A Bayesian Software Estimating Model Using a Generalized g-Prior Approach

Sunita Chulani
Research Assistant
Center for Software Engineering
University of Southern California
Los Angeles, CA 90089-0781, U.S.A.
+1-213-740-6470
sdevnani@sunset.usc.edu

Bert Steece
Deputy Dean of Faculty
Marshall School of Business
University of Southern California
Los Angeles, CA 90089-0781
+1-213-740-4823
berts@almaak.usc.edu

ABSTRACT

Created to provide a software cost estimation model suited for a rapidly evolving environment, the COCOMO II model is the result of a 1994 research effort to update the 1981 COnstructive COst MOdel and its 1987 Ada version. Boehm et al [3, 15] provided the initial definition and rationale for this model. The model's inputs include Source Lines of Code and/or Function Points as the sizing parameter, adjusted for both reuse and breakage; a set of 17 multiplicative effort multipliers and a set of 5 exponential scale factors [see appendix A]. They based their initial calibration of the model on expert judgement.

Soon after the initial publication of this model, the Center for Software Engineering (CSE) began an effort to empirically validate COCOMO II [14]. By January 1997, they had a dataset consisting of 83 completed projects collected from several Commercial, Aerospace, Government and FFRDC organizations. CSE used this dataset to calibrate the COCOMO II.1997 model parameters. Because of uncertainties in the data and / or respondents' misinterpretations of the rating scales, CSE developed a pragmatic calibration procedure for combining sample estimates with expert judgement. Specifically, the above model calibration for the COCOMO II.1997 parameters assigned a 10% weight to the regression estimates while expert-judgement estimates received a weight of 90%. This calibration procedure yielded effort predictions within 30% of the actuals 52% of the time.

CSE continued the data collection effort and the database grew from 83 datapoints in 1997 to 161 datapoints in 1998. Using this data and a Bayesian approach that can assign differential weights to the parameters based on the precision of the data, we provide an alternative calibration of COCOMO II. Intuitively, we prefer this approach to the uniform 10% weighted average approach described above because some of the effort multipliers and scale factors are more clearly understood than others. The sample information for well-defined cost drivers receives a higher weight than that given to the less precise cost drivers. This calibration procedure yielded significantly better predictions; that is our version of COCOMO II gives effort predictions within 30% of the actuals 76% of the time. The reader should note that these predictions are based on out-of-sample data (projects) as described in the 'Cross Validation' section (i.e. section 5).

This paper presents a generalized g-prior approach to calibrating the COCOMO II model. The paper shows that if the weights assigned to sample estimates versus expert judgement are allowed to vary according to precision, a superior predictive model will result.

Section 1 of this paper describes the calibration approach used on the 1997 dataset followed by four sections on approaches used on the 1998 dataset. Section 2 discusses the ordinary least squares approach on the 1998 dataset. Section 3 gives an overview of the Bayesian framework followed by section 4 where an overview of the generalized g-prior is provided. Then, section 5 discusses the application of the g-prior approach on the 1998 dataset of COCOMO II. Section 6 summarizes the results obtained by using the Bayesian approaches discussed in the earlier sections. And, we conclude that the Bayesian framework is well suited for calibrating software cost models and the generalized g-prior approach can be used to develop models with very good prediction accuracies.

Although this paper gives a synopsis of the COCOMO II model structure, the reader is urged to read [3] to attain a better understanding of COCOMO II and to ascertain the differences from its predecessors.

Keywords

Software cost estimation, project management, COCOMO, measurement, metrics, empirical modeling, Bayesian analysis, g prior, multiple regression, prediction accuracy.

1 COCOMO II.1997 CALIBRATION APPROACH

The initial definition of the COCOMO II model and its rationale based on expert-judgement results are provided in [3]. An effort to empirically validate this model began soon after the publication of [3]. By January 1997, CSE had a dataset of 83 completed projects collected from several Commercial, Aerospace, Government and FFRDC organizations.

The initial COCOMO II Post Architecture model had the following form:

$$Effort = A \times \left[Size \right]^{1.01+\sum_{i=1}^{5} SF_i} \times \prod_{i=1}^{17} EM_i \tag{1}$$

where, A = Multiplicative Constant
Size = Size of the software project measured in terms of KSLOC (thousands of Source Lines of Code [12], Function Points [9] or Object Points [1])
SF = Scale Factor
EM = Effort Multiplier [refer to 2, 3 for further explanation of COCOMO ’81 and COCOMO II respectively]

Equation 1 can be linearized by taking logarithms as follows:

$$\begin{aligned} ln(PM) = {} & \beta_0 + \beta_1 \cdot 1.01 \cdot \cdot ln(Size) + \beta_2 \cdot SF_1 \cdot ln(Size) \\ & + \cdots + \beta_6 \cdot SF_5 \cdot ln(Size) + \beta_7 \cdot ln(EM_1) + \\ & \beta_8 \cdot ln(EM_2) + \cdots + \beta_{23} \cdot ln(EM_{17}) \end{aligned} \tag{2}$$

Thus, equation 2 has the form of a multiple regression model:

$$y_t = \beta_0 + \beta_1 x_{t1} + ... + \beta_k x_{tk} + \varepsilon_t \tag{3}$$

where y_t (i.e. log Person Months), the response variable for the t_{th} observation, is a linear function of k predictors (i.e. log Source Lines of Code, etc.) where $x_{t1} \ldots x_{tk}$ are the values of the logged predictor (or regressor) variables for the t_{th} observation, $\beta_0 \ldots \beta_\kappa$ are the coefficients to be estimated, ε_t is the usual error term. Thus the parameters of equation 3 can be estimated by the ordinary least squares approach as discussed in numerous books such as [7, 16].

Using the 83 projects discussed above, Clark et al, [6] estimated coefficients associated with the 5 scale factors and the 17 effort multipliers. Because of the quality of the data, the regression produced some coefficient estimates that were counter-intuitive.

Clark et al developed a pragmatic procedure for addressing these counter-intuitive results by using a weighted average of the expert-judgement results (i.e. the initial model published in 1994) and the regression results (i.e. the data-determined results); with a 10% weight assigned to the regression estimate and a 90% weight assigned to the expert judgement estimate. Clark et al selected the 10% weighting factors because models with 40% and 25% weighting factors produced less accurate predictions. This pragmatic calibrating procedure moved the model parameters in the direction suggested by the sample data but retained the rationale contained within the expert-judgement values.

Using Proportional Error (PE) as their accuracy metric, Clark et al compared the actual efforts with those predicted by COCOMO II.1997 for their 83 projects. Table 1 presents the prediction accuracies they obtained.

Table 1: Effort Prediction Accuracies Using COCOMO II.1997

Prediction Accuracy	COCOMO II.1997
PRED(.20)	46%
PRED(.25)	49%
PRED(.30)	52%

They found that 52% of the predictions are within 30% of the actual software development efforts.

Given the quality and quantity of software development as well as the rapidly evolving nature of the development process, we believe the key issues for any cost estimation model for software development (and to improve the prediction accuracies of COCOMO II) are (a) how to properly assess expert judgement? (b) how to gather more sample information? and (c) how to appropriately combine expert opinion and sample data?

2 THE LEAST SQUARES CALIBRATION APPROACH ON THE 1998 DATABASE OF COCOMO II

The COCOMO II database grew from 83 datapoints in 1997 to 161 datapoints in 1998. As a result, the data-determined regression results for the 1998 database using RCode [7] are as shown:

```
Data set = COCOMOII.1998
Response      = log[PM]
Coefficient Estimates
Label          Estimate  Std.Err  t-value
Constant_A     0.962     0.103    9.304
log[SIZE]      0.922     0.046    20.015
PMAT*log[SIZE] 0.685     0.481    1.424
PREC*log[SIZE] 1.102     0.374    2.947
TEAM*log[SIZE] 0.323     0.497    0.650
FLEX*log[SIZE] 0.355     0.687    0.516
RESL*log[SIZE] 1.329     0.638    2.084
log[PCAP]      1.203     0.308    3.907
log[RELY]      0.641     0.246    2.602
log[CPLX]      1.035     0.233    4.448
log[TIME]      1.581     0.386    4.100
```

```
log[STOR]      0.784     0.352    2.225
log[ACAP]      0.926     0.272    3.400
log[PEXP]      0.755     0.357    2.119
log[LTEX]      0.172     0.416    0.412
log[DATA]      0.783     0.218    3.587
log[RUSE]     -0.34      0.286   -1.188
log[DOCU]      2.06      0.622    3.307
log[PVOL]      0.867     0.227    3.815
log[AEXP]      0.138     0.330    0.417
log[PCON]      0.488     0.322    1.517
log[TOOL]      0.551     0.222    2.488
log[SITE]      0.675     0.498    1.354
log[SCED]      1.119     0.275    4.063
```

These results provided the estimates for the β coefficients associated with each of the predictor variables as explained in equation 2). The t-value (ratio between the estimate and corresponding standard error) is the signal-to-noise ratio for each of the corresponding predictor variables. Hence, the higher the t-value, the stronger the signal (i.e. statistical significance) being sent by the predictor variable.

The regression results produced intuitively reasonable parameter estimates for all but five of the predictor variables. The negative coefficient estimate for RUSE (Develop for Reuse) is clearly counter intuitive [see results published in 13]. Furthermore, the magnitudes for the coefficients on some of the parameters, namely, AEXP (Applications Experience), LTEX (Language and Tool Experience), FLEX (Development Flexibility), and TEAM (Team Cohesion), conflict with our prior knowledge of the software development process. We believe that issues related to data quality (such as, lack of dispersion of values within some cost drivers and partial correlations among the cost drivers [11]) are major contributors to these counter intuitive results. For example, for the data collected on RUSE, a number of respondents answered “nominal” when they actually should have responded, “I don’t know”.

In the remainder of this paper, we will present a methodology for alleviating these problems.

3 THE BAYESIAN METHODOLOGY

While the 10% weighted-average procedure for combining expert judgement and sample information provided a workable initial model, we seek a formal framework which explicitly recognizes that information about individual effort multipliers and scale factors varies considerably. A Bayesian analysis with an informative prior provides such a framework.

Bayesian analysis [4,5], a mode of inductive reasoning used in many scientific disciplines, permits the investigator to combine sample (data) and prior (expert-judgement) information in a logically consistent manner. Using Bayes’ theorem, prior (or initial) values are transformed to post-data views of the model parameters. This transformation can be viewed as a learning process. The posterior distribution is determined by the variances of the prior and sample information. If the variance of the prior information is smaller than the variance of the sampling information, then a higher weight is assigned to the prior information. On the other hand, if the variance of the sample information is smaller than the variance of the prior information, then a higher weight is assigned to the sample information causing the posterior estimate to be closer to the sample information.

Using Bayes’ theorem, we can combine our two information sources as follows:

$$f(\beta|Y) = \frac{f(Y|\beta)\, f(\beta)}{f(Y)} \tag{4}$$

where ß is the vector of parameters in which we are interested and Y is the vector of sample observations from the joint density function $f(\beta|Y)$. $f(\beta|Y)$ is the posterior density function for ß summarizing all the information about ß, $f(Y|\beta)$ is the sample information and is algebraically equivalent to the likelihood function for ß, and $f(\beta)$ is the prior information summarizing the expert-judgement information about ß. Equation 4 can be rewritten as:

$$f(\beta|Y) \propto l(\beta|Y)\, f(\beta) \tag{5}$$

In words, equation 5 means

$$Posterior \propto Sample \times Prior$$

The Bayesian approach provides an optimal combination of the two independent sources of information. The posterior mean, b**, and variance, Var(b**), are defined as

$$b^{**} = [\frac{1}{\sigma^2} X'X + H^*]^{-1} \times \left[\frac{1}{\sigma^2} X'Xb + H^* b^*\right]$$

$$and\ Var(b^{**}) = \left[\frac{1}{\sigma^2} X'X + H^*\right]^{-1} \tag{6}$$

where X is the matrix of predictor variables, σ^2 is the variance of the data generating process; and H* and b* are the precision (inverse of variance) and mean of the prior information respectively.

Equation 6 shows that as the precision of the a-priori information (H*) increases (or the variance of the a-priori information decreases) relative to the precision (or the variance) of the sampling information $(1/\sigma^2\, X'X)$, the posterior parameter values move closer to the a-priori values.

4 THE GENERALIZED G-PRIOR METHODOLOGY

While there are numerous procedures for assessing prior knowledge about the model parameters (i.e. b* and H*), we chose Zellner's [17] g-prior approach because it does not require the difficult task of specifying the prior covariances for the elements of ß. The g-prior approach assumes that the prior covariances for ß are equal to those provided by the sample data. In other words, the prior precision matrix in equation 6 is given by

$$H^* = \frac{g}{\sigma^2} X' X \qquad (7)$$

Thus, the mean of the posterior distribution for ß is

$$b^{**} = \frac{b + gb^*}{1 + g} \qquad (8)$$

where b is the vector of the ordinary least squares values and b* is the vector of the values anticipated by the experts. The magnitude of g corresponds to the precision (i.e. relative weight) given to the prior. For example, when g = 0.1, the prior values b* receive a weight of one-ninth while the sample values receive a weight of eight-ninth. This version of the g-prior assumes that the same relative weight is assigned to each parameter.

Because the information about the individual effort multipliers and scale factors varies, we now extend Zellner's g-prior approach to assign differential weights to the parameters. We know that respondents to our data collection do not have a uniform understanding of each of the effort multipliers and scale factors. Hence it is only logical to assign different weights to some of the parameters. For example, for the data collected on RUSE, a number of respondents answered "nominal" when they actually should have responded, "I don't know". Other factors where we encountered some confusion in responses included AEXP, LTEX, FLEX, and TEAM. As a result, the prior estimates of these five factors are given greater weights than the weights assigned to the prior knowledge of the other factors where more precise data is available.

We can show that the mean of the posterior distribution of ß is given by:

$$b^{**} = [\, gZX' XZ + X' X \,]^{-1} \times \left[gZX' XZb + X' Xb^* \right] \qquad (9)$$

where Z is a diagonal matrix with elements $z_1 \ldots z_k$, $z_i \geq 0$. The z_i's are the differential weights for the effort multipliers and scale factors. When Z is an identity matrix (i.e. $z_i = 1 \; \forall i$) we have the usual g-prior case.

5 THE GENERALIZED G-PRIOR CALIBRATION APPROACH ON THE 1998 DATABASE OF COCOMO II

We used a two-round Delphi [8] exercise to determine the prior information for the coefficient vector (i.e. b*). Eight experts from the field of software estimation independently provided estimates of the numeric values associated with each COCOMO II predictor variable. After completion of the first round, the experts received the summarized results. They then had a second opportunity to independently revise their response based on the responses of the rest of the participants in round 1. These second round results provided the estimates for the vector b*.

Table 2: Ordinary Least Squares and Generalized G-prior Estimates

Parameter	Ordinary Least Squares	Generalized G-prior
Constant_A	0.962	0.994
log[SIZE]	0.922	0.918
PMAT*log[SIZE]	0.685	0.729
PREC*log[SIZE]	1.102	1.020
TEAM*log[SIZE]	0.323	0.498
FLEX*log[SIZE]	0.355	0.600
RESL*log[SIZE]	1.329	1.483
log[PCAP]	1.203	1.208
log[RELY]	0.641	0.682
log[CPLX]	1.035	1.034
log[TIME]	1.581	1.494
log[STOR]	0.784	0.807
log[ACAP]	0.926	0.903
log[PEXP]	0.755	0.732
log[LTEX]	0.172	0.492
log[DATA]	0.783	0.866
log[RUSE]	-0.34	0.145
log[DOCU]	2.06	2.022
log[PVOL]	0.867	0.890
log[AEXP]	0.138	0.450
log[PCON]	0.488	0.589
log[TOOL]	0.551	0.595
log[SITE]	0.675	0.878
log[SCED]	1.119	1.092

We initially assigned $z_i = \sqrt{5}$ for AEXP (Applications

Experience), LTEX (Language and Tool Experience), FLEX (Development Flexibility), and TEAM (Team Cohesion) and for RUSE (Develop for Reuse).[1] This caused the prior estimate of these parameters to receive a weight of one-third while the sample estimate received a weight of two-thirds. And, we assigned $z_i = 1$ for the rest of the parameters causing the prior estimate of these parameters to receive a weight of one-ninth while the sample estimate received a weight of eight-ninths. We tried several different combinations of g and the Z matrix. Using prediction accuracies as our model selection criteria, we found the best values to be: for the g-prior approach: $g = 0.1$, and for the generalized g-prior approach: $g = 0.1$, $z_i = \sqrt{5}$ for AEXP, LTEX , FLEX, TEAM, RUSE and $z_i = 1$ for the other cost drivers.

Our generalized g-prior and ordinary least-squares estimates are reported in table 2**.** The ordinary least-squares estimates for the five factors highlighted in the above discussion ranged from a negative value for RUSE (counter-intuitive) to values with insufficient magnitude for the other four factors. On the other hand, the generalized g-prior approach yields estimates that support prior expert opinion.

Figure 1 illustrates how our approach handles noisy data. Here our prior degree-of-belief receives a weight of one-third while the sample data receives a weight of two-thirds. As a consequence, the posterior mean moves closer to the prior mean than would have otherwise been the case. Remember that $z_i = 1$ for most of the effort multipliers and scale factors whereas $z_i = \sqrt{5}$ for AEXP and the other four parameters identified above.

The Productivity Ranges (ratio between the least productive parameter rating, i.e. the highest rating, and the most productive parameter rating, i.e. the lowest rating) that are a result of the above described generalized g-prior approach are illustrated in figure 2. Figure 2 gives an overall perspective of the relative Software Productivity Ranges (PRs) provided by the COCOMO II parameters.

Figure 1: A-Posteriori Generalized g-Prior Update in the Presence of Noisy Data (AEXP, Applications Experience)

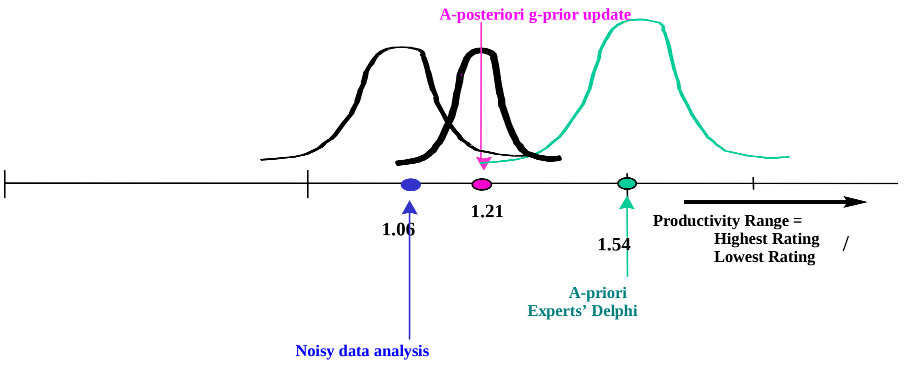
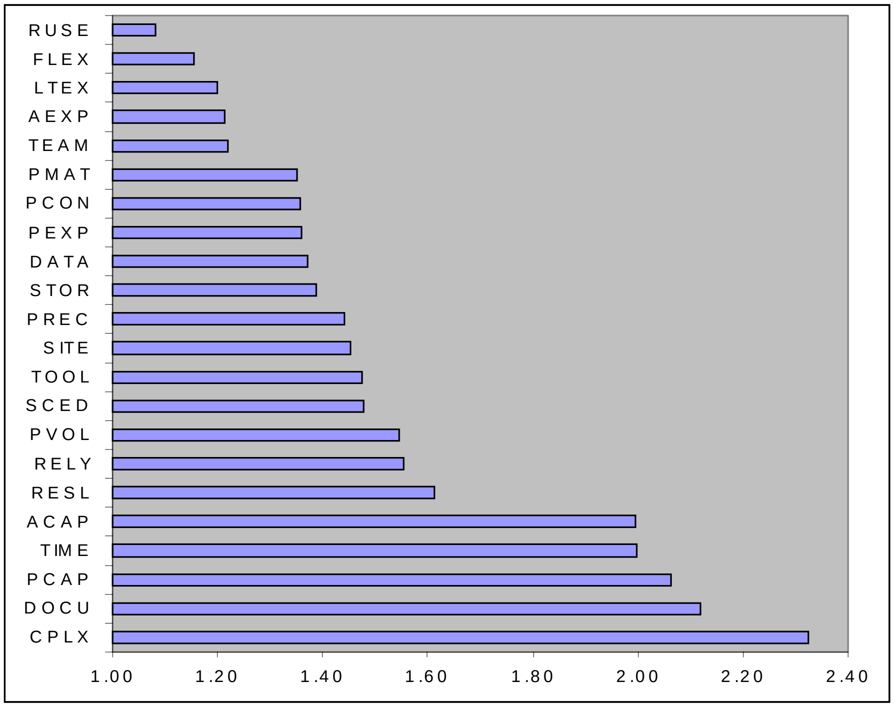


Figure 2: A-Posteriori Generalized G-Prior Productivity Ranges



The PRs provide insight on identifying the high payoff areas to focus on in a software productivity improvement activity. For example, CPLX (Product Complexity) is the highest payoff parameter in the g-prior calibration of the COCOMO II model. It should be noted that the results on the RUSE parameter are not consistent with all the published results [13]. The only way to empirically resolve

---

[1] The choice of these five parameters was due to the counter-intuitive data-determined results discussed in the last paragraph of section 2

for this problem is to collect better data. CSE has updated their data collection form [14] to ask more questions to make sure that the response “I don’t know” doesn’t get recorded as “Nominal”. Once, more data is available, further analysis can be done and the hope is that results consistent with published articles are obtained.

6 CROSS VALIDATION

It is not unusual in the analysis of software cost estimation data to have a model, which provides a good fit to the data used to develop the model. Unfortunately, such models sometimes give poor predictions of new data. Thus, we need to validate the prediction accuracy of our approach. While the collection of new data is clearly preferred, it is of course not practical in all situations nor was it feasible in our situation. Therefore, we ran a series of 15 separate cross-validation experiments where we randomly withheld 41 of the 161 observations. We used the data sets of size 120 each to obtain 15 prediction equations. We then used these equations to predict how well our models fit the withheld data.

Table 3: Prediction Accuracy of the Cross Validation

PRED(.30)			
	Average	**Minimum**	**Maximum**
Ordinary Least Squares	64%	54%	76%
g-Prior (g = 0.10)	70%	57%	79%
Generalized g-Prior (g = 0.10 and $z_i = 1$ for all but 5 factors where $z_i = \sqrt{5}$)	76%	62%	84%

The results are reported in table 3. We tried several different combinations of g and the Z matrix. Using prediction accuracies as our model selection criteria, we used the following values:
for the g-prior approach: g = 0.1, and
for the generalized g-prior approach: g = 0.1, $z_i = \sqrt{5}$ for AEXP, LTEX , FLEX, TEAM, RUSE and $z_i = 1$ for the other cost drivers.

The prediction accuracies within 30% of the actual (using the Proportional Error metric) for the 15 datasets are shown in table 3. Clearly, the generalized g-prior approach yields a model with the highest prediction accuracy. The average predicted estimates for the 15 datasets, using the models based on the generalized g-prior approach discussed in section 5, are within 30% of the actuals 76% of the time. The best model gives predictions within 30% of the actuals 84% of the time and the worst model gives predictions within 30% of the actuals 62% of the time. Note that these results are far superior than the results depicted in table 1 using COCOMO II.1997.

7 CONCLUSIONS

In this paper, we demonstrate the inappropriateness of the ordinary least squares approach to empirically validate software cost estimation models, specifically COCOMO II. We also ascertained a clear need for improving the data collection process and/or the model calibration process. We then moved on to develop a Bayesian framework for the software cost modeling domain. This framework has been successfully used in many other domains and we used to alleviate the problems we faced with the ordinary least squares approach. To avoid the difficult task of specifying prior covariances (the experts who participated in our 2-round Delphi lacked the understanding of the meaning of covariance), we chose two approaches (the g-prior and the generalized g-prior) within the Bayesian framework to apply on the 1998 database of COCOMO II. In addition to giving better prediction accuracies than those determined by the ordinary least squares approach, the g-prior approach resolves a few of the data-determined counter-intuitive results. We generalized this approach further by assigning differential weights to different cost drivers. The resulting generalized g-prior approach gave the best prediction accuracies on the 1998 database of COCOMO II. Although, it did result in some controversial estimates such as the Productivity Range for RUSE being less than 1.2. Unfortunately, given the desire to have a highly data-determined model we believe this cannot be resolved using the current dataset. Based on these results, we conclude that the generalized g-prior is a superior calibration methodology and should be further investigated when more data is available to calibrate a strong widely accepted version of COCOMO II.

ACKNOWLEDGEMENTS

This work is being supported both financially and technically under the AFRL Contract F30602-96-C-0274, "KBSA Life Cycle Evaluation" and by the COCOMO II Program Affiliates: Aerospace, Air Force Cost Analysis Agency, Allied Signal, AT&T, Bellcore, EDS, Raytheon E-Systems, GDE Systems, Hughes, IDA, JPL, Litton, Lockheed Martin, Loral, MCC, MDAC, Motorola, Northrop Grumman, Rational, Rockwell, SAIC, SEI, SPC, Sun, TI, TRW, USAF Rome Lab, US Army Research Labs, Xerox.

REFERENCES

1. Banker R., Kauffman R., Kumar R., “An Empirical Test of Object-based Output Measurement Metrics in a Computer Aided Software Engineering (CASE) Environment”, Journal of Management Information

Systems, Vo1. 8, No. 3, 1992.

2. Boehm B., Software Engineering Economics, Barry W. Boehm, Prentice-Hall, 1981.
3. Boehm B., Clark B., Horowitz E., Madachy R., Selby R., Westland C., "Cost Models for Future Software Live Cycle Processes: COCOMO 2.0", Annals of Software Engineering Special Volume on Software Process and Product Measurement, J.D. Arthur and S.M. Henry (Eds.) , J.C. Baltzer AG, Science Publishers, Amsterdam, The Netherlands, Vol 1, 1995.
4. Box G. and Tiao G, “Bayesian Inference in Statistical Analysis”, Addison Wesley, 1973.
5. Chulani S., “Incorporating Bayesian Analysis to Improve the Accuracy of COCOMO II and Its Quality Model Extension”, Qualifying Exam Report, Computer Science Department, USC Center for Software Engineering, February 1998.
6. Clark B., Chulani S., Boehm B., “Calibration Results of COCOMOII.1997”, International Conference on Software Engineering, April 1998.
7. Cook D. and Weisberg S., “An Introduction to Regression Graphics”, Wiley Series, 1994.
8. Helmer O.,” Social Technology”, Basic Books, New York, 1966.
9. International Function Point Users Group (IFPUG), Function Point Counting practices Manual, Release 4.0, 1994.
10. Judge G, Griffiths W., Hill C., “Learning and Practicing Econometrics”, Wiley, 1993.
11. Mullet M, “Why regression coefficients have the wrong sign”, Journal of Quality Technology, 1976.
12. Park R. et al, “Software Size Measurement: A Framework for Counting Source Statements”, CMU-SEI-92-TR-20, Software Engineering Institute, Pittsburg, PA, 1992.
13. Poulin J., “Measuring Software Reuse, Principles, Practices and Economic Models”, Addison Wesley, 1997.
14. Center for Software Engineering , “COCOMO II Cost Estimation Questionnaire,” Computer Science Department, USC Center for Software Engineering, http://sunset.usc.edu/, 1997
15. Center for Software Engineering , “COCOMO II Model Definition Manual,” Computer Science Department, USC Center for Software Engineering, http://sunset.usc.edu, 1997.
16. Weisberg S., “Applied Linear Regression”, 2nd Ed., John Wiley and Sons, New York, N.Y., 1985.
17. Zellner A., “Applications of Bayesian Analysis and Econometrics”, The Statistician, Vol. 132, p. 23-34, 1983.

APPENDIX A

This appendix has the acronyms and full forms of the 22 COCOMO II Post Architecture cost drivers. For a further explanation of these parameters, please refer to [3, 15]

PREC	Precendentedness
FLEX	Development Flexibility
RESL	Architecture and Risk Resolution
TEAM	Team cohesion
PMAT	Process Maturity
RELY	Required Software Reliability
DATA	Data Base Size
CPLX	Product Complexity
RUSE	Develop for Reuse
DOCU	Documentation Match to Life-cycle Needs
TIME	Time Constraint
STOR	Storage Constraint
PVOL	Platform Volatility
ACAP	Analyst Capability
PCAP	Programmer Capability
AEXP	Applications Experience
PEXP	Platform Experience
LTEX	Language and Tool Experience
PCON	Personnel Continuity
TOOL	Use of Software Tools
SITE	Multi-Site Development
SCED	Required Development Schedule