"The right to post a revised personal version of the text of the final journal article (to reflect changes made in the peer review process) on the author's personal or institutional web site or server, incorporating the complete citation and with a link to the Digital Object Identifier (DOI) of the article;"

The DOI is http://dx.doi.org/10.1016/j.infsof.2009.04.003

Sizing User Stories Using Paired Comparisons

© Eduardo Miranda, Institute for Software Research – Carnegie-Mellon University,

Pierre Bourque, Alain Abran, École de technologie supérieure – Université du Québec

Abstract

Agile estimation approaches usually start by sizing the user stories to be developed by comparing them to one another. Various techniques, with varying degrees of formality, are used to perform the comparisons – plain contrasts, triangulation, planning poker, and voting. This article proposes the use of a modified paired comparison method in which a reduced number of comparisons is selected according to an incomplete cyclic design. Using two sets of data, the authors show that the proposed method produces good estimates, even when the number of comparisons is reduced by half those required by the original formulation of the method.

Keywords

Agile estimation, triangulation, estimating by analogy, paired comparisons, software sizing, expert estimation, expert judgment, project planning, incomplete cyclic designs

1. Introduction

Agile estimation approaches typically comprise three steps: 1) comparison of the user stories to be developed to one another for the purpose of establishing their relative size; 2) conversion of the size estimates to lead times using an assumed team productivity; and 3) re-estimation of the project lead times using the team's actual productivity, once this becomes known after two or three iterations.

User story comparisons take the following form: "This story is like that story, so its size must be roughly the same," or "This story is a little bit bigger than that story which was estimated at 4, so its size should be around 5." The numbers 4 and 5 in the previous sentence are called "story points", which are numbers in ratio scale purportedly proportional to the effort it would take to develop each story based on its perceived size and complexity [1]. A 6-point user story is expected to require about twice as much effort as a 3-point user story. The degree of structure in the comparison process ranges from the ad hoc comparison of any two user stories, to triangulation – the comparison of a user story with two others, to a number of Delphi [2] like techniques such as the planning poker [3]. To avoid wasting time discussing insignificant differences between user stories, the use of a Fibonacci or power series is sometimes recommended, such as if the difference between two user stories is not as large as a following term in the series, the two user stories are assumed to be of the same size [4].

The project lead time is calculated using the concept of *velocity*, which is a proxy for the productivity of the team. At first, velocity is estimated or taken from a previous project, but, as work progresses, it is measured by tallying the number of story points completed during the counting period. Velocity is measured in story points per iteration, or story points per month. As an example, if the current team velocity is 30 story points per month, it will take the team 2 months to deliver 60 story points-worth of user stories.

As will be shown later, comparing one user story to another, or to two others, is not good enough to produce reliable estimates. The first reaction to this is to increase the number of comparisons, but this creates some problems of its own. As even the most devoted estimator gets tired after making a large number of comparisons, the question of how many comparisons to make becomes really important, as does the problem of dealing with the inconsistencies inherent to the judging process.

To address these problems, we propose the use of incomplete cyclic designs to identify which user stories to compare with which to reach a desired accuracy, and the use of the paired comparison method [5-7] to deal with judgment inconsistencies.

The rest of the paper is organized as follows: section 2 formalizes the triangulation concept, section 3 explains the basic paired comparison method, section 4 presents the modified process using incomplete cyclic designs, section 5 discusses the accuracy and precision of the resulting estimates, and section 6 provides a summary of the article.

2. Agile estimation and triangulation

Triangulation is defined in the Agile literature as the process of establishing the size of a user story relative to two other user stories with the purpose of increasing the reliability[1] of the estimate [3]. When using triangulation, the comparisons sound something like this: "I'm giving user story B 2 points, because it feels like its implementation will take somewhat more effort than user story A, which I already rated at 1 story point, and somewhat less effort than user story C, which I rated as a 4-point story." Despite its intuitive appeal, triangulation is not as simple as the sentence above makes it appear. First, there is the problem of consistency, which can be mathematically expressed as:

<1> $$a_{ij} \times a_{jk} = a_{ik} \forall i, j, k \in n$$

Equation <1> reads as follows: if user story$_i$ is a_{ij} times bigger[2] than user story$_j$, and user story$_j$ is a_{jk} times bigger than user story$_k$, then user story$_i$ must be $a_{ij} \times a_{jk}$ times bigger than user story$_k$. This is important, because lack of consistency among triangulations leads to inaccurate estimates.

Second, which two user stories should you choose as reference points? Does the choice affect the result?

The triangulation process can be visualized by arranging the user stories in a circular pattern and linking those being compared (see Figure 1). Given n user stories to be

[1] A reliable sizing method will yield estimates that are accurate, that is, close to their true value, and precise, that is estimates must be consistent across repeated observations in the same circumstances.

[2] The comparison can go both ways, i.e. replacing bigger for smaller.

estimated, there are $n(n-1)(n-2)/2$ possible configurations or designs which can be evaluated, but not all are equally good. A good design must have two properties: balance and connectedness [8-10]. A design is considered balanced when every user story appears in as many comparisons as any other user story. This ensures that one user story does not overly influence the estimation, while others are under-represented. Connectedness implies that any user story is compared, directly or indirectly, to every other user story. An unconnected graph is undesirable, because the size of some user stories relative to others would be completely indeterminate. Figure 1.b illustrates the problem: the user stories in the lower subset are never compared against those in the upper subset, so each subset could be accurately sized in itself but completely offset with respect to the other.

The number of times a user story appears in a comparison is called the replication factor (r) of the design. In all the designs shown in Figure 1, r is 2.

Balance and connectedness are necessary, but not sufficient conditions for a good estimation. As shown by Burton [8], a low r, such as that used in the triangulation approach ($r = 2$) is very sensitive to errors in judgment, and thus tends to produce unreliable results. In his experiments, Burton found that the correlation (ρ) between the actual and the estimated values using triangulation ranged from a low of 0.46 to a high of 0.92, with a mean value of 0.79. Similar variability was found by the authors using two sets of data, this is discussed later.

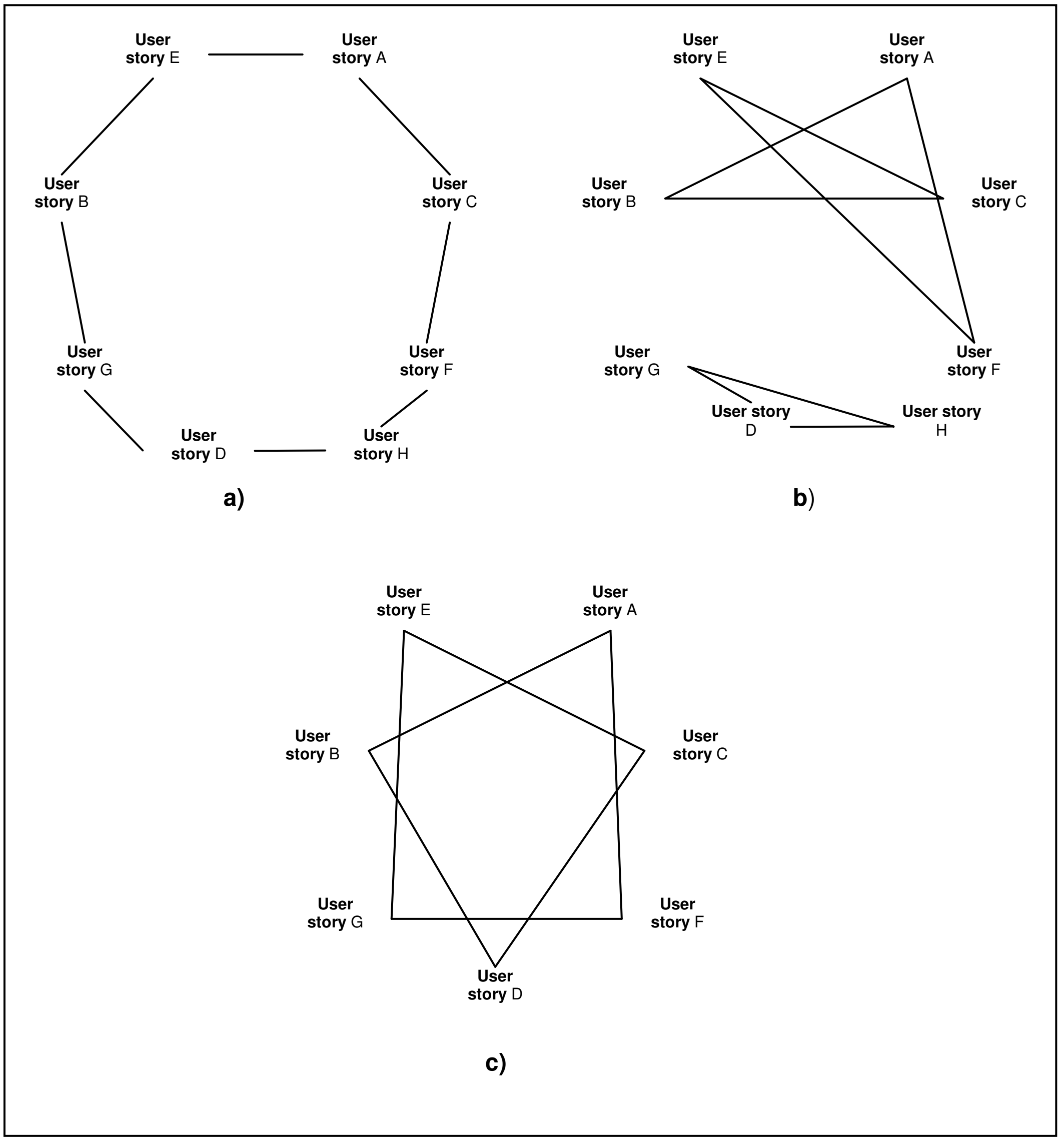


Figure 1 Three triangulation designs out of the $n(n-1)(n-2)/2$ possible ones

Note 1: 'a' and 'c' are good designs, but 'b' is not, as it consists of two disjoint subgraphs.

3. Paired comparison method basics

3.1. Overview

The idea behind the paired comparison method is to estimate the size of n user stories by asking one or more developers to judge their relative largeness rather than to provide absolute size values. After this is done, one of the n user stories is assigned an arbitrary number of story points. Using this story as reference, the sizes in story points, of all the other user stories are calculated. The process is called Full Factorial Pairwise Comparison because it compares all user stories (factors) against one another, see Figure 2.

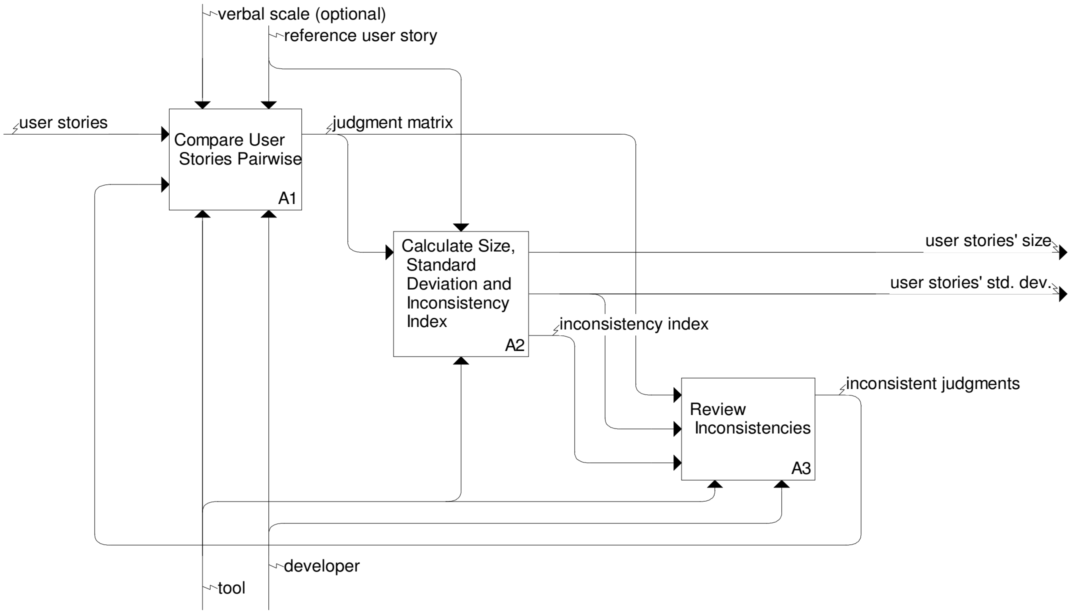


Figure 2 In the Full Factorial paired comparison process. Each user story is compared to every other user story.

Note 1: The optional verbal scale allows the user stories to be compared using an ordinal scale by labeling the comparison of two user stories with adjectives such as "equal", "a little bigger", "much bigger", etc.

Although the selection of the user story to be used as reference and the allocation of story points to it is arbitrary to a certain point[3], a consistent selection and allocation, i.e. two comparable user stories are not allocated 4 story points in one project and 10 in other, is useful for the developers to develop an intuition or sense for the effort required in the realization of a user story with so many story points.

It is also possible to use the method to estimate the effort required by each user story instead of their story points. In this case, a user story whose development effort, from either a previous project or a *spike*[4], is known will be brought in as reference story. For a more detailed description of the method, refer to [5, 6].

[3] The number zero must be reserved for "stories" with not content to preserve the properties of a ratio scale.

[4] In the Agile terminology a spike is an experiment that is performed to learn something. In this case the spike would consist on developing a user story tracking how much effort it required.

In the rest of the document we will work with story points to remain true to the title of the essay but all the same concepts apply to the calculations using effort.

3.2. The Pairwise comparison of user stories

Developers start the process by judging the relative size (a_{ij}) of each user story against every other user story, and recording these values in a matrix called the judgment matrix <2>.

$$\text{<2>}\quad A^{nxn} = \begin{cases} a_{ij} = \dfrac{sp_i}{sp_j} & \text{How much bigger (smaller) user story}_{\text{i}} \text{ is with respect to user story}_{\text{j}} \\ a_{ii} = 1 & \text{Every user story has the same size as itself} \\ a_{ji} = \dfrac{1}{a_{ij}} & \begin{array}{l}\text{If user story}_{\text{i}} \text{ is } a_{ij} \text{ times bigger (smaller) than user story}_{\text{j}}\text{, then user story}_{\text{j}} \\ \text{is } 1/a_{ij} \text{ times smaller (bigger) than user story}_{\text{i}}\end{array} \end{cases}$$

sp_i and sp_j are the as yet unknown numbers of story points for user story$_i$ and user story$_j$ to be derived from the a_{ij} judgments. Note that only the comparisons corresponding to the upper diagonal matrix have to be made, since the a_{ji} are the reciprocals of the a_{ij}.

3.3. Calculating the Size, the Inconsistency Index, and the Standard Deviation

Once all the a_{ij} judgments have been recorded in the judgment matrix, the mean relative size (mrs_i) of user story$_i$ is calculated as the geometric mean [11, 12] of the i^{th} row <3> of the judgment matrix. The size in story points of each user story is then computed by multiplying its mrs_i by the normalized size of the reference user story <4>. For a more detailed description of the method, refer to [5, 6].

$$\text{<3>}\qquad mrs_i = \left(\prod_{j=1}^{n} a_{ij} \right)^{\frac{1}{n}}$$

$$\text{<4>}\qquad sp_i = \frac{sp_{reference}}{mrs_{reference}} \times mrs_i$$

As inconsistencies are inherent to the judgment process, Crawford [12] and Aguaron [13] suggest the use of <5> as an unbiased estimator of the variance of the inconsistencies of the judgment matrix $A^{n\times n}$. The larger the inconsistencies between comparisons, the larger the variance will be. The square root of <5> is called the Inconsistency Index <6> of the judgment matrix.

<5> $$\sigma_A^2 = \frac{\sum_{i<j}^{n}\left(\ln a_{ij} - \ln\frac{mrs_i}{mrs_j}\right)^2}{\frac{n(n-1)}{2} - (n-1)}$$

<6> $$InconsistencyIndex = \sqrt{\sigma_A^2} = \sqrt{\frac{2\sum_{i<j}^{n}\left(\ln a_{ij} - \ln\frac{mrs_i}{mrs_j}\right)^2}{(n-1)(n-2)}}$$

While the Inconsistency Index gives an overall idea of the quality of the judgments, it is a quantity that is difficult to interpret. A much better alternative is to present the estimator with a range estimate – an interval within which the estimate will likely fall for a given degree of inconsistency.

To calculate the extremes of the interval, we start by assuming that each user story contributes equally to σ_A^2. This assumption allows us to write equation <7>, where σ_A^2 is shown to result from the sum of n individual inconsistencies σ_i^2 contributed by each user story[5]. The standard deviation of the size of each user story could then be calculated as the product of its estimated size and its individual inconsistency <8> . The range estimate is given by <9>.

<7> $$\sigma_A^2 = \sum_{i=1}^{n-1}\sigma_i^2 = n\sigma_i^2$$
$$\therefore$$
$$\sigma_i = \sqrt{\frac{\sigma_A^2}{n}} = \frac{InconsistencyIndex}{\sqrt{n}}$$

<8> $$\sigma_{sp_i} = sp_i \times \frac{InconsistencyIndex}{\sqrt{n}}$$

<9> $$RangeEstimate = [sp_i - \sigma_i, sp_i + \sigma_i]$$

Other approaches to calculating the standard deviation of the size exist. Hihn [14] proposes the use of a triangular distribution, where the developer or estimator judges the relation between two user stories in terms of the best case, the most likely case, and the worst-case scenarios, but, given the rather large number of user stories included in a typical project, we found this to be too taxing.

[5] If the reference story is brought in from another project or from a spike the denominator in equations <7> and <8> needs to be replaced by $n-1$ instead of n to account for inclusion of the known parameter.

3.4. Reviewing inconsistencies

At this point, the estimator will use the Inconsistency Index or the estimates' range as a guide to decide whether or not these are good enough and stop, or revise all or some of his judgments with the objective of reducing the inconsistencies.

By simulating a large number of judgment matrices and comparing them to perfectly consistent ones, Aguaron [13] determined that, for sets with four or more data points, an Inconsistency Index less than or equal to 0.35 would produce satisfactory results in most cases. To give the reader an idea of its meaning, an Inconsistency Index of 0.35 with 15 user stories being estimated would, under the assumption that all user stories contribute equally to it, result in a size range of ± 9% for each story. If fewer user stories are compared, the size range will be wider. If more user stories are compared, the interval will be narrower..

3.5. A numerical example

Suppose we wanted to estimate the story points of four user stories called A, B, C and D. If we judge the size of A to be three times that of B, five times that of C, and twice that of D, and then we assess B to be roughly a quarter of C and one-and-a-half times D, and C as being five times bigger than D, the resulting matrix is:

$$A^{4\times4}=\begin{array}{c|cccc|} & A & B & C & D \\ A & 1 & \boxed{3.0} & \boxed{5.0} & \boxed{2.0} \\ B & .33 & 1 & \boxed{.25} & \boxed{1.5} \\ C & 0.2 & 4.0 & 1 & \boxed{5.0} \\ D & .50 & .67 & .20 & 1 \end{array}$$

Only the relative size of the upper diagonal elements of the matrix needs to be judged, as all the other values can be derived using the definitions in <2>. Applying equation <3>, the mean relative size of each user story is:

$$mrs_i=\begin{cases}2.34\\0.59\\1.41\\0.50\end{cases}$$

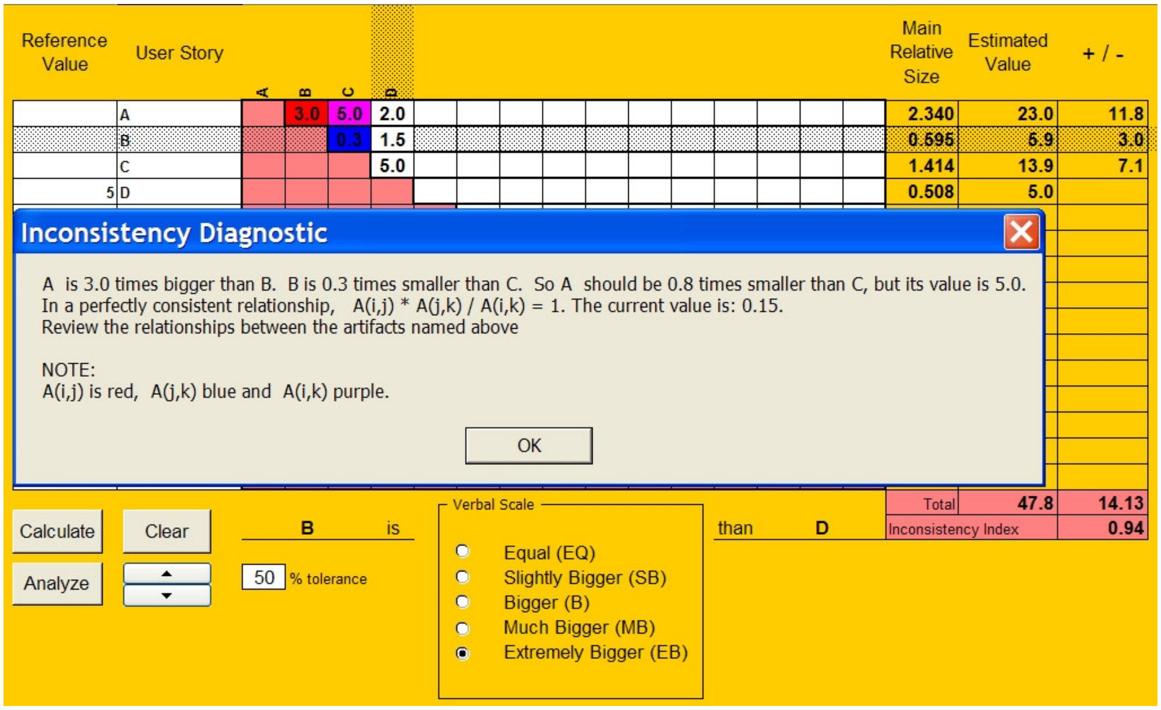
By designating D as the reference user story and assigning it 5 story points we anchor the scale above and are able to calculate the size of the other user stories using equation <4>

$$sp_i=\begin{cases}23.02\\5.85\\13.91\\5.0\end{cases}$$

The total size for the project is 47.78 story points, with an Inconsistency Index of 0.94 which yields a size range of ± 47% for each user story <8>.

As the estimator is not happy with such a range, he decides to review his judgments. To do this, he resorts to equation <1>, which states that, in a perfectly consistent matrix $a_{ij}\times a_{jk}=a_{ik}$, that is, the relative size of user story$_i$ with respect to user story$_j$ multiplied by the relative size of user story$_j$ with respect to user story$_k$ must be equal to the relative size of user story$_i$ with respect to user story$_k$. To operationalize this concept, we divide equation <1> by a_{ik} and obtain <10>. The amount by which equation <10> differs from 1 when the actual judgments are plugged in is used to identify the largest inconsistent judgments (see Figure 3).

<10> $$\frac{a_{ij} \times a_{jk}}{a_{ik}} = 1$$



Reference Value	User Story	A	B	C	D	Main Relative Size	Estimated Value	+ / -
	A		3.0	5.0	2.0	2.340	23.0	11.8
	B			0.3	1.5	0.595	5.9	3.0
	C				5.0	1.414	13.9	7.1
5	D					0.508	5.0	

Inconsistency Diagnostic

A is 3.0 times bigger than B. B is 0.3 times smaller than C. So A should be 0.8 times smaller than C, but its value is 5.0.
In a perfectly consistent relationship, A(i,j) * A(j,k) / A(i,k) = 1. The current value is: 0.15.
Review the relationships between the artifacts named above

NOTE:
A(i,j) is red, A(j,k) blue and A(i,k) purple.

OK

Total	47.8	14.13
Inconsistency Index		0.94

.

Figure 3 Tool interface for detecting the most inconsistent judgments.

Note 1: Each time the "Analyze" button is pressed, a new triad is displayed.

Note 2: The "spinner" is used to specify the amount over which a given triad is considered inconsistent .

Note 3: The values in the matrix and the estimated value are rounded to the nearest digit for display purposes.

After reviewing the estimates, the estimator decides that A is half the size of C and not five times larger as previously stated. So, he records the new judgment in the judgment matrix <11>.

<11> $$A^{4\times 4} = \begin{array}{c|cccc|} & A & B & C & D \\ A & 1 & 3.0 & \boxed{.50} & 2.0 \\ B & .33 & 1 & .25 & 1.5 \\ C & 2.0 & 4.0 & 1 & 5.0 \\ D & .50 & .67 & .20 & 1 \end{array}$$

This change results in the reduction of the Inconsistency Index from 0.94 to 0.27. The lower value indicates a more consistent evaluation of the relative size of the user stories. The new Inconsistency Index yields a size range for each user story of ± 14%, and the estimator decides to accept the results.

The new total size for the project is 48.7 story points, with individual sizes and standard deviations of:

$$sp_i = \begin{cases} 13.1 \pm 1.83 \\ 5.9 \pm 0.83 \\ 24.7 \pm 3.46 \\ 5.0 \pm 0.70 \end{cases}$$

4. Reducing the number of comparisons with Incomplete Cyclic Designs (ICD)

4.1. Overview

The Full Factorial Paired Comparison method provides a solution to the problem of dealing with inconsistent judgments. The problem is that the method does not scale up as the number of comparisons required grows with the square of the number of user stories being estimated.

To solve it, several authors [9, 10, 15] had proposed the use of Incomplete Cyclic Designs (ICDs) to select a subset of the $n(n-1)/2$ comparisons required by the Full Factorial method. Such designs are called fractional designs because they contain only a fraction of all possible comparisons. The ICD technique is used to select which stories are to be compared. Starting with one user story, successive comparisons are selected in a cyclical fashion using the arithmetic modulo n. This method is further developed in the following sections.

Table 1 shows the results of a series of experiments conducted by Burton [8] to evaluate the impact of a reduced number of comparisons in the reliability of the estimates. The experiment consisted of evaluating the correlation between the results of a Full Factorial estimation with the results of a number of fractional (ICD) designs, each with a different r for two groups with multiple respondents.

The closer to 1 the correlation between the full factorial and the fractional designs and the lower the spread between the lowest and the mean correlations, the more accurate and precise the values estimated using the ICD were.

The dataset used in the experiment consisted of 21 different concepts for which the test subjects needed to quantify their semantic similarity. The Full Factorial design required 210 comparisons. The experiment showed that a fractional design with only 63 comparisons displayed a high correlation (0.80 to 0.96) between the similarities estimated by the two methods.

Table 1 Results of the Burton experiments

Number of comparisons in which each concept was included (r)	Number of comparisons with respect to the complete design	Number of comparisons in the Full Factorial design	Number of comparisons in the ICD	Lowest correlation with results from the complete design	Mean correlation with results from the complete design	Comments
2	10%	210	21	.46	.79	Mean correlation is acceptable, but worst-case correlation is too low
4	20%		42	.58	.95	
6	30%		63	.80	.96	Mean correlation and worst-case correlation are acceptable
8	40%		84	.97	.98	Almost as good as the complete design

4.2. The Fractional Paired Comparison method

The Fractional Paired Comparison method requires that: (1) we decide which comparisons to make; and (2) we compensate for the missing values. This adds two new activities to the original process: Generate Incomplete Cyclic Designs and Impute Missing Values (see Figure 4), which are explained in later sections.

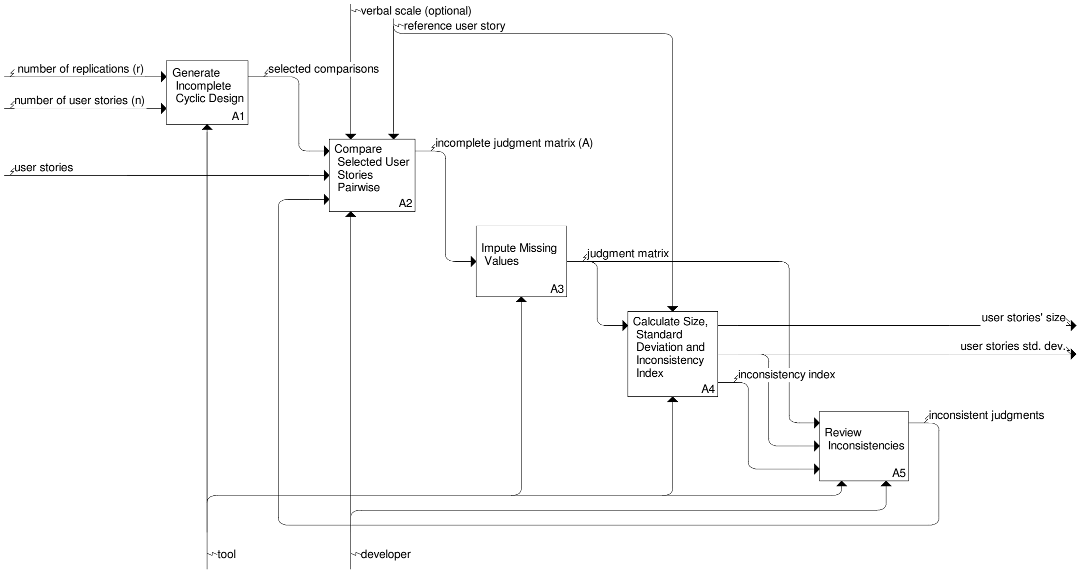


Figure 4 The Fractional Paired Comparison method. Each user story is only compared to those indicated by the ICD.

4.3. Generating Incomplete Cyclic Designs (ICD)

The proposed Incomplete Design Cycle (ICD) construction process starts by arranging, in a random order, the user stories along a circle, and joining adjacent user stories with a line. Each line corresponds to a comparison.

The design generated in this way (see Figure 5.a) consists of a total of 7 comparisons, with each user story appearing in two comparisons, one with the user story to its left and the other with the one to its right. The design's distance s is the minimum number of hops along the circle needed to reach the stories being compared. In Figure 5.a, $r = 2$ and $s = 1$.

Additional designs (see Figures 5.b and 5.c) are generated by increasing the distance between the user stories compared. ICDs with a higher r are obtained by merging simpler designs, as shown in Figure 5.d, which results from the juxtaposition of the designs in Figures 5.a and 5.c.

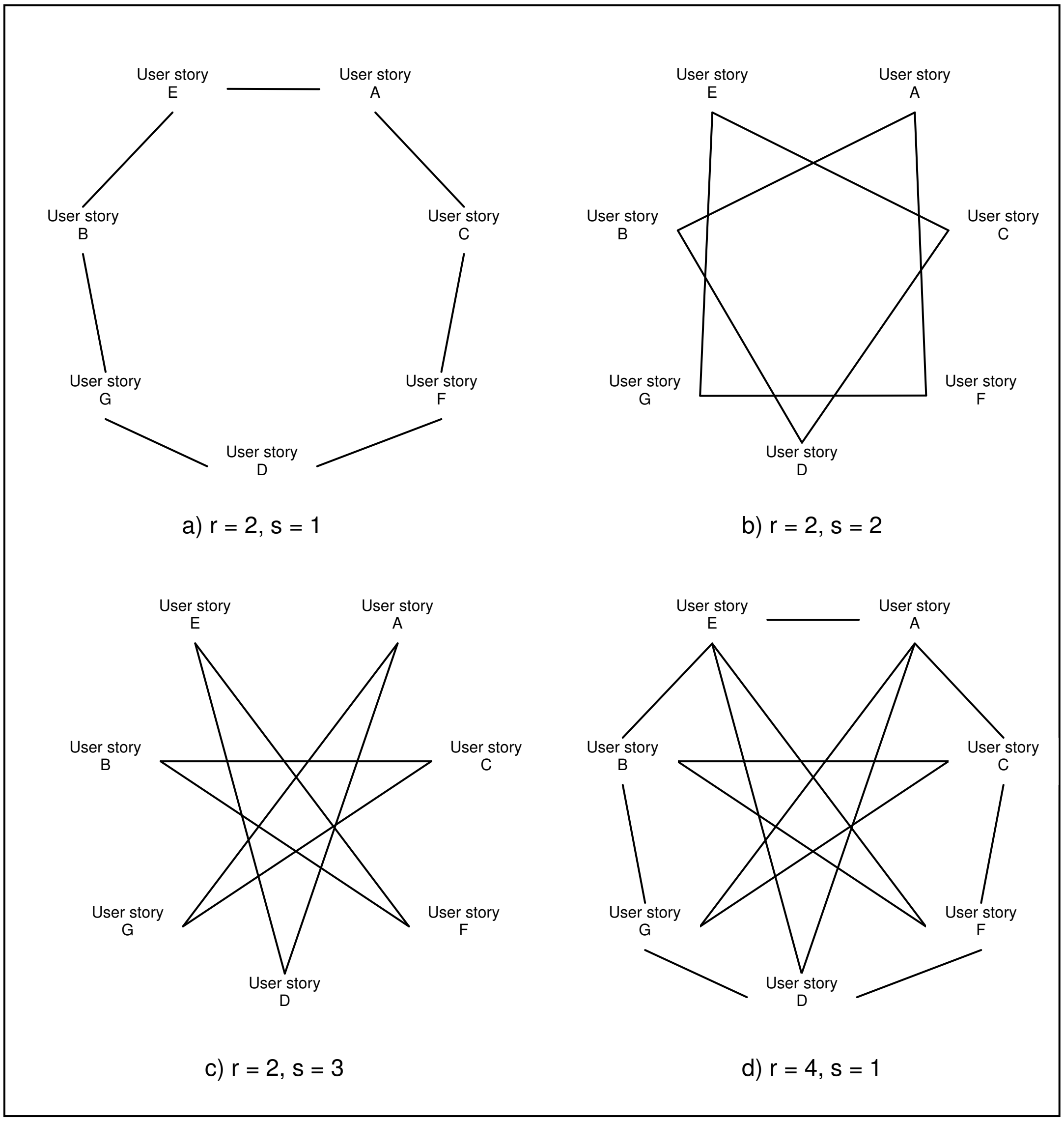


Figure 5 Four different Incomplete Cyclic Designs.

Note 1: Designs a, b, & c are created by increasing the distance (s) between user stories.

Note 2: Design d is the result of merging designs a & c.

To operationalize the generation of ICD, we use an adjacency matrix $G^{n \times n}$ and the algorithm in Figure 6. Table 2 shows the matrix representation of the designs in Figure 5.

```
1. Number the user stories to be sized $0,1,2,...,n-1$
   Create a matrix $G^{nxn}$; initially all the elements of the matrix are $False$
   $G^{nxn} = \begin{cases} g_{ij} = True \text{ if the comparison } a_{ij} \text{ is to be included in the design} \\ g_{ij} = False \text{ otherwise} \end{cases}$
2. For $s = 1$ to $\lceil r/2 \rceil$                    '$r$ is the desired replication factor
   2.1. For $i = 0$ to $n-1$
        2.1.1. $j = (s+i) \bmod n$
        2.1.2. If $Not(g_{ji})$ then $g_{ij} = True$    'this check prevents the inclusion of an
   2.2. Next $i$                                          'element if its reciprocal is already included
3. Next $s$
```

Figure 6 ICD-generating algorithm

Table 2 Matrix representation of the designs in Figure 5

User story	A	C	F	D	G	B	E
A		a, d	b	c, d			
C			a, d	b	c, d		
F				a, d	b	c, d	
D					a, d	b	c, d
G	c, d					a, d	b
B	b	c, d					a, d
E	a, d	b	c, d				
(a) r = 2, s = 1; (b) r = 2, s = 2; (c) r = 2, s = 3; (d) r = 4							

4.4. Imputing missing values

Before we can calculate the size of the user stories, we need to impute the missing values of the judgment matrix, that is, those corresponding to the comparisons that were skipped by the design, with a representative value. Note that the assignments $a_{ii} = 1$ and $a_{ji} = 1/a_{ij}$, the elements on the principal diagonal and the reciprocal of the judgments respectively, are not missing values and should be made before the imputing calculations are performed.

As the comparisons to be skipped were selected at random by the ICD construction procedure, we can impute them with the mean value [16, 17] of the row to which they would have belonged using the algorithm in Figure 7.

```
Assume the existence of a judgment matrix $A^{n \times n}$ filled according to the ICD represented
by the matrix $G^{n \times n}$ whose missing $a_{ij}$ values are zero

1. For $i = 1$ to $n$
    1.1. $meanvalue = 1; \quad c = 0;$
    1.2. For $j = 1$ to $n$
        1.2.1. If $a_{ij} \neq 0$ then
            1.2.1.1. $meanvalue = meanvalue \times a_{ij}$
            1.2.1.2. $c = c + 1$
        1.2.2. Endif
    1.3. Next $j$
    1.4. $meanvalue = meanvalue^{\frac{1}{c-1}}$    'this is the row's geometric mean, the $c-1$ in
                                                     'the denominator is used to compensate for the
                                                     'fact that an element is always equal to itself
    1.5. For $j = 1$ to $n$
        1.5.1. If $a_{ij} = 0$ then                  'we assume zero to be a missing value
            1.5.1.1. $a_{ij} = meanvalue$
            1.5.1.2. $a_{ji} = 1 / meanvalue$
        1.5.2. Endif
    1.6. Next $j$
2. Next $i$
```

Figure 7 Imputation algorithm.

Because the number of judgments made in an ICD with replication factor r is lower than in the case of the Full Factorial design, we need to change the formula for the Inconsistency Index to reflect the reduced number of degrees of freedom. To do this, we substitute in the denominator in <5> the number of judgments required by the Full Factorial - ($n(n-1)/2$) - with the number of judgments required by an ICD with a replication factor of r- ($r \times n / 2$).

<12> $$InconsistencyIndex = \sqrt{\frac{2\sum_{i<j}^{n}\left(\ln a_{ij} - \ln\frac{mrs_i}{mrs_j}\right)^2}{rn - 2n + 2}}$$

The computation of the standard deviation of the size of the user stories <8> remains unchanged.

5. Empirical verification

In this section, we compare the performance of the Fractional Paired Comparison method with the results generated by the Full Factorial method using a set of 15 user stories (see Table 3) derived from a popular Canadian job board [18]. The user stories are described in Table 3 using the template: “As <role> I would like to <action> so that <benefit>”.

For an evaluation of the paired comparison method against actuals, refer to [7, 19, 20].

5.1. Method set-up

The verification was conducted using two subsets of different sizes to explore the impact of the number of user stories in the reliability of the fractional method. As we only obtained 15 user stories from the website, two of them were included in both sets. The same user story was used as a reference to avoid differences originating from the use of different references. In Table 3, the numbers in the left-hand column indicate which user stories were included in which dataset (1 or 2, or both).

First, a Full Factorial estimation was performed on each dataset by a senior software developer. From these two datasets, the fractional designs were generated by excluding from the calculations those values that would have been skipped by the ICDs. This approach allowed us to control for the judgment errors associated with repeated questioning.

Figure 8 shows the data and the comparisons included in the first dataset for three different designs with r = 6 (full factorial – 100% of the comparisons), r = 4 (66% of the comparison), and r = 2 (33% of the comparisons). In Figure 8, only the shaded values need to be provided by the estimator. The user story ‘Notification’ is the reference user story, and its predetermined size is 10 story points.

The Full Factorial design required 21 comparisons, while the first and second ICDs required 14 and 7 respectively. The results are shown in Table 4.

The second dataset consisted of 10 user stories, which were estimated using a Full Factorial design and 4 different ICDs. The results are shown in Table 5.

Note that, for both datasets, the minimal replication ICDs, $r = 2$, correspond to the simple triangulation procedure proposed in the Agile literature.

Table 3 User stories derived from the job board [18]

Dataset (1 or 2)	Role	Action	Benefit
1	Job seeker	Login	I can use the system capabilities reserved for registered job seekers.
1		Logout	…to end a session and protect my data from being accessed by unauthorized people.
1,2		Register	I can make my data available to headhunters and use the system capabilities reserved for registered job seekers.
1		Search job announcements	I can find selected postings based on keywords or criteria, such as job category, location, industry, and city.
1		Create career alert	I will get email notifications whenever a new announcement matching the search criteria is first posted.
2		Suspend career alert	I will not receive notifications without deleting the career alert.
2		Delete career alert	I will not receive further notifications.
1		Upload resume	It can be searched and read by recruiters.
2		Delete resume	It is not longer available to recruiters.
2	Recruiter	Login	I can use the system capabilities reserved for registered recruiters.
2		Logout	…to end a session and protect my data from being accessed by unauthorized people.
2		Post	I can post a new job announcement.
2		Edit	I can modify an existing job announcement.
2		Delete	I can delete an existing job announcement.
1,2	System owner	Notify	I can email all job seekers re new postings, according to their career alert status.

Story Points	User Story	Registration	Notification	Create Alert	Search jobs	Login (job seeker)	Upload resume	Logout (job seeker)
	Registration		1.5	2	3			
10	Notification			1.5	2	3		
	Create alert				1.5	2	5	
	Search jobs					1.5	3	4.0
	Login job seeker	0.25					2.0	2.5
	Upload resume	0.11	0.17					1.2
	Logout job seeker	0.10	0.14	0.17				

a) Full Factorial design – r = 6

Story Points	User Story	Registration	Notification	Create Alert	Search jobs	Login (job seeker)	Upload resume	Logout (job seeker)
	Registration		1.5	2				
10	Notification			1.5	2			
	Create alert				1.5	2		
	Search jobs					1.5	3	
	Login job seeker						2.0	2.5
	Upload resume	0.11						1.2
	Logout job seeker	0.10	0.14					

b) Fractional design – r = 4

Story Points	User Story	Registration	Notification	Create Alert	Search jobs	Login (job seeker)	Upload resume	Logout (job seeker)
	Registration		1.5					
10	Notification			1.5				
	Create alert				1.5			
	Search jobs					1.5		
	Login job seeker						2.0	
	Upload resume							1.2
	Logout job seeker	0.10						

c) Fractional design -— $r = 2$

Figure 8 Empirical verification using three different designs for dataset 1

Note 1: Figure 8a) Design with $r = 6$ (full factorial – 100% of the comparisons),

Note 2: Figure 8b) Design with $r = 4$ (66% of the comparisons)

Note 3: Figure 8c) Design with $r = 2$ (33% of the comparisons).

Table 4 Dataset 1: Estimation results of 7 user stories for 3 different replication factors

User Story	r = 6 (Full Factorial) 21 comparisons		r = 4 14 comparisons		r = 2 7 comparisons	
	Estimated Story Points	Std. Dev.	Estimated Story Points	Std. Dev.	Estimated Story Points	Std. Dev.
Registration	14.4	0.3	18.8	4.97	31.9	20.86
Notification (Reference)	10.0	0.2	10.0	2.65	10.0	6.54
Create Alert	7.4	0.17	5.7	1.51	6.8	4.45
Search jobs	5.0	0.11	4.8	1.27	7.5	4.9
Login job seeker	3.4	0.08	4.5	1.19	9.0	5.88
Upload resume	1.6	0.04	3.0	0.79	8.6	5.62
Log out job seeker	1.3	0.0	2.7	0.71	8.0	5.23
Project Total	43.1		49.4		81.7	
Inconsistency Index	0.06		0.70		1.73	

Table 5 Dataset 2: Estimation results of 10 user stories for 5 different replication factors

User story	r = 9 (Full Factorial) 90 comparisons		r = 8 40 comparisons		r = 6 30 comparisons		r = 4 20 comparisons		r = 2 10 comparisons	
	Estimated Story Points	Std. Dev.	Estimated Story Points	Std. Dev.	Estimated Story Points	Std. Dev.	Estimated Story Points	Std. Dev.	Estimated Story Points	Std. Dev.
Registration	13.0	0.74	13.5	1.49	14.2	2.69	16.4	4.20	29.7	17.94
Notification (Reference)	10.0	0.57	10.0	1.11	10.0	1.90	10.0	2.56	10.0	6.04
Post announcement	6.1	0.35	6.0	0.66	5.2	0.99	4.1	1.05	5.4	3.26
Edit announcement	4.4	0.25	4.3	0.48	3.7	0.70	3.6	0.92	6.7	4.05
Login recruiter	3.0	0.17	2.8	0.31	2.7	0.51	3.1	0.79	7.0	4.23
Suspend alert	2.3	0.13	2.5	0.28	2.5	0.47	3.3	0.85	7.8	4.71
Delete alert	1.9	0.11	2.1	0.23	2.7	0.51	3.8	0.97	8.5	5.13
Delete announcement	1.5	0.09	1.7	0.19	2.4	0.46	3.8	0.97	8.4	5.07
Delete resume	1.3	0.07	1.5	0.17	2.2	0.42	3.7	0.95	8.7	5.25
Logout recruiter	1.0	0.06	1.2	0.13	1.8	0.34	3.1	0.79	7.5	4.53
Project Total	44.6		45.7		47.5		54.9		99.7	
Inconsistency Index	0.18		0.35		0.60		0.81		1.91	

5.2. Discussion of results

Each ICD's performance was evaluated using the Mean Magnitude Relative Error (*MMRE*) and the Predictive Quality Indicator (*Pred*) at the user story level and at the project levels. At the user story level, *MMRE* and *Pred* were calculated by comparing the estimated size of the user stories for a given replication against the value estimated by the full factorial method, and, at the project level, we compared the sum of the sizes of all the user stories in the project.

Evaluating the method on these two levels was important, because the planning of each iteration requires not only that the overall project size be reliable, but also that the estimation of each user story be acceptable.

The results were considered acceptable when their *MMRE* was less than or equal to 0.25 and their *Pred(.25)* was greater than or equal to 0.75 [6, 21].

Tables 6 and 7 summarize the performance for the estimation results presented in Tables 4 and 5. The values obtained show that the fractional designs produce acceptable results at the project level with very low r (r =4), but that good estimates at the user story level required higher degrees of replication or lower Inconsistency Index values. As expected, the influence of inconsistent judgments increased with a reduction in the number of comparisons, and this resulted in higher *MMREs* and lower *Pred(.25)s* in the experimental situation.

Note that, as a consequence of retaining the values from the Full Factorial design to control for judgment error in the experiment design, we did not correct any values to obtain an Inconsistency Index closer to the recommended 0.35. Had we allowed ourselves to perform one or two amendments, as we would normally do in practice, we would have brought the index down and improved the *MMREs* and *Pred(.25)s* in the low-replication estimations. Reasoning along this line, triangulating against two other user stories ($r=2$) would require almost perfect consistency on the judgments rendered for the extra comparisons to increase the estimate's reliability.

Table 6 Method evaluation at the user story level

Replication Factor (r)	Inconsistency Index	$MMRE^*$	$Pred(0.25)^*$	Comments
First dataset – 7 user stories				
4	0.70	0.45	0.33	MMRE and PRED do not meet the established criteria. Reducing the Inconsistency Index would help improve both measures, but, under the stated conditions, the method does not produce acceptable results at the user story level. Notice that the experiment with r = 2 corresponds to the triangulation approach recommended in the Agile literature.
2	1.73	2.10	0.17	
Second dataset – 10 user stories				
8	0.35	0.09	1	Estimates for individual user stories are acceptable.
6	0.60	0.34	0.56	As the number of comparisons is reduced, the MMRE and PRED for individual user story estimates start to deteriorate.
4	0.81	0.84	0.33	
2	1.91	2.85	0.11	Estimates for individual user stories are not acceptable. Note that this is the case for triangulation against 2 other user stories.

* The denominator used in both calculations is the number of user stories – 1, to account for the reference element.

$$MMRE = \frac{\sum abs\left(x_i - \hat{x}_i\right) / x_i}{n}$$

$Pred(0.25) = k / \left(n-1\right)$ is the proportion of observations (k) that fall within 25% of the actual.

Table 7 Method evaluation at the project level

Projects included in the calculation	$MMRE$	$Pred(0.25)$	Comments
All projects	0.43	0.67	MMRE and PRED do not meet the stated criteria.
Excluding projects with r = 2	0.11	1	By excluding the projects with a low replication factor, MMRE and PRED are brought to acceptable levels. Estimates produced with r > 2 are adequate at the project level.
Only projects with r = 2	1.06	0	This corroborates the comments made above.

6. Summary

Agile estimation approaches usually start by sizing the user stories to be developed by comparing them to one another. Various techniques, with varying degrees of formality, have been proposed by the Agile community to conduct the comparisons – plain contrasts, triangulation, planning poker, and voting. This article adds to these techniques by proposing the use of a modified paired comparison method, in which a reduced number of comparisons is selected according to an Incomplete Cyclic Design.

An empirical verification of this proposal was conducted using two datasets, showing that the proposed method produces good estimates, even when the number of comparisons is reduced by half those required by the original formulation of the paired comparison method. A byproduct of the evaluation is the conclusion that the simple triangulation advocated in the Agile literature does not to automatically result in more reliable estimates. Low-replication comparisons require a high degree of consistency among judgments.

To confirm these results the authors plan to conduct follow-up studies in their respective organizations.

Although we have used story points to illustrate the article, the techniques described could be equally applied using different size units such as lines of code or to the estimation of durations using ideal days or effort.

Those seeking to introduce the method in their organizations must be aware that, while people readily buy into the idea of comparing user stories to one another, they tend to become discouraged by the underlying mathematics. Therefore, two things are required for a successful deployment of the method: first, the careful selection of the number of details to be included in training presentations and process documentation; and second, the development of a simple spreadsheet to support these calculations.

7. Acknowledgment

We would like to thank Ahmed Bedhief for patiently going through all the comparisons required by the Full Factorial experiment.

8. References

[1] http://en.wikipedia.org/wiki/Story_points, "Story Points," Wikipedia, 4/5/2009.

[2] N. Dalkey, "The Delphi Method: An Experimental Study of Group Opinion," Rand Corporation, Santa Monica1969.

[3] M. Cohn, *Agile Estimating and Planning*. Upper Saddle River, NJ: Prentice Hall, 2006.

[4] M. Cohn, "Tutorial on Agile Estimating and Planning," in *Agile 2006 Conference*. vol. 2008 Minneapolis: Mountain Goat Software, 2006.

[5] G. Bozoki, "An Expert Judgment Based Software Sizing Model," *Journal of Parametrics,* vol. XIII, May 1993.

[6] E. Miranda, "Improving Subjective Estimates Using Paired Comparisons," *IEEE Software,* vol. 18, pp. 87-91, Jan/Feb 2001.

[7] M. Shepperd and M. Cartwright, "Predicting With Sparse Data," *IEEE Transactions on Software Engineering,* vol. 27, pp. 987 - 998, Nov 2001.

[8] M. L. Burton, "Too Many Questions? The Uses of Incomplete Cyclic Designs for Paired Comparisons," *Field Methods,* vol. 15, pp. 115-130, May 2003.

[9] I. Spence, "Incomplete Experimental Designs for Multidimensional Scaling," in *Proximity and preference: Problems in the multidimensional analysis of large data sets*, R. Golledge and J. Rayner, Eds. Minneapolis: University of Minnesota Press, 1982, pp. 29-45.

[10] I. Spence and D. Domoney, "Single Subject Incomplete Designs for Nonmetric Multidimensional Scaling," *Psychometrika,* vol. 39, 1974.

[11] J. Barzilai, W. Cook, and B. Golany, "Consistent Weights For Judgments Matrices Of The Relative Importance Of Alternatives," *Operations Research Letters,* vol. 6, Jul 1987.

[12] G. Crawford and C. Williams, "A Note on the Analysis of Subjective Judgment Matrices," *Journal Of Mathematical Psychology,* vol. 29, pp. 387-405, 1985.

[13] J. Aguaron and J. Moreno-Jimenez, "The Geometric Consistency Index: Approximated Thresholds," *European Journal of Operational Research* vol. 147, pp. 137-145, 2003.

[14] J. Hihn and K. Lum, "Improving Software Size Estimates by Using Probabilistic Pairwise Comparison Matrices," in *10th IEEE International Symposium on Software Metrics (METRICS'04)*, 2004, pp. 140 - 150.

[15] H. David, "Cyclic Designs," in *Encyclopedia of Statistical Sciences*, 2007 ed, S. Kotz, Ed.: John Wiley & Sons, 1965.

[16] I. Myrtveit, E. Stensrud, and U. Olsson, "Analyzing Data Sets with Missing Data: An Empirical Evaluation of Imputation Methods and Likelihood-Based Methods," *IEEE Transactions on Software Engineering,* vol. 27, pp. 999-1013, Nov 2001.

[17] Q. Song, M. Shepperd, and M. Cartwright, "A Short Note on Safest Default Missingness Mechanism Assumptions," *Empirical Software Engineering,* vol. 10, pp. 235-243, Apr 2005.

[18] www.workopolis.com, "Job Board," Toronto Star Newspapers, 2008.

[19] G. Bozoki, "Performance Simulation of SSM," in *ISPA 13th Annual Conference*, 1991.

[20] E. Miranda, "Evaluation of the Paired Comparisons Method for Software Sizing," in *22nd International Conference on Software Engineering*, Limerick, Ireland, 2000, pp. 597-604.

[21] S. Conte, H. Dunsmore, and V. Shen, *Software Engineering Metrics and Models*: Benjamin-Cummings Publishing Company, 1986.