International Journal of Reliability, Quality and Safety Engineering
© World Scientific Publishing Company

MAXIMUM LIKELIHOOD ESTIMATES FOR THE HYPERGEOMETRIC SOFTWARE RELIABILITY MODEL

FRANK PADBERG
Fakultät für Informatik, Universität Karlsruhe
Am Fasanengarten 5, 76131 Karlsruhe, Germany
E-mail: padberg @ira.uka.de

Received January 2002
Revised July 2002

We present a fast and exact novel algorithm to compute maximum likelihood estimates for the number of defects initially contained in a software, using the hypergeometric software reliability model. The algorithm is based on a rigorous and comprehensive mathematical analysis of the growth behavior of the likelihood function for the hypergeometric model. We also study a numerical example taken from the literature and compare the estimate obtained in the hypergeometric model with the estimates obtained in other reliability models. The hypergeometric estimate is highly accurate.

1. Introduction

The number of defects contained in a software is a quality attribute which can't be determined exactly. Therefore, it must be estimated in practice. Probabilistic software reliability modeling regards the software tests performed during development as a large random experiment. The random experiment is formally modelled as a stochastic process which is parameterized in some way by the number m of defects initally contained in the software. The results observed in the software tests then are used to estimate the value of m by applying suitable statistical techniques.

In the *hypergeometric software reliability model*, [5-14,17-20] testing is a stagewise process. The result of software test k consists of the number w_k of defects which were detected in the test and the number x_k of those defects which were newly detected. A defect is newly detected if it has not already been detected in one of the earlier tests. The hypergeometric model is based on an urn model: each individual test is modelled using a suitable hypergeometric probability distribution which depends on the number m of defects initially contained in the software and the number c_{k-1} of different defects detected in the previous $k-1$ tests. The distributions for the individual tests are combined into a stochastic process which turns out to be a Markov chain when properly formalized, see section 2.

The hypergeometric model is focused on defect content estimation instead of modelling the failure process.[21] Several features make the hypergeometric model useful and appealing to software engineers in practice: Being a conceptually simple and stagewise model, it is applicable to a wide range of development projects, including the important case of distributed development and testing. The model does not assume that defects are removed immediately after having been detected. Finally, the input data for the model is easily collected during testing. Thus, the hypergeometric model now is one of the main software reliability models.

Software reliability models often use maximum likelihood estimation to compute an estimate for the defect content m based on the test results.[4,15,21,22] To find a global maximum of the likelihood function $L(m)$ for the given test series result, one usually tries to solve for a zero of the derivative of the log likelihood function. In the hypergeometric model, the likelihood function contains a large number of binomials, see subsection 3.1. Thus, the maximum likelihood equation is far too complicated to be solved exactly. Attempts to approximately solve the maximum likelihood equation for the hypergeometric model in some special cases have had limited success, see below. As a consequence, least squares estimates are currently being used with the hypergeometric model, although minimizing the least squares sum is computationally expensive.[13,14,18]

In this paper, we *prove* that in almost all cases *maximum likelihood estimates exist and are unique in the hypergeometric reliability model.* This theoretical result is important because it provides a solid mathematical basis for highly efficient maximum likelihood estimation in the hypergeometric model and allows for a proper comparison of the hypergeometric model with other reliability models which use maximum likelihood. We also give a complete analysis of the exceptional cases where the maximum likelihood estimate does not exist or is not unique. From the main results, we derive a *novel algorithm* for computing the maximum likelihood estimate which is *simple, fast, and numerically stable.* In particular, computing the maximum likelihood estimate is computationally much cheaper than computing the least squares estimate. We also study a numerical example taken from the literature[6] and show that the maximum likelihood estimate is highly accurate and differs from the least squares estimate. For this example, the maximum likelihood estimate obtained in the hypergeometric model is as accurate as the estimates obtained with two S-shaped models[15,22] and clearly outperforms the estimate obtained with the exponential model.[4]

The idea underlying the paper is to study the *growth quotient*

$$Q(m) = \frac{L(m)}{L(m-1)}$$

of the likelihood function. Clearly, the likelihood function is increasing if and only if the growth quotient is greater than one. In the growth quotient, the binomials in the likelihood function cancel almost completely. The growth quotient turns out to be a rational function of m which depends on the w_k and the sum of the x_k, but not on the individual values x_k, see subsection 3.1. This result is important in practice, in particular for distributed testing, since it shows that the order in which defect samples are drawn does not matter for the value of the maximum likelihood estimate. A detailed mathematical analysis of the growth quotient shows that – in the typical case – the graph crosses the line $y = 1$ once, coming from above, and runs below the line beyond the intersection point, see Figure 1. It follows from the shape of the graph that a unique maximum likelihood estimate exists, being the greatest integer to the left of the intersection point. To compute this integer, we present a fast *iterative algorithm* which is numerically stable, see section 4.

For the special case that each test reveals the same number of defects, that is, w_k is constant, Tohma e.a.[20] use several methods to *approximate* the derivative of the log likelihood function for the hypergeometric model. For example, Tohma e.a. approximate the factorials in the likelihood function using Stirling's formula. Since the resulting approximate maximum likelihood equations are still complicated, Tohma e.a. apply Newton's method to find numerical solutions. Examples given by Tohma e.a. show that the approximation approach can fail to terminate. The question of existence and uniqueness of maximum likelihood estimates in the hypergeometric model has not been studied further by Tohma e.a. Our method is not restricted to special cases, does not involve derivatives or approximations, and always terminates with the solution.

An approach which is related to our work has been outlined by Darroch[3] and recalled in a recent book by Cai.[2] Darroch derives a polynomial equation which is equivalent to the equation $Q(m) = 1$. As opposed to our work, no proof is given that the polynomial equation has a (unique) zero; such a proof would follow the proof of our main theorem. The list of exceptions given by Darroch is incomplete. Cai solves for the zero of the polynomial equation using Newton's method, which involves the derivatives of the polynomials and is numerically unstable for long test series, because the degree of the polynomials equals the number of tests. For example, about half of the values listed in Table 4.7 of Cai's book are imprecise, even though the test series is short. Our iterative algorithm does not involve derivatives and is numerically stable even for long test series.

Our method of studying the growth quotient instead of the derivative of the likelihood function in order to compute maximum likelihood estimates might be applicable to other software reliability models. This is future work.

This paper is a revised version of a conference paper[16] and includes the proofs of the main results. The conference paper also discusses the recursion formula underlying least squares estimation for the hypergeometric model.

2. The Hypergeometric Model Revisited

We consider the outcome of test k to be the cumulative number

$$c_k = x_1 + x_2 + \ldots + x_k$$

of different defects detected in the first k tests. Using the c_k is equivalent to using the x_k, because $x_k = c_k - c_{k-1}$, but it leads to a convenient mathematical formulation of the hypergeometric model as a Markov chain, see Proposition 1 below. For $z \leq m$, $z - c \leq w$, and $w \leq z$, we set

$$p_{m,w}(z, c) = \frac{\binom{m-c}{z-c} \cdot \binom{c}{w-(z-c)}}{\binom{m}{w}} .$$

The hypergeometric model assumes that detecting a particular defect is stochastically independent of detecting the other defects. Thus, the outcome of test k may be modelled by the hypergeometric distribution

$$p_{m,w_k}(c_k, c_{k-1})$$

where the parameters m, w_k, and c_{k-1} are considered fixed.

To properly define the stochastic model for a whole test series, two questions must be answered: What is the sample space? How is the probability measure defined on the sample space?

Definition 1 *For a sequence of n tests, the sample space Ω_n consists of all n-tuples $(c_1, c_2, \ldots c_n)$ of natural numbers (zero included) such that the c_k are non-decreasing, $c_k \geq c_{k-1}$.*

For each value of n there is a separate sample space. This has not been made explicit in previous papers.

Definition 2 *Let $\underline{w}$ denote the sequence of numbers $w_1, w_2, \ldots w_n$. Suppose that m and the sequence $\underline{w}$ are given. The probability to observe the outcome $(c_1, c_2, \ldots c_n)$ in the test series is defined as*

$$\mathrm{P}_{m,\underline{w}}(c_1, c_2, \ldots c_n) =$$
$$p_{m,w_1}(c_1, 0) \cdot p_{m,w_2}(c_2, c_1) \cdot \ldots \cdot p_{m,w_n}(c_n, c_{n-1}).$$

It is well-known how to prove by induction on n that $\mathrm{P}_{m,\underline{w}}$ is a probability measure on the sample space Ω_n.

In the stochastic model just defined, the w_k are parameters and *not* part of the outcome of the random experiment, although in practice it is not known in advance how many defects will be detected in the next test. For each set of parameters $m, w_1, w_2, \ldots w_n$ there is a separate probability measure on the sample space. Strictly speaking, the hypergeometric reliability model thus consists of a whole family of probability measures for a given sample space Ω_n. This has not been made explicit in previous papers.

Denote by C_k the random variable which yields the cumulative number c_k of different defects detected in the first k tests. By construction of the probability measures, the probability to observe a total of c_k different defects after test k depends on c_{k-1}, but not on the earlier observations $c_{k-2}, c_{k-3}, \ldots c_1$.

Proposition 1 *For each probability measure* $\mathrm{P}_{m,\underline{w}}$ *on the sample space* Ω_n, *the stochastic process*

$$C_1,\ C_2,\ \ldots\ C_n$$

underlying the hypergeometric software reliability model is a Markov chain.

The stochastic process $X_1,\ X_2,\ \ldots\ X_n$ which yields for each test the number x_k of defects newly detected in the test is *not* a Markov chain.

3. Maximum Likelihood Estimates

3.1. *Growth quotient*

Suppose that a particular test series outcome $c_1,\ c_2,\ \ldots\ c_n$ and the parameters $w_1,\ w_2,\ \ldots\ w_n$ are given. We then get a family of probability measures $\mathrm{P}_{m,\underline{w}}$ on the sample space Ω_n which is parameterized by the number m of defects initially contained in the software. The *likelihood function* for the hypergeometric reliability model is defined as

$$L(m) \;=\; \mathrm{P}_{m,\underline{w}}(c_1,\ c_2,\ \ldots\ c_n).$$

The number $\widehat{m}$ is a *maximum likelihood estimate* for the number of defects initially contained in the software if it satisfies for all m the inequality $L(\widehat{m}) \geq L(m)$. A max likelihood estimate depends on the observed test series outcome, but we suppress the corresponding subscripts to simplify the notation.

Since the likelihood function for the hypergeometric model contains numerous factorials, it is not clear for which test series outcomes a max likelihood estimate exists. Instead of solving for the zeros of the maximum likelihood equation in order to find a global maximum of the likelihood function, we use an elementary approach to study the growth behavior of the likelihood function. Define the *growth quotient* of the likelihood function to be

$$Q(m) \;=\; \frac{L(m)}{L(m-1)}.$$

The likelihood function $L(m)$ is increasing when the growth quotient $Q(m)$ is greater than one and decreasing otherwise. The next two lemmas show that using the growth quotient has the advantage that the binomials in the formula for the likelihood function cancel almost completely. For $m \neq z$ and $m \neq 0$, we set

$$q_{m,w}(z, c) \;=\; \frac{(m-c)\cdot(m-w)}{m\cdot(m-z)}.$$

Lemma 1 *The growth quotient admits the presentation*

$$Q(m) = q_{m,w_1}(c_1, 0) \cdot q_{m,w_2}(c_2, c_1) \cdot q_{m,w_3}(c_3, c_2) \cdot \ldots \cdot q_{m,w_n}(c_n, c_{n-1}).$$

Lemma 1 shows that the growth quotient is a *rational* function with a nice presentation where both the numerator and the denominator are given as a product of linear factors. At least half of the factors cancel:

Lemma 2 *The growth quotient admits the cancelled presentation*

$$Q(m) = \frac{(m - w_1) \cdot (m - w_2) \cdot \ldots \cdot (m - w_n)}{m^{n-1} \cdot (m - c_n)}.$$

If $c_n = w_k$ for some k, then the factor $m - c_n$ in the presentation cancels. If $w_j = 0$ for some j, then the factor $m - w_j$ cancels.

Lemma 2 shows that the degree of the polynomials in both the numerator and denominator of the growth quotient $Q(m)$ is bounded by the length n of the test series. More important, the lemma shows that the order in which defect samples are drawn does not matter for the value of the maximum likelihood estimate. The same is true for the numbers x_k of defects newly detected in the tests. Only the total number c_n of different defects and the sizes w_k of the defect samples enter the formula for the growth quotient.

3.2. *Main theorem*

We view the growth quotient as a function $Q(x)$ of a real-valued variable x. The number of defects initially contained in a software must be greater or equal to the total number c_n of different defects detected during the test series. To compute maximum likelihood estimates, we must analyze the behavior of $Q(x)$ relative to the line $y = 1$ for $x > c_n$.

Proposition 2 *Suppose that* $n \geq 2$.

(1) *If* $c_n > w_k$ *for all* k, *then the growth quotient is greater than one near* c_n,

$$Q(x) > 1$$

for all x *in a suitable right neighborhood of* c_n.

(2) Special Case A. *If* $c_n = w_k$ *for some* k *and* $w_j > 0$ *for at least one* $j \neq k$, *then the maximum likelihood estimate* $\widehat{m}$ *is equal to* c_n,

$$\widehat{m} = c_n.$$

(3) Special Case B. *If* $c_n = w_k$ *for some* k *and* $w_j = 0$ *for all* $j \neq k$, *then any number* $m \geq c_n$ *is a maximum likelihood estimate.*

Proof. Refer to the cancelled presentation of $Q(x)$ given in Lemma 2.

(1) Since $c_n > w_k$ for all k, the factor

$$\frac{x - w_n}{x - c_n}$$

in the presentation does not cancel, but grows arbitrarily large as x approaches c_n from the right. The other factors remain bounded.

(2) Since $c_n = w_k$ for some k, the term $x - c_n$ in the presentation cancels and the growth quotient is a product of factors

$$\frac{x - w_j}{x} \leq 1.$$

Since $w_j > 0$ for at least one $j \neq k$, one of these factors is strictly less than one. Thus, $Q(x) < 1$ for all $x > c_n$.

(3) Since $c_n = w_k$ for some k and $w_j = 0$ for all $j \neq k$, all the terms in the presentation cancel. Thus, $Q(x) = 1$ for all $x > c_n$. □

Parts (2) and (3) of Proposition 2 are special cases. The condition $c_n = w_k$ means that the effort for the first $k - 1$ and the last $n - k$ tests could have been saved, because test k would have sufficed to detect all c_n defects which are known by the end of the full test series. Typically, no single test will uncover *all* defects known to be in the software by the end of the test series. Thus, the special cases can be disregarded in software testing practice and part (1) of Proposition 2 is the typical case.

Proposition 3 *Suppose that $c_n > w_j$ for all j and that $n \geq 2$.*

(1) *If $w_k > c_k - c_{k-1}$ for at least one index k, then the growth quotient is finally less than one,*

$$Q(x) < 1$$

for all sufficiently large $x > c_n$, and approaches one as x grows large.

(2) Special Case C. *If $w_k = c_k - c_{k-1}$ for all k, then no maximum likelihood estimate exists.*

Proof. Since $c_n > w_j$ for all j, there are at least two indices j with $w_j > 0$.

(1) Suppose that $w_k > c_k - c_{k-1} \geq 0$ for a particular index k. Refer to the cancelled presentation of $Q(x)$ given in Lemma 2. Since $c_n > w_j$ for all j, the term $x - c_n$ does not cancel. Since there is at least one index $j \neq k$ such that $w_j > 0$, the numerator and the denominator in the presentation are polynomials of degree at least two, even after fully cancelling. The numerator can be written as

$$x^{n} - \left(\sum_{j=1}^{n} w_j\right) \cdot x^{n-1} + r(x)$$

where $r(x)$ is a polynomial of degree at most $n-2$. The denominator equals

$$x^{n} - c_n \cdot x^{n-1}.$$

By assumption, $w_k > c_k - c_{k-1}$. Since $w_j \geq c_j - c_{j-1}$ for any j, we have

$$\sum_{j=1}^{n} w_j > \sum_{j=1}^{n} (c_j - c_{j-1}) = c_n.$$

It follows that the numerator is strictly dominated by the denominator for sufficiently large values of x and approaches one as x increases.

(2) Suppose that $w_k = c_k - c_{k-1}$ for all k. Refer to the presentation of $Q(x)$ given in Lemma 1. If $w_j = 0$, the factor $q_{x,w_j}(c_j, c_{j-1})$ equals one. On the other hand, if $w_j = c_j - c_{j-1} > 0$, it is straightforward to check that

$$q_{x,w_j}(c_j, c_{j-1}) > 1$$

for all $x > c_j$. Since there are at least two $w_j > 0$, it follows that $Q(x) > 1$ for all $x > c_n$. □

Part (2) of Proposition 3 is a special case. The condition $w_k = c_k - c_{k-1}$ for all k means that in each test all detected defects are new. This outcome indicates a high number of defects in the software. In the hypergeometric reliability model, such an outcome is the more likely the larger the number of defects contained in the software is. Typically though, defects will be re-detected in later tests of a series. Thus, the special case is unlikely to occur in software testing practice and part (1) of Proposition 3 is the typical case.

Theorem 1 *Suppose that $c_n > w_j$ for all j and $w_k > c_k - c_{k-1}$ for at least one index k. Then, the growth quotient has only a single local extreme point for $x > c_n$. The extreme point $\overline{x}$ is a local minimum smaller than one.*

Proof. Refer to the cancelled presentation of $Q(x)$ given in Lemma 2. Since $c_n > w_j$ for all j, the term $x - c_n$ does not cancel, but for each $w_j = 0$ a factor x cancels. After fully cancelling and re-numbering the (at least two) $w_j > 0$ suitably, we are left with the presentation

$$Q(x) = \frac{(x-w_1)\cdot(x-w_2)\cdot \ \ldots \ \cdot(x-w_\lambda)}{x^{\lambda-1}\cdot(x-c_n)}$$

where $2 \leq \lambda \leq n$ and $w_j > 0$ for $1 \leq j \leq \lambda$.

For values $x > c_n$, the growth quotient $Q(x)$ is a differentiable function. By Proposition 2, the growth quotient is greater than one near c_n. By Proposition 3, the growth quotient is less than one for large x and approaches one as x grows larger. Therefore, the growth quotient must have a local minimum $\overline{x} > c_n$ and the derivative $Q'(x)$ must be zero at that point,

$$Q'(\overline{x}) = 0.$$

We must show that $\overline{x}$ is the only zero of the derivative for $x > c_n$.

A short computation yields

$$Q'(x) = \frac{A(x) \cdot x \cdot (x - c_n) - (\lambda \cdot x - (\lambda - 1) \cdot c_n) \cdot B(x)}{x^{\lambda} \cdot (x - c_n)^2}$$

with the functions

$$\begin{aligned} A(x) &= (x - w_2) \cdot (x - w_3) \cdot \ \ldots \ \cdot (x - w_\lambda) + \\ &\quad (x - w_1) \cdot (x - w_3) \cdot \ \ldots \ \cdot (x - w_\lambda) + \\ &\quad \ldots \\ &\quad (x - w_1) \cdot (x - w_2) \cdot \ \ldots \ \cdot (x - w_{\lambda - 1}) \\ B(x) &= (x - w_1) \cdot (x - w_2) \cdot (x - w_3) \cdot \ \ldots \ \cdot (x - w_\lambda). \end{aligned}$$

Fully expanding the numerator of $Q'(x)$ shows that it is not a polynomial of degree $\lambda + 1$, but actually of degree at most λ. Thus, $Q'(x)$ can have at most λ zeroes.

Suppose that σ of the λ numbers $w_j > 0$ are pairwise different. Re-order the numbers w_j such that

$$w_j > w_{j-1} \quad \text{for } 1 < j \leq \sigma.$$

Suppose that w_j is a zero of multiplicity $\mu(j) \geq 1$ of the growth quotient $Q(x)$. A closer look at the terms $A(x)$ and $B(x)$ in the derivative $Q'(x)$ shows that w_j then is a zero of multiplicity $\mu(j) - 1$ of the derivative. Therefore, we can write the numerator of the derivative as

$$(x - w_1)^{\mu(1) - 1} \cdot (x - w_2)^{\mu(2) - 1} \cdot \ \ldots \ \cdot (x - w_\sigma)^{\mu(\sigma) - 1} \cdot r(x)$$

with a suitable polynomial $r(x)$. Since the derivative $Q'(x)$ has at most λ zeroes, the polynomial $r(x)$ has degree at most σ.

Recall that $c_n > w_j$ for all j and that $Q(x)$ is continuously differentiable for $0 < x < c_n$. The theorem of Rolle from advanced calculus[1] implies that $Q'(x)$ must have a zero between each pair (w_{j-1}, w_j) of different zeroes of $Q(x)$. Since we have re-ordered the numbers w_j in such a way that $w_1, w_2, \ldots w_\sigma$ are all different, there are $\sigma - 1$ such pairs. We also have $r(\overline{x}) = 0$ because $Q'(\overline{x}) = 0$. Since $r(x)$ has degree at most σ, the derivative has no zero for $x > c_n$ other than $\overline{x}$. □

4. Algorithm and Examples

The results proved in subsection 3.2 describe the behavior of the growth quotient completely for $x > c_n$. The conditions of Theorem 1 are the typical case. In the typical case, the intermediate-value theorem for continuous functions[1] implies that the graph of the growth quotient must cross the line $y = 1$ at least once. If the graph would cross (or touch) the line $y = 1$ more than once, then the graph would have a local maximum between these two intersection points, which would contradict Theorem 1. It follows that for a typical test series, *the graph of the growth quotient $Q(x)$ crosses the line $y = 1$ only once for $x > c_n$ and runs below the line afterwards,* see Figure 1. In particular, the likelihood function $L(m)$ is increasing only until the graph of $Q(x)$ falls below the line $y = 1$ for the first time. This fact gives us a *fast iterative algorithm* for computing the maximum likelihood estimate in the hypergeometric reliability model:

Step 1. Check whether one of the special cases A, B, or C applies. If no, continue with Step 2.

Step 2. Compute the maximum likelihood estimate $\widehat{m}$ iteratively as follows.

Step 2a. Set $x := c_n + 1$.

Step 2b. Check whether $Q(x) > 1$. If yes, set $x := x + 1$ and repeat Step 2b. If no, return the maximum likelihood estimate $\widehat{m} = x - 1$.

Step 2 b of the algorithm selects the greatest integer to the left of the intersection point as the maximum likelihood estimate $\widehat{m}$. By definition, $Q(x) < 1$ if and only if $L(x) < L(x-1)$. Therefore, the integer to the right of the intersection point is *not* the maximum likelihood estimate since it has a smaller value of the likelihood function than the integer to the left of the intersection point.

The growth quotient must be computed in Step 2 b of the algorithm in a way which avoids problems with the precision of the floating point arithmetic when the number n of tests is large. For example, one might take advantage of the cancelled presentation of the growth quotient given in Lemma 2: Re-order the w_k according to their size so that w_n is largest. Compute

$$\frac{x - w_n}{x - c_n}$$

and then multiply with the factors

$$\frac{x - w_j}{x}$$

until the product is less than one, or, all w_j have been used up. When computing $Q(x)$ this way, all factors have a similar order of magnitude.

In our previous conference paper,[16] we have computed the maximum likelihood estimate for an example with more than a hundred tests without any numerical problems in less than a second on a standard personal computer. This result is in sharp contrast to the difficulties reported by Tohma e.a.[20] who use approximations of the log likelihood function. In addition, our algorithm is simpler and faster than minimizing the *least squares sum* in the hypergeometric model by exhaustive search[18] or application of genetic algorithms.[14]

A typical graph of the growth quotient $Q(x)$ for $x > c_n$ is shown in Figure 1.

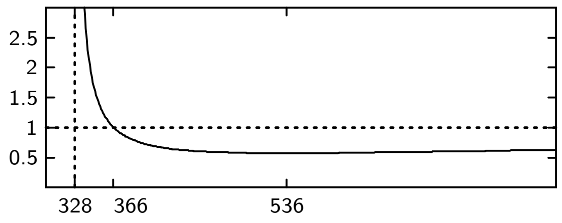

Figure 1: a typical graph of the growth quotient

The example is taken from a paper by Hou e.a.[6] A total of $c_n = 328$ different defects were detected in a series of $n = 19$ tests. The other input data are listed in Table 1. In the table, t_k denotes the execution time of test k in milliseconds, from which the numbers w_k were computed using a linear function.[6,16]

Table 1: input data for the example

k	x_k	t_k	w_k	k	x_k	t_k	w_k	k	x_k	t_k	w_k
1	15	2450	15	8	29	3370	48	15	6	3840	72
2	29	2450	29	9	4	960	15	16	7	2300	45
3	22	1960	22	10	27	1920	30	17	9	1760	36
4	37	980	37	11	27	2880	46	18	5	1990	41
5	2	1680	21	12	22	1440	24	19	3	2990	64
6	5	3370	43	13	21	3260	57				
7	36	4210	56	14	22	3840	69				

In the example, $Q(366) = 1.0089$ and $Q(367) = 0.9922$. It follows that the maximum likelihood estimate is $\widehat{m} = 366$. The local minimum $\overline{x}$ of the growth quotient is located between 536 and 537. The least squares estimate for this example is equal to 388. In particular, the maximum likelihood estimate can be different from the least squares estimate in the hypergeometric model.

Hou e.a.[6] also provide the estimates obtained for this example when using other software reliability models instead of the hypergeometric model: the exponential model of Goel e.a.,[4] the delayed S-shaped model of Yamada e.a.,[22] and the inflection S-shaped model of Ohba.[15] Most importantly, Hou e.a. report that 30 additional defects have been detected *after* testing. Therefore, a total of at least 358 defects were contained in this software. This data allows us to compare the estimation accuracy of the various reliability models for this example. Table 2 shows the relative estimation errors in percent:

Table 2: estimation errors for the example

model	estimate	error (percent)
exponential MLE	455	$+27.1$
delayed S-shaped MLE	351	-2.0
inflection S-shaped MLE	347	-3.1
hypergeometric MLE	366	$+2.2$
hypergeometric LSE	388	$+8.4$

The hypergeometric model and the S-shaped models yield highly accurate maximum likelihood estimates in this example and clearly outperform the exponential model. As compared to the S-shaped models, the maximum likelihood estimate in the hypergeometric model shows a slight tendency to overestimate. Since the number of defects in the software could actually be higher than the known 358 defects, the hypergeometric estimate might be preferable in this case. Also, the maximum likelihood estimate in the hypergeometric model is more accurate than the corresponding least squares estimate.

Acknowledgements

I'd like to thank Kai-Yuan Cai from Beijing University of Aeronautics and Astronautics for pointing out the relevant section in his book and for good discussions about the subject.

References

1. T. M. Apostol: *Mathematical Analysis.* Addison-Wesley, 1974
2. K.-Y. Cai: *Software Defect and Operational Profile Modeling.* Kluwer, 1998
3. J. N. Darroch: "The Multiple-Recapture Census I.", Biometrika 45 (1958) 343-359
4. A. L. Goel, K. Okumoto: "Time-Dependent Error-Detection Rate Model for Software Reliability and Other Performance Measures", IEEE Transactions on Reliability 28:3 (1979) 206-211
5. R.-H. Hou, I.-Y. Chen, Y.-P. Chang, S.-Y. Kuo: "Optimal Release Policies for Hyper-Geometric Distribution Software Reliability Growth Model with Scheduled Delivery Time", Proceedings Asia-Pacific Software Engineering Conference APSEC (1994) 445-452
6. R.-H. Hou, S.-Y. Kuo, Y.-P. Chang: "Applying Various Learning Curves to Hyper-Geometric Distribution Software Reliability Growth Model", Proceedings International Symposium on Software Reliability Engineering ISSRE 5 (1994) 8-17
7. R.-H. Hou, S.-Y. Kuo, Y.-P. Chang: "Hyper-Geometric Distribution Software Reliability Growth Model with Imperfect Debugging", Proceedings International Symposium on Software Reliability Engineering ISSRE 6 (1995) 195-200
8. R.-H. Hou, S.-Y. Kuo, Y.-P. Chang: "Efficient Allocation of Testing Resources for Software Module Testing Based on the Hyper-Geometric Distribution Software Relia-

bility Growth Model" , Proceedings International Symposium on Software Reliability Engineering ISSRE 7 (1996) 289-298
9. R.-H. Hou, S.-Y. Kuo, Y.-P. Chang : "Needed Resources for Software Module Test, Using the Hyper-Geometric Software Reliability Growth Model" , IEEE Transactions on Reliability 45:4 (1996) 541-549
10. R.-H. Hou, S.-Y. Kuo, Y.-P. Chang : "Optimal Release Policy for Hyper-Geometric Distribution Software-Reliability Growth Model" , IEEE Transactions on Reliability 45:4 (1996) 646-651
11. R.-H. Hou, S.-Y. Kuo, Y.-P. Chang : "Optimal Release Times for Software Systems with Scheduled Delivery Time Based on HGDM" , IEEE Transactions on Computers 46:2 (1997) 216-221
12. R. Jacoby, Y. Tohma : "The Hypergeometric Distribution Software Reliability Growth Model: Precise Formulation and Applicability" , Proceedings International Computer Software and Applications Conference COMPSAC (1990) 13-19
13. R. Jacoby, Y. Tohma : "Parameter Value Computation by Least Squares Method and Evaluation of Software Availability and Reliability at Service-Operation by the Hyper-Geometric Distribution Software Reliability Growth Model" , Proceedings International Conference on Software Engineering ICSE 13 (1991) 226-237
14. T. Minohara, Y. Tohma : "Parameter Estimation of Hyper-Geometric Distribution Software Reliability Growth Model by Genetic Algorithms" , Proceedings International Symposium on Software Reliability Engineering ISSRE 6 (1995) 324-329
15. M. Ohba : "Software Reliability Analysis Models" , IBM Journal of Research and Development 28:4 (1984) 428-443
16. F. Padberg : "A Fast Algorithm to Compute Maximum Likelihood Estimates for the Hypergeometric Software Reliability Model" , Proceedings Asia-Pacific Conference on Quality Software APAQS 2 (2001) 40-49
17. Y. Tohma, R. Jacoby, Y. Murata, M. Yamamoto : "Hypergeometric Distribution Model to Estimate the Number of Residual Software Faults" , Proceedings International Computer Software and Applications Conference COMPSAC (1989) 610-617
18. Y. Tohma, K. Tokunaga, S. Nagase, Y. Murata : "Structural Approach to the Estimation of the Number of Residual Software Faults Based on the Hypergeometric Distribution" , IEEE Transactions on Software Engineering 15:3 (1989) 345-355
19. Y. Tohma, H. Yamano, M. Ohba, R. Jacoby : "Parameter Estimation of the Hyper-Geometric Distribution Model for Real Test/Debug Data" , Proceedings International Symposium on Software Reliability Engineering ISSRE 2 (1991) 28-34
20. Y. Tohma, H. Yamano, M. Ohba, R. Jacoby : "The Estimation of Parameters of the Hypergeometric Distribution and Its Application to the Software Reliability Growth Model" , IEEE Transactions on Software Engineering 17:5 (1991) 483-489
21. M. Xie : *Software Reliability Modeling*. World Scientific, 1991
22. S. Yamada, M. Ohba, S. Osaki : "S-Shaped Reliability Growth Modeling for Software Error Detection" , IEEE Transactions on Reliability 32:5 (1983) 475-485

About the Author

Frank Padberg is a senior researcher at the Faculty of Informatics, University of Karlsruhe, Germany. He holds a master in mathematics from the University of Erlangen, Germany, and a Ph.D. in computer science from the University of Saarbrücken, Germany. He has been supported by fellowships and research grants from the Deutsche Forschungsgemeinschaft DFG. Before joining the university, he has given seminars in industry on networking technology and operating systems for several years. He has published papers on a number of different topics in software engineering.