ABSTRACT

In this keynote I explore what exactly do we mean by data quality, techniques to assess data quality and the very significant challenges that poor data quality can pose. I believe we neglect data quality at our peril since —whether we like it or not — our research results are founded upon data and our assumptions that data quality issues do not confound our results. A systematic review of the literature suggests that it is a minority practice to even explicitly discuss data quality. I therefore suggest that this topic should become a higher priority amongst empirical software engineering researchers.

Categories and Subject Descriptors
D.2.8 [Software Engineering]: Metrics—complexity measures, performance measures, process metrics, product metrics

General Terms
Measurement, Management

Keywords
Software metrics, empirical research, data quality

1. INTRODUCTION

There has been extensive research over the past 40 years into software metrics, that is quantitative approaches to understand, manage and improve software engineering. The bulk of this research can be characterised as data driven, for example metrics are validated by observing relationships between the putative metrics and phenomena of interest or predictive models are tested in terms of their goodness of fit. Indeed, given the central role of data, there has been a move to encourage the publishing and sharing of data sets. This is clearly helpful as it assists replication of results.

Surprisingly, as a community, we do not seem to systematically or explicitly deal with data quality nor are there agreed reporting protocols for such issues. A systematic literature review [9, 8] found that it was a minority practice to even discuss data quality, let alone taking steps to check for quality and quarantine potentially problematic cases.

What is data quality? There are a range of definitions however, quite a few authors adopt the Crosby-style perspective of fitness for purpose, e.g. Strong et al. [17]. The idea is that in order to examine data quality we need to consider what the data are to be used for. Thus, data quality is not seen in any absolute sense but is relative to a particular purpose. However, I prefer the simpler notion of (high) data quality being the absence of noise (i.e. the recorded value of a data item is the same as its true value). Note that noise is not the same as the presence of outliers which might be true outliers (i.e. not noisy) or the consequence of misreported (i.e. noisy and therefore possibly not an outlier). However, noise in the sense that I intend it, underpins other views of quality.

A significant problem is that in general the ‘true’ value of a data item will be unknown, most notably for secondary data. For this reason it is helpful to differentiate between implausible values e.g. a defect count is negative and plausible but still potentially noisy values e.g. the defect count is 11 (which may or may not be so). This is illustrated in Fig. 1. Note that not all outliers are due to noise. Missing values could be viewed as a particular case of an implausible value although data imputation, that is dealing with missingness, that is dealing with missingness is generally regarded as a separate research topic. For a general overview see Little and Rubin [11] and applications specifically within software engineering see [14, 16].

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Figure 1: A data quality taxonomy

For a more general discussion of the impact of poor data quality in a range of disciplines see DeVeaux and Hand [3].
2. A SYSTEMATIC REVIEW

So how have empirical software engineering researchers viewed data quality? A systematic literature review\(^1\) was recently conducted by Liebchen and myself. An early version may be found in [9] and an extended and updated version in Liebchen’s doctoral thesis [8]. The review sought answers to the following questions:

- **Q1:** How significant do the community consider noise to be (in principle and in practice)?
- **Q2:** How do empirical analysts address this problem? Are there techniques that might be deployed to independently assess the quality of a given data set?

The inclusion criteria of the search were that the article had to (i) relate to some aspect of empirical software engineering, (ii) address data noise explicitly, (iii) be refereed, (iv) be written in English and (v) where multiple accounts were given of the same underlying primary study we utilised the most recent version. We searched the bibliographic databases ACM Digital Library, IEEE Xplore, ScienceDirect, SCOPUS and SpringerLink databases. No limit to the years searched was imposed, however the earliest relevant article we located was from 1993. The searches were conducted in August 2010.

The search term is given below. Whilst it might seem somewhat convoluted we devised it via the capture-recapture technique. Specifically, we polled members of the Promise 2008 Conference program committee and asked them to indicate any relevant papers they had authored. This was augmented by a number of papers we were already aware of. This then constituted our target list of known papers that any search query could be tested against. Where the search failed to locate a known target paper we revisited and extended the query:

```plaintext
software AND ‘noisy data’ OR ‘data quality’ OR noise OR ‘inconsistent pieces’ OR ‘erroneous data’ OR ‘clean the data’ OR ‘data cleaning’ OR ‘Issues arising from the data collection’ OR ‘recorded with enough consistency and completeness’ OR ‘not very consistent’
```

This query actually retrieved many hundreds of papers that we then hand searched for relevance. Eventually we identified 161 papers that satisfied all five inclusion criteria. To give a sense of the proportion, a search of the same databases for empirical AND ‘software engineering’ yielded in excess of 17000 hits, so as a ballpark figure one might argue about 1% of papers makes some explicit reference to data quality. I should stress that there are no doubt other papers about 1% of papers makes some explicit reference to data quality. I should stress that there are no doubt other papers.

**Q1:** Is data quality a problem? Yes, a substantial majority of papers, 138 out of 161, considered poor data quality to be a threat to empirical data analysis. The majority of papers, 122 (76%), focussed on quantitative data and 45 papers (28%) were concerned with qualitative data quality. Note that some papers addressed both quantitative and qualitative data and a few made no explicit reference to data type.

A good example of an empirical assessment of data quality comes from Johnson and Disney [5]. They report that as part of the data recording process of the Personal Software Process for 89 projects completed by ten participants they discovered 1539 primary errors! However, nearly half of these errors were incorrect calculations and so potentially could be addressed by the provision of better tool support.

Bachmann and Bernstein [1] investigated the quality of software process data in change data repositories. They concluded that data quality differs substantially between projects. More specifically they found that the bug reports in all the data sets they studied were often poorly linked and source code changes were untraceable “due to empty messages or missing bug report links”. Invalid and duplicate bug reports for open source projects varied between 12% and 34% (duplicates) and from 4% and 34% (invalid reports). This suggests significantly poor quality bug report information.

**Q2:** How was poor data quality addressed? We can broadly classify the studies as follows (in each case the count is out of 161):

1. (50 studies) Prevention through improved data collection procedures e.g. special purpose data collection tools, input validation or improved education for those responsible for the collection are attractive strategies. This is a problem though when dealing with secondary data as is increasingly the norm.
2. (35 studies) Manual data quality checking which typically involves increasing one’s confidence in a data set by some manual intervention such as independent scrutiny or the use of triangulation. However, again this is not always possible because many researchers are not directly involved in the actual data collection process and therefore have to work with secondary data.
3. (18 studies) Use of meta-data (i.e. data to describe the data, specifically quality). This approach has mainly centred on the ISBSG dataset which contains a data quality field (though it actually is a proxy for completeness which is not necessarily the same).
4. (15 studies) Use of some automated data quality checking procedure that is able to work in the absence of input from those originally associated with the data collection (e.g. rule induction and cluster analysis). The idea is to identify and filter suspect data. Some approaches go beyond this with some form of polishing (i.e. repair or impute a new value for the problem data item).
5. (4 studies) Tolerate low data quality through robust analysis e.g. Colombo et al. [2] employed a

---

\(^1\)A systematic literature review aims to locate all studies relevant to a specified research question, in an unbiased and repeatable fashion in order to synthesise an answer. They are popular in medicine, social policy and many other disciplines. See Kitchenham et al. [7] for examples in software engineering.
Bayesian approach to handle noisy software process data. Other researchers have used robust statistical techniques or data transformations such as natural logs or fuzzy methods. However, Teng [18] reported that robust algorithms were outperformed by filter and polish techniques (i.e. explicitly identifying noisy instances and imputing new values). Nevertheless the two approaches are not mutually exclusive. Robust techniques are relatively common and it is quite likely that the motivation — to handle noise — is not always explicitly given, hence the number of papers identified is almost certainly an under-estimate.

2.1 Examples of Automated Procedures
Moses [13] describes the use of a Bayesian probability model to assess the correctness of subjective categorisations of inter-modular cohesion. He used the cohesion classifications of 163 students to show that data quality could be inferred from agreement established using a Bayesian probability model.

Another approach is Pairwise Attribute Noise Detection Algorithm (PANDA) [20] which ranks instances according to noisiness, summing the combination of a distance measure extracted from the mean and standard deviation of a distribution, which the authors term a ‘noise factor’. The authors found that PANDA identified more noisy instances, fewer outliers and fewer clean instances than an alternative nearest neighbour outlier detection technique. They used a software metrics expert to corroborate the identification of noisy instances.

Liebchen et al. [10] used a range of techniques based upon rule induction which appeared to give useful results on a large industrial data set when checked by a local metrics expert. Unfortunately subsequent work based on simulation produced less satisfactory results [8]. This illustrates the difficulty of satisfyingly validating these algorithms because in general the true values will be unknown without simulating data using a true model and then injecting known noise according to some probability distribution.

Most recently (published after the systematic review), is the work of Yoon and Bae [21] who make the point that human judgement is needed to determine how vulnerable our studies are to potential data quality problems. Or to put it the other way around, how heroic our assumptions must be about the absence of noise.

Fourth, we need to progress towards the goal of effective algorithms to detect, filter and even polish noisy data items. Presently there are no simple solutions. As indicated earlier even evaluating techniques is fraught with problems since it is seldom possible to definitively know the true value for all items in a data set. Simulating noise is possible but clearly requires the making of many assumptions some of which are hard to justify from a real-world standpoint. Nevertheless, headway would be invaluable in improving the quality of our empirical research.

Acknowledgements
I would like to thank Gernot Liebchen for many of the ideas contained within this keynote and for doing the lion’s share of the systematic review.

4. REFERENCES