS has a plot that does not significantly deviate from the line of unit slope (although MOS, DUS, LNHPPS, and LVS are only just significant).

While the u-plots for MOS and DUS improved a great deal, there is little change in the medians (Figures 2b and 2e). This is because the raw medians were very accurate, but other points on the raw predictive distributions were not, and these have been improved by recalibration.

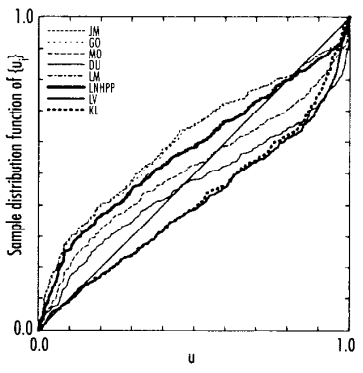
Figure 2g shows a steady increase in all log(PLR) plots and confirms that, in all



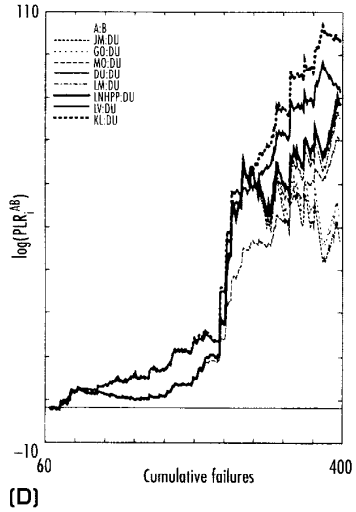
(A)



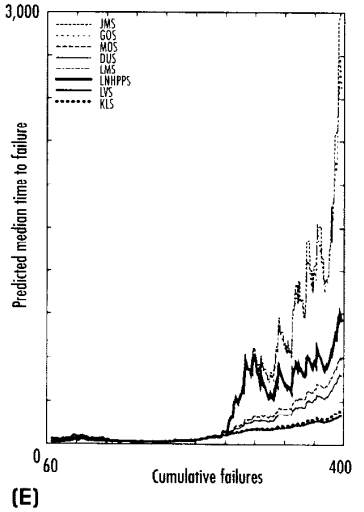
(B)



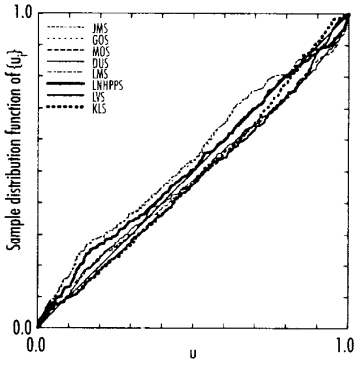
(C)



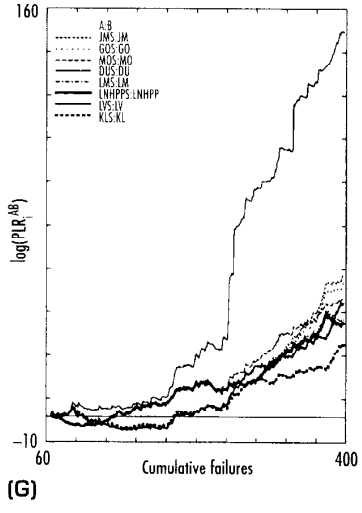
(D)



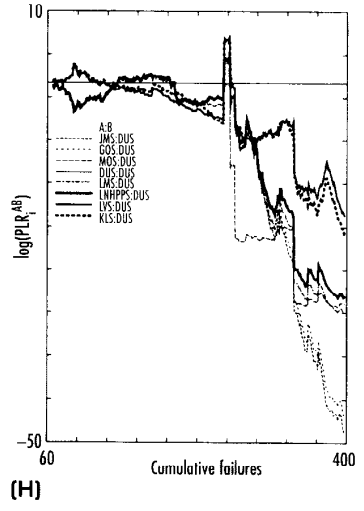
(E)



(F)



(G)



(H)

Figure 2. *Analysis of the CSR1 data set.* ***(A)*** *The cumulative failure plot for the raw data and* ***(B)*** *the median predictions from the eight models. There is marked disagreement among the models from when reliability growth starts, about halfway through the data.* ***(C)*** *The u-plots show that some models are too optimistic, others too pessimistic, and the S-shaped curves of two models indicate they predict the median time to failure accurately, but neither small nor large times to failure.* ***(D)*** *The PLR analysis indicates LK and LV are giving the most accurate predictions.* ***(E)*** *The recalibrated models' median predictions. The change from (B) is in the right direction, according to the raw u-plots.* ***(F)*** *Improvement is confirmed by the u-plots of the recalibrated predictions and the* ***(G)*** *steady increase in all log (PLR) plots. In all cases, the recalibrated predictions are superior to the raw ones, with* ***(H)*** *the best predictions coming from DUS.*

EIGHT MODELS

In applying our techniques, we used these eight reliability models:

♦ *Jelinski-Moranda* (JM): One of the earliest models, it assumes that failures occur purely at random and that all faults contribute equally to unreliability. It also assumes that fixes are perfect; thus, a program's failure rate improves by the same amount at each fix. (Z. Jelinski and P.B. Moranda, "Software Reliability Research," in *Statistical Computer Performance Evaluation*, W. Freiberger, ed., Academic Press, New York, 1972, pp. 465-484).

♦ *Goel-Okumoto* (GO): Similar to JM, except it assumes the failure rate improves continuously in time. (A.L. Goel and K. Okumoto, "Time-Dependent Error-Detection Rate Model for Software and Other Performance Measures," *IEEE Trans. Reliability*, Aug. 1979, pp. 206-211.)

♦ *Musa-Okumoto* (MO): Similar to GO, except it attempts to consider that later fixes have a smaller effect on a program's reliability than earlier ones. (J.D. Musa and K. Okumoto, "A Logarithmic Poisson Execution Time Model for Software Reliability Measurement," *Proc. Int'l Conf. Software Eng.*, IEEE CS Press, Los Alamitos, Calif., 1984, pp. 230-238.)

♦ *Duane* (DU): Developed for burn-in hardware testing, in which defective components are detected and replaced with good ones in the early days of use. Again, it assumes the failure rate changes continuously in time. (L.H. Crow, "Confidence Interval Procedures for Reliability Growth Analysis," Tech. Report 197, US Army Materiel Systems Analysis Activity, Aberdeen, Md., 1977.)

♦ *Littlewood* (LM): Similar to JM, except it assumes that different faults have different sizes (they contribute unequally to unreliability). It assumes larger faults tend to be removed early, so there is a law of diminishing returns in debugging. (B. Littlewood, "Stochastic Reliability Growth: A Model for Fault-Removal in Computer Programs and Hardware Designs," *IEEE Trans. Reliability*, Oct. 1981, pp. 313-320.)

♦ *Littlewood Nonhomogeneous Poisson Process* (LNHPP): Similar to LM, but assumes a continuous change in failure rate (instead of discrete jumps) when fixes take place. (D.R. Miller, "Exponential Order Statistic Models of Software Reliability Growth," *IEEE Trans. Software Eng.*, Jan. 1986, pp. 12-24.)

♦ *Littlewood-Verrall* (LV): Lets the size of the improvement in the failure rate at a fix vary randomly, representing the uncertainty about fault size and the efficacy of the fix. (B. Littlewood and J.L. Verrall, "A Bayesian Reliability Growth Model for Computer Software," *J. Royal Statistical Soc. C.*, Vol. 22, 1973, pp. 332-346.)

♦ *Keiller-Littlewood* (KL): Similar to LV, but uses a different mathematical form for reliability growth. (P.A. Keiller et al., "Comparison of Software Reliability Predictions," *Proc. IEEE Int'l Symp. Fault-Tolerant Computing*, IEEE CS Press, Los Alamitos, Calif., 1983, pp. 128-134.)

cases, the recalibrated predictions are superior to the raw ones. The greatest improvement is in DU, but this is largely because this model performs so poorly without recalibration. Figure 2h shows that after recalibration, the best predictions are coming from DUS, with KLS and LVS next best.

In this case, you would be advised to use DUS, bearing in mind that you must repeat this analysis at future stages in case there should be a reversal in fortunes among the models. Here, recalibration has turned the worst-performing model, DU, into the best, DUS.

Users should never believe claims that a particular model is universally reliable.

In this analysis, we have deliberately taken no account of the fact that there seems to be little evidence of reliability growth until quite late — we have blindly applied the models and the recalibration procedure as a naive user would. Clearly, you could do some simple preprocessing to detect the early stationarity of the data, such as applying simple tests for trend. You could then exclude the early part of the data in applying the growth models. However, we do not introduce this complication here to save space.

CSR2 data set. Our third data set is a subset of the second — failures that are known to be due to software faults. The data displayed in Figure 3a is notable for several extremely large interfailure times, including one of 9,549 minutes. Although you might think that these large numbers are outliers that should be excluded from the data, such exclusion can only be justified in the face of real evidence.

Unfortunately, there is no obvious statistical test for outliers for nonstationary data. Because the problem of outliers may arise frequently in practice — and it has no obvious solution — we retained the possibly extreme values to show their effect on the models and the analysis.

The median predictions in Figure 3b show that the models respond to the large times differently. Four models (JM, LNHPP, GO, and LM) respond dramatically to a large time by making the next predicted median much larger, and this relative optimism continues for many predictions before dying away. Two models (LV and KL) seem completely unaffected by large times, exhibiting a much smoother and steadier median growth. Two others (MO and DU) are affected only slightly.

The *u*-plots in Figure 3c confirm that the models most affected by large times give very optimistic predictions. LV and KL are too pessimistic. The best *u*-plots come from MO and DU, but even these are poor. Once again, we have objective evidence that all these models are giving inaccurate results.

The PLR analysis in Figure 3d is greatly influenced by the ability of the models to cope with a very large time. LV and KL cope best because they give a predictive distribution that assigns fairly high probability to large times; DU is worst. If we were to omit the 9,549 observation and join the plots, KL and LV are still superior and there is little difference among the others.

The main point, though, is that on this evidence none of the predictions can be trusted and recalibration is appropriate. Figure 3e shows the recalibrated medians, which are in much closer agreement than the raw ones, although the large observa-

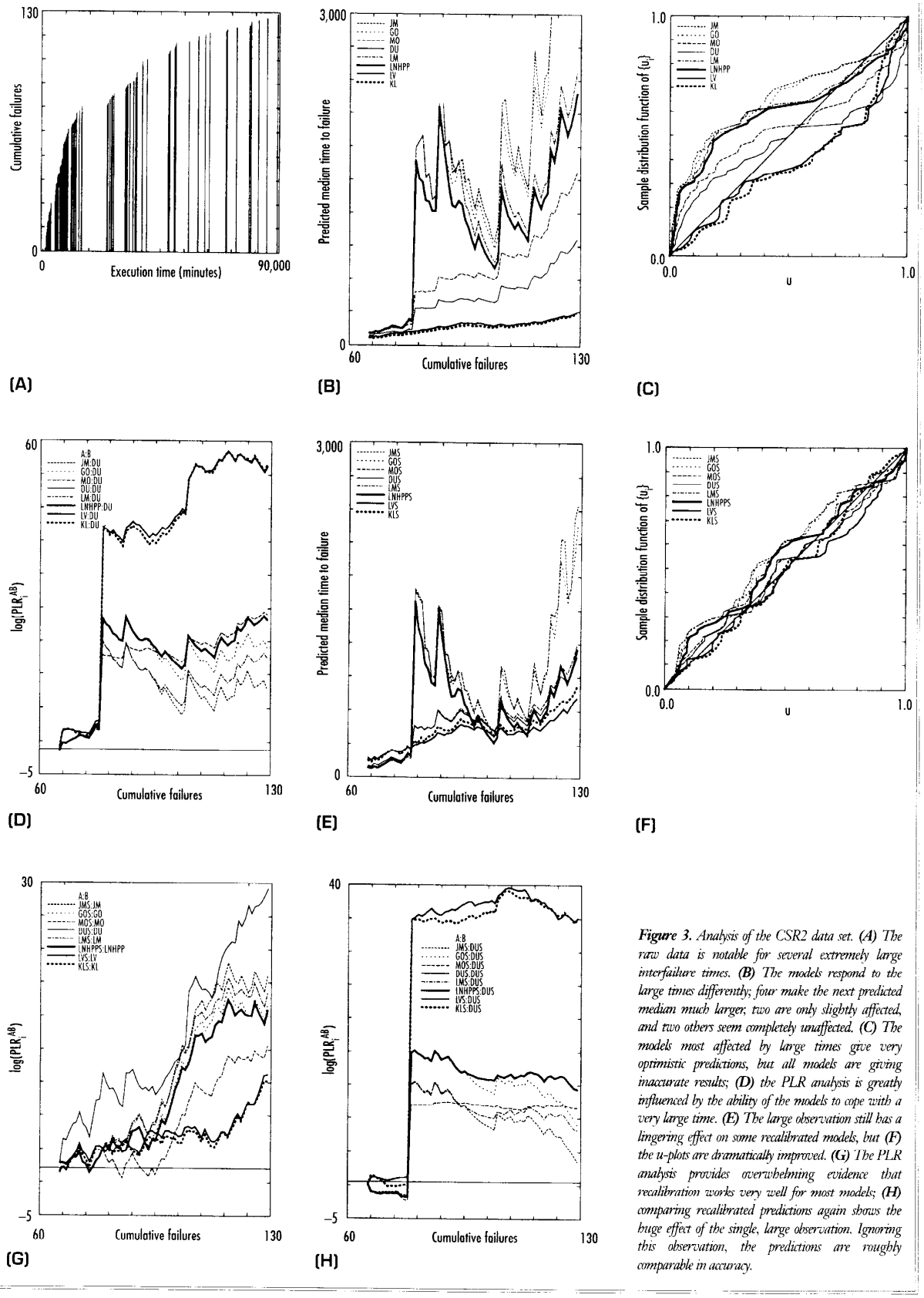


Figure 3. *Analysis of the CSR2 data set.* ***(A)*** *The raw data is notable for several extremely large interfailure times.* ***(B)*** *The models respond to the large times differently; four make the next predicted median much larger, two are only slightly affected, and two others seem completely unaffected.* ***(C)*** *The models most affected by large times give very optimistic predictions, but all models are giving inaccurate results;* ***(D)*** *the PLR analysis is greatly influenced by the ability of the models to cope with a very large time.* ***(E)*** *The large observation still has a lingering effect on some recalibrated models, but* ***(F)*** *the u-plots are dramatically improved.* ***(G)*** *The PLR analysis provides overwhelming evidence that recalibration works very well for most models;* ***(H)*** *comparing recalibrated predictions again shows the huge effect of the single, large observation. Ignoring this observation, the predictions are roughly comparable in accuracy.*

tion still has a lingering effect on some models. The *u*-plots in Figure 3f show a dramatic improvement — no deviations are statistically significant at the 10 percent level.

There is overwhelming evidence from the PLR analysis of recalibrated versus raw predictions in Figure 3g that recalibration is working very well for most models. It provides the least improvement for KL and LV, but these models needed the least improvement.

Finally, the comparison of recalibrated predictions in Figure 3h, shows again the huge effect of the single, large observation. If we take account of this single prediction, then once again the recalibrated LV and KL models are best, but all the evidence for their superiority comes merely from their ability to cope with this single observation. The different recalibrated predictions seem to be of roughly comparable accuracy if this observation is ignored, and that would seem to be a sensible procedure for anyone wishing to make further predictions on this data set.

The techniques we have described are important because they largely resolve a reliability modeling dilemma: Users are faced with a plethora of models, but none can be trusted to give accurate results always, and there is no way to select beforehand the model most appropriate for a particular application.

We cannot overemphasize that users should never believe claims for the universal validity of a particular reliability model. Indeed, we believe that the relatively poor adoption of reliability modeling has been caused in part by certain models being sold as panaceas.

We think our techniques provide a way to overcome these difficulties. We also think it is now possible to measure and predict reliability for the relatively modest levels needed in the vast majority of applications. Most important, the techniques give the user confidence that the results are sufficiently accurate for the program under examination. Users need not subscribe to dubious claims about a model's inherent plausibility to trust the reliability figures it generates.

In the examples we chose, the raw models perform badly. We deliberately chose these examples to show the power of the recalibration technique, but sometimes a model will perform reasonably well before recalibration. From a user's point of view, this is immaterial. Recalibration is easy to do and is genuinely predictive, so it should be applied as a matter of course. Then it is easy to use the analytical methods to find which version (raw or recalibrated) is performing best.

Although these techniques depend on rather novel and subtle statistical methods, we think their actual use and interpretation are comparatively straightforward. At the Centre for Software Reliability, we've developed software — available from us — to do these analyses.

Our approach is suitable for anyone contemplating measuring and predicting software reliability. Most of the time, the results are trustworthy. In rare cases in which none of the models work before or after recalibration, our techniques will serve as a warning.

Finally, a word of caution. Software is being used increasingly in safety-critical applications that demand a very high reliability. This poses enormous — possibly insurmountable — problems for system validation. We emphasize that all our techniques are designed for fairly modest reliability levels. Techniques that depend on reliability growth cannot assure very high reliability without infeasibly large observation periods. It has been argued that assuring ultrahigh reliability is even harder than we have suggested — that essentially it is impossible.[5] ♦

ACKNOWLEDGMENTS

This work was supported by the Commission of the European Communities' Strategic Programme for Research in Information Technology under project 3092 PDCS (Predictably Dependable Computing Systems).

REFERENCES

1. A.A. Abdel-Ghaly, P.Y. Chan, and B. Littlewood, "Evaluation of Competing Software Reliability Predictions," *IEEE Trans. Software Eng.* Sept. 1986, pp. 950-967.
2. S. Brocklehurst et al., "Recalibrating Software Reliability Models," *IEEE Trans. Software Eng.*, Apr. 1990, pp. 458-470.
3. A.P. Dawid, "Statistical Theory: The Prequential Approach," *J. Royal Statistical Soc. A*, Vol. 147, 1984, pp. 278-292.
4. J. Musa, "Software Reliability Data," tech. report, Data and Analysis Center for Software, Rome Air Development Center, Griffis Air Force Base, N.Y., 1979.
5. B. Littlewood, "Limits to Evaluation of Software Dependability," *Proc. 7th Annual Centre for Software Reliability Conf.*, B. Littlewood and N.E. Fenton, eds., Elsevier, London, 1991, pp. 81-110.

Sarah Brocklehurst is a research assistant at the Centre for Software Reliability at the City University, London, where she is working on the ESPRIT project. Predictably Dependable Computing Systems. Her research interests are reliability modeling and prediction.

Brocklehurst received a BSc in mathematics with statistics from the City University and is now doing doctoral work there.

Bev Littlewood is a professor of software engineering and director of the CSR at the City University. His research interests are stochastic modeling, especially as it relates to software reliability.

Littlewood received a PhD in statistics from the City University. He serves on the HSC Advisory Committee on the Safety of Nuclear Installations and the IFIP working group on reliable computing and fault tolerance. He is a fellow of the Royal Statistical Society and a member of the IEEE Computer Society.

Address questions about this article to Brocklehurst at the Centre for Software Reliability, City University, London, EC1V OHB, UK; Internet s.brocklehurst@city.ac.uk.