Department of Computer Science

Enhancing Accuracy of Software Reliability Prediction

Naixin Li and Yashwant K. Malaiya
Technical Report CS-93-112
August 20, 1993

Colorado State University
Abstract

The measurement and prediction of software reliability require the use of the Software Reliability Growth Models (SRGMs). The predictive quality can be measured by the average end-point projection error [9]. In this paper, the effects of two orthogonal classes of approaches to improve prediction capability of a SRM have been examined using a large number of data sets. The first approach is preprocessing of data to filter out short term noise. The second is to overcome the bias inherent in the model. The results show that proper application of these two approaches can be more important than the selection of the model.

1 Introduction

In order to achieve high reliability at an acceptable cost, developers need to be able to estimate the reliability of software under development, and, for management and planning purposes, they should be able to project the additional effort needed for their software to reach a certain reliability level. The reliability of software can be estimated statically or dynamically. The static approach is based on the software size, complexity and other static parameters and can be used even before the software being tested. It can however provide only preliminary estimates. The dynamic approach is based on the software failure data, which is collected during the system test phase. It can be used when the software has been tested for a while and some failure data have been collected. This approach, however, has the advantage of being able to predict the general trend of the software reliability improvement during the test phase.

Quite a few SRGMs have been proposed for the dynamic approach [1, 9, 17, 20]. The steps involved in the process of using a model to make prediction about a system’s reliability include: collecting software failure data during the testing phase and/or operational use; preprocessing the failure data to filter out noise; selecting the best SRGM for the data (project); applying the failure data to drive the model; applying the fitted model to make claims about the reliability level of the software and/or to make projection about the additional effort needed to achieve a desired reliability level. The accuracy of projection can be affected by each activity during the process, so appropriate decisions must be taken at each step.

First we consider the problem of the noise inherent in software failure data. Different inputs applied can have significantly different defect detection capabilities. Since the order in which inputs are applied is never truly random, it is possible for a few “weak” or a few “strong” inputs to be applied close to each other causing a relatively low or high failure intensity for a short duration. Ideally, a preprocessing step should filter out the short-term variations as noise while preserving the longer trend. As we would expect some smoothing generally improves predictability but excessive smoothing makes it worse. Malaiya et al [10] tried to smooth the noise by data grouping. By grouping a few adjacent failure data points, they noticed that the predictability of SRGMs improves initially as the group size increases, and then gets worse as larger group sizes are involved. For the four data sets used, they found that there was an optimal grouping size, given by the total number of defects divided by 20, or equivalently the optimal number of groups is about 20.

In this study, we did more detailed experiments with a much larger number of data sets. We also tried some other methods to smooth the noise, including windowing and data dependent grouping. Currently we have
more than forty data sets from various sources, which cover a wide range of software, including system software, real-time control, and database applications. Since grouping is not appropriate if only a small number of data points are available, we used 21 software failure data sets with 73 to over 800 data points. Table 1 is a summary of the data sets used in our experiments.

<table>
<thead>
<tr>
<th>Data</th>
<th>Ref.</th>
<th>Code Size</th>
<th>#Bugs</th>
<th>Project Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>SS1A</td>
<td>[2]</td>
<td>100,000+</td>
<td>112</td>
<td>O.S.</td>
</tr>
<tr>
<td>SS1B</td>
<td>[2]</td>
<td>100,000+</td>
<td>375</td>
<td>O.S.</td>
</tr>
<tr>
<td>SS1C</td>
<td>[2]</td>
<td>100,000+</td>
<td>277</td>
<td>O.S.</td>
</tr>
<tr>
<td>SS2</td>
<td>[2]</td>
<td>100,000+</td>
<td>192</td>
<td>Time Sharing</td>
</tr>
<tr>
<td>SS4</td>
<td>[2]</td>
<td>100,000+</td>
<td>196</td>
<td>O.S.</td>
</tr>
<tr>
<td>T1</td>
<td>[2]</td>
<td>21,700</td>
<td>136</td>
<td>Realtime</td>
</tr>
<tr>
<td>T2</td>
<td>[1]</td>
<td>N/A</td>
<td>86</td>
<td>Realtime</td>
</tr>
<tr>
<td>T3</td>
<td>[2]</td>
<td>N/A</td>
<td>207</td>
<td>N/A</td>
</tr>
<tr>
<td>T4</td>
<td>[2]</td>
<td>24,450,000</td>
<td>831</td>
<td>Realtime</td>
</tr>
<tr>
<td>T6</td>
<td>[2]</td>
<td>5,700</td>
<td>73</td>
<td>Commercial</td>
</tr>
<tr>
<td>T18</td>
<td>[3]</td>
<td>N/A</td>
<td>163</td>
<td>Military</td>
</tr>
<tr>
<td>T40</td>
<td>[2, 19]</td>
<td>180,000</td>
<td>101</td>
<td>Military</td>
</tr>
<tr>
<td>Proj1</td>
<td>[22]</td>
<td>14,000</td>
<td>132</td>
<td>Space System</td>
</tr>
<tr>
<td>Proj3</td>
<td>[22]</td>
<td>N/A</td>
<td>210</td>
<td>N/A</td>
</tr>
<tr>
<td>Proj4</td>
<td>[22]</td>
<td>N/A</td>
<td>196</td>
<td>N/A</td>
</tr>
<tr>
<td>Y1T</td>
<td>[20]</td>
<td>200,000</td>
<td>111</td>
<td>Realtime</td>
</tr>
<tr>
<td>Y1T3</td>
<td>[21]</td>
<td>870,000</td>
<td>535</td>
<td>Realtime</td>
</tr>
<tr>
<td>RP2</td>
<td>[14]</td>
<td>N/A</td>
<td>74</td>
<td>N/A</td>
</tr>
<tr>
<td>TSW</td>
<td>[5]</td>
<td>N/A</td>
<td>129</td>
<td>N/A</td>
</tr>
<tr>
<td>Usbar</td>
<td>[5]</td>
<td>N/A</td>
<td>397</td>
<td>N/A</td>
</tr>
</tbody>
</table>

The second concern is the bias that all major models have [4, 11]. While some specific models have been shown to have better fit or predictability, none of them describes the fault detection process exactly. In addition, the characteristics of individual data sets may vary because of different testing practices. One possible way to overcome this is to emphasize the most recent data points by using a weighted parameter estimation. The other approach is to adaptively adjust the projections. The results suggest that using such recalibration improves the accuracy.

Finally we have also examined combinations of preprocessing and recalibration. The results show that if near-optimal choices are made, the accuracy can be greatly enhanced.

Section 2 describes the experiments performed. Section 3 presents a summary of the results obtained using real data sets, and the observations are discussed.

### 2 The Experiments

#### 2.1 The SRGMs

<table>
<thead>
<tr>
<th>Models</th>
<th>Mean Value Function</th>
<th>Failure Intensity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exponential</td>
<td>$\beta_0 [1 - \exp(-\beta_1 t)]$</td>
<td>$\beta_0 \beta_1 e^{-\beta_1 t}$</td>
</tr>
<tr>
<td>Logarithmic</td>
<td>$\beta_0 \ln(1 + \beta_1 t)$</td>
<td>$\beta_0 \beta_1 t e^{-\beta_1 t}$</td>
</tr>
<tr>
<td>Delayed S</td>
<td>$\beta_0 [1 - (1 + \beta_1 t)^{-\beta_1}]$</td>
<td>$\beta_0 \beta_1 t e^{-\beta_1 t}$</td>
</tr>
<tr>
<td>Power</td>
<td>$\beta_0 t^{\beta_1}$</td>
<td>$\beta_0 t^{\beta_1 - 1}$</td>
</tr>
</tbody>
</table>

Four of the major SRGMs are considered here: Exponential model, logarithmic model, delayed S-shaped model, and power model. Table 2 is a summary of these models. All of these are two parameter models, and thus can be compared [11].

The past software failure data is generally in the form of defect number, failure time, all the data sets used contain the points: $<m_1, t_1>, <m_2, t_2>, <m_3, t_3>, ..., <m_n, t_n>$, where $m_i$ is the number of defects found by time $t_i$. We can use $\frac{\sum_{i=1}^{n} t_i m_i}{\sum_{i=1}^{n} t_i}$ as an estimation of the actual failure intensity $\lambda_i$ at time $t_i$. With this, we can fit the SRGMs and evaluate the model parameters $\beta_0$ and $\beta_1$. For each data set, we repeated this with subsets of the data set containing 2, 3, ..., n data points. In each case, using a SRGM, we predict the number of defects $p_i$ to be observed at the time of last failure data point $t_n$ and compare the predicted number $p_i$ with the actual number of defects found by that time $m_n$. This allows us to obtain the relative error, $\frac{|p_i - m_i|}{\lambda_i}$, corresponding to that fitted model. By averaging the relative error over time points from $t_2$ to $t_n$, we get the average error (AE) as introduced in [9, 11]. We will use AE as a measure to evaluate the predictive accuracy.

#### 2.2 Data Preprocessing

As mentioned earlier, the noise in the software failure data affects the predictability of SRGMs. The main objective here is to find some guideline that will allow us to select the smoothing method and the degree of smoothing to improve the predictability of SRGMs. Five different smoothing schemes were considered.

**Grouping with fixed group size** By grouping a fixed number of data points into one point, we expected that the noise values may compensate each other for that period and thus the noise inherent in the failure data reduced significantly. This is carried out by selecting data points $<m_1, t_1>, <m_2, t_2>, ..., <m_{i+g}, t_{i+g}>, <m_{i+g}, t_{i+g}>, ..., <m_{i+g}, t_{i+g}>$, where $g$ is the group size, which is the number of defects grouped for most of data sets we experimented.
This scheme was experimentally examined in [10], where the fitness of the projected number of defects $p_n$ against $\mu_n$ at time $t_n$ was used as a measure to evaluate the goodness of different group sizes. Here we use the variable-step predictability measure, $AE$ [9], as the measure which is a better predictability measure.

```
PROCEDURE lump_grouping();
  i: INTEGER; /* loop variable */
  data_size: INTEGER; /* #data points before lumping*/
  data_cnt: INTEGER; /* #data points after lumping */
  rising: BOOLEAN; /* rising failure intensity */
  rfail, rtime, lemda: ARRAY [1..N] of REAL;
  /* failure data before lumping*/
  fail, ftime: ARRAY [1..N] of REAL;
/* failure data after lumping */
BEGIN
  data_cnt := 0;
  rising := TRUE;
  FOR i := 1 TO data_size DO
    BEGIN
      IF lemda[i] < lemda[i+1] THEN
        BEGIN
          IF rising THEN continue ELSE
            BEGIN
              fail[data_cnt] := rfail[i];
              ftime[data_cnt] := rtime[i];
              data_cnt := data_cnt + 1;
              rising := TRUE;
            END;
        END;
      END ELSE rising := FALSE;
    END;
  END;
  fail[data_cnt] := rfail[data_size];
  /* always keep the last data point */
  ftime[data_cnt] := rtime[data_size];
END;
```

Figure 1: Procedure for lump grouping

**Grouping failure intensity lumps** When testing proceeds by focusing on one kind of defects at a time, or by applying one testing technique during a stage, the failure intensity would typically increase initially and then decrease until the next stage. When one finds a special type of fault, it is possible that he will soon find other faults of the same kind, and then as he proceeds further, since fewer such faults exist, the failure intensity decreases. Thus each lump in failure intensity could be associated with the transition from one testing stage to another during the testing process. By grouping failure intensity lumps in the data, we hope to minimize the noise associated with transition points during testing.

The data points are selected according to neighbouring failure intensities. For example, if $\lambda_i < \lambda_{i+1} < ... < \lambda_j$, and $\lambda_j > \lambda_{j+1} > ... > \lambda_k$ then all the data points between $t_i$ and $t_k$, excluding $< \mu_i, t_i >$ and $< \mu_k, t_k >$, would be dropped due to grouping for model fitting. The procedure in Figure 1 describes lump grouping.

**Windowing** Instead of grouping a few data points into one, in this method all the data points are kept, but the failure intensity is approximated differently at each point. $\lambda_i$ is estimated as $\frac{\mu_{i+1} - \mu_i}{t_{i+1} - t_i}$ where $w$ is an adjustable factor called window width. Clearly the failure intensity calculated in this way will be smoother than the earlier approaches.

### 2.3 Weighted least square estimation

Normal least square parameter estimation approach gives equal weight to each data point when a model is fitted. This results in a fitted model that best fits the data. When reliability projection or reliability estimation is our major concern, we expect that if we get a better fit to the later data point, we will get better predictions for the future points in time. If during the process of fitting a model, we give more weights to the recent data points, we would expect a better predictive accuracy. It can be done as described below.

Suppose $(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)$ are the data points and $Y = a + bX$ is a model to be fitted to this data set. Then the actual value of $y_k$ at each data point can be written as $y_k = a + b \times x_k + e_i$

where $e_i$ is the error related to the model at point $x_i$.

With weighted least square, the weighted sum of squares of errors as given by:

$$\sum_{i=1}^{n} \beta_i \times \beta_i$$

where $\beta_i$ is the weight associated with data point $x_i$. For normal least square approach, $\beta_i = 1$ for all $i \in [1..n]$.

The weight $\beta_i$ can be chosen in numerous ways. We experimented with weights as a linear function of time $t_i$, as a linear function of cumulative number of failures $\mu_i$, and as a linear function of data points $i$. The last one was found to be better than the other two schemes, which is described by:

$$\beta_i = c + \epsilon \times i$$

Let the total weights be the same as unweighted scheme, i.e.:

$$\sum_{i=1}^{n} c + \epsilon \times i = \sum_{i=1}^{n} 1 = n$$

then $c$ is determined once $c$ is chosen. The parameter $c$ controls the weighting scheme. Setting $c = 1$ results in normal least square; setting $c = 0$ results in biggest differences in weights. In our experiments we adjusted the value of $c$ from 1 to 0 in step of -0.1 for each data set.
2.4 Adaptive Approach

It was noticed [4, 6, 11, 23] that some models have a tendency to overestimate or underestimate the reliability, which make them good candidates for the adaptive approach (also called recalibration [4]). Once a model is fitted, it can be used to predict the reliability of software at some future time. However there is no way to tell exactly how close a prediction will be to the actual value. We can, however, measure the bias of the estimation of the software reliability at each past point of time. These biases can then be used to adjust the projection made using the model. We tried to adjust the prediction by using linear regression to project the trend of bias with respect to time \( t_i \) or the cumulative number of failures \( \mu_i \). We also tried to simply take the average of bias values during the past and deducting this average bias from the model projection. This later approach was found to be superior to the other two more complicated methods.

To observe the effects of the adaptive approach, we repeat each experiment with and without the adjustment. This also allows us to see the effect of combining this approach with the earlier noise smoothing techniques. For each data set in Table 1, we experimented with all the methods mentioned in this section. The result are discussed in next section.

3 Results and Observations

The results of the experiments are shown in various plots. Since there are many sets of data, four SRGMs, and a few different methods for preprocessing and bias adjustment, there are too many plots to be included here. We show only some representative plots and present the summary information along with the observations and discussions.

While some SRGMs usually works better than others, there is no single SRGM that will work best for every possible software failure data all the time. It is also true that there is no single scheme introduced in section 2 will work best for every case all the time. If a scheme works best in a few situations but does poorly in others, we cannot claim that scheme as a good scheme. The first thing we tried to observe is to find a guideline to be followed for each scheme to get an overall good result for that scheme. Only then can we compare the performance of different schemes using their specific guidelines. Our objective is to to develop approaches that are independent of individual data set and, will generally give good predictive quality.

**Fixed size grouping** Three possible scenarios appear in this case as represented by Figure 2. The X-axis represents the number of data points (in most cases the number of defects) grouped together. The Y-axis represents the average relative error in percent for all the plots in this paper. Figure 2(a,d) are plots...
Figure 3: Representative plots for lump grouping

for data set T1. These also represents the observations for data sets T18, SS1A and some other data sets. In this class, the projection error decreases initially as the group size increases. As group size further increases, the projection error starts to increase. This increases can be very significant for some SRGMs for some data sets. Figure 2(b,e) are plots for data set TSW and they also represent the result for data sets SS4 and some other data sets. For this class the projection error decreases initially as the group size increases, and then remains relatively flat, and hence there is a wide range of preferred group sizes. Figure 2(c,f) are plots for T2 and representative of data sets HP2 and some other data sets. There the initial projection error is good and there is little significant improvement through grouping. Considering all the three class of data sets, the optimal grouping which consistently results in smaller prediction error is to have about 50 groups after grouping if no recalibration is involved, or to have about 20 groups if recalibration is used. The effect of recalibration on the range of optimal group size is obvious from the plots.

The logarithmic model gives the best predictions if no grouping is made. It is also least sensitive to grouping, i.e., the predictive quality changes little when the grouping size varies. The Delayed S-shape model is the most sensitive to group size. A little grouping can help a lot but grouping too many points together can severely reduce the predictive accuracy.

Without grouping, the performance of four models differ quite significantly for some data sets, and there is no model which works best in all the cases, though “on the average” the logarithmic model is apparently superior to the others and the exponential model is second to the logarithmic model. If the above mentioned optimal grouping is followed, and then the differences between the models become less significant.

**Grouping lumps of failure intensity** Figure 3 shows the effect of lump grouping on AE for data sets T18, TSW and T2. The number of lumping steps possible is data dependent. In general, larger data sets allow more lumping steps. Without recalibration, lump grouping twice gives overall best predictions. With recalibration, more lump grouping can enhance the prediction accuracy further for most of the data sets. In our experiments, we did as many lump groupings as allowed, which provides only a few data points in most cases for these plots.

**Windowing** Figure 4 shows representative plots for the effect of windowing. Here the X-axis represents the window width in the number of data points. The improvement in predictive accuracy is obvious when the window width increases initially, but then the average prediction error stays relatively flat with little fluctuation. The best window width can be determined by the total number of defects found (data points) divided by 20, or about the same as the best group size with recalibration.
Like fixed size grouping and lump grouping, the windowing approach gives significant improvements over the raw data and is more stable than the other two approaches in that a wider range of window width can be regarded as optimal.

Recalibration enhances the power model significantly for some data sets, but weakens it for some other data sets. Overall there seems to be no need to combine recalibration with the windowing approach. The performance achieved with windowing is similar to that with fixed-size grouping and is not as good as with lump grouping.

**Weighted least square** Figure 5 shows representative plots on the effect of the weighted least square approach. The origin on the X axis corresponds to equal weights for every data point; moving towards the 1 point on X axis corresponds to the recent past data points being given increasing weights. It is clear from the plots that the weighted least square approach helps little in predictive accuracy. It is not recommended based on the experiments we have done.

**Adaptive approach** The effect of adaptive approach (recalibration) can be observed in Figures 2 through Figure 5. This scheme reduces the average predictive error as seen by comparing the plots with and without recalibration. This is true especially for situations with large prediction errors, which are probably caused by bias in prediction and are hence more suitable for the adaptive approach.

The improvement by the adaptive approach alone is less than that achieved by the grouping and windowing approaches alone. Based on our observations, recalibration is specially recommended for use with fixed size grouping. In addition to reducing AE, it makes overgrouping less likely. It is not effective in conjunction with the windowing approach.

**Best Model and Best Approach** Earlier work [11] showed that the logarithmic model is superior to the other models, and the exponential model is close to the logarithmic model. It also showed that the power model is very inconsistent across different data set, and that the delayed S-shape model performed the worst. This is again observed in our experiments when no enhancing techniques are used (plots with horizontal axis at 0). However this study shows that, different models have different degrees of enhancement possible when the above techniques are used. We should take this into consideration when comparing different models.

Table 3 shows the effect of preprocessing and bias adjustment on the prediction accuracy for different data sets. In Table 3, all the values are relative AEs in percent; column Raw represents the AE obtained with-
out using any enhancing techniques; FSG corresponds to fixed-size grouping using 50 grouped data points; LG corresponds to lump grouping twice; Win. corresponds to windowing with the window width given by the initial number of data points divided by 20; Wei corresponds to the weighted scheme with half of the total weights evenly distributed; Rec. stands for Recalibration; for fixed-size grouping with recalibration (F&R), 20 data points were used with recalibration; for lump grouping with recalibration (L&R), we did lumping as many times as possible followed by recalibration.

Table 4 gives the average AEs over all the data sets. Windowing, fixed-size grouping and lump grouping with recalibration can dramatically improve the prediction accuracy for “weak” SRGMs. When the enhancing techniques are properly applied, the different models perform close to each other. For instance, without enhancing, the Power model is much worse than the Logarithmic model; but with lumping and recalibration, the overall prediction error with the Power model is even lower than that with the Logarithmic model. Thus the proper application of the enhancing techniques can be more important than the selection of a SRGM.

Figure 6 depicts the effect of preprocessing (fixed-size grouping and lump grouping) followed by recalibration on AE. For comparison the effect of recalibration alone without grouping is also shown. It is easy to see that lump grouping with recalibration is the best way to achieve low prediction error for all four models.

Since lump grouping is superior to the other preprocessing methods considered, the intuitive motivation behind lump grouping is provided here. The very purpose of software failure data preprocessing is to filter out the noise and yet to retain the general trend in the data. By grouping, the noise manifested as fluctuations in failure intensities are averaged out by grouping a few data points together. With fixed size grouping, this is done somehow blindly in the sense that it does not take into consideration any characteristics of individual data. If we are grouping a data set by hand to smooth the noise, it is natural to group the most noisy data points, which are seen as either sharp peaks or deep valleys in the failure intensity plots. We did try to group a few data points manually in this way and found that it worked well. Then we came up with the idea of grouping data points with rising or decreasing failure intensities and lump grouping. Since the other two methods were not as effective, only lump grouping was reported here.
Table 3: Summary of the Relative AEs

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>S51A</td>
<td>EKP</td>
<td>150 155 165 170</td>
<td>180 190 200 210</td>
<td>20 21 22 23</td>
<td>10 11 12 13</td>
<td>9 10 11 12</td>
<td>7 8 9 10</td>
<td>5 6 7 8</td>
<td>3 4 5 6</td>
<td>2 3 4 5</td>
<td>1 2 3 4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>POW</td>
<td>253 249 245 241</td>
<td>220 216 212 208</td>
<td>18 17 16 15</td>
<td>14 13 12 11</td>
<td>10 9 8 7</td>
<td>6 5 4 3</td>
<td>2 1 2 3</td>
<td>1 0 1 2</td>
<td>0 1 2 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>DelS</td>
<td>15 20 25 30</td>
<td>26 31 36 41</td>
<td>22 27 32 37</td>
<td>21 26 31 36</td>
<td>20 25 30 35</td>
<td>19 24 29 34</td>
<td>18 23 28 33</td>
<td>17 22 27 32</td>
<td>16 21 26 31</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>LOG</td>
<td>11 10 9 8</td>
<td>7 6 5 4</td>
<td>5 4 3 2</td>
<td>4 3 2 1</td>
<td>3 2 1 0</td>
<td>2 1 0 0</td>
<td>1 1 0 0</td>
<td>0 0 0 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S51B</td>
<td>EKP</td>
<td>150 155 165 170</td>
<td>180 190 200 210</td>
<td>20 21 22 23</td>
<td>10 11 12 13</td>
<td>9 10 11 12</td>
<td>7 8 9 10</td>
<td>5 6 7 8</td>
<td>3 4 5 6</td>
<td>2 3 4 5</td>
<td>1 2 3 4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>POW</td>
<td>253 249 245 241</td>
<td>220 216 212 208</td>
<td>18 17 16 15</td>
<td>14 13 12 11</td>
<td>10 9 8 7</td>
<td>6 5 4 3</td>
<td>2 1 2 3</td>
<td>1 0 1 2</td>
<td>0 1 2 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>DelS</td>
<td>15 20 25 30</td>
<td>26 31 36 41</td>
<td>22 27 32 37</td>
<td>21 26 31 36</td>
<td>20 25 30 35</td>
<td>19 24 29 34</td>
<td>18 23 28 33</td>
<td>17 22 27 32</td>
<td>16 21 26 31</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>LOG</td>
<td>11 10 9 8</td>
<td>7 6 5 4</td>
<td>5 4 3 2</td>
<td>4 3 2 1</td>
<td>3 2 1 0</td>
<td>2 1 0 0</td>
<td>1 1 0 0</td>
<td>0 0 0 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>S51C</td>
<td>EKP</td>
<td>150 155 165 170</td>
<td>180 190 200 210</td>
<td>20 21 22 23</td>
<td>10 11 12 13</td>
<td>9 10 11 12</td>
<td>7 8 9 10</td>
<td>5 6 7 8</td>
<td>3 4 5 6</td>
<td>2 3 4 5</td>
<td>1 2 3 4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>POW</td>
<td>253 249 245 241</td>
<td>220 216 212 208</td>
<td>18 17 16 15</td>
<td>14 13 12 11</td>
<td>10 9 8 7</td>
<td>6 5 4 3</td>
<td>2 1 2 3</td>
<td>1 0 1 2</td>
<td>0 1 2 3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>DelS</td>
<td>15 20 25 30</td>
<td>26 31 36 41</td>
<td>22 27 32 37</td>
<td>21 26 31 36</td>
<td>20 25 30 35</td>
<td>19 24 29 34</td>
<td>18 23 28 33</td>
<td>17 22 27 32</td>
<td>16 21 26 31</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>LOG</td>
<td>11 10 9 8</td>
<td>7 6 5 4</td>
<td>5 4 3 2</td>
<td>4 3 2 1</td>
<td>3 2 1 0</td>
<td>2 1 0 0</td>
<td>1 1 0 0</td>
<td>0 0 0 0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Notes on Table 3 and 4:
1. Raw: AE obtained without using any enhancing techniques.
2. LG: Fixed-size grouping using 50 grouped points.
3. LG & R: Zoom grouping with recalibration.
4. LG & R: Zoom grouping with recalibration.
5. LG: Zoom grouping with recalibration.
6. LG & R: Zoom grouping with recalibration.
7. LG & R: Zoom grouping with recalibration.
8. LG & R: Zoom grouping with recalibration.

Table 4: Average AEs Over All the Data Sets

<table>
<thead>
<tr>
<th>Model</th>
<th>Raw</th>
<th>LG</th>
<th>LG</th>
<th>Rec.</th>
<th>LG &amp; R</th>
<th>LG &amp; R</th>
<th>LG &amp; R</th>
<th>LG &amp; R</th>
</tr>
</thead>
<tbody>
<tr>
<td>EKP</td>
<td>111</td>
<td>114</td>
<td>115</td>
<td>118</td>
<td>119</td>
<td>120</td>
<td>121</td>
<td>122</td>
</tr>
<tr>
<td>POW</td>
<td>174</td>
<td>177</td>
<td>180</td>
<td>183</td>
<td>186</td>
<td>189</td>
<td>192</td>
<td>195</td>
</tr>
<tr>
<td>DsS</td>
<td>126</td>
<td>130</td>
<td>134</td>
<td>138</td>
<td>142</td>
<td>146</td>
<td>150</td>
<td>154</td>
</tr>
<tr>
<td>LOG</td>
<td>46</td>
<td>49</td>
<td>52</td>
<td>55</td>
<td>58</td>
<td>61</td>
<td>64</td>
<td>67</td>
</tr>
</tbody>
</table>

Figure 6: The effect of pre-processing and recalibration

4 Concluding Remarks

As we can see from the experimental results, different SRGMs have different characteristics with respect to the enhancing techniques. Logarithmic model gives best or close to the best predictions in most of the cases with raw data. Its performance is improved only modestly using the techniques considered here. For other models, windowing, zoom grouping or fixed size grouping with recalibration can significantly reduce the average prediction error. Zoom grouping combined with recalibration leads to the best prediction.

One may argue that a comparison is fair only if different SRGMs are compared using their optimal achievable performance. The logarithmic model is indeed superior to raw software failure data and the model are used directly to make projections. However proper application of the enhancing techniques can render the difference among the models to such a small degree that it can be more important to select proper combination of the enhancing techniques than to choose a SRGM.
5 Acknowledgements

We would like to thank A. Nikora, S. Brocklehurst, N. Schneidewind, Y. Tohma for providing us some of the data sets used in this study.

References