# Obvious Outliers in ISBSG Repository of Software Projects: Exploratory Research

Dominic Paré Alain Abran
*École de Technologie Supérieure*
*dominic.pare.1@ens.etsmtl.ca* *Alain.abran@etsmtl.ca*

## Abstract

*This paper discusses the issue of outliers in the repository of software projects of the International Software Benchmarking Standards Group - ISBSG. The criteria used for the identification of outliers is whether the productivity is significantly lower and higher, that is with significant economies or dis-economies of scale, in relatively homogeneous samples. Once the outliers identified, other project variables are investigated by heuristics to identify candidate explanatory variables that might explain such outliers' behaviors.*

## 1. Introduction

In software engineering, software projects productivity can vary considerably. It is then interesting to analyze the cause of these significant variations in order to be able to explain why the productivity of these projects is much higher or much lower than the average. The International Software Benchmarking Standards Group (ISBSG) [1] designed and maintains a repository of software projects. For productivity analysis and for estimation purposes, it is important on the one hand to identify outliers which have productivity behaviors significantly different from all other projects and, on the other hand, to try to discover next which factors have such a large influence (positive or negative) on the productivity of these projects.

This article identifies outliers in the ISBSG repository as well as candidate variables which could explain major differences in productivity by comparison to other projects in the same samples. This paper is structured as follows: section 2 presents on overview of the ISBSG repository, section 3 the identification of outliers for the samples selected, section 4 a discussion on these outliers and, section 5, a summary and discussion.

## 2. ISBSG repository

ISBSG makes available to industry and researchers, at a reasonable cost, an Excel data file which contains 92 variables for each of the projects in its repository, such as effort (in hours), functional size of the software (measured according to various standards: Function Points, COSMIC-FFP - ISO 19761, MKII), programming language, etc. [2].

The ISBSG repository is a multi-organizational, multi-application domain and multi-environment data repository, that is, its data content is fairly heterogeneous in projects characteristics. Data from either Release 8 (R8) with 2027 projects or Release 9 (R9) with 3024 projects are used for the various analyses reported here. Obviously, the analysis should not be carried out on all the projects simultaneously. To get a minimum of homogeneity in the samples to be analyzed, the following two criteria are taken into account: same functional sizing method and same programming language.

For the first criterium, projects measured with the IFPUG function points method have been selected since in ISBSG R8, close to 90% of the projects had been measured with the IFPUG method.

For the second criterium, the projects with the same programming language were grouped together in distinct samples. In ISBSG R8, there were only 6 programming languages with more than 30 projects, 30 being the number of points for considering a sample of a reasonable size for statistical purposes; only these samples were kept for further analysis. Table 1 presents the number of projects for each of the following programming languages with over 30 projects: COBOL, C, Visual BASIC, C++, SQL and Oracle[1]. For all other alternative programming languages within the ISBSG repository, there was an insufficient number of projects for our purposes.

| Programming language | Number of projects |
|---|---|
| Cobol | 413 |
| C | 139 |
| Visual Basic | 103 |
| C++ | 101 |
| SQL | 90 |
| Oracle | 87 |
| Total | 933 |

Table 1. ISBSG R8 -Programming language with over 30 projects

[1] These are the programming languages as recorded in the ISBSG repository. Some data collectors might have associated an environment (eg. ORACLE) to a programming language.

## 3. Identification of Outliers

In Figure 1, the functional size in function points (FP) is on the X-axis and the effort in hours on the y-axis. Figure 1 is typical of data sets available in software engineering, that is with an increasing dispersion of data, (referred to as heteroscedasticity) [3,4,5].

A number of outliers can been observed in Figure 1, with either very high productivity while others have very low productivity for projects of equivalent size.

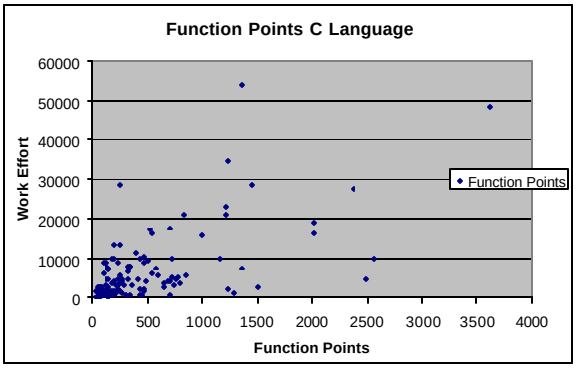


Figure 1. Data set with heteroscedasticity

Figure 2 points outsome projects in COBOL2 – R9 that have a large functional size with almost no corresponding effort: for illustrative purposes, seven (7) outliers were selected which appear to have very large economies of scale. These 7 outliers within a functional size range of 1000 to 2500 FP did not cost more than many projects 10 to 20 times smaller, thereby appearing to benefit from very large economies of scale (by a factor in the 10 to 20 range). The most probable cause is that there are some other variables that could explain such a minimal effort for such large size for these projects

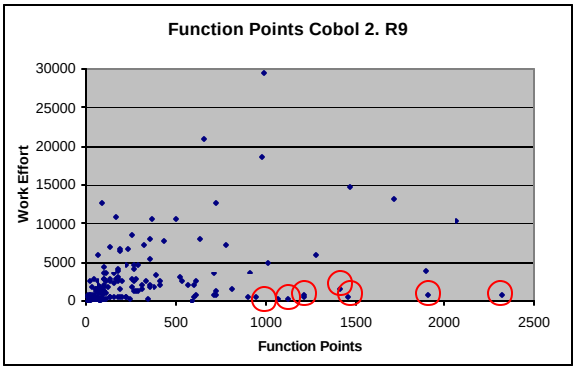


Figure 2. Visual identification of outliers with very large economies of scale

Figure 3 points out next to some projects in C language with large effort with relatively small functional size. Again for illustrative purposes, 3 outliers were selected that could qualify as having somewhat large dis-economies of scale, in particular for the outlier in the 300 FP range with a cost at least 10 times more than projects of similar size. The other two outliers identified graphically do not have such a large effort discrepancy, while still present.

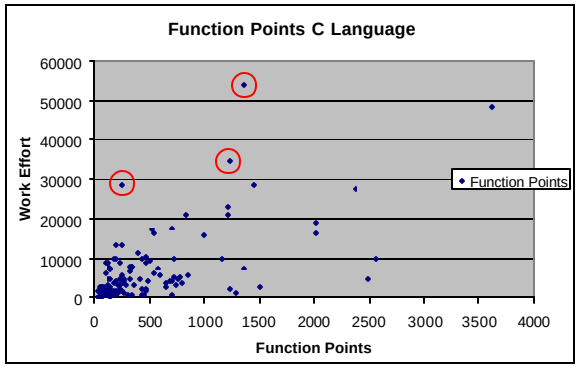


Figure 3. Visual identification of outliers with dis-economies of scale

## 4. Investigation of Outliers

Once the outliers identified, they are next compared to other projects of similar size or effort to explore if there exists patterns in the other variables recorded that might explain such outliers.

For the analysis of the ISBSG data repository, a good number of distinct tests selected by heuristics on some of the variables available in the repository were carried out on both R8 and R9 releases. In practice, only 8 tests gave results allowing a practical interpretation leading to the formulation of candidate hypotheses to be tested later with more robust statistical analyses.

### 4.1. Outliers with economies of scale

The analyses of the outliers with very large economies of scale are presented in tables 2 to 4, by programming language:

Table 2: COBOL - R8: 53 projects, including 10 outliers.
Table 3: C - R9: 118 projects, including 7 outliers.
Table 4: COBOL2 - R9: 115 projects, 14 outliers.

In these tables, the variables tested by heuristics are on the left hand-side column, and the value most often observed in the outliers for such a variable tested, in the next column. The other two columns present the ratio of observations of this value over the samples, first within the subset of outliers, and finally within the sample to the exclusion of the outliers.

For instance, in Table 2 for COBOL projects, the first variable tested is the Data Quality Rating assigned to a project by the ISBSG repository manager. It can then be observed that the worst value for this

variable, that is D = poor quality (column 2) is present in 10 out of 10 outliers (column 3) and only in 1 out of the other 43 projects Column 4) within the sample of projects in COBOL; that is 100% of the outliers have data considered of very poor quality, while only 2% of the other projects in COBOL have a poor data quality rating.

| Variables tested | Value observed | Ratio of Outliers | Ratio of Non-outliers |
|---|---|---|---|
| Data Quality Rating | D | **10 / 10** (100%) | **1 / 43** 2,3% |
| Resource Level | 2 | **10 / 10** (100%) | **12 / 43** (28%) |
| Organization type | Insurance | **10 / 10** (100%) | **12 / 43** 28% |
| Reference table approach | Counted as inputs | **10 / 10** (100%) | **7 / 43** (16,3%) |

Table 2. Economies of scale : COBOL - R8 (N=53).

In tables 2 to 4, several variables have been identified by heuristics as partially responsible for the outliers behavior in terms of project productivity ratios. The ISBSG definitions of these various variables are presented next :

- Data Quality rating: Quality of the data, as evaluated by the ISBSG repository manager.
- Resource Level: Personnel included in the recording of effort.
- Organization type: Type of organization which sent the data.
- Reference table approach: IFPUG Function Points version used to count the tables of codes in the software[2].
- Operating system: Operating system (O/S) on which the software measured runs.
- Primary database system: The main database management system (DBMS) for the software measured.

The values admissible for the "Data Quality Rating" are:
A = data submitted was assessed as being sound.
B = appears fundamentally sound but there are some factors which could affect the integrity of the data.
C = Due to significant data not being provided, it was not possible to assess the integrity of the submitted data.
D = Due to one factor or a combination of factors, little credibility should be given to the submitted data.

The values admissible for the Resource Level are:
1 = development team only
2 = development + support teams
3 = development + support teams + operators
4 = development + support teams + operators + customers

In Table 2, all of the outliers share the same values for the 4 variables identified: they all (eg. 100%) have a poor data quality rating, their effort include hours for both direct development staff and support staff, are insurance projects and they have used for size measurement an IFPUG version that takes into account each code table.

For the non outliers (Table 2), these characteristics are much less frequent (from 2 to 28 % of the projects).

For the sample with the projects in C (Table 3), there are two candidate explanatory variables for the economies of scale: the AIX Operating System and Sysbase as the primary DBMS which appear in around 50% of the outliers, and only 4% of the non outliers.

| Variable tested | Valeur observed | Ratio of Outliers | Ratio of Non-outliers |
|---|---|---|---|
| Operating System | AIX | **3 / 7** (42,9%) | **4 / 89** (4,5%) |
| Primary Database System | Sybase | **4 / 7** (57,1%) | **4 / 111** (3,6%) |

Table 3. Economies of scale : C - R9 (N=118).

For the sample with the projects in COBOL2 (Table 4), there are again four candidate explanatory variables for the economies of scale: they are the same as for the C sample.

| Variable tested | Value Observed | Ratio of Outliers | Ratio of Non-outliers |
|---|---|---|---|
| Data Quality Rating | D | **13 / 14** (92,9%) | **8 / 101** (7,9%) |
| Resource Level | 2 | **14 / 14** (100%) | **36 / 101** (35,6%) |
| Organization type | Insurance | **14 / 14** (100%) | **21 / 101** (20,7%) |
| Reference table approach | Counted as inputs | **14 / 14** (100%) | **21 / 101** (20,7%) |

Table 4. Economies of scale : COBOL2 - R9 (N=115).

[2] This is a peculiarity of the IFPUG method: depending on which IFPUG version is selected for the measurement of Tables of code, there can be large differences in the number of Function Points.

### 4.2. Outliers with dis-economies of scale

The results of the analyses of the outliers with dis-economies of scale, that is with a very high effort for comparable projects of smaller functional size, are presented in Tables 5 to 9.

Table 5: C - R8: 40 projects, 6 outliers
Table 6: Java - R9: 24 projects, 4 outliers
Table 7: COBOL - R8: 412 projects, 7 outliers
Table 8: C - R9: 16 projects, 4 outliers
Table 9: SQL - R9: 26 projects, 4 outliers.

In tables 5 to 9, four additional variables have been identified by heuristics as partially responsible for the outliers' behavior in terms of project productivity ratios. The ISBSG definitions of these variables are presented next:

- Standard FP: IFPUG standard used to count the points of function.
- Max TEAM size: Maximum number people who worked on the project at the same time (peak time).
- Lines of code: Number of lines of source code produced by the project.
- Project elapsed time: Duration, in months, to complete the development of the project.

In Table 5 for the C sample, the two most discriminative variables for dis-economies of scale are the Max team size greater than 10 people and Lines of code greater than 100 000, that is projects of relatively large size when compare to the full sample of C projects.

| Variable tested | Value Observed | Ratio of Outliers | Ratio of Non-outliers |
|---|---|---|---|
| Data Quality Rating | B | **6 / 6** (100%) | **24 / 34** (70,6%) |
| FP Standard | CPM 4.0 | **3 / 6** (50%) | **7 / 34** (20,6%) |
| Max team size | > 10 | **4 / 6** (66,7%) | **4 / 34** (11,8%) |
| Lines of code | > 100 000 | **2 / 6** (33,3%) | **2 / 34** (5,8%) |

Table 5. Dis-economies of scale : C - R8 (N=40).

In Tables 6 and 7 for the Java and COBOL samples, a single discriminative variable has been identified by heuristics for dis-economies of scale for both COBOL and C samples, that is, projects with a Max team size greater than 10 people.

| Variable tested | Value Observed | Ratio of Outliers | Ratio of Non-outliers |
|---|---|---|---|
| FP Standard | IFPUG 4 | **4 / 4** (100%) | **2 / 20** (10%) |

Table 6. Dis-economies of scale : Java - R9 (N=24).

| Variable tested | Value Observed | Ratio of Outliers | Ratio of Non-outliers |
|---|---|---|---|
| Max team size | > 10 | **5 / 7** (71,4%) | **27 / 405** (6,7%) |

Table 7. Dis-economies of scale : COBOL - R8 (N=412).

| Variable tested | Value Observe | Ratio of Outliers | Ratio of Non-outliers |
|---|---|---|---|
| Max team size | > 10 | 3 / 4 (75%) | 3 / 12 |

Table 8. Dis-economies of scale : C - R9 (N-16).

Finally, in Table 9 for the SQL sample, the two most discriminative variables for dis-economies of scale are a resource level that includes staff in addition to the development and support teams and a project elapsed time of over 15 months in duration.

| Variable tested | Value observed | Ratio of Outliers | Ratio of Non-outliers |
|---|---|---|---|
| Resource Level | > 2 | 3 / 4 (75%) | 1 / 22 (4,5%) |
| Project Elapsed time | > 15 months | 3 / 4 (75%) | 2 / 22 (9,1%) |

Table 9. Dis-economies of scale : SQL - R9 (N=26).

## 5. Summary & Discussion

This paper has discussed the issue of outliers in the repository of software projects of the International Software Benchmarking Standards Group - ISBSG. The criteria used for the identification of outliers is whether the productivity is significantly lower and higher in relatively homogeneous samples, that is projects with significant economies or dis-economies of scale. Once the outliers identified, other project variables were investigated by heuristics to identify candidate explanatory variables that might explain such outliers' behaviors.

Candidate variables identified as potentially related to large economies of scale in the ISBSG repository for some programming languages have been identified as: resource level 2, insurance as the organization type and the peculiarity of the Reference table approach in the IFPUG Function Points sizing method. The D rating for the data quality assigned to the outliers project is a somewhat confounding factor:

it is not a data collected by an organization, but rather a judgment of the ISBSG repository manager who has indeed identified an unusual effort relationship with respect to size, but which does not provide any clue into the whys of such a pattern nor does it provide confirmation that the data is erroneous.

Candidate variables identified as potentially related to large dis-economies of scale in the ISBSG repository for some programming languages have been identified as: maximum team size larger than 10 people, lines of code greater than 100 000, project duration greater than 15 months and effort data which includes not only development and support staff, but as well operators and customers project related effort. The specific version of the IFPUG Function Points method is also a variable identified as a candidate explanatory variable.

Of course, this list of candidate explanatory variables is far from being exhaustive: further research is required on the one hand for more robust methods for identifying in a systematic manner the outliers and, on the other hand, for investigating causes of such outliers' behaviors. Such further research will be challenging and time consuming.

Practitioners, however, can derive immediate benefits from this exploratory research in the following way: monitoring of the candidate explanatory variables can provide valuable clues for early detection of potential project outliers for which most probable estimates should be selected not within a close range of values predicted by an estimation model, but rather at their upper or lower limits: that is, the selection of either the most optimist or most pessimist value that can be predicted by the estimation model being used.

## References


[1] ISBSG, *Estimating, Benchmarking & Research Suite Release 8 & 9*, International Software Benchmarking Standards Group – ISBSG, Australia, 2005.

[2] ISBSG, International Software Benchmarking Standards Group, www.isbsg.org

[3] B.A Kitchenham, N.R. Taylor, "Software Cost Models", *ICL technical journal*, Vol. 4, no 1, May 1984, pp. 73-102., B.,

[4] A. Abran, P.N. Robillard, "Function Points Analysis: An Empirical Study of its Measurement Processes," *IEEE Transactions on Software Engineering*, Vol. 22, no 12, 1996, pp. 895-909.

[5] A. Abran**,** I. Silva, L. Primera, "Estimation Models for Functional Maintenance Projects – Field Studies", in Journal of Software Maintenance: Research and Practice, Vol. 14, 2002, pp. 31-64.