Computing and Informatics, Vol. 26, 2007, 1–15

AN ALGORITHM FOR THE GENERATION OF SEGMENTED PARAMETRIC SOFTWARE ESTIMATION MODELS AND ITS EMPIRICAL EVALUATION

Juan J. Cuadrado-Gallego, Miguel-Angel Sicilia

Computer Science Department, Polytechnic Building, University of Alcalá
Ctra. Barcelona km. 33.6, 288 71 Alcalá de Henares, Madrid, Spain
e-mail: {jjcg, msicilia}@uah.es

Manuscript received 21 April 2005; revised 4 July 2006
Communicated by Clark Thomborson

Abstract. Parametric software effort estimation techniques use mathematical cost-estimation relationships derived from historical project databases, usually obtained through standard curve regression techniques. Nonetheless, project databases – especially in the case of consortium-created compilations like the ISBSG –, collect highly heterogeneous data, coming from projects that diverge in size, process and personnel skills, among other factors. This results in that a single parametric model is seldom able to capture the diversity of the sources, in turn resulting in poor overall quality. Segmented parametric estimation models use local regression to derive one model per each segment of data with similar characteristics, improving the overall predictive quality of parametrics. Further, the process of obtaining segmented models can be expressed in the form of a generic algorithm that can be used to produce candidate models in an automated process of calibration from the project database at hand. This paper describes the rationale for such algorithmic scheme along with the empirical evaluation of a concrete version that uses the EM clustering algorithm combined with the common parametric exponential model of size-effort, and standard quality-of-adjustment criteria. Results point out to the adequacy of the technique as an extension of existing single-relation models.

Keywords: Parametric software estimation, software project databases, clustering algorithms, EM algorithm

1 INTRODUCTION

Parametric estimation techniques are nowadays widely used to measure and/or estimate the cost associated to software development [1]. The *Parametric Estimating Handbook* (PEH) [17] defines parametric estimation as "a technique employing one or more cost estimating relationships (CERs) and associated mathematical relationships and logic". Parametric techniques are based on identifying significant CERs that obtain numerical estimates from main *cost drivers* that are known to affect the effort or time spent in development. Parametrics uses the few important parameters that have the most significant cost impact on the software being estimated.

One important aspect of the process of deriving models from databases is that of the heterogeneity of data. Heteroscedasticity (non-uniform variance) is known to be a problem affecting data sets that combine data from heterogeneous sources [18]. When using such databases, traditional application of curve regression algorithms to derive a single mathematical model results in poor adjustment to data and subsequent potential high deviations in the results of the estimation process. This is due to the fact that a single model can not capture the diversity of distribution of different segments of the database points. As an illustrative example, the straightforward application of a standard least squares regression algorithm to the points used in the Reality tool of the ISBSG 8 database[1] distribution results in measures of MMRE = 2.8 and PRED(.3) = 23 % (these measures are introduced later), which are poor figures of predictive quality. Other studies have used much smaller and more homogeneous datasets than the ISBSG, as those surveyed in [8], and thus they are less likely to be negatively influenced by the difficulties posed by heterogeneity. Wittig and Finnie used ISBSG 5.0 in their experiments, but the data amounted to 136 samples [3]. Even in such small data set, the authors considered the "large" ranges of system size, development effort and productivity as a complication in the development effort estimation process, and eliminated extreme points as a way to overcome the effect of heterogeneity.

In addition to the considerations above, the use of a single model intuitively goes against the notion of "cost realism" described in the PEH, since it attempts to use the same model for elements in which the CERs may have not the same characteristics. These issues point out to the possible adequacy of partitioning data sets before the adjustment of the models. Since current databases as the ISBSG and internal project databases grow continuously, it is desirable to attain a level of automated segmentation in the process of adjusting the final parametric model.

The use of clustering techniques has been described as a solution to provide more realism to parametric models by decomposing the model in a number of submodels, one per segment, that are used to estimate points that are near them [11, 5], yielding improved predictive characteristics in empirical evaluations. The resulting predictive schemes have been called *segmented* models.

[1] http://www.isbsg.org/

Related work includes the use of different clustering approaches to several aspects of software management, including software estimation, software quality and software metrics. Concretely, Xu and Khoshgoftaar [21] use the *fuzzy c-means* algorithm for variable, the partitioning of the data into a number of clusters based on experiences. Pedrycz and Succi [19] also use fuzzy c-means as a tool to derive prototypes related to software code measurements. Dick et al. [7] use the same algorithm for a similar setting in a knowledge discovery study. Gray and MacDonell use fuzzy rules combined with clustering [12]. Nonetheless, none of these approaches deal explicitly with the heterogeneity of the project databases they use as a first step for a standard parametric model, but use a different, non-parametric approach.

Lung, Zaman, and Nandi [13] have used the numerical taxonomy method for the clustering of software components at several development phases, but these analyses are driven by the structure of the code, which is rarely available in public historical software project databases. Oligny et al. [16] approach estimation studies by the partitioning of the project database into "more homogeneous subsets". This study can be considered as supporting evidence for the segmentation approach described in this paper, even though the partitioning of the data is carried out without using a clustering algorithm.

One of the principal benefits of segmented parametric techniques is the fact that the search of segmented models that satisfy some pre-established quality conditions can be automated through existing clustering methods. From a pragmatical perspective, this would entail that common software estimation tools or software packages like COCOMO or PRICE-S could include a module for the generation of candidate segmented models based on in-house historical project databases, as recommended by current practice [8]. This would be analogous to the *calibration* procedures included in the USC-COCOMO tool [2], and would enable the tailoring of models to the specificities of the organization. Oligny et al [16] approach estimation studies by the partitioning of the project database into "more homogeneous subsets". This study can be considered as supporting evidence for the segmentation approach described in this paper, even though the partitioning of the data is carried out without using a clustering algorithm. Preliminary data for the use of clustering following the same considerations is described in [11]. Even though the automated solution would require the assessment of a parametric analyst for quality, the use of an algorithm to generate candidate segmented models eases the analysis process in a process similar to common knowledge discovery activity cycles.

The rest of this paper is structured as follows. Section 2 describes a general account of parametric segmented models in software effort estimation. A generic recursive algorithmic scheme is discussed in Section 3. Then, Section 4 reports the evaluation of an instance of the generic algorithm that uses common quality criteria and a size-based CER as the input for a clustering and local-regression procedure. Finally, conclusions and future research directions are provided in Section 4.

2 SEGMENTED MODELS FOR PARAMETRIC SOFTWARE ESTIMATION

Standard parametric models are usually obtained from the entire historical project database using conventional curve regression techniques, relating effort or schedule predictions to a number of cost drivers $c_i \in \mathcal{C}$. Expression (1) shows one of the most usual concrete models for the relationship between size (expressed in function point estimates [10]) and total effort measured for example in total hours or effort spent.

$$e = a \cdot fp^b \text{ generally } e = f(c_i) \ \mathcal{C} = \{c_i\} \tag{1}$$

The exponential model was used for the sake of comparability, since most previous studies used it, see e.g. [8].

Segmented models replace the single-equation approach with a collection of mathematical models f_j, each of them associated to the definition of a segment $s_i \in \mathcal{S}$, as expressed in (2).

$$e = f_j(c_i) \ \ j = \gamma(c_i) \text{ with } segment(f_j) = s_j \tag{2}$$

Segment definitions may be expressed in different ways, depending on the clustering technique used with the project database. The mapping function $\gamma(c_i)$ is responsible for selecting the function for each particular project being estimated, and it proceeds by finding out the segment (cluster) that best characterizes the project under consideration. Expression (2) could be generalized by introducing the possibility for a given point to be assigned to several segments to a given degree, and using some form of aggregation for the contribution of the relevant local estimating functions. Nonetheless, such extension is out of the scope of this paper.

Related work includes the use of different clustering approaches to several aspects of software management, including software estimation, software quality and software metrics. Concretely, Xu and Khoshgoftaar [21] use the *fuzzy c-means* algorithm for variable, the partitioning of the data into a number of clusters based on experiences. Pedrycz and Succi [19] also use fuzzy c-means as a tool to derive prototypes related to software code measurements. Dick et al. [7] use the same algorithm for a similar setting in a knowledge discovery study. Nonetheless, these approaches do not deal with the heterogeneity of the project databases they use. Lung, Zaman, and Nandi [13] have used the numerical taxonomy method for the clustering of software components at several development phases. Nonetheless, no previous research has addressed a generic mechanism for the recursive selection of segmented models for parametric software estimation.

Our aim in this paper is that of providing a generic algorithmic schema for the process of selecting candidate segmented models given a historical project dataset and a number of quality criteria regarding the final predictive value. Such algorithm is recursive in nature, since the same rationale used to divide the initial dataset could be applied in subsequent iterations whenever certain conditions are met, as that of not reducing the size of the clusters below a given reasonable threshold. Such

approach would enable further experimentation with different quality criteria, clustering models and consideration of cost drivers, eventually leading to a meta-analysis that would help in deciding which schemes fit better each estimation context.

3 AN ALGORITHM FOR THE GENERATION OF SEGMENTED ESTIMATION MODELS

The algorithm for the generation of segmented estimation models should be framed in the general process of obtaining parametric estimation models. According to the guidance of the PEH, the main elements of the development of a parametric model entails a first phase of data collection and normalization followed by the development of the cost model. The latter includes calibration and the validation of the model. Figure 1 depicts these main phases. The algorithm proposed here can be considered as a tool for the iterative calibration-validation of CERs to the available data, since it is intrinsically driven by a concept of accuracy comparison that is typically used in the validation step.

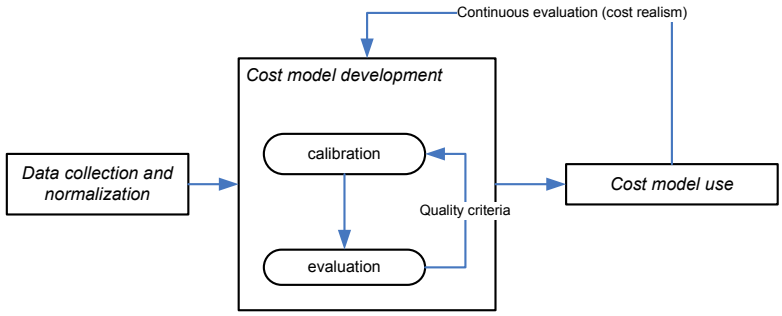

Fig. 1. Main steps in the development of parametric CERs

The SEARCH-SMODEL algorithm provided below in pseudocode captures the main elements to be considered in the automated search for calibrations driven by segmentation criteria.

SEARCH-SMODEL($\mathcal{D}$, $\mathcal{C}$,$\mathcal{Q}$, m, A)
- $\triangleright$ $\mathcal{D}$ is the historical dataset
- $\triangleright$ $\mathcal{C}$ is the set of relevant cost drivers
- $\triangleright$ $\mathcal{Q}$ are the (desired) quality constraints on the result
- $\triangleright$ m is the unadjusted mathematical model
- $\triangleright$ A is the algorithm for finding segments

```
1  f ← FIND-LOCAL-MODEL(D, C, m)
2  if SATISFIES(D, Q, f) ∨ ¬SEGMENTABLE(D, C)
       then
3          return f
       else
4          ⟨s_1 ... s_k⟩ ← A(D, C)
5          if k > 1
6          for each segment s_i ∈ ⟨s_1 ... s_k⟩
               do
7                  f_i ← SEARCH-SMODEL(s_i, C, Q, m, A)

8          return JOIN-MODELS(⟨f_1 ... f_k⟩)
```

The following are preconditions to the algorithm:

1. The dataset $\mathcal{D}$ is a relation that contains in its domain the elements included in the set of cost drivers $\mathcal{C}$
2. The segmentation algorithm must be able to deal with the type of cost drivers in $\mathcal{C}$, be it nominal, ordinal or any other.
3. The mathematical model m must be coherent with the relevant cost drivers selected.
4. The segment-finding algorithm must be capable of dealing with the data types of the relevant cost drivers selected. For example, if a clustering algorithm is used, it must be capable of dealing with the type (nominal, ordinal, real) of the cost drivers, or type mappings like discretizations must be provided *ad hoc*.

The assumption that the cost drivers considered constraint the attributes to be used both in the finding of local models (FIND-LOCAL-MODEL) and in the process of split (A) is considered. This is common practice but may be otherwise in some specific situations.

Related work has used genetic algorithms for the selection of the mathematical model itself [8]. Nonetheless, the scope of the algorithm discussed here is that of adjusting the divergences in variance to the well-known exponential function used in parametric estimation, and not postulating different models.

As an illustration, the following algorithm (that was used for the empirical evaluation) is a concrete instantiation of the previous one, which considers common practices in parametric estimation and uses the EM algorithm [6] as the segmentation procedure. The expectation-maximization (EM) algorithm is an algorithm for finding maximum likelihood estimates of parameters in probabilistic models, where the model depends on unobserved latent variables. EM is frequently used for data clustering in machine learning and computer vision. EM alternates between performing an expectation (E) step, which computes an expectation of the likelihood by including the latent variables as if they were observed, and a maximization (M)

step, which computes the maximum likelihood estimates of the parameters by maximizing the expected likelihood found in the E step. The parameters found in the M step are then used to begin another E step, and the process is repeated.

SEARCH-SMODEL-BASIC-EM($\mathcal{D}$, $\{effort, size\}$, q_1, q_2)

```
1  f, tr, ts ← NON-LINEAR-REGRESSION(D, {effort, size})
2  if (MMRE(f, ts) < q1 ∧ PRED25(f, ts, q2) > q2) ∨ |D| < θ)
       then
3          return f
       else
4          ⟨s1 ... sk⟩ ← EM(D, {effort, size})
5          if k > 1
6          for each segment si ∈ ⟨s1 ... sk⟩
               do
7                  fi ← SEARCH-SMODEL(si, C, Q, m, A)

8          return COMPOSE-MODELS(⟨f1 ... fk⟩)
```

In the SEARCH-SMODEL-BASIC-EM algorithm, the model used for local regression is the standard function $effort = a \cdot size^b$. Quality criteria are represented by the levels q_1 and q_2 that refer to figures of quality of the well-known adjustment measures MMRE and PRED[2] that will be discussed later.

To account for the common practice of cross-validation, the non-linear regression procedure used to find local parametric models returns the dataset randomly divided in two parts tr and ts, where tr is used for the regression algorithm, and ts is reserved for computing the quality measures. The condition for "segmentability" is that the size of the dataset is large enough (as determined by the threshold θ) for a clustering algorithm to operate reasonably. This provides a trade-off between over-segmenting and exploring the possibilities for an additional level of decomposition. In addition, the method to obtain segments used is the EM algorithm. This entails that the recursive structure resulting from the joining of the local models would have associated metadata with the probabilistic information describing the EM-generated clusters (normal distribution descriptions).

Our EM-based algorithm is a concrete scheme that uses common practice in software estimation. Nonetheless, many other parameterizations are possible, which are left to further studies. For example, the quality parameters q_1, q_2 may be more restrictive as the recursion proceeds, as a requirement on better adjustment for more homogeneous segments. Another interesting direction is providing heuristics, different clustering schemes or some degree of back-tracking in the exploration of the possibilities of division.

[2] PRED25 is used in the algorithm since it is a common value in software estimation studies, but the PRED level could also be considered as another parameter to the algorithm.

4 EMPIRICAL EVALUATION

The empirical evaluation has been carried out with two different kinds of data sources. The large ISBSG-8 database has been used as a case that clearly suffers from the problems of divergences of variances mentioned in the introduction to this paper [5]. Additionally, the smaller public data sets used in [8] have been used for comparison purposes with the results of other methods, even though these data sets are small-sized and thus less likely to require segmentation due to heterogeneity in the collection of the data.

The implementation of the concrete instantiation of the algorithm uses the open source WEKA libraries [20], and the non-linear general regression model implemented in the Java libraries of Dr. Michael Thomas Flanagan[3], which implement the Nelder and Mead procedure [14]. Such procedures use logarithmic transformations internally. For the regression, initial values of zero and steps of 0.1 have been used.

In both parts of the study, the models obtained from regression techniques were subject to cross-validation following standard practices. The data assigned to each cluster was randomly split into two sets called training (t) and validation (v), respectively, containing 70 % and a 30 % of the data. It must be emphasized that the quality measures that will be presented below are applied to test data not used in the regression process, to avoid the effects of overfitting. The selection of the t set is automatically done by the program at each step of Search-SModel.

The measures of prediction accuracy used were standard MMRE and PRED(.25) which are commonly accepted measures that reflect different aspects of the models [8]. Mean magnitude of relative error (MMRE) is defined as [4]:

$$MMRE = \frac{1}{n}\sum_{i=1}^{n}|\frac{e_i - \widehat{e}_i}{e_i}| \tag{3}$$

where e_i is the actual value of the variable and $\widehat{e}_i$ its corresponding estimate, and n is the number of observations. Thus if MMRE is small, then the predictions can be considered as good.

Prediction at Level p where p is a percentage, is defined as the quotient of number of cases in which the estimates are within the p absolute limit of the actual values, divided by the total number of cases. For example, PRED(0.2) = 70 means that 70 % of the cases have estimates within the 20 % range of their actual values.

4.1 ISBSG Evaluation Results and Discussion

The entire ISBSG-8 database[4] containing information about 2028 projects was used as the project database. The database contained information about size, effort and

[3] www.ee.ucl.ac.uk/ mflanaga

[4] http://www.isbsg.org/

many other project characteristics. The first cleaning step was that of removing the projects with null or invalid numerical values for the fields effort ("Summary Work Effort" in ISBSG-8) and size ("Function Points"). Then, the projects with "Recording method" for total effort other than "Staff hours" were removed. The rationale for this is that the other methods for recording were considered to be subject to subjectivity. For example, "productive time" is a rather difficult magnitude to assess in a organizational context.

Since size measurements were considered the main driver of project effort, the database was further cleaned for homogeneity in such aspect. Concretely, the projects that used other size estimating method ("Derived count approach") than IFPUG, NESMA, Albretch or Dreger were removed, since they represented smaller portions of the database. The differences between IFPUG and NESMA methods are considered to have a negligible impact on the results of function point counts [15]. Counts based on Albretch techniques were not removed since in fact IFPUG is a revision of these techniques; similarly, the Dreger method refers to the book [9], which is simply a guide to IFPUG counts.

For comparison purposes, an overall model was obtained from the entire ISBSG-8 database. The measures of adjustment for this model with and without cross-validation are showed in Table 1.

	MMRE	PRED(.3)	a	b
with c.v.	2.81	0.23	7.6	1.07
without c.v.	0.88	0.27	14.5	0.4615

Table 1. Characteristics of the model for the entire database (without clustering)

The results of one of the executions of the algorithm are provided in Table 2. Column labeled "#" is the cluster number, and column "#-inst" is the number of points or instances in each cluster (it should be noted that the points reserved for cross-validation are not counted in the table). For each cluster (row), the parameters of the exponential model are provided, along with the MMRE and PRED values per cluster. The overall parameters are the average of these figures.

Since the selection of "training" and "test" data is random, and the WEKA-EM implementation also was configured to determine the number of clusters through cross-validation, the results may vary in different executions. Nonetheless, a process of one thousand repetitions of the whole algorithm execution with the same data was carried out without finding relevant divergences from the overall data provided below. The number of clusters obtained was in the range 25 to 40 in all the repetitions, and the depth of the recursion in the algorithm did not exceeded the value of 15. As a matter of coherence with the assumptions embodied in the algorithm itself, clusters with size below 20 projects were discarded[5]. A simple analysis of proximity of that points to the distributions obtained by the EM algorithm for other remaining

[5] This is a minimum size that fits some extreme data ranges in the ISBSG in which there is a sparse distribution of points.

clusters evidenced that this did not affect the overall results. As can be appreciated in Table 2, the overall values of adjustment are much better than the single-model approach (Table 1).

#	#-inst	param.[a,b]	MMRE	PRED25(%)	level
1	110	[1905.97, -0.010511]	0.079	100	1
2	97	[13631.462707,-0.515552]	0.962	48.3	2
3	94	[518.949111,0.078785]	0.359287	53.6	3
4	47	[2001.248634,0.044032]	0.09348	100	4
5	187	[4076.565265,-0.040429]	0.177622	89.3	5
6	27	[158.312169,0.406403]	0.095601	100	1
7	58	[13147.90087,-0.261441]	0.18446	88.2	2
8	48	[169.34605,0.440456]	0.178966	78.6	3
9	44	[894.218296,-0.111508]	0.649	53.8	1
10	84	[506.140546,-0.05302]	0.153637	96	2
12	103	[1201.35927,0.006415]	0.095388	100	4
13	61	[271.048788,-0.129842]	0.904536	27.7	5
14	90	[786.252424,-0.006614]	0.163166	85.2	6
16	46	[742.018383,-0.036715]	0.12064	100	8
17	26	[520.565331,-0.169616]	0.15351	100	9
19	27	[45500.109931,-0.518142]	0.929531	12.5	2
20	35	[59050.864818,-0.468531]	0.22197	80	3
21	90	[4128.474511,0.043318]	0.079869	100	2
22	117	[197.907091,0.416854]	0.866377	40	3
23	78	[8743.367059,-0.023768]	0.099939	100	4
24	87	[19918.250148,-0.115967]	0.143495	92.3	5
26	62	[110.238818,0.498503]	1.51495	16.7	1
27	77	[14226.777306,0.013624]	0.103796	100	2
28	71	[26454.358784,0.003797]	0.179631	85.7	3
29	49	[6.068873E9,-1.9740551]	1.125719	35.7	4
30	71	[2775092.985303,-0.54002]	1.259482	47.6	5
			0.406	71.6	

Table 2. Results of the segmentation algorithm for the ISBSG-8 database

The clusters obtained are of a considerable size, comparable to those that were reported in the literature before large databases as the ISBSG were available [8], so that the solution appears to conform with established practice. Table 2 can also be used to identify the data points that are more difficult to model, which are the clusters that in the Table have worse figures of quality of measurement.

The process of post-assessment of the segmented model in Table 2 was carried out by implementing a simple algorithm that yielded effort predictions by using the model in the cluster that best matched the input point. The EM algorithm yields models based on normal distributions for which a value of "probability of membership in a cluster" can be computed. The evaluation process consisted of

random selection of one of the input values in the ISBSG database, which was contrasted with the predictions resulting from the segmented and non-segmented models. The results of the segmented versions were systematically better in about 10–20 % of relative error. This should be complemented in the future with the assessment that uses additional data, e.g. by using new data as is provided in further versions of the ISBSG database.

4.2 Public Datasets Evaluation Results and Discussion

Table 3 provides the results of the algorithm for the collection of public datasets used in a previous study [8]. The table mentions the dataset (references can be found in [8]) and the model reported in the literature for each of them ("curve estimation" column). The overall properties of the model obtained from the algorithm is provided in the column "Segmented model", and the properties per cluster, when applicable, are provided in the column "Clusters".

The repeated execution of the algorithm yielded in some cases non-segmented models of similar or even better predictive quality than those reported in the literature. These are marked in Table 3 with a "n.a." value in the column "clusters". In the other cases, the segmentation algorithm yielded a significantly better model. The sizes of each cluster are provided in the table as well.

Since the datasets in Table 3 were all of a size below 100 points, the minimum segmentable size was reduced to the value of 20. The results for some datasets were still not satisfactory from the viewpoint of quality of adjustment. For example, the Belady & Lehman dataset presents poor quality of adjustment even when searching for a model by repeated execution of the algorithm. These cases can be considered to have a complex combination of projects which are elusive to conventional regression and also to segmented models. Nonetheless, for other datasets, the decomposition in clusters helps in capturing groupings of data with different characteristics. Figure 2 provides an example for the Desharnais data model, which illustrates this kind of situations.

Figure 2 illustrates how different local models help in better modeling the CER. The clusters obtained lead to a function for c_1 with a clear difference from the functions of c_2 and c_3. The latter two clusters have models that bear some similarity, and thus could be merged. The assessment of the clusters and models obtained can be used to gain new insight about the data in addition to serve as parametric models with better overall predictive capabilities.

5 CONCLUSIONS

Segmented models of parametric software effort estimation allow for the explicit consideration of heterogeneity that is inherent to many historical project databases. An algorithmic scheme for the search of such models has been described, which is flexible enough to accommodate various views on the quality of local models, the cost

Data set	**Curve estimation**	Segmented model	Clusters
Abran & Robilard	$e = 1.89 \cdot fp^{1.07}$ PRED(.25)=75 MMRE=0.19	$e = 7.081644 \cdot fp^{0.805606}$ PRED(.25)=83.3 MMRE=0.14	n.a.
Albretch & Gaffney	$0.02 \cdot fp^{1.04}$ PRED(.25)=47.37 MMRE=0.27	$2.581985E{-}4 \cdot fp^{1.70153}$ PRED(.25)=71.43 MMRE=0.225	n.a.
Bailey & Basili	$1.94 \cdot kloc^{0.86}$ PRED(.25)=66.67 MMRE=0.26	PRED(.25)=83.3 MMRE=0.19	$\|c_1\|$=9 MMRE[c_1]=0.31877 PRED(.25)[c_1]=50 a[c_1]= 1035306.01277 b[c_1]=-1.87266 $\|c_2\|$=7 MMRE[c_2]=0.0282447 PRED(.25)[c_2]=100 a[c_2]= 1.154344E-4 b[c_2]=1.95316 $\|c_3\|$=7 MMRE[c_3]=0.225822 PRED(.25)[c_3]=100 a[c_3]= 0.025885 b[c_3]=0.983568
Belady & Lehman	$0.003 \cdot loc^{1.06}$ PRED(.25)=33.33 MMRE=0.64	PRED(0.25)=58.3 MMRE=0.46	$\|c_1\|$=8 MMRE[c_1]=0.24638 PRED(.25)[c_1]=100 a[c_1]= 63379.1613958 b[c_1]=-0.417028 $\|c_2\|$=25 MMRE[c_2]=0.686929 PRED(.25)[c_2]=16.6 a[c_2]= 7.6154748 b[c_2]=0.2524346
Boehm	$0.0018 \cdot adjkdsi^{1.108}$ PRED(.25)=17.46 MMRE=1.13	$7.4050E{-}4, \cdot adjkdsi^{1.20445}$ PRED(0.25)= 44 MMRE=0.49	n.a.
Desharnais	$97.09 \cdot fp^{0.68}$ PRED(.25)=54.29 MMRE=0.27	PRED(.25)=83.3 MMRE=0.192	$\|c_1\|$=9 MMRE[c_1]=0.13544 PRED(.25)[c_1]=100 a[c_1]= 2589.58308 b[c_1]=-0.002607 $\|c_2\|$=19 MMRE[c_2]=0.14953 PRED(.25)[c_2]=100 a[c_2]= 3894.3803274 b[c_2]=0.0283768 $\|c_3\|$=7 MMRE[c_3]=0.291042 PRED(.25)[c_3]=50 a[c_3]= 2.28532 b[c_3]=1.447182
Heiat & Heiat	$0.009 \cdot reio^{1.44}$ PRED(.25)=93.93 MMRE=0.106	$0.020789 \cdot reio^{1.29279}$ PRED(0.25)=100 MMRE=0.075	n.a.
Kemerer	$0.35 \cdot fp^{0.903}$ PRED(.25)=33.33 MMRE=0.44	$0.69388 \cdot fp^{0.78925}$ PRED(0.25)= 50 MMRE=0.333	n.a.
Kitchenham & Taylor	$0.0077 \cdot fp^{0.89}$ PRED(.25)=34.62 MMRE=0.54	$0.0087 \cdot fp^{0.8772}$ PRED(.25)=57.1 MMRE=0.3	n.a.
Miyazaki et al.	$1.307 \cdot kloc^{0.89}$ PRED(.25)=37.5 MMRE=0.39	$1.78759 \cdot kloc^{0.83628}$ PRED(.25)=77.7 MMRE=0.2424	n.a.
Shepperd & Schofield	$13.49 \cdot files^{0.65}$ PRED(.25)=50 MMRE=0.44	$8.46813 \cdot files^{0.756352}$ PRED(.25)=60 MMRE=0.29	n.a.

Table 3. Error and predictive values for each data set, compared to curve estimation

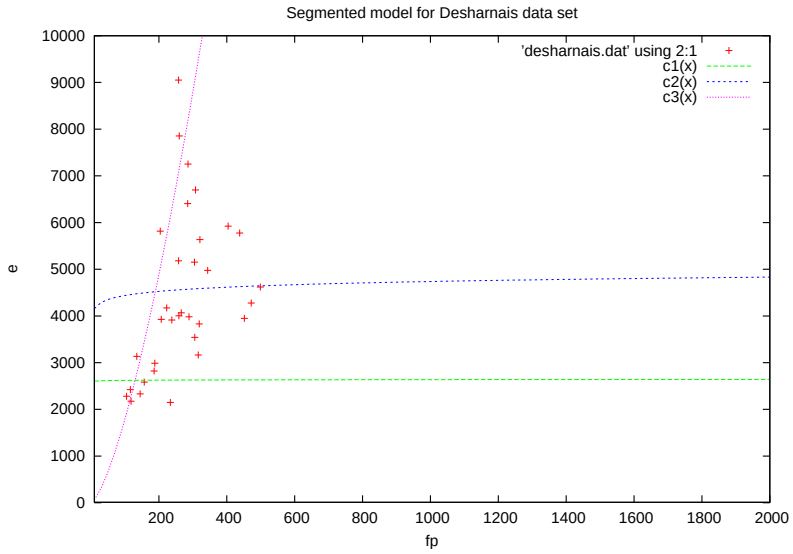

Fig. 2. Segmented model for the Desharnais data set

drivers considered and the associated mathematical models. A concrete instantiation of the algorithm has been used for the purpose of evaluating the potential of the segmentation to derive effort estimation models with reasonable local and overall quality of adjustment metrics.

Future work should deal with metrics of overlapping between segments as a way to discover possibilities for "merging" segments or to re-shape the underlying clusters to adjust to empirical evidence. This in combination with heuristics and a degree of backtracking could provide a more generalized model for the cases in which empirical evidence suggests that additional adjusting at the local segment level. Another aspect of the problem that has not been dealt in this paper is that of the selection of cost drivers other than size, which could be subject to a prior selection and analysis whenever project databases with enough information are available.

Acknowledgements

This work has been supported by the project MCYT TIN2004-06689-C03 (IN2GESOFT). The authors of the paper are listed in alphabetical order with no particular precedence.

REFERENCES

[1] Boehm, B.—Abts, C.—Chulani, S.: Software Development Cost Estimation Approaches – a Survey. USC Center for Software Engineering Technical Report # USC-CSE-2000-505, 2000.

[2] Boehm, B.—Abts, C.—Winsor Brown, A.—Chulani, S.—Clark, B.—Horowitz, E.—Madachy, R.—Reifer, D.—Steece, B.: Software Cost Estimation with Cocomo II. Prentice Hall, 2000.

[3] Boetticher, G.: When Will it Be Done? The 300 Billion Dollar Question, Machine Learner Answers. IEEE Intelligent Systems, May/June 2003, pp. 2–4.

[4] Conte, S. D.—Dunsmore, H. E.—Shen, V. Y.: Software Engineering Metrics and Models. Benjamin/Cummings, Menlo Park, CA, 1986.

[5] Cuadrado-Gallego, J. J.—Sicilia, M. A.—Rodríguez, D.—Garre M.: An Empirical Study of Process-Related Attributes in Segmented Software Cost-Estimation Relationships. Journal of Systems and Software, Vol. 3, 2006, No. 79, pp. 351–361.

[6] Dempster, A. P.—Laird, N. M.—Rubin, D. B.: Maximum-Likelihood from Incomplete Data Via the em Algorithm. J. Royal Statist. Soc. Ser. B., Vol. 39, 1977.

[7] Dick, S.—Meeks, A.—Last, M.—Bunke, H.—Kandel, A.: Data Mining in Software Metrics Databases. Fuzzy Sets and Systems, Vol. 145, 2004, No. 1, pp. 81–110.

[8] Dolado, J.: On the problem of the software cost function. Information and Software Technology, Vol. 43, 2001, No. 1, pp. 61–72.

[9] Dreger, J. B.: Function Point Analysis. Englewood Cliffs, NJ: Prentice Hall, 1989.

[10] Garmus, D.—Herron, D.: Function Point Analysis: Measurement Practices for Successful Software Projects, Addison-Wesley, 2000.

[11] Garre, M.—Cuadrado, J. J.—Sicilia, M. A.: Recursive Segmentation of Software Projects or the Estimation of Development Effort. Proceedings of the V ADIS 2004 Workshop on Decision Support in Software Engineering, CEUR Workshop Proceedings, Vol. 120, available at `http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-120/`.

[12] Gray, A.—MacDonell, S. G.: Applications of Fuzzy Logic To Software Metric Models for Development Effort Estimation. Proceedings of the 1997 Annual Meeting of the North American Fuzzy Information Processing Society NAFIPS, IEEE, Syracuse NY, 1997, pp. 394–399.

[13] Lung, C. H.—Zaman, M.—Nandi, A.: Applications of Clustering Techniques to Software Partitioning, Recovery and Restructuring. Journal of Systems and Software, Vol. 73, 2004, No. 2, pp. 227–244.

[14] Nelder, J. A.—Mead, R.: Computer Journal. Vol. 7, 1965, pp. 308–313.

[15] NESMA: NESMA FPA Counting Practices Manual CPM 2.0, 1996.

[16] Oligny, S.—Bourque, P.—Abran, A.—Fournier, B.: Exploring the Relation Between Effort and Duration in Software Engineering Projects in World Computer Congress. Beijing, China, August 21–25, 2000, pp. 175–178.

[17] Parametric Estimating Initiative: Parametric estimating handbook. 2nd edition, 1999.

[18] Stensrud, E.—Foss, T.—Kitchenham, B.—Myrtveit, I.: An Empirical Validation of the Relationship Between the Magnitude of Relative Error and Project Size. In Proceedings of the Eighth IEEE Symposium on Software Metrics, 2002.

[19] Pedrycz, W., Succi, G.: Genetic granular classifiers in modeling software quality. The Journal of Systems and Software, Vol. 76, 2002, pp. 277–285.

[20] Witten, I. H.—Frank, E.: Data Mining, Practical Machine Learning Tools and Techniques with Java Implementations. Morgan Kaufmann Publishers, San Francisco, California, 1999.

[21] Xu, Z.—Khoshgoftaar, T.: Identification of Fuzzy Models of Software Cost Estimation. Fuzzy Sets and Systems, Vol. 145, 2004, No. 1, pp. 141–163.

Juan J. Cuadrado-Gallego is currently a Profesor Titular de Universidad at the Computer Science Department of the University of Alcalá, Madrid, Spain, and the Director of the doctorate studies in computer sciences at this University. He also is a consultant at the University Oberta of Catalunya, Barcelona, Spain. His expertise area is software engineering and especially software measurement. He has been teaching in this subject in all the universities where he has been as permanent staff and at the University Roma Tre, Roma, Italy, as teaching staff.

Miguel-Angel Sicilia obtained his university degree in computer science from the Pontifical University of Salamanca in Madrid, Spain (1996) and his Ph. D. from Carlos III University in Madrid, Spain (2002). In 1997 he joined an object-technology consulting firm, after enjoying a research grant at the Instituto de Automatica Industrial (Spanish Research Council). From 1997 to 1999 he worked as Assistant Professor at the Pontifical University. Since 2000 to October 2003, he worked as a full-time lecturer at Carlos III University working actively in the area of adaptive hypermedia and e-learning systems. Currently, he works as an Associate Professor at the Computer Science Department, University of Alcalá (Madrid). His research interests focus primarily on semantic metadata, software engineering and learning technology.