Empirical Interval Estimates for the Defect Content After an Inspection

Frank Padberg
Fakultät für Informatik
Universität Karlsruhe
Germany
padberg@ira.uka.de

ABSTRACT

We present a novel method for estimating the number of defects contained in a document using the results of an inspection of the document. The method is empirical, being based on observations made during past inspections of comparable documents. The method yields an interval estimate, that is, a whole range of values which is likely to contain the true value of the number of defects in the document. We also derive point estimates from the interval estimate. The method is validated using a known empirical inspection dataset and clearly outperforms existing approaches for estimating the defect content after inspections.

1. INTRODUCTION

Inspections are a successful technique to detect defects in software documents. In an inspection run, several reviewers inspect the same document to find defects. The outcome of an inspection run is a matrix having entries zero and one showing which reviewer detected which defect. Some defects will be detected by more than one reviewer, but usually not all the defects contained in a document are detected in an inspection. Thus, after an inspection management must decide whether to re-inspect the document to find additional defects or to pass the document on to the next development step. Since it is unknown how many defects actually are contained in a document, management must have a *reliable estimate* for the number of remaining defects as a basis for the decision.

Currently, defect content estimation methods for software inspections fall into two categories : capture-recapture methods and curve-fitting methods. Both approaches have attracted considerable attention over the past years, and we shall briefly describe the approaches here.

Capture-recapture methods have been applied to software inspections for the first time in [6]. Recent extensions are [5, 9]. Capture-recapture methods as a first step compute certain summarized data from the the 0/1 matrix of the inspection run to be estimated. For example, one particular method computes for each reviewer the number of defects which were detected by that reviewer. As a second step, each capture-recapture method defines a particular stochastic process to describe the inspection run. The stochastic process is parameterized by the (unknown) number N of actual defects contained in the document. Finally, some statistical technique is applied to the stochastic model to estimate the value of N taking into account the observed summarized data. For example, some methods use maximum likelihood estimation as the statistical technique. The various capture-recapture methods differ in the summarized data computed from the results of the inspection run, the assumptions underlying the stochastic model, and the statistical estimation technique which is used. For a comparison of different capture-recapture methods see [4].

Two curve-fitting methods have been developed for software inspections in [11]. The methods as a first step compute certain summarized data from the 0/1 matrix of the inspection run to be estimated. For example, the "detection profile method" computes for each defect the number of reviewers who have detected that particular defect. As a second step, the summarized data is sorted and a curve is fitted through the sorted datapoints. For example, in the detection profile method the datapoints are sorted in decreasing order according to their size and an exponentially decreasing curve is fitted through the sorted datapoints using linear regression on the log values. Finally, the number of defects contained in the document is estimated as the x-value of the crossing point of the fitted curve with a suitably defined horizontal line. The detection profile method has been combined with a capture-recapture method in [3].

Several studies indicate that the capture-recapture methods and the curve-fitting methods yield estimates which are too unreliable to be used in practice [2, 3, 4, 9, 10]. The capture-recapture methods show a tendency to underestimate the true number of defects. They also show outliers and a high variation in the relative error of the estimates. The curve-fitting methods show outliers and a high variation in the relative error of the estimates, too. Neither approach

provides an indicator for how reliable a particular estimate is. In particular, a user of the methods can't tell in advance whether an estimate is an outlier or not.

Both the capture-recapture methods and the curve-fitting methods use only the results of the inspection run to be estimated as input. They do not take into account the experience made in past inspections. In this paper, we present an estimation method which is much different from existing approaches : besides data about the run to be estimated we also use data from past inspections. The new method has several steps :

- pre-processing the empirical data from past inspections;
- constructing a probabilistic model for the outcome of inspection runs from the empirical database;
- computing an interval estimate for the inspection run to be estimated;
- computing several point estimates from the interval estimate.

Pre-processing the empirical database means to compute summarized data for each inspection run in the database. For each run in the database, its *signature* is computed. The signature of a run combines a measure for the variation among the inspection results of the individual reviewers with a measure for the overall efficiency of the run, see subsection 2.3. The signature of a run depends on the true number N of defects contained in the document, which is an additional required input to the new method for each run in the empirical database. In many cases the number N of defects contained in an inspected document is known some time after an inspection, at least approximately; see subsection 2.1 for a discussion. For practical purposes, adding the defects detected in a document during later development phases (including maintenance) to the defects found during the inspection run yields a reasonable approximate value for the true number of defects in the document. The empirical database in an organization grows with each inspection carried out in the organization.

The probabilistic model is constructed directly from the empirical database, see subsection 2.4. The probability measure tells for each possible signature how likely it is in view of past experience that this signature is the outcome of an inspection run. Combining the probabilistic model with a maximum likelihood approach, the new method yields an *interval estimate* for the number of defects in the document to be estimated. The interval estimate is a range of numbers which is likely to contain the true value for the number of defects in the document (subsections 2.6 and 2.7). This is in contrast to other estimation methods which yield point estimates. The new estimation method yields a whole range of estimates because the signature introduces an equivalence relation on runs which maps different pairs of inspection results and values of N to the same signature, see subsections 2.3 and 2.7 (recall that the signature depends on N). Introducing an equivalence relation is necessary to bundle up the datapoints. Otherwise, one would end up with a probability distribution which has many isolated bars.

In the new method, each interval estimate comes with a *confidence level* which indicates how reliable the interval estimate is in view of past experience (subsection 2.9). Finally, in subsection 2.8 several different point estimates are derived from the interval estimate.

The new method is validated using a known empirical inspection dataset in section 3. Step by step, a run from the database is left out and then estimated using the runs remaining in the database ("jackknife"). The confidence levels for the resulting interval estimates show that this dataset must be subdivided into two subsets according to the application domain of the document, see subsection 3.3. After splitting the dataset the estimates get re-computed. On one half of the dataset, using a particular point estimate derived from the interval estimate the new method achieves a mean of absolute relative errors of only 6.6 percent. In particular, the new method shows no outliers for this half of the dataset. On the other half of the dataset, however, the new method indicates a low confidence level for the interval estimates which therefore are discarded. Although one would like to have an estimate in each case, we consider it to be much better to have the method indicate that a particular estimate is likely to be invalid than to base a management decision on an invalid estimate without knowing.

In section 4, the new estimation method is compared with a particular capture-recapture method [6] and the detection profile method [11] using the dataset from section 3. The new method clearly outperforms the other approaches on the half of the dataset where the interval estimates get a high confidence level. On the other half of the dataset where the new method doesn't provide estimates, both the capture-recapture method and the detection profile method show outliers and a high variation in the relative error of their estimates and thus are unreliable. As opposed to the capture-recapture method, the interval estimates computed with the new method show a tendency to overestimate the number of defects in the document. Overestimation is preferable to underestimation because it is better to re-inspect a good quality document than to pass a low quality document on to the next development step.

2. ESTIMATION MODEL

2.1 Input Data

The new estimation method requires a database containing empirical data about past inspection runs as input. For each inspection run in the database as well as for the inspection run to be estimated the following empirical data are required :

- the number of defects found by each reviewer;
- the total number of different defects found in the run.

The empirical data about an inspection run can be computed directly from the 0/1 matrix of the run. For a particular run, entry (k, j) of its matrix is equal to 1 if reviewer k detected defect j and equal to 0 if he did not. There is a separate 0/1 matrix for each run in the empirical database and a 0/1 matrix for the run to be estimated.

In addition to the empirical data just described, for each run in the database the true number N of defects contained in

the inspected document is required (but not for the run to be estimated, of course). In many cases, the total number N of defects contained in an inspected document is known some time after the inspection, at least approximately. If the document was used in a *controlled experiment* where the defects have been seeded into the document, the total number of defects in the document is known exactly. On the other hand, if the document was inspected during a development project, usually not all the defects in the document are known right after the inspection, but later development phases (including maintenance) will reveal additional defects in the document besides those which were found during the inspection. For practical purposes, adding the defects detected during later development phases to the defects found during the inspection run yields a reasonable approximate value for the true number of defects in a document. One might also argue that defects which were not detected during deployment of the product obviously are not relevant for the operational profile of the product and can thus be neglected.

2.2 Empirical Dataset

To validate the new method, we shall apply it in later subsections to a known inspection dataset. The dataset consists of 16 inspection runs which were conducted during two controlled experiments in 1994 and 1995 [1].

The inspected documents were specifications. There were four documents: two from NASA (NASA-A and NASA-B, 27 pages each) and two generic ones (the specification for a parking garage system PG, 16 pages, and for an automatic teller machine ATM, 17 pages). All defects contained in the documents were known since the defects had been seeded into the documents. The NASA documents contained 18, respectively, 15 defects, while the generic documents contained 30, respectively, 28 defects.

For each run and document, the reviewers had about two hours to complete their review. The number of reviewers for each run varies between 6 and 8 reviewers; in most cases, there were 6 reviewers. The inspections were conducted using two different reading techniques, namely, perspective-based reading and ad-hoc reading. In both experiments, the reviewers were software professionals from the NASA Software Engineering Laboratory SEL. For later use, the inspection runs are labelled as follows:

run	document	year	technique	defects	detected
A1	ATM	1994	perspective	30	23
A2	ATM	1994	ad-hoc	30	20
A3	ATM	1995	perspective	30	24
A4	ATM	1995	ad-hoc	30	22
B1	PG	1994	perspective	28	20
B2	PG	1994	ad-hoc	28	17
B3	PG	1995	perspective	28	24
B4	PG	1995	ad-hoc	28	21
C1	NASA-A	1994	perspective	18	10
C2	NASA-A	1994	ad-hoc	18	6
C3	NASA-A	1995	perspective	15	15
C4	NASA-A	1995	ad-hoc	15	15
D1	NASA-B	1994	perspective	15	6
D2	NASA-B	1994	ad-hoc	15	9
D3	NASA-B	1995	perspective	15	14
D4	NASA-B	1995	ad-hoc	15	13

Except for the 1994 inspection runs on the NASA documents, a large fraction of the defects detected in a run were detected by more than one reviewer. For more details on the inspection experiments see [1].

2.3 Preprocessing the Dataset

The empirical database must be pre-processed by computing the *signature* for each run in the database. The signature of a run has two parts:

- the "span", which is a measure for the variation among the inspection results of the individual reviewers;
- the "efficiency class", which is a measure for the overall efficiency of the inspection run.

For a given inspection run, denote by w_k the number of defects found by reviewer k. Denote by d the total number of different defects detected in the run. The w_k and d get computed directly from the 0/1 matrix of the run. The results of a run with m reviewers are then described as a vector

$$(\, w_1,\ \ldots\ w_m;\ d\,).$$

For example, the inspection results of run A1 with six reviewers ($m = 6$) are

$$(\, 9,\, 7,\, 6,\, 13,\, 9,\, 6;\ 23\,).$$

To compute the first part of the signature of a run, its "span", compute for each reviewer k the *individual detection ratio*

$$r_k \;=\; \frac{w_k}{N}.$$

Recall that N denotes the number of defects contained in the document and is assumed to be known for each run in the database. Subdivide the range of 0 to 100 percent of possible detection ratios into intervals and number the intervals in ascending order. The subdivision may be arbitrary, but usually the range will be subdivided into intervals of equal length. For example, the range might be subdivided into the intervals

$$[\,0,\,10\,],\quad (\,10,\,20\,],\qquad \ldots\qquad (\,90,\,100\,]$$

which are then numbered from 1 to 10. After having subdivided the range, compute for each reviewer the *detection ratio class,* that is, the number c_k of that interval to which the reviewer's detection ratio r_k belongs. For example, reviewer 2 of run A1 detected 7 out of 30 defects contained in the document. Therefore, the second reviewer's detection ratio is

$$r_2 \;=\; \frac{7}{30},$$

or 23.4 percent. This detection ratio falls into the third interval of the given subdivision, whence

$$c_2 \;=\; 3.$$

Finally, compute from the individual detection ratio classes c_k the *span* of the run as

$$s \;=\; \max_k\, c_k \;-\; \min_k\, c_k \;+\; 1.$$

The span of a run is a measure for the variation among the individual reviewers' detection ratios. For run A1, the detection ratios range between 20.0 and 43.4 percent. Thus,

$$s \;=\; 4$$

is the span for run A1.

In the current version of the method, only the span is used in the computation of the estimates, not the full vector of individual detection ratios. Thus, computing the largest and the smallest individual detection ratio would suffice to compute the span.

The second part of a run's signature, its "efficiency class", is computed similarly. Compute the *overall detection ratio*

$$r \;=\; \frac{d}{N}.$$

In run A1, 23 out of 30 defects in the document were detected by the reviewers, whence the overall detection ratio is

$$r \;=\; \frac{23}{30},$$

or 76.7 percent. Subdivide the range of 0 to 100 percent of possible overall detection ratios into intervals and number the intervals in ascending order. The subdivision need not be the same as the one chosen for the individual detection ratios. For example, the range might be subdivided into the intervals

$$[\,0,\,20\,], \quad (\,20,\,40\,], \quad \ldots \quad (\,80,\,100\,]$$

which are then numbered from 1 to 5. The *efficiency class* of a run is the number c of that interval to which the run's detection ratio r belongs. The efficiency class of a run is a measure for the overall efficiency of the inspection run. For example,

$$c \;=\; 4$$

is the efficiency class of run A1 for the given subdivision.

To sum up, the *signature* of an inspection run is defined as

$$\sigma \;=\; (\,c,\,s\,).$$

For example, the signature of run A1 equals $(\,4,\,4\,)$. The following table shows the signatures of all runs in the empirical dataset from subsection 2.2, using the subdivisions chosen above:

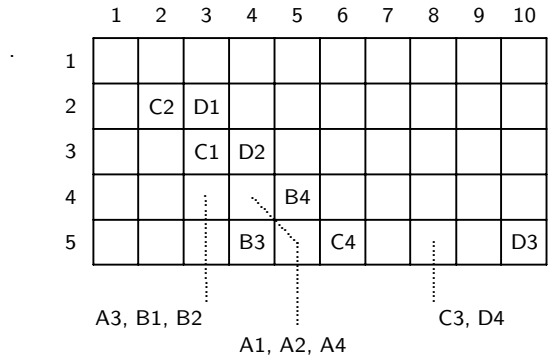
run	c	s
A1	4	4
A2	4	4
A3	4	3
A4	4	4
B1	4	3
B2	4	3
B3	5	4
B4	4	5

run	c	s
C1	3	3
C2	2	2
C3	5	8
C4	5	6
D1	2	3
D2	3	4
D3	5	10
D4	5	8

The subdivision of the range of possible individual detection ratios should be chosen as finegrained as possible, subject to the constraint that the resulting probability distribution for the span doesn't have many holes, see the next subsection. From that point of view, the choice of the appropriate subdivision depends on the dataset at hand. The same holds for the subdivision of the range of possible overall detection ratios.

2.4 Probabilistic Model

The signature classifies the runs in an empirical dataset according to two criteria, namely, the span of the runs and the efficiency class of the runs. Here is a visualization of the classification of the 16 runs in the empirical dataset from subsection 2.2:



The rows correspond to the efficiency class, the columns to the span. One can see that different runs may map to the same signature.

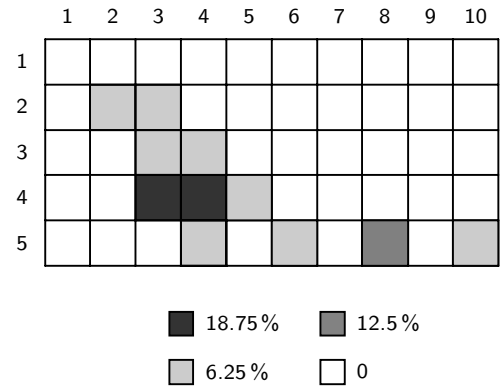
Using the classification of the runs in an empirical dataset, an empirical *probability measure* on the set of all possible signatures can be constructed as follows:

- compute for each run in the dataset its signature;
- compute the relative frequency of each signature in the dataset;
- assign to each signature its relative frequency as its probability.

The probability measure embodies a probabilistic model for the outcome of inspection runs which is based on the experience made in past inspections. For the empirical dataset from subsection 2.2, the probability measure is given by the following table:

signature	probability
$(\,2,\,2\,)$	6.25 %
$(\,2,\,3\,)$	6.25 %
$(\,3,\,3\,)$	6.25 %
$(\,3,\,4\,)$	6.25 %
$(\,4,\,3\,)$	18.75 %
$(\,4,\,4\,)$	18.75 %
$(\,4,\,5\,)$	6.25 %
$(\,5,\,4\,)$	6.25 %
$(\,5,\,6\,)$	6.25 %
$(\,5,\,8\,)$	12.5 %
$(\,5,\,10\,)$	6.25 %

The probabilities are multiples of $\frac{1}{16}$. The signatures not listed in the table have probability zero for that dataset. The probability measure for the 16 runs can be visualized like this:



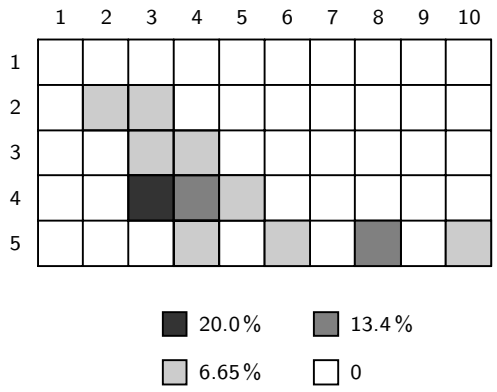
The two subdivisions of the range of possible detection ratios introduce an equivalence relation on inspection runs. The equivalence relation bundles up the datapoints in the database and thus avoids ending up with a probability measure which has many isolated bars. The difficulty is to choose the equivalence relation in such a way that it keeps enough information about the inspection results to be useful for estimating the defect content. The probability measure for the running example still has a few isolated bars, but this is natural for empirical data.

2.5 Estimation Procedure

In the preceding subsection, we have constructed a probability measure from an empirical database of inspection runs. The probability measure can be used to estimate the unknown number N of defects contained in some document given the observed result $(w_1, \ldots w_m; d)$ of a recent inspection of the document. The estimation procedure consists of three steps:

1. compute the likelihood function corresponding to the observed result of the inspection;
2. compute the maximum likelihood interval estimate from the likelihood function;
3. compute various point estimates from the interval estimate.

In the next three subsections, we shall explain each step in the estimation procedure using the following *running example.* Assume that the true number of defects in some document were unknown. Also assume that the document gets inspected with the results $(9, 7, 6, 13, 9, 6; 23)$ (the results correspond to run A1). We then estimate the number of defects in the document using the observed results of the run and the probability measure constructed from an empirical database of runs, where – in this example – the database consists of the 15 runs A2, A3 ... D4. Since run A1 is left out in the database, the probability measure changes slightly as compared to the preceding subsection:



The probabilities are now multiples of $\frac{1}{15}$. Since the new database has one run less than before, field $(4, 4)$ is shaded lighter than before.

2.6 Likelihood Function

Suppose for a moment that n were the true number of defects in the document under consideration. Then, the observed inspection result $(9, 7, 6, 13, 9, 6; 23)$ corresponds to a particular signature

$$\sigma \;=\; (c(n), s(n)).$$

The steps to compute the signature are the same as when pre-processing an inspection run from the empirical database, see subsection 2.3, where N is substituted by n. We write $c(n)$ and $s(n)$ to clearly indicate that the signature depends on the value for n. For example, for the result of run A1 we have $c(25) = 5$ and $s(25) = 4$, whereas $c(35) = 4$ and $s(35) = 3$. When computing the span and efficiency class of the run to be estimated the same subdivisions of the 0 to 100 percent range must be used as were used when pre-processing the empirical database.

By the probability measure constructed from the database, the signature $(c(n), s(n))$ occurs with the probability

$$\mathrm{P}(c(n), s(n)).$$

For example, for the run A1 we have $\mathrm{P}(c(25), s(25)) = \mathrm{P}(5, 4) = 6.7\,\%$.

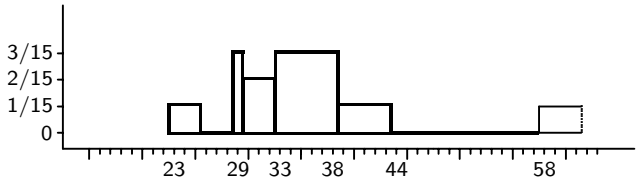
The probabilities $\mathrm{P}(c(n), s(n))$ are used to construct the *likelihood function*

$$\mathrm{L}: \; n \;\mapsto\; \mathrm{P}(c(n), s(n)).$$

The likelihood function associates with each n the probability to observe the given inspection result $(w_1, \ldots w_m; d)$ when assuming that n defects are contained in the document. The inspection result is assumed to be fixed. Note that the likelihood function is just a function and not a probability distribution.

Here are the values and the graph of the likelihood function for our running example:

n	c	s	likelihood
23 – 25	5	4	6.65 %
26 – 28	5	3	0
29	4	3	20.0 %
30 – 32	4	4	13.4 %
33 – 38	4	3	20.0 %
39 – 43	3	3	6.65 %
44 – 57	3	2	0
58 – 59	2	2	6.65 %
60 – 64	2	3	6.65 %
65 – 114	2	2	6.65 %
115 – 129	1	2	0
130 – ...	1	1	0



The likelihood function for run A1 hits the dark shaded field $(4, 3)$ twice and the medium shaded field $(4, 4)$ once. Some fields are not hit, such as $(5, 8)$. The efficiency class c decreases as the number n increases, but this does not always hold for the span s.

2.7 Interval Estimates

From the likelihood function corresponding to a particular observed inspection result one can compute estimates for the number of defects contained in the document in a standard way. The number $\widehat{n}$ is called a *maximum likelihood estimate* if it satisfies for all n the inequality

$$\mathrm{L}(\widehat{n}) \;\geq\; \mathrm{L}(n).$$

In other words, $\widehat{n}$ is a maximum likelihood estimate if the likelihood function assumes a global maximum for $\widehat{n}$. The value of a maximum likelihood estimate depends on the results of the inspection run, of course. For our running example, the likelihood function takes its maximum value of $\frac{3}{15}$ for the numbers 29 and 33 to 38. Therefore, each number $\widehat{n}$ in the range

$$e_p \;=\; \{29\} \cup [33, 38]$$

is a maximum likelihood estimate for the running example. The range e_p is called the *interval estimate* for the inspection run. The interval estimate is the range of numbers which most likely contains the true number N of defects in the document.

The example shows that different values of n often result in the same value of the likelihood function. The reason is that we map different overall detection ratios to the same efficiency class, respectively, different sets of individual detection ratios to the same span by using the range subdivisions. Therefore, one often gets a whole interval of numbers n where the likelihood function takes on its maximum value; in some cases, one even gets a union of non-adjacent intervals. This way, the maximum likelihood approach yields an interval estimate for the number of defects in the document instead of a point estimate.

2.8 Point Estimates

One might think of several ways to get a point estimate for the number of defects in a document from the interval estimate constructed in the preceding subsection. We study three obvious candidates here:

- the upper boundary b of the interval estimate;
- the lower boundary a of the interval estimate;
- the median $\frac{1}{2}(a + b)$ of the interval estimate.

Here are the results for the running example, including the relative error for each point estimate. The true number of defects is 30:

estimate	value	error
b	38	+ 26.7 %
a	29	− 3.4 %
$\frac{1}{2}(a + b)$	34	+ 13.4 %

The lower boundary a of the interval estimate e_p gives a good point estimate for this example.

2.9 Confidence Levels

The dataset under study is too small and we know too little yet about the shape of the empirical probability distributions occuring in software inspections to assume that the distributions come from a particular family of distributions. In particular, there is not much use in computing the usual confidence intervals, because such computations are based on normal distributions. As a result, we don't give confidence intervals in this paper. Yet, the values of the likelihood function provide a simple sort of *confidence level* for the interval estimates. For the running example, the maximum value of 20.0 % of the likelihood function is three times as high as its lowest non-zero value of 6.7 %. Thus, the interval estimate e_p is given a high confidence level for the running example.

In the next section, we provide examples where the interval estimate gets only a low confidence level. In such a case it is preferable to re-inspect the document or do some other testing of the document than to assume that the estimate is valid.

3. VALIDATION

To validate the method, we apply it to the empirical inspection dataset from subsection 2.2. For the validation, we step by step

- leave out a run from the database,
- compute the probability measure for the remaining 15 runs,
- compute the interval and point estimates for the run which was left out,
- and compare the estimates with the true value of the number of defects in the document.

For each of the 16 examples generated this way, we perform the same estimation procedure as for the running example in

the preceding section. In fact, last section's running example corresponds to the first validation example.

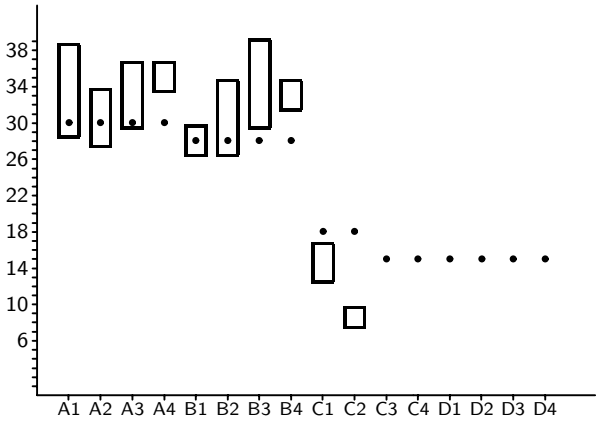
Since the dataset is small, we use all of the remaining 15 inspection runs in the empirical dataset when computing the probability measure. In particular, for the time being we do not distinguish between different document domains or reading techniques, but see subsection 3.3.

3.1 Results

The following table shows for each of the 16 validation examples the interval estimate e_p and the corresponding maximum value $\mathrm{L}(e_p)$ of the likelihood function:

run	N	interval estimate e_p	$\mathrm{L}(e_p)$
A1	30	$\{29\} \cup [33, 38]$	20.0 %
A2	30	$[28, 33]$	13.4 %
A3	30	$[30, 36]$	13.4 %
A4	30	$[34, 36]$	20.0 %
B1	28	$[27, 29]$	13.4 %
B2	28	$[27, 34]$	13.4 %
B3	28	$[30, 39]$	20.0 %
B4	28	$[32, 34]$	20.0 %
C1	18	$[13, 16]$	20.0 %
C2	18	$[8, 9]$	20.0 %
C3	15	$[15, 16] \cup [22, 24] \cup [26, 29] \cup [33, 37] \cup [44, 74]$	6.65 %
C4	15	$[19, 74]$	6.65 %
D1	18	$[7, 14] \cup [20, 29]$	6.65 %
D2	18	$[10, 14] \cup [20, 44]$	6.65 %
D3	15	$[47, 69]$	6.65 %
D4	15	$[13, 16] \cup [26, 29] \cup [44, 64]$	6.65 %

Only the first 10 runs (A1 to C2) show a high or medium confidence level. Therefore, the interval estimates computed for the last 6 runs are not considered valid and are to be discarded. Here is a picture showing for each of the 16 runs the true number of defects in the document as a black dot and the interval estimates for the 10 runs with a high or medium confidence level as bars:



For the first 8 runs the interval estimates contain the true value for the number of defects, or, overestimate that number. The situation is different for the runs C1 and C2 which are studied in detail in subsection 3.3. In the remainder of this subsection, we study how the various point estimates perform which are derived from the interval estimate as shown in subsection 2.8. For reasons that will become apparent in subsection 3.3, we focus on the first 8 runs in the discussion.

The upper boundary b of the interval estimate e_p overestimates the number of defects in a document by 3.6 to 39.3 percent for the runs A1 to B4:

run	N	b	error
A1	30	38	+ 26.7 %
A2	30	33	+ 10.0 %
A3	30	36	+ 20.0 %
A4	30	36	+ 20.0 %
B1	28	29	+ 3.6 %
B2	28	34	+ 21.5 %
B3	28	39	+ 39.3 %
B4	28	34	+ 21.5 %

The mean of the absolute relative errors for the upper boundary estimate b equals 20.3 percent for the runs A1 to B4.

The lower boundary a of the interval estimate e_p results in an error of -6.7 to $+14.3$ percent for the runs A1 to B4:

run	N	a	error
A1	30	29	− 3.4 %
A2	30	28	− 6.7 %
A3	30	30	0 %
A4	30	34	+ 13.4 %
B1	28	27	− 3.6 %
B2	28	27	− 3.6 %
B3	28	30	+ 7.2 %
B4	28	32	+ 14.3 %

The corresponding mean of the absolute relative errors is only 6.6 percent for the runs A1 to B4. Clearly, the lower boundary a of the interval estimate e_p is a good estimator for the number N of defects in a document for the dataset studied here.

The median $\frac{1}{2}(a + b)$ of the interval estimate e_p results in an error of 0 to $+25.0$ percent for the runs A1 to B4:

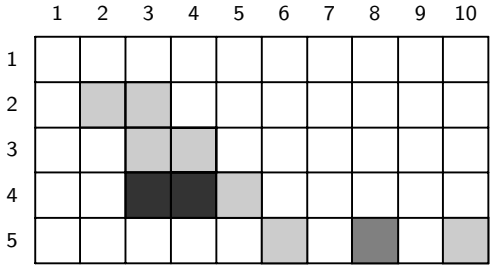
run	N	$\frac{1}{2}(a + b)$	error
A1	30	34	+ 13.4 %
A2	30	31	+ 3.4 %
A3	30	33	+ 10.0 %
A4	30	35	+ 16.7 %
B1	28	28	0 %
B2	28	31	+ 10.8 %
B3	28	35	+ 25.0 %
B4	28	33	+ 17.9 %

The median shows a tendency to overestimate. The mean of the absolute relative errors for the median estimate is 12.2 percent for the runs A1 to B4.

3.2 Typical Cases

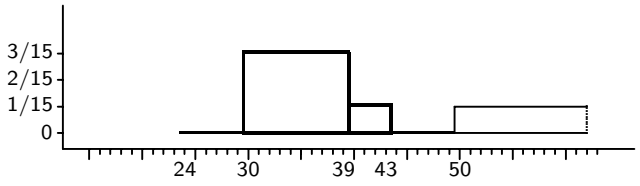
In this subsection, we take a closer look at the runs B3, C3, and C4 to illustrate typical examples of possible estimation results. The meaning of the grey colors in the pictures for the probability measures is fixed for this subsection:

The probability measure for the dataset with run B3 left out looks like this:



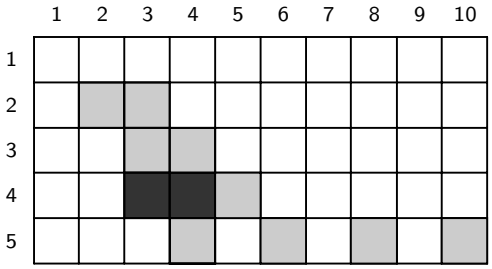
For run B3, the values of the likelihood function for different values of n and the corresponding graph are:

n	c	s	likelihood
24	5	4	0
25	5	5	0
26 – 29	5	4	0
30 – 32	4	4	20.0 %
33 – 39	4	3	20.0 %
40 – 43	3	3	6.65 %
44 – 49	3	2	0
50 – 59	3	3	6.65 %
60 – 64	2	3	6.65 %
65 – 119	2	2	6.65 %
120 – 129	1	2	0
130 – ...	1	1	0



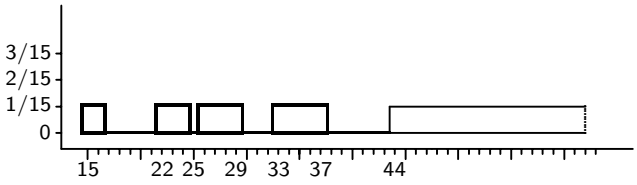
As opposed to run A1, the likelihood function for run B3 has only a single interval where it takes on its maximum value. Since the maximum value is $\frac{3}{15}$, the interval estimate has a high confidence level. The upper boundary estimate b overestimates the true number of defects in the document by 39.3 percent, the lower boundary estimate a by only 7.2 percent.

The probability measure for the dataset with run C3 left out looks like this:



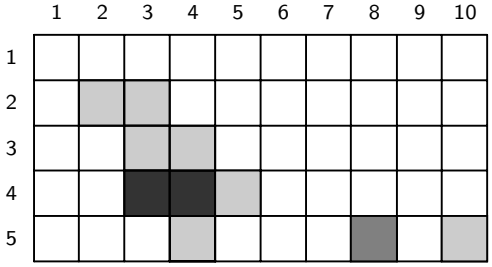
For run C3, the values of the likelihood function for different values of n and the corresponding graph are:

n	c	s	likelihood
15 – 16	5	8	6.65 %
17 – 18	5	7	0
19 – 21	4	6	0
22 – 24	4	5	6.65 %
25	3	5	0
26 – 29	3	4	6.65 %
30 – 32	3	5	0
33 – 37	3	4	6.65 %
38 – 43	2	4	0
44 – 64	2	3	6.65 %
65 – 74	2	2	6.65 %
75 – 129	1	2	0
130 – ...	1	1	0



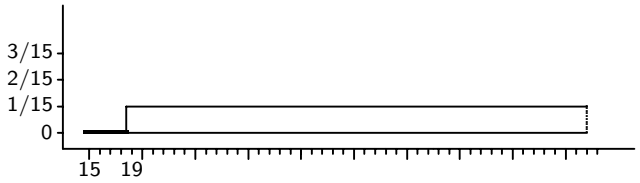
The likelihood function does not hit the dark shaded fields $(4,3)$ and $(4,4)$. Therefore, the likelihood function takes on the values zero and $\frac{1}{15}$ only. There are several non-adjacent intervals where the likelihood function takes on its maximum value: from the likelihood function alone one can't tell which interval contains the true value for the number of defects in the document. Since the confidence level is low, the interval estimate is to be discarded.

The probability measure for the dataset with run C4 left out looks like this:



For run C4, the values of the likelihood function for different values of n and the corresponding graph are:

n	c	s	likelihood
15 – 16	5	6	0
17 – 18	5	5	0
19 – 24	4	5	1/15
25 – 33	3	4	1/15
34 – 37	3	3	1/15
38 – 49	2	3	1/15
50 – 74	2	2	1/15
75 – 99	1	2	0
100 – ...	1	1	0



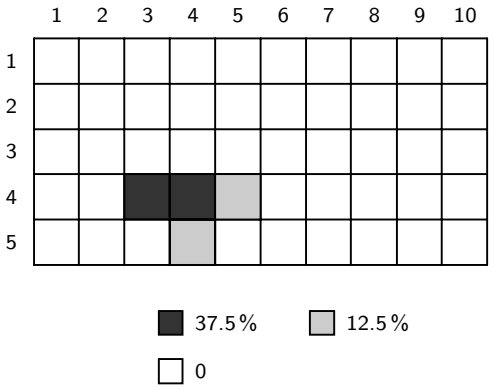
Again, the likelihood function does not hit the dark shaded fields $(4,3)$ and $(4,4)$. As opposed to run C3, there is a large contiguous range where the likelihood function takes on its maximum value of $\frac{1}{15}$. Again, the interval estimate is to be discarded.

3.3 Domain Dependence

The interval estimates for the runs C1 and C2 have a high confidence level, similar to the runs A1 to B4. Also, the estimation procedure outputs a single, fairly small interval for runs C1 and C2 as interval estimate. Despite this, the upper boundary b of the interval estimate $[8, 9]$ for run C2 is 50 percent *below* the true number of 18 defects in the document; the situation is similar for run C1. What has happened? There are several observations pointing to a possible explanation for the outliers:

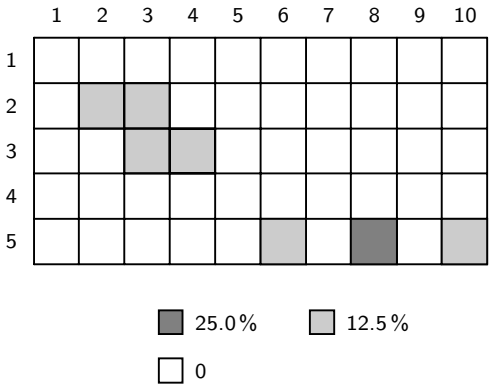
- Using a different reading technique (perspective-based reading instead of ad-hoc reading) yields a better defect detection ratio of $\frac{10}{18}$ and a better interval estimate of $[13, 16]$ for run C1 as compared to run C2. This might point to the reading technique as being an important factor for the estimation.
- The total number of 6 defects detected by the reviewers in run C2 is the lowest among all inspection runs. This is a warning sign that run C2 might be special and that its interval estimate should be taken with caution. The two next smallest numbers of detected defects are 6 (out of 15) in run D1 and 9 (out of 15) in run D2. In those two cases, the estimation procedure indicates only a low level of confidence for the interval estimates.
- Runs C1 and C2 come from a different application domain (NASA domain) than the runs A1 to B4 (generic domain). Only the eight runs from the generic domain contribute to the high value of 20.0 percent for the signatures $(4, 3)$ and $(4, 4)$ in the probability measure.

The last observation indicates a possible *domain dependence* of the estimation procedure. To study this hypothesis, we split the dataset into two subsets: one subset contains the 8 runs A1 to B4 from the generic domain, the other subset contains the 8 runs C1 to D4 from the NASA domain. After splitting the database according to the domain, we repeat the validation for each domain separately. Here is the full probability measure (all 8 runs) for the generic domain:



The values of the likelihood functions for the runs in the generic domain change after splitting the database, but *all interval estimates for the runs in the generic domain remain unchanged.* The confidence levels also don't change. Therefore, all results about the various estimates for the runs A1 to B4 that we have described in the preceding subsections still hold after splitting the database. In particular, the lower boundary estimate remains to be a good estimator for the number of defects in the generic documents. Note that the picture might be quite different for other examples.

The full probability measure (all 8 runs) looks differently in the NASA domain:



In the NASA domain, recomputing the likelihood function for runs C1 to D4 using the NASA half of the database shows that the likelihood functions take on only two different values, namely, zero and $\frac{1}{7}$. Again, not every shaded field is hit by the likelihood function. Since the value of the likelihood function is small compared to the values in the generic domain, there is only low confidence in the interval estimates computed for the runs in the NASA domain. Therefore, the interval estimates are all discarded. In particular, *no outliers* occur for the runs C1 and C2 after splitting the database.

Clearly, for this particular dataset one must distinguish between different domains for the documents. If the dataset were larger, one should also distinguish between different reading techniques. The NASA domain subset is too small and inhomogeneous to derive valid estimates with the new method. Although the subset for the generic domain is small, too, the results still have some validity because the runs correspond to different documents, groups of inspectors, inspection dates, and reading techniques.

4. COMPARISON

4.1 Capture-Recapture Methods

For the comparison, we use the capture-recapture method described in [6]; the other capture-recapture methods show similar performance [4]. The capture-recapture method requires as input the total number d of different defects detected in the run, and, for each reviewer k the number w_k of defects which were detected by that reviewer. The probabilistic model underlying the method assumes that the defects are probabilistically identical, but different reviewers may have different probabilities of detecting defects. By construction, the probabilistic model is parameterized by the (unknown) true number N of defects in the document. Based on the observed inspection data, a maximum likelihood estimation for N is performed. This capture-recapture method is frequently referred to as Mt(MLE). Here are the estimates $\widehat{m}$ computed using this capture-recapture method and the corresponding estimation error for each of the runs in the database:

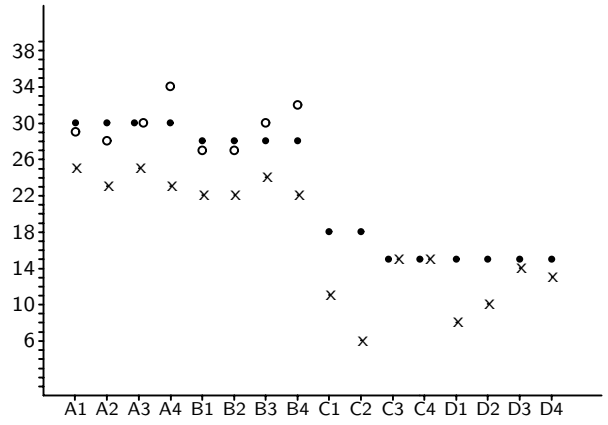
run	N	$\widehat{m}$	error
A1	30	24	−20.0 %
A2	30	22	−26.7 %
A3	30	25	−16.7 %
A4	30	23	−23.4 %
B1	28	22	−21.5 %
B2	28	21	−25.0 %
B3	28	24	−14.3 %
B4	28	21	−25.0 %

run	N	$\widehat{m}$	error
C1	18	11	−38.9 %
C2	18	6	−66.7 %
C3	15	15	0 %
C4	15	15	0 %
D1	15	8	−46.7 %
D2	15	10	−33.4 %
D3	15	14	−6.7 %
D4	15	13	−13.4 %

The capture-recapture estimates show a strong tendency to underestimate the true number of defects in the document. For the runs A1 to B4 in the generic domain, the estimation error ranges between −26.7 % and −14.3 %. The mean of the absolute relative errors is 21.6 % for the runs in the generic domain. Therefore, the capture-recapture method is clearly outperformed by the lower boundary estimate a given in subsection 3.1 which has a mean of absolute relative errors of only 6.6 % for the runs in the generic domain.

For the runs C1 to D4 in the NASA domain, the estimation error of the capture-recature method ranges between −66.7 % and 0 %. The mean of the absolute relative errors is 25.8 % for the runs in the NASA domain. Recall that the interval estimates e_p for the runs in the NASA domain had low confidence levels and were discarded in subsection 3.3. Thus, no lower boundary estimates are given for the NASA domain.

Here is a picture showing the true number of defects in the documents as black dots, the capture-recapture estimates as small crosses, and the lower boundary estimates as hollow dots:



Interestingly, the capture-recapture method yields good estimates for the four runs C3, C4, D3 and D4 in the NASA domain. The reason is that in all four cases the total number of defects detected during the inspection run is close to the true number of defects contained in the document, and capture-recapture methods in general tend to produce estimates which are not much larger than the number of defects detected in the run; see [7] where we have studied the behavior of maximum likelihood estimates for the hypergeometric capture-recapture model. From that point of view, the four good estimates of the capture-recapture method are coincidental.

4.2 Detection Profile Method

The detection profile method is a curve-fitting method and has been described in [11]. The method requires as input for each defect j the number n_j of reviewers who have detected that defect. The numbers n_j are sorted in descending order according to their size and an exponential curve is fitted through the sorted datapoints using linear regression on the log values. The estimate is computed as the x-value of the crossing point of the exponential curve with the horizontal line $y = 0.5$. Here are the estimates $\widetilde{m}$ computed using the detection profile method and the corresponding estimation error for each of the runs in the database:

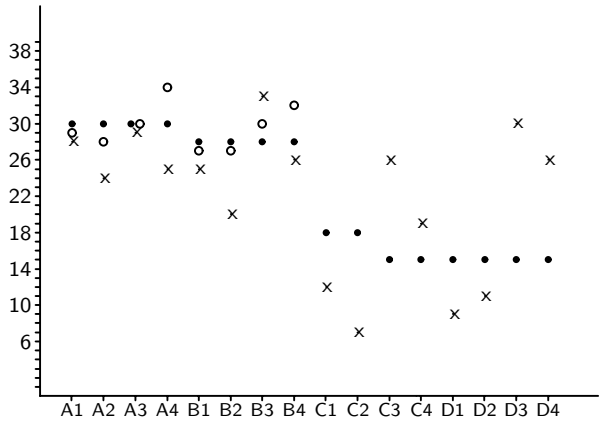
run	N	$\widetilde{m}$	error
A1	30	28	−6.7 %
A2	30	24	−20.0 %
A3	30	29	−3.4 %
A4	30	25	−16.7 %
B1	28	25	−10.8 %
B2	28	20	−28.6 %
B3	28	33	+17.9 %
B4	28	26	−7.2 %

run	N	$\widetilde{m}$	error
C1	18	12	−33.4 %
C2	18	7	−61.2 %
C3	15	26	+73.4 %
C4	15	19	+26.7 %
D1	15	9	−40.0 %
D2	15	11	−26.7 %
D3	15	32	+113.4 %
D4	15	28	+86.7 %

For the runs in the generic domain, the estimation error ranges between −28.6 % and +17.9 %. The mean of the absolute relative errors is 13.9 % for the runs in the generic domain. For all runs except B4 in the generic domain,

the detection profile method is outperformed by the lower boundary estimate a given in subsection 3.1. For run B4, the detection profile method underestimates the number of defects in the document whereas the new method overestimates, which is preferable.

Here is a picture showing the true number of defects in the documents as black dots, the estimates computed with the detection profile method as small crosses, and the lower boundary estimates as hollow dots:



For the runs in the NASA domain, the detection profile method estimates show a high variation in their relative errors of $-61.2\,\%$ to $+113.4\,\%$. The mean of the absolute relative errors is $57.7\,\%$ for the runs in the NASA domain. Therefore, the detection profile method estimates are highly unreliable for the runs in the NASA domain.

5. CONCLUSIONS

The new estimation method presented in the paper uses as input not only the observed results of the inspection of the document to be estimated, but also empirical data collected in past inspections. Before applying the new method to estimate the defect content of a document, one should perform a jackknife validation on the empirical dataset, as we have done in section 3. The results will show which values of the likelihood function correspond to valid interval estimates and which do not. The results will also show which one of the three point estimates derived from the interval estimate in subsection 2.8 should be used and what relative error to expect. For the dataset that we used in the paper, the lower boundary estimate performed best.

It might be necessary to subdivide the empirical dataset according to, for example,

- different document types and sizes;
- the reading techniques used in the inspections;
- the number of reviewers.

For a study of the impact of different such factors on the quality of an inspection see [8]. For the dataset used in the paper, the new method indicates through outlier estimates that one must subdivide the dataset according to the domain of the documents. After subdividing the dataset into two subsets and applying the new method to each subset separately, no more outliers occur. On one subset (generic domain), the new estimation method provides highly accurate estimates and clearly outperforms existing approaches. On the other subset (NASA domain), the new method indicates through low confidence levels that the estimates should be discarded. The estimates provided by existing approaches on the NASA domain subset are highly unreliable. Thus, we consider it to be a clear advantage that the new method indicates whether an estimate should be used or not.

The dataset used in the paper is small and the results might be biased by the fact that the inspections were performed on only a few different documents. The next step will be to validate the new method using other empirical datasets.

6. ACKNOWLEDGEMENTS

I'd like to thank Vic Basili and Forrest Shull from the University of Maryland who kindly provided the 0/1 matrices from their 1994 and 1995 inspection experiments.

7. REFERENCES

[1] Basili, Green, Laitenberger, Lanubile, Shull, Sorumgard, Zelkowitz: "The Empirical Investigation of Perspective-Based Reading", Empirical Software Engineering 1:2 (1996) 133-164

[2] Biffl, Grossmann: "Evaluating the Accuracy of Defect Estimation Models Based on Inspection Data From Two Inspection Cycles", Proceedings International Conference on Software Engineering ICSE 23 (2001) 145-154

[3] Briand, El-Emam, Freimut: "A Comparison and Integration of Capture-Recapture Models and the Detection Profile Method", Proceedings International Symposium on Software Reliability Engineering ISSRE 9 (1998) 32-41

[4] Briand, El-Emam, Freimut, Laitenberger: "A Comprehensive Evaluation of Capture-Recapture Models for Estimating Software Defect Content", IEEE Transactions on Software Engineering 26:6 (2000) 518-540

[5] Ebrahimi: "On the Statistical Analysis of the Number of Errors Remaining in a Software Design Document after Inspection", IEEE Transactions on Software Engineering 23:8 (1997) 529-532

[6] Eick, Loader, Long, Votta, Vander Wiel: "Estimating Software Fault Content Before Coding", Proceedings International Conference on Software Engineering ICSE 14 (1992) 59-65

[7] Padberg: "A Fast Algorithm to Compute Maximum Likelihood Estimates for the Hypergeometric Software Reliability Model", Asia-Pacific Conference on Quality Software APAQS 2 (2001) 40-49

[8] Porter, Siy, Mockus, Votta: "Understanding the Sources of Variation in Software Inspections", ACM Transactions on Software Engineering and Methodology 7:1 (1998) 41-79

[9] Runeson, Wohlin: "An Experimental Evaluation of an Experience-Based Capture-Recapture Method in Software Code Inspections", Empirical Software Engineering 3:3 (1998) 381-406

[10] Vander Wiel, Votta: "Assessing Software Designs Using Capture-Recapture Methods", IEEE Transactions on Software Engineering 19:11 (1993) 1045-1054

[11] Wohlin, Runeson: "Defect Content Estimations from Review Data", Proceedings International Conference on Software Engineering ICSE 20 (1998) 400-409