Chapter 7
Missing Data in Software Engineering

Audris Mockus

Abstract The collection of valid software engineering data involves substantial effort and is not a priority in most software production environments. This often leads to missing or otherwise invalid data. This fact tends to be overlooked by most software engineering researchers and may lead to a biased analysis. This chapter reviews missing data methods and applies them on a software engineering data set to illustrate a variety of practical contexts where such techniques are needed and to highlight the pitfalls of ignoring the missing data problem.

1. Introduction

The goal of this chapter is to increase the awareness of missing data techniques among people performing studies in software engineering. Three primary reasons for this presentation are:

1. The “quick-fix” techniques that drop the cases with missing values may yield biased or inconclusive results. Such techniques are still widely (and often implicitly) used in software engineering
2. Dealing with missing values is no longer a burden for a practitioner, because easy to use statistical software is now available on popular platforms
3. Software represents a distinct data source with unique reasons and patterns for missing data. For example, software studies tend not to have the luxury of large sample sizes requiring analysis methods that use all available data, including incomplete cases. Many properties of software can not be measured directly, therefore investigators have to get the necessary information from people who create and maintain a particular piece of software, leading to frequent and complex patterns of missing data

Section 2 discusses sources of software data. The next section introduces an illustrative example evaluating how a software process influences development time. Section 4 presents a general statistical perspective for dealing with missing data with an illustrative example. Section 5 discusses non traditional missing data problems specific to the field of software engineering. A summary is provided in Sect. 6.

© Springer 2008
2. Sources of Software Data

Software engineering data come from several distinct sources. The three primary sources are:

- Data collected through experimental, observational, and retrospective studies
- Software metrics or reported project management data including effort, size, and project milestone estimates
- Software artifacts including requirements, design, and inspection documents, source code and its change history, fault tracking, and testing databases

To narrow the scope of the presentation we did not include data sources produced directly by software with little or no human involvement, such as program execution and performance logs or the output of program analysis tools. Such data sources tend to produce tool specific patterns of missing data that are of limited use in other domains.

Surveys in an industrial environment are usually small and expensive to conduct. The primary reasons are the lack of subjects with required knowledge and the minimal availability of expert developers who, it appears, are always working toward a likely-to-be-missed deadline. The small sample size limits the applicability of deletion techniques that reduce the sample size even further. This may lead to an inconclusive analysis, because the sample of complete cases may be too small to detect statistically significant trends. If, on the other hand, the sample sizes are large and only a small percentage of data are missing, a deletion technique (a technique that removes missing observations) may work quite well.

The values in survey data may be missing if a survey respondent declines to fill the survey, ignores a question, or does not know the answer to some of the questions.

Reported data on software metrics often contain the desired measurements on quality and productivity. Unfortunately, the reported data are often not comparable across distinct projects (Herbsleb and Grinter, 1998). The reasons include numerous social and organizational factors related to intended use and potential misuse of metrics, and serious difficulties involved in defining, measuring, and interpreting a conceptual measure in different projects.

Reported data need extensive validation to confirm that it reflects the quantities an analyst is interested in. Data collection is rarely a priority in software organizations (Goldenson et al., 1999). The priority of validating collected data is even lower, often leading to unreliable and misleading software measures. In addition, some software measures are difficult to obtain or have large uncertainty. Examples of such measures include function point estimates or size and effort estimates in the early stages of a project. Frequently data values are missing because some metrics are not collected for the entire period of the study or for a subset of projects.

Software artifacts are large, highly structured, and require substantial effort to interpret. Measures derived from software artifacts tend to be more precise and consistent over time than measures derived from surveys and reported data. They