Estimation of f-COCOMO Model Parameters Using Optimization Techniques

Leonard J. Jowers[a], James J. Buckley[b], and Kevin D. Reilly[a]

October 30, 2006

University of Alabama at Birmingham
[a]Department of Computer and Information Sciences, JowersL,Reilly@cis.uab.edu,
[b]Department of Mathematics, buckley@math.uab.edu,
Birmingham, Alabama, 35294, USA

Abstract

The COCOMO Model is well known as the currently predominate model for software cost estimation. It allows one to work from linguistic variables to estimate software project effort and schedule. This basis in linguistic variables encourages research of the COCOMO Model as a fuzzy system. As is known in fuzzy circles and is shown here, fuzzy arithmetic based on the popular fuzzy extension principle may produce unacceptable results under fuzzy multiplication. This makes fuzzy results of some computations too fuzzy to be useful. Nevertheless, in the case of software cost estimation using COCOMO, we find and show that this characteristic of fuzzy arithmetic may be used to advantage.

If a project parameter is fuzzy, the associated COCOMO Model becomes a fuzzy COCOMO Model (f-COCOMO Model) with a fuzzy result (schedule and effort). Most software projects have deadlines dictated by management or market. However, even deadlines are fuzzy objects. For example, if a project must be complete by March 15, there is some possibility that it will not complete until the end of March. There is some possibility that it will complete in February. With a fuzzy project schedule, or effort (budget), some one or more parameters necessarily must be fuzzy. Of course this is all sensible, since standard COCOMO parameters begin with linguistic variables.

An example which demonstrates how fuzzy parameters affect f-COCOMO results is presented. When a project is planned, software management uses prior history and software engineers' opinions to estimate some parameters of the proposed project. Current resources and/or policy may determine other parameters. Other forces, such as time-to-market pressure or corporate goals, determine an estimated (fuzzy) schedule and budget even before conceptualization is complete. It is not likely that the fuzziness of the project parameters (linguistic variables) will produce a COCOMO result that is contained within the forces-estimated schedule or budget. This paper demonstrates how one may use a dictated fuzzy schedule and budget to improve an f-COCOMO Model, or plan a project. By application of constraints created by dictated fuzzy results, and back propagation, better estimates of project parameters are obtainable. Such a project scenario is presented and the method is applied to demonstrate its use.

Given such a project for which the method is applied, one may ask whether some augmentation of one or more parameters might optimize the COCOMO result toward a desired combination of schedule and effort. Specification of appropriate constraint functions and an objective function allows application of fuzzy optimization methods. The example provided continues the project scenario to demonstrate optimization of parameters to satisfy the objective.

Keywords: COCOMO, fuzzy, optimization.

1 Introduction

This paper is concerned with a method of improving project outcome using Constructive Cost Models (COCOMO). In our studies of fuzzy numbers (FN) and of COCOMO, we recognized an opportunity for improvement of COCOMO models through embracing uncertainty in software cost estimation. We begin by providing an introduction to fuzzy numbers (FN)in Section 2. In Section 3 we describe how uncertainty in evaluating COCOMO parameter values translates into a fuzzy COCOMO ($f-$COCOMO). Finally as introduction, in Section 4, we describe the issue we address with respect to improving estimates of COCOMO parameters.

In Section 5, we explain a method of using optimization to reduce uncertainty in a COCOMO model. Lastly, our results are summarized and future work is proposed.

2 Fuzzy Logic, Sets, and Numbers

Good, published reference books for fuzzy sets and fuzzy logic are [BE02] and [KY95]. Chapter 2 of [BJ05] is used as a model for this section. Most of the information here is available from several sources, where any two of [BE02], [KY95], and [Mat05] could have been used as explicit references.

Lotfi Zadeh's 1965 paper on fuzzy sets [Zad65] is the document which began the field of fuzzy logic. One earlier paper by R. Bellman, R. Kalaba and L.A. Zadeh [BKZ64] is considered the oldest paper about fuzzy sets, but Zadeh's 1965 paper is considered the seminal paper on fuzzy sets and fuzzy logic. Zadeh and his subsequent doctoral students became, and still are, the first tier of scientists in the field of fuzzy logic. The second tier was formed as Zadeh sought to sell fuzzy logic world wide; some of the familiar names are Mamdami, Sugeno, Klir, Dubois and Prade.

Fuzzy logic initially met with resistance to its acceptance and use. With success stories using fuzzy logic mounting [Juu99][GU+04], fuzzy logic is becoming more accepted. Some of the better known commercial successes are the bullet train between Tokyo and Osaka Japan, video camera features, and ABS brakes. Fuzzy logic is closely linked with soft computing (a Zadeh creation at the Berkeley Initiative in Soft Computing)[Nik05]. Recently Zadeh has expanded his fuzzy thinking with The Generalized Theory of Uncertainty [Zad05], which addresses reasoning under uncertainty using possibilistic and probabilistic reasoning. Still, after forty years, fuzzy logic can still be designated as under-recognized for its contributions [Nik05]; a survey of fuzzy opportunities in [GN97] and [BVH99] suggest a promising future for fuzzy logic.

The notation used here to specify a fuzzy set is to place a "bar" over a letter; e.g. $\overline{X}$ and $\overline{x}$.

2.1 Fuzzy Logic

Fuzzy logic is based in a principle recently reiterated by Lotfi Zadeh, "everything is a matter of degree" [Nik05]. Whereas Boolean logic postulates the concept of truth as a function from a linguistic expression onto the set $\{0, 1\}$, fuzzy logic postulates the concept of truth as a function from a linguistic expression onto the interval [0,1]. In other words, a Boolean truth ("the switch is ON") is either 1 (ON) or 0 (OFF). In fuzzy logic, the truth of the same statement might be 0.1 ("the switch is OFF in the mechanical sense but a trickle of current is still flowing, or may be flowing!"). Substitute "transistor" for "switch" and applicability of fuzzy logic to everyday computer possibilities is evident.

There is more than one way in which fuzziness occurs in computation, perception-based (linguistic) and measurement-based (numerical)[Zad03] fuzziness. Perception based (linguistic) fuzziness is one evident to the layperson. Consider an example, "That woman is beautiful." "Beautiful" is a linguistic variable which, for individuals, is subjective. The statement is definitely true in the instance of the speaker, but possibly not true from some (or most) listener(s). The degree of truthfulness of the statement for an instance is called its *membership value* for that instance. *Matter of degree* applies to both nominal (e.g., beautiful) and ordinal (e.g., low) linguistic expressions. Linguistically, membership values may be modified by quantifiers (aka hedges); e.g., "That woman is somewhat beautiful." One of the grand challenges of fuzzy logic is to create algorithms to compute with linguistic variables. The state of the art is to map linguistic expressions into mathematical representations (fuzzify), apply appropriate algorithms, then convert results back into a linguistic expression (for a good graphical example see MATLAB's Fuzzy Inference System [Mat05]).

A different type of fuzziness is a result of precision issues for interval or absolute (ratio) data. For example, 3.14 and 355/113 are both representations of π, but neither is π. In textbook numerical computing, one is normally satisfied with several digits of significance and can estimate error caused by estimation. However, in other situations, constants and/or initial conditions are derived from measurements, inherently imprecise, for which a "true" value is not known. Sufficiently bounded error estimates may not be possible. The extent of uncertainty may make computation intractable. Fuzzy logic seeks to estimate a solution, or neighborhood of a solution.

In some cases measurements are simply not available, or cannot be made. In those cases, parameter estimates from an expert may be used, or composed from several estimates from experts. An example might be a large new software project for a new company. Experts can estimate from prior, possibly marginally-related, experiences. Using fuzzy logic, one can evaluate the possible schedule and cost, and evaluate strategies to improve the project's outcome.

2.2 Fuzzy Sets

Consider some non-empty set Ω. A fuzzy subset $\overline{A}$ of Ω is defined by a *membership function*, written $\overline{A}(x)$, which produces values in some closed interval [0,z], $z > 0$, for all x in Ω. Most commonly $[0, 1]$ is the interval used. That is, in most cases, $\overline{A}(x)$ is a function mapping Ω into $[0, 1]$. Bi-valued logic is not required. For $\overline{A}(x_a) = 0$, x_a does not belong to $\overline{A}$. For $\overline{A}(x_b) = y$, the membership value of x_b in $\overline{A}$ is y. When $\overline{A}(x)$ is always equal to one or zero we obtain a crisp (non-fuzzy) subset of Ω. For all fuzzy sets $\overline{B}$, $\overline{C}, \ldots$ we use $\overline{B}(x)$, $\overline{C}(x), \ldots$ for the value of their membership function at x.

"Crisp" means "not fuzzy". A crisp set is a set for which there is no uncertainty as to membership of its components. A crisp number is just a real number (but rational representations of irrationals are fuzzy). A crisp function maps real numbers (or real vectors) into real numbers. A crisp solution to a problem is a solution involving crisp sets, crisp numbers, crisp functions, etc.

2.3 Fuzzy Numbers

Fuzzy numbers are fuzzy sets on which certain restrictions and distinctive definitions have been applied. For fuzzy set $\overline{A}$,

1. $\overline{A}(x)$ is a function mapping Ω, often a subset of $\Re$, into [0,1].
2. The value of $\overline{A}$(x) is called its *membership value* in $\overline{A}$, denoted by α.
3. For some x ϵ Ω, $\overline{A}(x)$=1; that is, fuzzy numbers are *normalized* to 1. The set {x | $\overline{A}(x)$=1} is called the *core*. If x is a singleton it may be called a *vertex*.
4. The membership function $\overline{A}$ must have C^0 continuity; i.e., be connected, have no breaks.
5. The interval for which $\overline{A}(x) > 0$, say [a,c], $a \leq$the core$\leq c$, is called the *support* of the fuzzy number. In this case, a (c) is called the left (right) support. Some authors allow the support

to be an open interval. The membership function from a to the core (the core to c) must be monotonically increasing (decreasing).

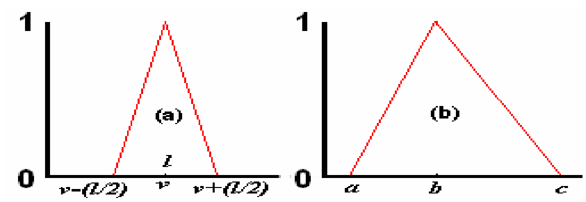

Figure 1: (a) Symmetric TFN; (b) Asymmetric TFN

One commonly found representation of fuzzy numbers is the triangular fuzzy number (TFN). One may find symmetric TFNs represented by a couplet, (v, l), where v is the position of the vertex, and l is the length of the support bisected by a perpendicular from the vertex. Other representations for TFNs exist. The notation a/b/c, where b the vertex, and [a,c] is the support (base of the triangle) is very commonly used. See Figure 1 This notation allows for both symmetric and asymmetric TFNs.

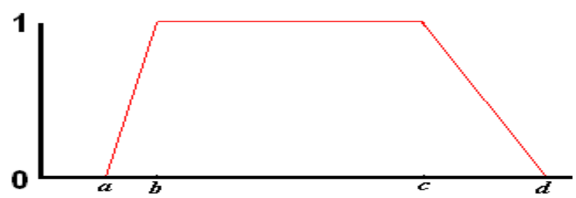

Figure 2: Trapezoidal Fuzzy Number

In cases where the core is not a singleton, one has a trapezoidal fuzzy number (TrFN). See Figure 2. Though less frequently discussed than TFNs, the notation used is a/b/c/d; [a,d] is the support and [b,c] is the core.

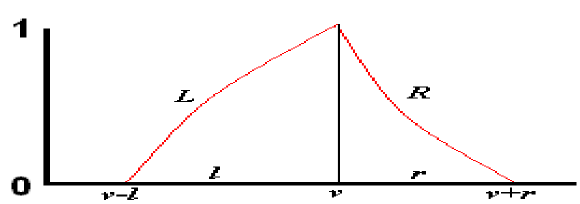

Figure 3: Triangular-shaped Fuzzy Number

For TFNs and TrFNs the sides of the membership function are straight lines. For some purposes curves are used for the legs; in those cases triangular-shaped (TsFN) and trapezoidal-shaped FNs are the result. The notation is changed to allow for description of the membership functions. One form is the notation $\overline{P} \approx (a/b/c)$ to show that it is partially defined by the three numbers a, b, and c [Buc06].

Another descriptive notation is $L - R$ notation: $(v, l, r)_{LR}$, where v is the vertex, l (r) is the extent of the support to the left (right) of the vertex, and L (R) is the left (right) shape function. L and R may represent any membership functions which satisfy FN requirements [MM01]. If L and R are quadratic functions, the FN may be called a parabolic fuzzy number. Conic fuzzy numbers are those for which L and R describe a hyperbola, ellipse, parabola, or line (Figure 3).

Finally for this description, and less commonly found, are S-shaped or Gaussian FNs. The membership functions for them are higher order functions. In some cases, the membership functions appear to asymptotically approach the horizontal axis; however, requirements of FNs must still hold; i.e., particularly here, there must be a support and core.

2.3.1 Alpha-Cuts

Alpha-cuts are slices through a fuzzy set producing crisp (non-fuzzy) sets. If $\overline{A}$ is a fuzzy subset of some set Ω, then an α-cut of $\overline{A}$, written $\overline{A}[\alpha]$, is defined as

$$\overline{A}[\alpha] = \{x \in \Omega | \overline{A}(x) \geq \alpha\}, \tag{1}$$

for all α, $0 < \alpha \leq 1$. The $\alpha = 0$ cut, or $\overline{A}[0]$, must be defined separately.

Note that every α-cut of a FN is a closed, bounded interval. Also note the use of brackets to indicate α-cuts. If $q_L(\alpha)$ $(q_R(\alpha))$ are functions defining legs of $\overline{A}$,

$$\overline{A}[\alpha] = [q_L(\alpha), q_R(\alpha)], \tag{2}$$

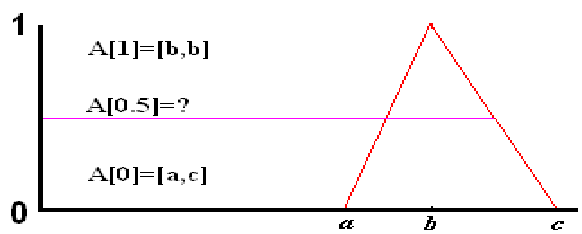

Figure 4: Alpha-Cuts

The α-cut at 0 is the support. The α-cut at 1 is the core. In Figure 4, the α-cut at 0.5 is $[\frac{a+b}{2}, \frac{b+c}{2}]$, from

$$q_L(\alpha) = \frac{(x - a)}{(b - a)} \tag{3}$$

$$q_R(\alpha) = 1 - \frac{(x - b)}{(c - b)} \tag{4}$$

2.3.2 Fuzzy Arithmetic

To compute with fuzzy numbers, one needs fuzzy operations. There are two commonly accepted equivalent methods of defining the binary operations of addition, subtraction, multiplication, and division, for fuzzy numbers. Both are presented here because both are frequently used. They are the *extension principle* and *interval arithmetic using α-cuts*. These operations are closed in the fuzzy numbers, but preservation of shape is not guaranteed. For example, the sum or difference of two TFNs is a TFN, but the product or quotient is generally not a TFN. In Figure 5, two TFNs and their binary compositions are shown.

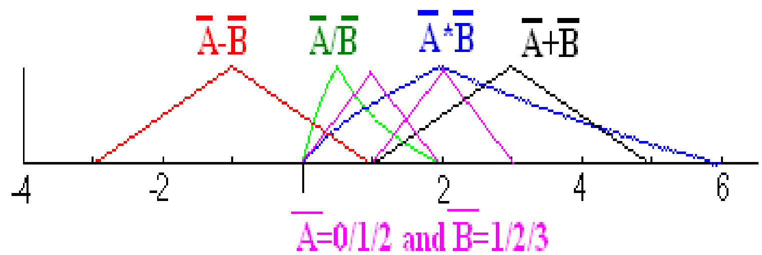

Figure 5: Example Using Fuzzy Arithmetic

Extension Principle: Let $\overline{A}$ and $\overline{B}$ be two fuzzy numbers. Compute the membership function for

$$\overline{C} = \overline{A} + \overline{B}: \ \overline{C}(z) = \sup_{x,y}\{\min(\overline{A}(x), \overline{B}(y)) \mid x + y = z\} \ . \tag{5}$$

$$\overline{C} = \overline{A} - \overline{B}: \ \overline{C}(z) = \sup_{x,y}\{\min(\overline{A}(x), \overline{B}(y)) \mid x - y = z\} \ . \tag{6}$$

$$\overline{C} = \overline{A} \cdot \overline{B}: \ \overline{C}(z) = \sup_{x,y}\{\min(\overline{A}(x), \overline{B}(y)) \mid x \cdot y = z\}, \tag{7}$$

and if 0 is not in the support of $\overline{B}$,

$$\overline{C} = \overline{A}/\overline{B}: \ \overline{C}(z) = \sup_{x,y}\{\min(\overline{A}(x), \overline{B}(y)) \mid x/y = z\} \ . \tag{8}$$

Interval Arithmetic Using α-Cuts: Again, let $\overline{A}$ and $\overline{B}$ be fuzzy numbers. Moving from $\overline{A}[0]$ and $\overline{B}[0]$ to $\overline{A}[1]$ and $\overline{B}[1]$, pairs of intervals are composed to create α-cuts of the result. Let $\overline{A}[\alpha]$=$[a_L,a_R]$ and $\overline{B}[\alpha]$=$[b_L, b_R]$ be α-cuts of $\overline{A}$ and $\overline{B}$.

Compute α-cuts of the membership function for

$$\overline{C} = \overline{A} + \overline{B}: \ \overline{C}[\alpha] = [c_L, c_R] = [a_L, a_R] + [b_L, b_R] = [a_L + b_L, a_R + b_R]. \tag{9}$$

$$\overline{C} = \overline{A} - \overline{B}: \ \overline{C}[\alpha] = [c_L, c_R] = [a_L, a_R] - [b_L, b_R] = [a_L - b_R, a_R - b_L]. \tag{10}$$

$$\overline{C} = \overline{A} \cdot \overline{B}: \ \overline{C}[\alpha] = [c_L, c_R] = [a_L, a_R] \cdot [b_L, b_R] = [\beta_L, \beta_R]. \tag{11}$$

where

$$\beta_L = \min\{a_L b_L, a_L b_R, a_R b_L, a_R b_R\}, \tag{12}$$

$$\beta_R = \max\{a_L b_L, a_L b_R, a_R b_L, a_R b_R\} \ . \tag{13}$$

and if 0 is not in the support of $\overline{B}$,

$$\overline{C} = \overline{A}/\overline{B}: \ \overline{C}[\alpha] = [c_L, c_R] = [a_L, a_R]/[b_L, b_R] = [a_L, a_R] \cdot [\frac{1}{b_R}, \frac{1}{b_L}], \tag{14}$$

Fuzzy Functions and Soft Computing: Fuzzy functions are usually fuzzifications of crisp functions. In the case of polynomials, for example, one may have fuzzy coefficients, or fuzzy variables, or fuzzy operators. Note however, the fuzzy community believes it is not necessary to start with a crisp function in order to create a fuzzy function.

One-Step/Two-Step: Working with fuzzy functions properly require awareness of the *one-step/two-step* issue, aka *one-step/multi-step*. Consider a crisp expression, say $x(1-x)$. A one-step fuzzification, $\overline{x(1-x)}$, does not produce the same result as evaluating using $\overline{X}$; that is, $\overline{X}(1-\overline{X})$. Though the latter may involve multiple operations, we often call it two-step.

There is no difference between the two methods applied as binary operations; that is, $\overline{X}(1-\overline{X})$ produces the same result by either method. Fuzzification, $\overline{x(1-x)}$, using α-cuts ($x=f^{-1}(\alpha)$), and alternatively $\alpha=f(x)$, are also consistent with each other. However, a difficulty arises if one does not realize that fuzzifying a variable and computing a result, is different from fuzzifying a result.

2.4 Fuzzy Estimators

In this paper we consider only one method of fuzzy estimation: expert opinion. More information on fuzzy estimators is in [BJ05]. We may obtain a value for the b from some group of experts (software analysts). This group may consist of only one expert. First assume we have only one expert and he/she is to estimate the value of b. We can solicit this estimate from the expert as is done in estimating job times in project scheduling ([Tah92], Chapter 13). Let b_1 = the "pessimistic" value of b, or the smallest possible value, let b_3 = be the "optimistic" value of b, or the highest possible value, and let b_2 = the most likely value of b. We then ask the expert to give values for b_1, b_2, b_3 and we construct the triangular fuzzy number $\overline{b} = (b_1/b_2/b_3)$ for b. If we have a group of N experts all to estimate the value of b we solicit the b_{1i}, b_{2i} and b_{3i}, $1 \leq i \leq N$, from them. Let b_1 be the average of the b_{1i}, b_2 is the mean of the b_{2i} and b_3 is the average of the b_{3i}. The simplest thing to do is to use $(b_1/b_2/b_3)$ for $\overline{b}$. We now assume, when necessary, this is how we employ expert opinion to obtain fuzzy estimators.

2.5 Simulation Optimization

Simulation optimization is the process of finding an optimal solution for a system from output of a simulation model of the system [aK02]. Let x be the solution for a parameter variable, in a crisp COCOMO model that we wish to study. Fuzzy estimation of parameters for the model leads to a fuzzy COCOMO model. Having a single fuzzy parameter $\overline{x}$ results in a fuzzy model.

3 Fuzzy COCOMO

COCOMO-II is a currently popular model for Software Cost Estimation (SCE). It is designed to be generally applicable to most software development. It is tailor-able to an organization. It has specific and complex modeling equations (consider simplified equations, Equations 15 and 16, and their table of partial definitions Table 1, for COCOMO-II). It has published results, computer models, and datasets for study. Because COCOMO-II embraces uncertainty in specification of SCE, we see opportunity in it with respect to fuzzy logic. Others have briefly addressed COCOMO-II as a fuzzy issue [MPSR00].

$$PM = A \times Size^E \times \prod_{i=1}^{n} EM_i + PM_{Auto}, \tag{15}$$

$$E = B + 0.01 \times \sum_{j=1}^{5} SF_j. \tag{16}$$

Reference	Description
A	calibrated effort coefficient (2.94 for COCOMO II.2000)
B	calibrated base scaling exponent (0.91 for COCOMO II.2000)
E	computed scaling exponent for effort
EM	effort multipliers (n)
n	number of effort multipliers used (up to 17)
PM	person months
SF	scale factors (5)
$Size$	source lines of code (thousands)

Table 1: Definition of COCOMO-II Terms

Reference	Description
PREC	Precedentedness
FLEX	Development Flexibility
RESL	Architecture / Risk Resolution
TEAM	Team Cohesion
PMAT	Process Maturity

Table 2: Scale Factors

Use of COCOMO for software cost estimation begins with measurement of resources. Parameters are estimated from data on prior projects. Some parameters are estimated from supervisors' evaluations of employees and other resources. In either case, these estimates are usually not crisp estimates. An estimate may be just a linguistic value (very low, low, nominal, high, very high, extra high) for a linguistic variable (some scale factor or effort multiplier). For classical COCOMO, each linguistic variable is mapped to a crisp number (defuzzified) according to tables and explanations given in the COCOMO book [BAB$^+$00]. Uncertainty in estimates are lost.

Linguistic variables may be represented by fuzzy numbers. Expert opinion or confidence intervals for the value of a variable may be used to build fuzzy numbers [BJ05]. Creating a fuzzy number to represent a linguistic variable results in loss of less information than with defuzzification.

4 An Opportunity for Improvement

Fuzzy arithmetic is usually done using one of two equivalent methods, extension principle or interval arithmetic on α-cuts (Section 2.3.2). Looking back at Figure 5, notice how for the resultant fuzzy numbers the left and right supports are farther apart; that is, the resultant fuzzy numbers are more fuzzy.

As a baseline simplistic example, consider a Size of $5,000$, with nominal values for Scale Factors and for Effort Multipliers. As the product of nominal values for Effort Multipliers is unity, the equation we shall work with is

$$PM = A \times Size^{B+0.01\times\sum_{j=1}^{5} SF_j} = 17.26 \tag{17}$$

Reference	Range	Description
RELY	0.82 – 1.26	Required Software Reliability
DATA	0.90 – 1.28	Database Size
CPLX	0.73 – 1.74	Product Complexity
RUSE	0.95 – 1.24	Developed for Reusability
DOCU	0.81 – 1.23	Documentation Match to Life-Cycle Needs
TIME	1.00 – 1.63	Execution Time Constraint
STOR	1.00 – 1.46	Main Storage Constraint
PVOL	0.87 – 1.30	Platform Volatility
ACAP	1.42 – 0.71	Analyst Capability
PCAP	1.34 – 0.76	Programmer Capability
PCON	1.29 – 0.81	Personnel Continuity
APEX	1.22 – 0.81	Applications Experience
PLEX	1.19 – 0.85	Platform Experience
LTEX	1.20 – 0.84	Language and Tool Experience
TOOL	1.17 – 0.78	Use of Software Tools
SITE	1.22 – 0.80	Multisite Development
SCED	1.43 – 1.00	Required Development Schedule

Table 3: Effort Multipliers

We build an f-COCOMO example. Suppose Team Cohesion (TEAM) is inexactly known. It can treated as a fuzzy parameter about nominal; in this case represent it by a TFN 2.74/3.29/3.84. This fuzzification of TEAM yields a TsFN $PM = 17.11/17.26/17.41$.

Similarly suppose Process Maturity (PMAT) is inexactly known. It can treated as a fuzzy parameter about nominal; in this case represent it by a TFN 3.90/4.68/5.46. This fuzzification of PMAT yields a TsFN $PM = 17.04/17.26/17.48$.

And finally for our simplistic example, if both are inexactly known $PM = 16.89/17.26/17.64$. Note that uncertainty in PM is expanded. One may imagine effects of fuzziness applied to all of the Scale Factors. Fuzziness in Effort Multipliers would also increase fuzziness in PM.

However, this is a feature of fuzzy arithmetic we believe may be exploited. Using simulation optimization, we can consider the inverse problem of, “Given an effort budget, where should resources be directed to increase possibility of staying within budget?”

For a COCOMO model, we have two fuzzy results we may take into consideration. Before beginning a project, one has estimates for what a project is allowed to take to deliver, and for what a project may cost in person months. Allowable boundaries of the answer are known; thus, that information may be used to optimize parameters.

5 The Inverse Problem

As a continuation of our simple example, consider that management has allotted no more than 16 person months to the project. With that additional information, one may determine what tightening or loosing of parameters are needed or possible. For our example, if one could target Team Cohesion and/or Process Maturity for improvement, what kind of improvement is indicated to stay with a budget of 16 Person Months? The method we chose follows from research on fuzzy Monte Carlo methods [BEJ07]. This is a maximization problem; our objective is to maximize

$$PM = A \times Size^{B+0.01\times\sum_{j=1}^{5} SF_j} < 16. \quad (18)$$

however with a logarithm transformation we can use

$$ln(PM) = ln(A) + (B + 0.01 \times \sum_{j=1}^{5} SF_j) \times ln(S) < ln(16) = 2.7726. \quad (19)$$

$$ln(PM) = ln(A) + B \times ln(S) + 0.01 \times ln(S) \times \sum_{j=1}^{5} SF_j < 2.7726. \quad (20)$$

$$ln(PM) = 0.0161 \times TEAM + 0.0161 \times PMAT < 0.0526. \quad (21)$$

For this model we pretend that our Team Cohesion is about Nominal, but could possibly be raised to Very High; we allow the same improvement for Process Maturity. For our simple example we find that it is possible to stay within the PM budget. If Team Cohesion and Process Maturity can be improve to near Very High, (1.43/1.51/2.62) and (1.62/1.76/2.99) respectively, outlook is 'good'. Additional evaluations are possible with a defined defuzzification scheme.

Reference	Description	Evaluation	SF_i
PREC	Precedentedness	nominal	3.72
FLEX	Development Flexibility	nominal	3.04
RESL	Architecture / Risk Resolution	nominal	4.20
TEAM	Team Cohesion	nominal to very high	3.29 to 1.10
PMAT	Process Maturity	nominal to very high	4.58 to 1.56

Table 4: Input Scale Factors for Simple Example Optimization

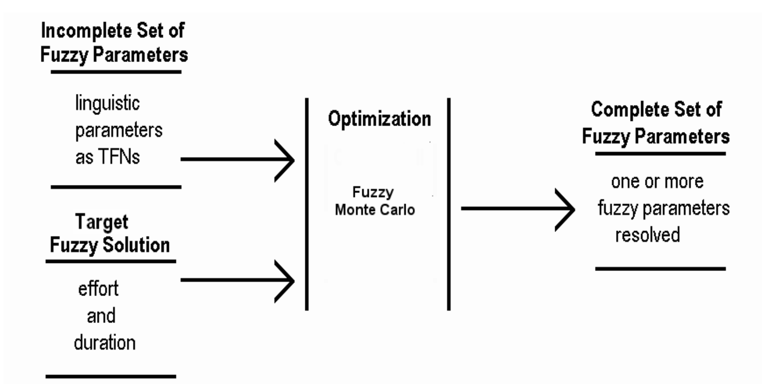

Figure 6: COCOMO Optimization Using Monte Carlo

For our simple simulation, the inequality (18) provides both the objective and the constraint. However, it is clear that additional constraints may be supplied; e.g., cost to improve a Scale Factor.

6 Summary and Future Work

This paper continues our research into simulating fuzzy systems and particularly in application of fuzzy logic and systems to software cost estimation. We start with a crisp COCOMO model which depends on interpretation of a set of linguistic variables to create a set of crisp parameters. There are always a number of parameters in the system whose values are not known precisely. These parameters need to be estimated and their estimators contain uncertainties. We model these uncertainties using fuzzy numbers. Using fuzzy numbers for these parameters changes the COCOMO model into a fuzzy COCOMO model. This uncertainty in the parameter values computes a larger uncertainty in the COCOMO result. Usually a schedule deadline or a project cost estimate is known before commencement of a project. We show how, by addressing the inverse problem, one may use such an estimate to improve one or more COCOMO parameters. This has value in that it should enhance the fidelity of the model to the real project, or be used as an additional tool to manage a project.

This research has attempted to show a method for improving a single-objective COCOMO model through use of fuzzy logic. Although a simple example was used to demonstrate the concept, the method is extensible to other COCOMO parameters. Future research will be concerned with extending this method to a multi-objective fuzzy COCOMO model; one for which fuzzy project parameters may be optimized simultaneously to a schedule objective and cost objective.

References

[aK02] Sifurdur Ólafsson and Jumi Kim, *Simulation optimization*, WSC '02: Proceedings of the 2002 Winter Simulation Conference, Dec 2002, pp. 79–84.

[BAB+00] Barry W. Boehm, Chris Abts, A. Windsor Brown, Sunita Chulani, Bradford K. Clark, Ellis Horowitz, Ray Madachy, Donald Reifer, and Bert Steece, *Software Cost Estimation with COCOMO II*, Prentice Hall PTR, Upper Saddle River, NJ, 2000.

[BE02] J.J. Buckley and E. Eslami, *Introduction to fuzzy logic and fuzzy sets*, Physica-Verlag, Heidelberg, Germany, 2002.

[BEJ07] J.J. Buckley, E. Eslami, and L.J. Jowers, *Monte carlo studies with random fuzzy numbers*, 2007, Work-in-progress.

[BJ05] J.J. Buckley and L.J. Jowers, *Simulating continuous fuzzy systems*, Springer-Verlag, Heidelberg, Germany, 2005.

[BKZ64] R. Bellman, R. Kalaba, and L.A. Zadeh, *RAND Memorandum RM-4307-PR, "Abstraction and Pattern Classification"*, October 1964.

[Buc06] J.J. Buckley, *Fuzzy probability and statistics*, Springer, Heidelberg, Germany, 2006, To appear.

[BVH99] R. Babuška, H.B. Verbruggen, and H. Hellendoorn, *Promising Fuzzy Modeling and Control Methodologies for Industrial Applications*, Proceedings of ESIT '99 - European Symposium on Intelligent Techniques, June 3, 1999, Available: http://www.erudit.de/erudit/events/esit99/12643_P.pdf.

[GN97] Alfred L. Guiffrida and Rakesh Nagi, *Fuzzy Set Theory Applications in Production Management Research: A Literature Survey*, 1997, or late 1996, http://www.eng.buffalo.edu/~nagi/papers/fuzzy.pdf.

[GU+04] GU+04, *several articles*, Research Journal / Graz University of Technology (2004), 00–00, http://dokumentation.htu.tugraz.at/tug-forschung/pdfs/2004a.pdf.

[Juu99] Esko K. Juuso, *Modelling and Control with Linguistic Equations*, Proceedings of ESIT '99 - European Symposium on Intelligent Techniques, June 3, 1999, Available: http://www.erudit.de/erudit/events/esit99/12536_p.pdf.

[KY95] George J. Klir and Bo Yuan, *Fuzzy sets and fuzzy logic: Theory and applications*, Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1995.

[Mat05] Matlab, *Fuzzy Logic Toolbox: Fuzzy Inference Systems*, 2005, http://www.mathworks.com/access/helpdesk/help/toolbox/fuzzy/.

[MM01] B. Matarazzo and G. Munda, *New approaches for the comparison of L-R fuzzy numbers: a theoretical and operational analysis*, Fuzzy Sets and Systems **118** (2001), 407–418.

[MPSR00] Petr Musílek, Witold Pedrycz, Giancarlo Succi, and Marek Reformat, *Software cost estimation with fuzzy models*, SIGAPP Appl. Comput. Rev. **8** (2000), no. 2, 24–29.

[Nik05] M. Nikravesh, *2005 Berkeley Initiative in Soft Computing International Special Event (BISCSE05)*, http://www-bisc.cs.berkeley.edu/BISCSE2005/.

[Tah92] H.A. Taha, *Operations research, fifth edition*, Macmillan, New York, NY, 1992.

[Zad65] L. Zadeh, *Fuzzy sets*, Inf. Control **8** (1965), 338–353.

[Zad03] Lotfi A. Zadeh, *Computing with Words and Its Application to Information Processing, Decision and Control*, The 2003 IEEE International Conference on Information Reuse and Integration (2003), Keynote speech.

[Zad05] ———, *Toward a Generalized Theory of Uncertainty (GTU)An Outline*, January 20, 2005, To appear in *Information Sciences*, http://www-bisc.cs.berkeley.edu/BISCSE2005/Zadeh2005.pdf.