Approximation techniques for measuring Function Points

Jean-Marc Desharnais, Alain Abran
École de Technologie Supérieure - ETS
1100 Notre-Dame Ouest, Montréal, Canada H3C 1K3
jmdeshar@ele.etsmtl.ca
aabran@ele.etsmtl.ca

Abstract: *The generic concepts of Function Points Analysis were published in the late 1970s, and later more detailed measurement rules were developed to improve consistency of measurement. Due to lack of good software documentation, it is not always possible to apply all the detailed rules, and measurers must fall back on approximation techniques. This paper presents an analysis of two such techniques: Function Points Simplified and "backfiring" with a ratio of lines of code per Function Point. Two verification criteria were selected from ISO 14143-3: accuracy and convertibility. Results from empirical studies with five data sets are reported.*

1 Introduction

The generic concepts of Function Points Analysis (FPA) were published initially in the late 1970s to derive the functional size of a software application. Subsequently, detailed measurement procedures were developed to transform these concepts into a measurement method aimed at improving the consistency of measurement results independently of contexts and measurers. The most recent detailed standards on Function Points are contained in the Counting Practices Manual, version 4.1, published by the International Function Point Users Group (IFPUG) [1]; this manual has over three hundred pages, with a large number of measurement rules and examples.

Five function types are recognized in FPA, and to each type corresponds a distinct set of three weights assigning a specific size in Function Points to the functions being measured – Table 1.

Function Type	Weights		
	Low	Medium	High
External Input – EI	3	4	6
External Output – EO	4	5	7
External Enquiry – EQ	3	4	6
Internal Logical File – ILF	7	10	15
External Interface File – EIF	5	7	10

Table 1: Set of FPA weights – in number of Function Points

To ensure that different measurers make a repeatable selection, detailed decision tables have been added to the initial design of FPA. The use of these decision

tables requires detailed analysis of each of the functions being measured, and the identification, in each function, of:

- the number of data element types (DET);
- the number of record element types (RET);
- the number of file types referenced (FTR).

For the application of these detailed measurement standards, the software documentation must be very detailed and of high quality. Such data are usually only available once the detailed project specifications have been prepared, which is much too late for estimation purposes. Similarly, this level of information is not available for measuring legacy software which lacks documentation.

It has been observed in practice that there are a number of instances where these detailed measurement rules are not being used:

- The documentation is not precise enough for the application of the detailed measurement rules;
- The amount of work required to apply the detailed measurement rules to obtain precise measures of the software, and the work required subsequently to update these measures, is perceived by management as being too expensive;
- Qualified measurers are not available.

To deal with incomplete documentation, and other related measurement difficulties, organizations have devised various approximation techniques to quantify the functional size of their software: that is, to approximate their size. Of course, such approximation techniques have to deal with the poor quality of the inputs to the measurement process, and produce measurements which cannot be as reliable as they would have been, had the detailed measurement standards been used.

The approximation techniques are sometimes referred to as rapid counts; there are no standards recognized for such approximation techniques and they will vary not only across organizations, but can also vary within the same organization.

This study presents an analysis of two types of techniques often used in industry to approximate the functional size of a software application: Function Points Simplified and the so-called “backfiring” technique.

An overview of the two types of approximation techniques is presented in section 2, and the verification criteria of accuracy and convertibility in section 3.

Empirical designs for testing the accuracy and convertibility criteria are presented in section 4. Verification results for the simplified technique are presented in section 4, and results for the backfiring technique in section 5. Observations and conclusions are presented in section 6.

2 The approximation techniques

2.1 Function Points Simplified - FPS

In the Function Points Simplified (FPS) approximation technique, a single set of weights is used for the different Function Types, instead of choosing among the 3 weights given in FPA (i.e. any row in Table 1). There are, of course, a number of ways to apply this technique: for example, using the set of medium weights [2], or determining the median weights obtained from a sample of projects in a specific organization [3]. In simplified approximation techniques such as these, a single weight is pre-determined to a function type, independently of the characteristics of the particular function type. There is no discrimination based on differences in size of the various functions, and so there is no need to obtain the detailed information required to capture such differences. The measurement process is, of course, much less time-consuming in this case, but it is correspondingly less precise in its measurement of the individual functions.

2.2 Backfiring

The backfiring technique was developed by Jones in the 1980s [2] who published a table of conversion ratios, by type of programming language, between a Function Point and a quantity (i.e. a number) of lines of code (LOC). When a software application is coded in multiple programming languages, the backfiring formula for deriving the size in Function Points on the basis of the number of lines of code is the following: (Number of LOC for language A / conversion ratio for language A) + (Number of LOC for language B / conversion ratio for language B) + …. + (Number of LOC for language X / conversion ratio for language X).

2.3 Previous studies

2.3.1 Function Points Simplified

The Bock and Klepper [3] assumption is that the mix of low, medium and high functions across systems is quite stable within one development site, “regardless of the size or the complexity of the systems.” If their assumption that the “function type mix stability between systems is valid over time, then the process for classifying functions… can be replaced by a procedure that uses a single multiplicative adjustment factor for each of the five function types.” Bock and

Klepper derived such a set of adjustment factors through a multiple regression procedure [3].

2.3.2 Backfiring

This approximation technique is based on the following assumptions:

- that there is a direct relationship between the number of LOC and the number of Function Points,
- that such a relationship is constant across contexts (i.e. stable),
- that there is a fairly limited variation across this ratio.

However, there is no documented evidence to support such assumptions: Jones has not published any data on either the size and the demographics of his samples for each programming languages or statistical details for each conversion ratio published (for instance, if the ratios correspond to averages, what are the standard deviation, distribution shape of the data samples, etc.). There is no discussion either on the representativeness of the data samples and ratios for their use outside their initial contexts of derivation.

Henderson & Garland [4] investigated the backfiring technique using the following two data sets:

A) Commercial data set: from two distinct databases of MIS projects reported in Albrecht [5] and Kemerer [6]. Of these 39 projects, 31 were written in COBOL and could be used for statistical analysis, while the other 8 projects were written in three other languages and did not have enough data points each for statistical analysis.

B) Military data set: MIS projects completed for the Air Force.. Of the 61 projects, 21 were in COBOL, while the others were in many different languages with not enough observations each to be considered a statistically valid sample [4, p. 41-42].

Henderson and Garland used the statistical regression techniques and related criteria: the coefficient of determination (R^2) measures the strength of the relationship between the independent and dependent variables (here FP and SLOC), and the coefficient of variation multiplied by two gives the 95% prediction bounds (in percentage) around the center of the data, if it is distributed normally. The coefficient of variation should be less than 50%.

Henderson's results are summarized in Tables 2 and 3. While the R^2 are relatively high for both data sets (Table 2), these results should be used with

caution since the coefficients of variation are greater than 50%, and both regression models are lacking in predictive power.

Indicators	Commercial Data Set (31 COBOL Projects)	Military MIS Data Set (21 COBOL Projects)
Regression Equation	SLOC = 69.5 + 13.4 x FP	SLOC = 116.1 + 178.4 x FP
R^2	0.96	0.72
Coefficient of variation	64.6	53.2

Table 2: Regular Regression Models – Henderson results

To obtain the direct conversion factors derived from these data sets, and compare them with the published conversion ratios of Jones [9], Henderson used regression models with the intercept forced to 0. The results are presented in Table 3.

Indicators	Commercial Data Set (31 COBOL Projects)	Military MIS Data Set (21 COBOL Projects)
Regression Equation	SLOC = 13.7 FP	165.1 FP
R^2	0.96	0.72
Coefficient of variation	69.5	53.2

Table 3: Regression Models with Intercept = 0 – Henderson results

When compared to the 105 COBOL SLOC/FP, the ratios reported by Jones and the 100 COBOL SLOC/FP reported by Reifer [7, p. 97], the ratios found by Henderson differ significantly, that is, 13.7 COBOL SLOC/FP for the commercial data set and 165 COBOL SLOC/FP for the military data set.

Henderson's conclusions are as follows:

- for both data sets, the SLOC/FP published conversion ratios should not be used;
- with such a large range of variation in the SLOC/FP ratio (13 to 165 for COBOL), conversion ratios as useful estimating tools for Function Points are tenuous at best [4, p. 97].

2.4 Verification criteria

Since the publication of these studies, ISO has developed a verification guide for the assessment of Functional Size Measurement methods: ISO 14143-3 (2003) [8]. Two criteria from this ISO Guide have been selected for the study reported here: accuracy and convertibility.

2.4.1 Accuracy

ISO 14143-3 defines accuracy of measurement as "the closeness of the agreement between the result of a measurement and the true value of the measurand."

A true value, as defined in the ISO Vocabulary on basic and general terms in metrology [9], is a value consistent with the definition of a given particular quantity, and this is a value that would be obtained by a perfect measurement. In contexts where perfect measurement is not feasible, a conventional true value is a value attributed to a particular quantity and accepted, sometimes by convention, as having an uncertainty appropriate for a given purpose. 'Conventional true value', in the same reference, is sometimes called an assigned value, best estimate of the value, conventional value or reference value.

The accuracy of the FPS approximation technique will be verified against the true value obtained from the standard for these types of measurement methods, that is, the IFPUG Function Points Analysis method. It is recommended in ISO 14143-3 [8] that accuracy should be expressed in terms of the Mean Magnitude of Relative Error.

The set of tests used in this study has been derived from the test procedures described for verification of the accuracy criteria of Functional Size Measurement methods in ISO 14143-3 [8].

2.4.2 Convertibility

Convertibility is defined in ISO 14143-3 [8] as the ability to convert the results of applying two or more FSM methods in the measurement of the same set of Functional User Requirements.

The convertibility criteria will be studied for the backfiring technique, which is basically a conversion technique from one unit type (lines of code) to another (Function Point units). The accuracy criterion was not used for the backfiring technique since it is not the relevant verification criterion in this context.

The other type of approximation technique, that is, Function Points Simplified, produces units of the same type as FPA, and so there is no convertibility issue in this case.

3 Empirical designs

3.1 IFPUG Standard and LOC standard

The generally recognized standard for measurement in Function Point units is currently IFPUG version 4.1. Measurements carried out using this standard will

be used as the benchmark to assess the results of the approximation techniques on both the accuracy and convertibility criteria.

There is no generally recognized standard for counting lines of code (LOC). The logical definition put forward by Boehm [10] (source instructions) is used here: LOC "includes all program instructions created by project personnel and processed into machine code by some combination of preprocessors, compilers, and assemblers. It excludes comment cards and unmodified utility software…" [10, p. 59].

3.2 Data sets

Five distinct data sets were available for this study, and these are briefly described below.

- Data Set A consists of detailed measures of 13 software applications (a portfolio) from a North American government agency. IFPUG version 4.0 was used for the measurements at the time of data collection, and all detailed measurements were available for this study. In addition, the number of LOC was available for each software application, and COBOL was the only programming language used. This development environment can be characterized as fairly homogeneous and stable, and the size of the software group was within the 200 staff range. The software systems were all of the MIS type and transaction-based. The Function Points were counted manually using IFPUG version 4.0 by an external team of 3 counters under the direct supervision of a certified Function Point specialist who controlled the quality of the counting process. The count was based on software documentation of remarkable quality in that it consisted of what was considered to be accurate, complete and up-to-date systems documentation, including data models and data-flow diagrams. Most of this documentation was produced, and maintained up-to-date, with upper CASE tools. The LOC count was obtained through the same automated line counter tool for all software. The actual data collection process took place in 1996.

- Data Set B consists of detailed measures of 11 development projects from another North American government department. IFPUG version 4.0 was used for the measurements at the time of data collection, and all detailed measurements were available for this study. In addition, the number of LOC was available for each software application, and COBOL was again the only language used. This development environment can be similarly characterized as fairly homogeneous and stable, and the size of the software group was within the 500 staff range.

- Data Set C consists of detailed measures of 9 development projects from a European banking organization. IFPUG version 4.0 was used for the measurements at the time of data collection, and all detailed measurements were available for this study. In addition, the number of LOC was available for each software application, and COBOL was again the only language used. This development environment can be similarly characterized as fairly homogeneous and stable, and the size of the software group was within the 500 staff range.

- Data Set D consisted of 8 development projects from a North American telecom organization. IFPUG version 3.4 was used for the measurements at the time of data collection, and the detailed measurements were not available for this study. In addition, the number of LOC was available for each software application, but a wide variety of programming languages was used. The software environment can be characterized as fairly diversified and the size of the organization was within the 500 staff range.

- Data Set E consists of 90 development projects measured in 5 different North American organizations (Appendix B). These projects were measured by an expert certified in functional size measurement. The information on LOC was not available.

4 Function Points Simplified – FPS – Empirical results

For verification of the accuracy of the simplified technique as an approximation technique, the following empirical verification procedure was designed, using two data sets.

Data set E, containing 90 projects from 5 organizations was split approximately in half with data in each from the same organizations. This resulted in an initial set of 47 projects from 3 organizations (referred to as Sample1), and 43 projects from 2 organizations (referred to as Sample 2) (Appendix A). Sample 1 from data set E was used to derive the simplified FP weights per function type, and then these weights were applied to the second sample from data set D, and the results compared against the results from the application of the detailed measurement standard at the time of the original data collection.

This empirical design can be compared to the Bock approach [3]. We used a slightly modified approach, as described above, instead of sampling our projects randomly. Our selection was made on the basis of distinct organizations. Another difference in the empirical design is that our regression analysis was carried out on all 5 function types instead of 4 (Bock leaves EIF out of the regression analysis).

To derive the simplified FP weights per function type, for each of the 47 projects in Sample 1, the number of points per function type was divided by the number of functions. Then, an average weight per function type was calculated using the full set of 47 projects – see Table 4.

Function Type	Average Weight
External Input – EI	4.2
External Output – EO	5.4
External Enquiry – EQ	4.4
Internal Logical File – ILF	7.9
External Interface File – EIF	5.8

Table 4: Average weights for Sample 1 of Datas Set E

Then, this set of average weights was applied to the second sample of 43 different projects. The detailed results of using this approximation technique for each of the 43 projects are presented in Appendix B, together with a comparison against the true values obtained by manual measurement using the detailed measurement standards. This is also illustrated in Figure 1, which gives both measurements for each of the 43 projects.

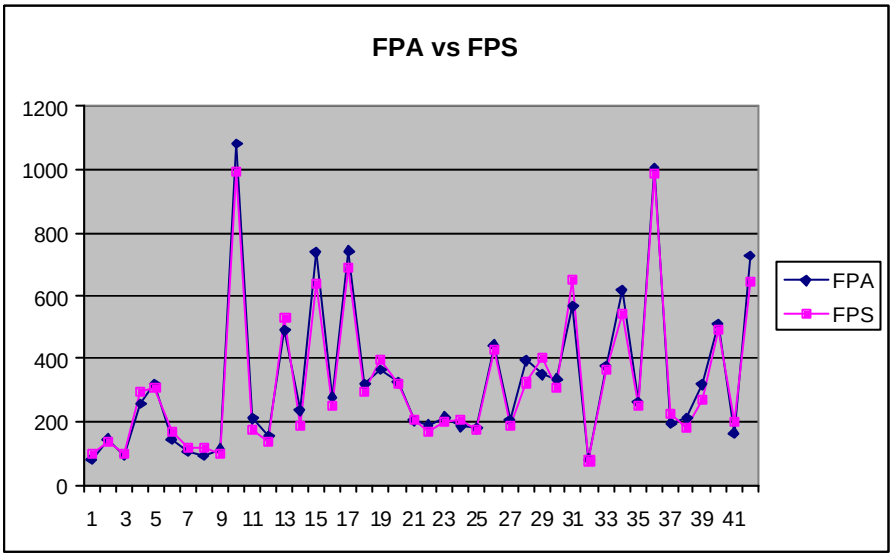

Figure 1: Detailed (FPA) and simplified (FPS) measurement results – Sample 2 of Data Set E

The correlation between the two types of measurement results is very high, at 98%, and the average difference between the two measures is less than 5% (Table 5).

The Measure of Goodness recommended by Conte et al. [11] was also investigated. It includes four criteria: R^2, Mean Relative Error, Mean Magnitude of Relative Error and the Prediction level. The results from Sample 2 are presented in Table 6, together with the thresholds recommended by Conte et al. The results obtained exceed the recommended thresholds on all four criteria.

Criteria	Results on Sample 2	Conte et al. criteria thresholds
R^2 – coefficient of determination	0.98	Greater than 0.95
RE – Mean Relative Error	10.17	Less than 25
MRE – Mean Magnitude of Relative Error	22.80	
PRED (.15) – Prediction at level 1	0.87	Less than .25 for 75% of the time

Table 5: FPS Accuracy - Comparison of Measures of Goodness

This means that, for Sample 2, the simplified technique can be used instead of the detailed FPA measurement method, with an accuracy range within 5%.

Of course, these conclusions are valid for these samples. Generalization to other contexts requires caution and an analysis of similarities and representativeness across samples.

5 Backfiring technique – Empirical results

With four distinct data sets using FPA and LOC, we produced the R^2 between the FPA and LOC, the average of LOC on FPA, the min and max numbers of LOC on FPA and the order of magnitude between the minimum and the maximum, as suggested by Henderson [4].

For verification of the convertibility criteria of the backfiring technique, the Henderson approach [4] and regression models were used. In regression models, the coefficient of determination (R^2) measures the strength of the relationship between variables (here, FP and SLOC), while the coefficient of variation multiplied by two gives the 95% prediction bounds (in percentage) around the center of the data, if it is distributed normally. The coefficient of variation should be less than 50%.

For data set A, the R^2 obtained is 98.5% between FPA and LOC. The average number of LOC on FPA for COBOL is 209. However, Figure 2 illustrates that, even with a very high R^2, there is a large variation between the minimum LOC per FPA (= 77) and the maximum LOC per FPA (= 393), which is an order of magnitude of more than 400%.

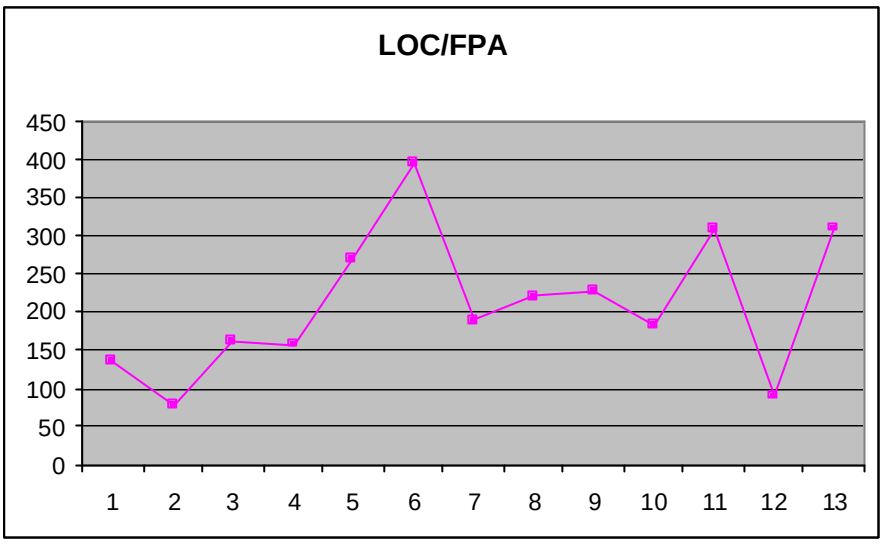

Figure 2: COBOL Convertibility Ratio (LOC/FPA) – data set A

For data set B, the R^2 is 51.9% between FPA and LOC. The average number of LOC on FPA for COBOL is 111. R^2 is lower (51.9%), and the average is about half the value of that found in data set A. Figure 3 again illustrates a large variation between the minimum LOC per FPA (= 53) and the maximum LOC per FPA (= 315), an order of magnitude of more than 600%.

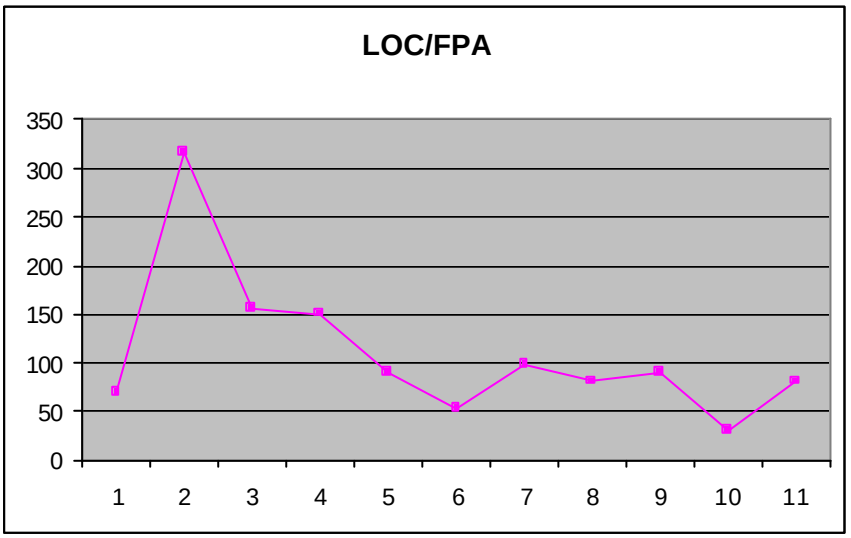

Figure 3: COBOL Convertibility Ratio (LOC/FPA) – data set B

For data set C, R^2 is 6.7% between the FPA and LOC. The average number of LOC on FPA for COBOL is 105. The R^2 is very low (6.7%), the average is about half that found in data sample A and about the same as that found in data sample B. Figure 4 illustrates again a large variation between the minimum LOC per FPA (= 26) and the maximum LOC per FPA (= 274), an order of magnitude of more than 1000%.

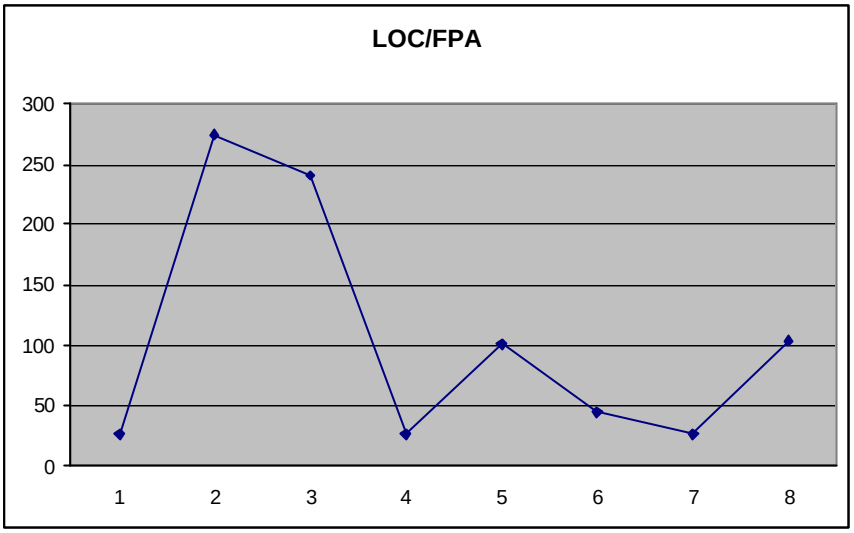

Figure 4: COBOL Convertibility Ratio (LOC/FPA) – data set C

For data set D, R^2 is 72.1% between the FPA and LOC when only the COBOL projects are considered. The average number of LOC on FPA for COBOL is 229. If we add the other languages, the average is 270. The R^2 is good (72.1%), the average being about the same as that found in Data Set A and twice that of the other two data sets (B and C). Figure 5 illustrates again a large variation between the minimum LOC per FPA (= 35) and the maximum LOC per FPA (= 382), an order of magnitude of more than 1100%.

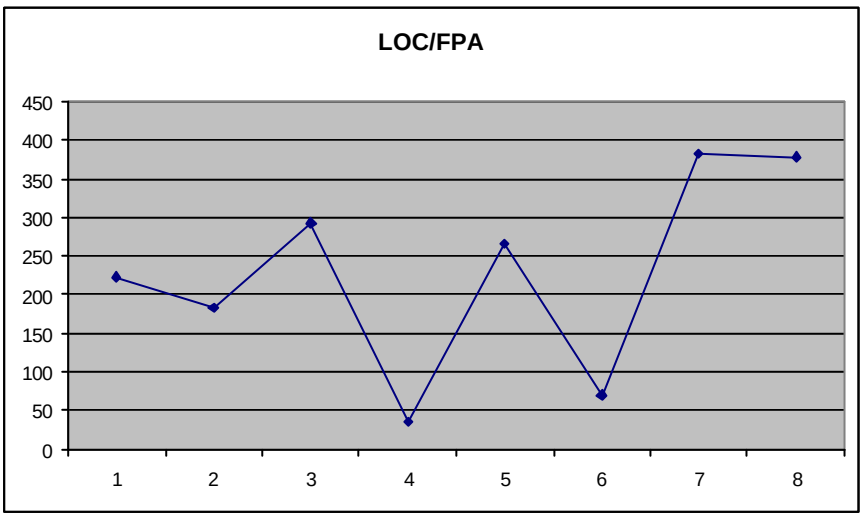

Figure 5: COBOL Convertibility Ratio (LOC/FPA) – data set D

The details of the LOC/FPA ratios at the project level are presented in Appendix A.

For the four data sets analyzed, the R^2 varies from 6.7% through 98.5%, the average LOC/FPA ratios vary from 105 through 229 LOC per FPA and the order of magnitude between the Min LOC/FPA and Max LOC/FPA varies from 510% through 1102%. It is obvious that there is large variability in these results

across samples. It would thus be unwise for an organization to rely on those conversion ratios. This also supports Henderson, who stated that "the best models for each environment contain much variability from the actual LOC values" [8, P. 104].

Data Set	Min LOC/FPA	Max LOC/FPA	Magnitude	AVG LOC/FPA	Rsquare
A	77	393	510%	209	0,985
B	31	317	1023%	111	0,519
C	26	274	1061%	105	0,067
D	35	382	1102%	229	0,721

Table 6: Summary of conversion ratios – data sets A to D

6 Observations and conclusions

It has often been observed in practice that, due to a lack of good software documentation, it is not always possible to apply all the detailed measurement rules of Function Points Analysis. Measurers must fall back on approximation techniques. This paper has presented an analysis of two such approximation techniques: Function Points Simplified and the so-called "backfiring" technique based on ratios of lines of code per Function Point. Two verification criteria were selected from ISO 14143-3: accuracy and convertibility. Results from empirical studies with five data sets were reported.

The empirical results indicate that, in the organizational contexts reported, the simplified technique can be used instead of the detailed measurement method, with an accuracy range within 5%. In such a context, the simplified technique represents a much less time-consuming measurement process, with an acceptable loss of accuracy. The extrapolation of these findings to other contexts will require further investigation.

The empirical results for the convertibility criteria for the 'backfiring' technique indicate that, in the organizational contexts reported, the backfiring technique did not meet any of the criteria recommended in the literature. Such an approximation technique should not be used, therefore, and this recommendation is consistent with previous findings by Henderson [4].

References

[1] IFPUG, "Function Point Counting Practices Manual, Release 4.1," International Function Point Users Group (IFPUG), Mequon, Wisconsin Release 4.1, January 1999.

[2] Jones C., Applied Software Measurement, Assuring Productivity and Quality, Second ed. New York, NY: McGraw Hill, 1997.

[3] Bock D. B. Klepper R., "FP-S: a simplified function point counting method," Journal of Systems and Software, vol. 18, pp. 245-54, 1992.

[4] Henderson G. Garland S., "The application of function points to predict source lines of code for software development," in Faculty of the School and Logistics of the Air Force Institute of Technology: Air University, 1992, pp. 190.

[5] Albrecht A.J. Gaffney J.E. Jr., "Software Function, Source Lines of Code, and Development Effort Prediction: A Software Science Validation," IEEE Transactions on Software Engineering, vol. SE-9, pp. 639-648, 1983.

[6] Kemerer C. F., "An empirical validation of software cost estimation models," Communications of the ACM, vol. 30, pp. 406-429, 1986.

[7] Reifer D.J., "Asset-R: A Function Point Sizing Tool for Scientific and Real-Time Systems," Journal of Systems Software, vol. 11, pp. 159-171, 1990.

[8] ISO 2003 ISO/IEC 14143-3: 2003, "Functional Sized Measurement Methods - Verification of Functional Sized Measurement Methods," International Organization for Standardization, Geneve 2003, 2003.

[9] ISO, "ISO Vocabulary on Basic and General Terms in Metrology," International Organization for Standardization, Geneva 1993.

[10] Boehm B.W., Software engineering economics. Englewood Cliff, New Jersey: Prentice-Hall Inc., 1981.

[11] S. D. Conte, H. E. Dunsmore, and V. Y. Shen, Software engineering metrics and models. Menlo Park: The Benjamin/Cummings Publishing Company, Inc., 1986.

Appendix A: Data Sets A, B, C and D

Data Sample A

NO	FPA	LOC	LOC/FPA
1	474	64255	136
2	239	18437	77
3	137	22116	161
4	252	39731	158
5	1041	279141	268
6	471	185327	393
7	252	47572	189
8	126	27874	221
9	152	34556	227
10	197	35785	182
11	2664	817880	307
12	116	10341	89
13	1296	402828	311
		Average	209

Data Sample B

NO	FPA	LOC	LOC/FPA
1	1287	90122	70
2	249	78979	317
3	734	114772	156
4	280	42166	151
5	742	66558	90
6	319	17000	53
7	366	35455	97
8	327	26569	81
9	208	18767	90
10	189	5806	31
11	218	17516	80
		Average	111

Data Sample C

NO	FPA	LOC	LOC/FPA
1	107	2763	26
2	268	73445	274
3	196	47187	241
4	1024	26996	26
5	456	45973	101
7	213	9412	44
8	80	2120	27
9	87	8967	103
		Average	105

Data Sample D

NO	FPA	LOC	LOC/FPA
1	1408	315236	224
2	4888	888388	182
3	2998	875851	292
4	2148	74426	35
5	767	203869	266
6	258	17977	70
7	896	342065	382
8	192	72787	379
		Average	229

Appendix B : Data Set E and detailed calculations of sample 2

Data Sample 1	
No	FPA
1	107
2	268
3	201
4	456
5	260
6	213
7	86
8	228
9	288
10	157
11	329
12	475
13	146
14	156
15	106
16	145
17	273
18	579
19	93
20	169
21	199
22	269
23	176
24	285
25	514
26	183
27	505
28	175
29	793
30	377
31	447
32	252
33	259
34	217
35	118
36	436
37	499
38	308
39	404
40	114
41	189
42	200
43	198
44	261
45	252
46	368
47	489

Data Sample 2 and FPS based on the avg		
No	FPA	FPS
1	81	98
2	145	139
3	91	100
4	255	295
5	317	309
6	144	169
7	109	118
8	95	117
9	111	99
10	1084	993
11	213	173
12	154	140
13	488	529
14	239	186
15	734	640
16	280	252
17	742	691
18	319	299
19	366	394
20	327	320
21	208	208
22	189	169
23	218	198
24	185	206
25	181	175
26	442	426
27	204	184
28	395	327
29	350	401
30	336	307
31	565	649
32	80	78
33	378	366
34	619	539
35	263	251
36	1002	983
37	194	229
38	213	182
39	321	271
40	510	493
41	165	200
42	728	645