Compound Poisson and Non homogeneous Software Reliability Models

ASSE 2010 - 11th Argentine Symposium on Software Engineering

Néstor R. Barraza

School of Engineering, University of Buenos Aires
Software Development Phases

Analysis

Design

Coding

Testing

Operation and Maintenance

Quality:
Standards and Methods

ISO-IEC 9126 (1991)
CMMI (1991)

Software Analysis

Reliability

Metrics
Software Development Phases. Reliability

- Analysis
- Design
- Coding
- Testing
- Operation

Software Reliability
- Growth Models
- Markov Chains [7]
- Clusters [21]
- ...
Probability of failures is a stochastic process $P(N(t) = n)$.

Number of failures are predicted as the expected value $\mu(t) = E[N(t)]$. 
## Software Reliability Growth Models

<table>
<thead>
<tr>
<th>Model</th>
<th>$\mu(t)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delayed S-shaped</td>
<td>$a(1 - (1 + b \cdot t)e^{-b \cdot t})$</td>
</tr>
<tr>
<td>Log Power</td>
<td>$\alpha \ln^\beta(1 + t)$</td>
</tr>
<tr>
<td>Gomperts</td>
<td>$a(b^c^t)$</td>
</tr>
<tr>
<td>Yamada Exponential</td>
<td>$a(1 - e^{-b \cdot c \cdot (1 - e^{-d \cdot t})})$</td>
</tr>
</tbody>
</table>
Non-homogeneous Poisson process models

\[
P(N(t) = n) = \frac{\lambda(t)^n}{n!} \exp(-\lambda(t))
\]

\[
\lambda(t) = a(1 - \exp(-bt)) \quad \text{Goel – Okumoto}
\]

\[
\lambda(t) = \frac{1}{\theta} \ln(\lambda_0 \theta t + 1) \quad \text{Musa – Okumoto}
\]

\[
E[N(t)] = \lambda(t)
\]
Compound Poisson process models

\[
P(N(t) = n) = \sum_{k=1}^{m} \frac{(\lambda t)^k}{k!} \exp(-\lambda t) f^k(X_1 + X_2 + \cdots + X_k = n)
\]

\[
E[N(t)] = \lambda t E[X]
\]
Compound Poisson. Compounding Distribution

\[ P(X) = (1 - r)r^{x-1} \] Geometric. Sahinoglu’s first proposal.

\[ P(X) = \frac{a^X}{X! \frac{\exp(-a)}{1+\exp(a)}} \] Poisson Truncated at Zero. This proposal.
Compound Poisson. Parameters Estimation

Mean value unbiased estimator

\[ \hat{\lambda} = \frac{m}{\Delta t} \]

\[ E[\hat{X}] = \frac{n}{m} \]

\[ E[N(t)] = \hat{\lambda} t \hat{E}[X] = \frac{n}{\Delta t} t \]

Simple failure rate
Poisson Truncated at Zero. Parameter Estimation

\[ \hat{a} = \frac{1}{m} \sum_{i=1}^{m} x_i \quad \text{Plackett} \]

\[ \hat{a} = \frac{n^{\{n-1\}}}{\{n\} \{m\}} \quad \text{Tate and Goen (Unbiased Minimum Variance)} \]

where \( \{n\} \{m\} \) is the Stirling number of the second kind
Poisson Truncated at Zero. Mode Estimator

Diminution in the cluster size as the testing phase progresses is better taken into account by the mode estimator. Also has faster adaptation.
Software Reliability models prediction

![Graph showing actual and predicted remaining failures](image1)

![Graph showing data cluster size progress](image2)
Software Reliability models prediction

Actual and predicted remaining failures

Data 8 cluster sizes progress
### Software Reliability Growth Models Comparison

<table>
<thead>
<tr>
<th>Non-Homogeneous</th>
<th>Compound</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Non-linear $m(t)$</strong></td>
<td><strong>Linear $m(t)$</strong></td>
</tr>
<tr>
<td>ML convergence problem</td>
<td>Available for several Estimation methods</td>
</tr>
<tr>
<td>Input: Mean time between failures</td>
<td>Input: Arrival rate and failures clusters size</td>
</tr>
<tr>
<td>$m(t)$ adjusted by inhomogeneity</td>
<td>$m(t)$ adjusted by the cluster size expectation</td>
</tr>
<tr>
<td>Parameters are estimated simultaneously</td>
<td>Parameters are estimated independently</td>
</tr>
</tbody>
</table>
## Software Reliability Growth Models Comparison

### Non-Homogeneous
- **Non linear** $m(t)$
- **ML convergence problem**
  - Input: Mean time between failures
  - $m(t)$ adjusted by inhomogeneity
  - Parameters are estimated simultaneously

### Compound
- **Linear** $m(t)$
- Available for several Estimation methods
  - Input: Arrival rate and failures clusters size
  - $m(t)$ adjusted by the cluster size expectation
  - Parameters are estimated independently
## Software Reliability Growth Models Comparison

### Non-Homogeneous
- Non-linear $m(t)$
- ML convergence problem
- Input: Mean time between failures
  - $m(t)$ adjusted by inhomogeneity
- Parameters are estimated simultaneously

### Compound
- Linear $m(t)$
- Available for several Estimation methods
- Input: Arrival rate and failures clusters size
  - $m(t)$ adjusted by the cluster size expectation
- Parameters are estimated independently
## Software Reliability Growth Models Comparison

<table>
<thead>
<tr>
<th>Non-Homogeneous</th>
<th>Compound</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1" alt="Non linear $m(t)$" /></td>
<td><img src="image2" alt="Linear $m(t)$" /></td>
</tr>
<tr>
<td><img src="image3" alt="ML convergence problem" /></td>
<td><img src="image4" alt="Available for several Estimation methods" /></td>
</tr>
<tr>
<td><img src="image5" alt="Input: Mean time between failures" /></td>
<td><img src="image6" alt="Input: Arrival rate and failures clusters size" /></td>
</tr>
<tr>
<td><img src="image7" alt="Input: $m(t)$ adjusted by inhomogeneity" /></td>
<td><img src="image8" alt="Input: $m(t)$ adjusted by the cluster size expectation" /></td>
</tr>
<tr>
<td><img src="image9" alt="Parameters are estimated simultaneously" /></td>
<td><img src="image10" alt="Parameters are estimated independently" /></td>
</tr>
</tbody>
</table>
Software Reliability Growth Models Comparison

Non-Homogeneous

- Non linear $m(t)$
- ML convergence problem
- Input: Mean time between failures
- $m(t)$ adjusted by inhomogeneity
- Parameters are estimated simultaneously

Compound

- Linear $m(t)$
- Available for several Estimation methods
- Input: Arrival rate and failures clusters size
- $m(t)$ adjusted by the cluster size expectation
- Parameters are estimated independently
# Software Reliability Growth Models Comparison

<table>
<thead>
<tr>
<th>Non-Homogeneous</th>
<th>Compound</th>
</tr>
</thead>
<tbody>
<tr>
<td>Non-linear $m(t)$</td>
<td>Linear $m(t)$</td>
</tr>
<tr>
<td>ML convergence problem</td>
<td>Available for several Estimation methods</td>
</tr>
<tr>
<td>Input: Mean time between failures</td>
<td>Input: Arrival rate and failures clusters size</td>
</tr>
<tr>
<td>$m(t)$ adjusted by inhomogeneity</td>
<td>$m(t)$ adjusted by the cluster size expectation</td>
</tr>
<tr>
<td>Parameters are estimated simultaneously</td>
<td>Parameters are estimated independently</td>
</tr>
</tbody>
</table>
### Software Reliability Growth Models Comparison

<table>
<thead>
<tr>
<th>Non-Homogeneous</th>
<th>Compound</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Non linear $m(t)$</td>
<td>- Linear $m(t)$</td>
</tr>
<tr>
<td>- ML convergence problem</td>
<td>- Available for several Estimation methods</td>
</tr>
<tr>
<td>- Input: Mean time between failures</td>
<td>- Input: Arrival rate and failures clusters size</td>
</tr>
<tr>
<td>- $m(t)$ adjusted by inhomogeneity</td>
<td>- $m(t)$ adjusted by the cluster size expectation</td>
</tr>
<tr>
<td>- Parameters are estimated simultaneously</td>
<td>- Parameters are estimated independently</td>
</tr>
</tbody>
</table>
## Software Reliability Growth Models Comparison

### Non-Homogeneous
- Non linear $m(t)$
- ML convergence problem
- Input: Mean time between failures
- $m(t)$ adjusted by inhomogeneity
- Parameters are estimated simultaneously

### Compound
- Linear $m(t)$
- Available for several Estimation methods
- Input: Arrival rate and failures clusters size
- $m(t)$ adjusted by the cluster size expectation
- Parameters are estimated independently
Software Reliability Growth Models Comparison

**Non-Homogeneous**
- Non-linear $m(t)$
- ML convergence problem
- Input: Mean time between failures
- $m(t)$ adjusted by inhomogenity
- Parameters are estimated simultaneously

**Compound**
- Linear $m(t)$
- Available for several Estimation methods
- Input: Arrival rate and failures clusters size
- $m(t)$ adjusted by the cluster size expectation
- Parameters are estimated independently
Software Reliability Growth Models Comparison

**Non-Homogeneous**
- Non linear $m(t)$
- ML convergence problem
- Input: Mean time between failures
- $m(t)$ adjusted by inhomogeneity
- Parameters are estimated simultaneously

**Compound**
- Linear $m(t)$
- Available for several Estimation methods
- Input: Arrival rate and failures clusters size
- $m(t)$ adjusted by the cluster size expectation
- Parameters are estimated independently
Compound Non Homogeneous Poisson?

\[ P(N(t) = n) = \sum_{k=1}^{m} \frac{(\lambda(t))^k}{k!} \exp(-\lambda(t)) f^k(X_1 + X_2 + \cdots + X_k = n) \]

\[ E[N(t)] = \hat{\lambda}(t) \hat{E}[X] \]
# Software Reliability and Quality Models

<table>
<thead>
<tr>
<th>SEI CMM Level</th>
<th>Multiplier</th>
</tr>
</thead>
<tbody>
<tr>
<td>Level 1</td>
<td>1.5</td>
</tr>
<tr>
<td>Level 2</td>
<td>1</td>
</tr>
<tr>
<td>Level 3</td>
<td>0.4</td>
</tr>
<tr>
<td>Level 4</td>
<td>0.1</td>
</tr>
<tr>
<td>Level 5</td>
<td>0.05</td>
</tr>
</tbody>
</table>

Parameter adjustment as proposed in [11].
Software Reliability Actual Practice

- Predicted by experts (too conservative or too optimistic)
- Lack of failures reports
- Failures history ignored
- Released time badly estimated
- Released when major bugs have been fixed
Software Reliability Actual Practice

- Predicted by experts (too conservative or too optimistic)
- Lack of failures reports
  - Failures history ignored
  - Released time badly estimated
  - Released when major bugs have been fixed
Software Reliability Actual Practice

- Predicted by experts (too conservative or too optimistic)
- Lack of failures reports
- Failures history ignored
- Released time badly estimated
- Released when major bugs have been fixed
Software Reliability Actual Practice

- Predicted by experts (too conservative or too optimistic)
- Lack of failures reports
- Failures history ignored
- Released time badly estimated
- Released when major bugs have been fixed
Software Reliability Actual Practice

- Predicted by experts (too conservative or too optimistic)
- Lack of failures reports
- Failures history ignored
- Released time badly estimated
- Released when major bugs have been fixed
Conclusions

- A comparison between two Poisson based models has been shown
- The behavior of the mode estimator has been studied
- Results for real data were analyzed
- Advantages and Disadvantages has been pointed out
**Bibliography**


- Barraza N. R., Pfefferman J. D., Cernuschi-Frías B., Cernuschi F.: An application of the chains-of-rare-events model to software development failure prediction, in Proc. 5th Int. Conf. Reliable Software Technologies. ser. Lecture


Goel N. L., Okumoto K., Time-dependent error detection rate model for software reliability and other performances measures, IEEE Trans. on Reliability 28, 206-211 (1979)


The Data & Analysis Center for Software, http://www.thedacs.com
