Empirical Analysis of Factors Affecting Confirmation Bias Levels of Software Engineers

Journal Item

Noname manuscript No.
(will be inserted by the editor)

Empirical Analysis of Factors Affecting Confirmation Bias Levels of Software Engineers

Gul Calikli · Ayse Bener

Received: date / Accepted: date

Abstract Confirmation bias is defined as the tendency of people to seek evidence that verifies a hypothesis rather than seeking evidence to falsify it. Due to the confirmation bias, defects may be introduced in a software product during requirements analysis, design, implementation and/or testing phases. For instance, testers may exhibit confirmatory behaviour in the form of a tendency to make the code run rather than employing a strategic approach to make it fail. As a result, most of the defects that have been introduced in the earlier phases of software development may be overlooked leading to an increase in software defect density. In this paper, we quantify confirmation bias levels in terms of a single derived metric. However, the main focus of this paper is the analysis of factors affecting confirmation bias levels of software engineers. Identification of these factors can guide project managers to circumvent negative effects of confirmation bias; as well as providing guidance for the recruitment and effective allocation of software engineers. In this empirical study, we observed low confirmation bias levels among participants with logical reasoning and hypothesis testing skills.

Keywords Confirmation bias · Human factors · Software psychology

1 Introduction

In cognitive psychology, confirmation bias is defined as the tendency of people to seek evidence that could verify their hypotheses rather than seeking evidence

G. Calikli
Department of Computing
Faculty of Maths, Computing and Technology
The Open University, Milton Keynes, United Kingdom E-mail: gul.calikli@open.ac.uk

A. Bener
Data Science Laboratory
Mechanical and Industrial Engineering
Ryerson University, Toronto, Canada E-mail: ayse.bener@ryerson.ca

that could falsify them. The term confirmation bias was first used by Peter Wason in his rule discovery task [1] and later in his selection task [2].

One can observe the effects of confirmation bias during any phase of the software development process. Due to confirmation bias, conditions which have the potential to make the software fail may be overlooked during requirements analysis and design phases of SDLC. This, in turn, leads to an increase in software defect density. In addition to the defects which are introduced during early phases of the SDLC, significant number of defects may also be introduced and/or overlooked during the implementation phase. Due to confirmation bias, developers might prefer only the unit tests to make their code run rather than unit tests, which aim to make their code fail. We argue that testers with high confirmation bias levels will employ testing strategies, which do not include adequate attempts to fail the code. As a result, most of the defects that have propagated from earlier phases of the SDLC (e.g., requirements analysis, design, and implementation phases) may be overlooked leading to a significant increase in the post-release defect density.

To the best of our knowledge, Stacy and MacMillian are the two pioneers who recognized the potential effects of confirmation bias on software engineering [3]. There is also empirical evidence showing the existence of confirmation bias among testers. According to the results obtained by Teasley et. al, in both naturalistic and laboratory studies of test case selection, testers are four times more likely to choose positive tests (i.e., tests that make the code run) than negative tests (i.e, tests that break the code).

In our previous empirical studies, we found a positive correlation between developers' confirmation biases and software defect density [4], [5]. We also discovered that more post-release defects are overlooked in the modules that are tested by software engineers with high confirmation bias [9]. In another research study, we compared confirmation bias values of developer groups from three different projects and two different companies [6]. We defined developer groups as the set of developers who contribute to the development of a common set of source code files. Developer groups were identified for each project by mining log files that were obtained from version management systems. Two of the project groups, Telecom1 and Telecom2 belonged to a large-scale telecommunication company and the third project group ERP belonged to an independent software vendor specialized in Enterprise-Resource Planning. Developer groups of Telecom2 and ERP project groups had confirmation bias values that were much closer to ideal values compared to those of developer groups belonging to project group Telecom1. As a result of the interviews we conducted with developers and projects managers, we discovered that negative tests were part of testing routine of developers in project groups Telecom2 and ERP besides positive tests, while only positive tests were conducted by developers in project group ERP.

In our previous studies, we defined a methodology to quantify confirmation bias levels that resulted in the formation of a confirmation bias metrics set [4],[5], [7], [8], [9]. Having found a correlation between confirmation biases of developers and software defect density, we built predictive models to iden-

Table 1 The hypotheses which are derived from the objectives of this research

Hypothesis	Explanation
H_0^1	Confirmation bias levels of software engineers are not affected by their educational background (undergraduate level).
H_0^2	Confirmation bias levels of software engineers are not affected by their level of education (Bachelor's vs. Master's degree).
H_0^3	Logical reasoning and strategic hypothesis testing skills are not differentiating factors in low confirmation bias levels.
H_0^4:	Confirmation bias levels of a software engineer is not related to his/her role (e.g. analyst, developer, tester) in the software development process.
H_0^5	Confirmation bias level of a software engineer is not affected by his/her industrial software development experience (in years).
H_0^6	Confirmation bias level of a software engineer is not affected by his/her industrial software testing experience (in years).
H_0^7	The methodology that is used for the development of a software product (e.g. incremental, agile and TDD) does not affect confirmation bias levels of the software engineers.
H_0^8	There is no significant difference between confirmation bias levels of software engineers who work for large-scale software development companies and confirmation bias levels of those who work for Small-Medium Enterprises (SMEs).

tify defect prone parts of software. The performance of our prediction model, which we built by using confirmation bias metrics was comparable with the performance of the models, which we built by using product (i.e. static code) and process (i.e. churn) metrics [5].

In order to circumvent negative effects of confirmation bias as well as providing guidance to recruit new software engineers and to allocate existing ones to positions where they can perform most effectively, it is crucial to identify factors having significant impact on confirmation bias. In this paper, we present a methodology to quantify confirmation bias levels in terms of a single derived metric and investigate the factors which affect confirmation bias levels of software engineers.

The rest of the paper is organized as follows: In Section 2, we explain our research goal and the corresponding hypotheses. Our methodology to quantify/measure confirmation bias levels of software engineers is explained in Section 3. We give the details of our empirical analysis in Section 4. Results of our empirical study and discussions are presented in Section 5. Section 6 addresses threats to validity. Finally, we conclude and mention future work in Section 7.

2 Goals and Hypotheses

The goal of this research is to investigate the factors which significantly affect confirmation bias levels of software engineers. The hypotheses that are derived from the factors of interest are listed in Table 1 as well as being mentioned in this section. During our empirical analysis, in addition to the factors of interest, confounding factors also had to be taken into account. We defined "confounding factor" as an extraneous variable (i.e., a variable that is not the focus of the study) that is statistically related to one or more of the independent variables whose effects on the response variable (i.e., dependent variable) are analyzed. We explored the factors of interest and the confounding factors under the following three categories:

2.1 Characteristics of the Software Engineers

The characteristics of software engineers that we took into consideration are *education*, *job title*, *years of experience in development* and *years of experience in testing*, respectively. As confounding factors, we included *age* and *gender* into our empirical analysis.

In cognitive psychology literature, there are studies indicating the effects of *education* and *profession* (i.e., *job title*) on confirmation bias. In a study conducted by Inglish and Simpson [10], mathematics students, mathematicians and history students were compared. It turned out that mathematicians and mathematics students employ more disconfirmatory strategies compared to history students and hence are less prone to confirmation bias. Similar studies were conducted with samples from engineers, scientists and statisticians, who were found to be less prone to confirmation bias compared to the subjects from the rest of the population [11], [12]. In this paper, one of our goals is to analyze the effect of *education* and *professions* on confirmation bias within the context of software engineering. Therefore, we aim to test the validity of the following hypotheses:

H_0^1 : Confirmation bias levels of software engineers are not affected by their educational background (undergraduate study).

H_0^2 : Confirmation bias levels of software engineers are not affected by their level of education (Bachelor's and Master's degree).

Information about the effect of education on confirmation bias levels of software engineers may guide the human resources in the recruitment process. Moreover, we would like to investigate whether obtaining logical reasoning and hypotheses testing skills lead to lower confirmation bias levels. For this purpose, we would like to test the validity of the following hypothesis:

H_0^3 : Logical reasoning and strategic hypotheses testing skills are bot differentiating factors in low confirmation bias levels.

In the long run, information, which we gain testing hypotheses H_0^1, H_0^2 and H_0^3 may lead to the modification of cirriculum to integrate de-biasing techniques into the fields of education, if necessary. Moreover, further investigation of practices and problem solving techniques, which are unique to certain professions and areas of expertise may guide us to develop certain de-biasing techniques. There are opportunities to use recent results from meta-cognition research to circumvent negative effects of confirmation bias and other cognitive biases. Although meta-cognitive skills can be taught [20], these are not necessarily automatically transferred to another context. As Mair and Shepperd suggest it is required to design context-specific materials and techniques for use by software professionals [21]. Moreover, there are techniques and tools, which aid circumvention of the negative effects of cognitive biases, such as decision analysis, expert systems and various programming techniques [22]. Some of these techniques and tools are part of the daily practice of specific areas of expertise in software engineering (i.e., developer, tester, analyst). In other words, there might be a correlation between the role of a software engineer (i.e., developer, tester, analysts) and confirmation bias levels. Therefore, it is crucial to analyze the effects of these specific roles on confirmation bias within the context of software engineering by testing the validity of the following hypothesis:

H_0^4 : Confirmation bias level of a software engineer is not related to his/her role in the software development (i.e., developer, tester and analysts).

We took experience in development/testing into consideration as another factor, since information about whether expertise level affects cognitive biases has the potential to give us insight about circumventing the negative effects of confirmation bias. In cognitive psychology literature, most studies have shown that experts also have cognitive biases as much as non-experts. Kahneman found out that highly experienced financial managers performed no better than chance compared to less experienced ones due to confirmation bias [13]. In another study, expertise level of stock analysts and traders made them highly resistant to signals that did not conform to their beliefs while making predictions about the stock market [14]. Existence of cognitive biases have also been observed in scientists as well as experts in statistics and logic [15]. An accumulating body of research on clinical judgment also report existence of confirmatory behaviour [16], [17]. In their laboratory and field studies, Teasley et al. also observed that experienced programmers/testers exhibit confirmatory behaviour as much as novices [18], [19]. Our previous findings within the context of software engineering were inline with all these above mentioned studies. In our previous research, we could not find any significant difference between confirmation bias levels of experienced software developers/testers and less experienced ones [7], [8], [9]. In this paper, we extend our dataset in order to test the validity of our claim stating that experience in development and testing does not have a significant positive impact on confirmation bias. Therefore, we aim to test the validity of the following hypotheses:

H_0^5 : Confirmation bias level of a software engineer is not affected by his/her industrial software development experience (in years).

H_0^6 : Confirmation bias level of a software engineer is not affected by his/her industrial software testing experience (in years).

If experience in testing and development do not have a significant effect on circumventing negative effects of confirmation bias, then it might be crucial to design and conduct training sessions for software professionals. Such training sessions should inherit findings from research in meta-cognition and they can only be designed by multi-disciplinary teams that include cognitive psychologists as well as computer scientists [21].

2.2 Characteristics of the Software Product

Among software product characteristics, the factor which we investigated is the *software development methodology*. The confounding factors are the type of the *development language* and the level of *domain expertise* which is required to take part in the development process of a software product.

Some software development methodologies such as TDD (Test Driven Development) [23] involve de-biasing strategies. Test Driven Development (TDD) helps to understand the underlying requirements and to consider test case scenarios before the implementation of the code itself. Practicing TDD is likely to result in low confirmation bias levels, since it helps the developer to have a critical view of his/her own code. In this paper, we investigate possible effects of incremental, agile and TDD methodologies on confirmation bias levels of software engineers. In other words, we test the validity of the following hypothesis:

H_0^7 : The methodology that is used for the development of a software product (e.g., incremental, agile and TDD) does not affect confirmation bias levels of software engineers.

2.3 Characteristics of the Software Company

In order to investigate the possible effects of *company size* on confirmation bias, data is collected from both large-scale software development companies and Small-Medium Enterprizes (SMEs). One of the companies that took part in this research is located in Canada, while the rest of the companies are located in Turkey. Therefore, we took *geographical location* (i.e.*country*) into consideration as a confounding factor. Moreover, each software development company, which took part in this research, is either in-house development company or an independent software vendor (ISV). For this reason, we included the type of a software development company regarding its target customers as another confounding factor.

During our field studies [4], [5], [7], [8], [9], we observed that large scale companies are process-driven, while individual performance of software engineers play a significant role in the success of a software product that is developed by a SME. Confirmation bias is one of the characteristics of an individual. Therefore, it is probable that software quality in a SME might be affected by confirmation bias more, whereas having defined software development processes in large-scale companies may assist circumvention of the negative effects of confirmation bias. In order to test this hypothesis, in our previous study [8], we compared confirmation bias levels of software engineers working for a large-scale company with confirmation bias levels of software engineers who work for SMEs. Results of our previous study did not indicate any significant effect of company size on confirmation bias levels of software engineers. In this research, having extended the dataset, we test the validity of our previous findings and hence that of the following hypothesis:

H_0^8 : There is no significant difference between confirmation bias levels of software engineers who work for large-scale software development companies and confirmation bias levels of those, who work for SMEs.

3 Measurement/Quantification of Confirmation Bias

Our methodology to measure/quantify confirmation bias consists of the following steps:

3.1 Preparation of the Confirmation Bias Test

The *confirmation bias test*, which we prepared, consists of the interactive question and the written question set. Interactive question is Wason's Rule Discovery Task [1] itself, whereas written question set is based on Wason's Selection Task [2]. Details about these two psychological experiments, which were designed by Wason, can be found in our previous research, [4], [5], [7], [8], [9]. Our previous research [5] also contains a discussion of the analogy of Wason's Rule Discovery Task and Selection Task with software unit testing.

Written question set consists of 8 abstract and 7 thematic questions. Abstract questions are based on the original Wason's Selection Task. In cognitive psychology literature, there are variants of the Selection Task [24], [25], [26], [27], [28], [29], [30], [31], [32]. Thematic questions in our confirmation bias test are based on these psychological experiments which were inspired by Wason's original work [2].

3.2 Administration of the Confirmation Bias Test

We prepared both Turkish and English versions of the confirmation bias test. Details about the standard procedure, which is followed during the admin-

Table 2 List of interactive question metrics

Metric	Explanation	Values for Ideal Case	Values for Worst Case
N_A	Number of rule announcements	1	N_A^{abort}
$Ind_{elim/enum}$	Eliminative/enumerative index by Wason	$Ind_{elim/enum}^{max}$	0
$F_{negative}$	Frequency of negative instances	$F_{negative}^{max}$	0
F_{IR}	Immediate rule announcement frequency	0	F_{IR}^{max}
$avgL_{IR}$	Average length of immediate rule announcements	0	$avgL_{IR}^{max}$
$UnqRs/Time$	Number of unique reasons which are given per unit time	$UnqRs/Time^{max}$	0

istration of the confirmation bias test, was explained in our previous paper [5].

3.3 Definition and Extraction of the Confirmation Bias Metrics

Interactive question metrics and written question metrics are briefly explained in Tables 2 and 3, respectively. Our previous paper [5] contains extensive information about confirmation bias metrics as well as the analogy between software testing and Wason's experiments.

In Tables 2 and 3, ideal case and worst case values for each confirmation bias metric are also listed. Ideal case values are among the indications of low confirmation bias values, while worst case values imply high confirmation bias levels.

High values of the metrics $Ind_{elim/enum}$, $F_{negative}$ and $UnqRs/Time$ are among the indications of low confirmation bias. Wason discovered that participants with high $Ind_{elim/enum}$ and $F_{negative}$ values performed better in his rule discovery task [1]. Our previous empirical findings showed that developers with low $Ind_{elim/enum}$, $F_{negative}$ and $UnqRs/Time$ values are more inclined to select positive tests to verify their code which, in turn, leads to an increase in software defect density [5]. Therefore, as shown in Table 2, we indicate the ideal values of these metrics by $Ind_{elim/enum}^{max}$, $F_{negative}^{max}$ and $UnqRs/Time^{max}$, respectively.

On the contrary, high values of the metrics N_A, F_{IR} and $avgL_{IR}$ are among the indications of high confirmation bias. In our previous empirical analysis, we found a significant correlation between the values of these three metrics and software defect density [5]. In Table 2, the worst case values of the metrics F_{IR} and $avgL_{IR}$ are denoted by F_{IR}^{max} and $avgL_{IR}^{max}$, respectively.

In this research, except for the metric N_A, the maximum value of each metric M (M^{max}) is assigned the corresponding maximum value obtained from the data that has been collected so far [4], [5], [7], [8], [9]. N_A^{abort} is used

Table 3 List of written question set metrics

Metric	Explanation	Values for Ideal Case	Values for Worst Case
S_{Abs}	Score in abstract questions	1	0
S_{Abs}/S_{Th}	Ratio of the score in abstract questions to score in thematic questions		
$ABS_{CompleteInsight}$	Ratio of the number of abstract questions that are answered with *complete insight* to total number of positive abstract questions	1	0
$ABS_{PartialInsight}$	Ratio of the number of abstract questions that are answered with *partial insight* to total number of positive abstract questions	0	1
$ABS_{NoInsight}$	Ratio of the number of abstract questions that are answered with *no insight* to total number of positive abstract questions	0	1
$Th_{CompleteInsight}$	Ratio of the number of thematic questions that are answered with *complete insight* to total number of thematic questions	1	0
$Th_{PartialInsight}$	Ratio of the number of thematic questions that are answered with *partial insight* to total number of thematic questions	0	1
$Th_{NoInsight}$	Ratio of the number of thematic questions that are answered with *no insight* to total number of thematic questions	0	1
$R_{Falsifier}$	Ratio of the number of Reich and Ruth's tendency questions that are answered with *only* falsifying tendency to total number of tendency questions	1	0
$R_{Verifier}$	Ratio of the number of Reich and Ruth's tendency questions that are answered with *only* verifying tendency to total number of tendency questions	0	1
$R_{Matcher}$	Ratio of the number of Reich and Ruth's tendency questions that are answered with *only* matching tendency to total number of tendency questions	0	1

to denote the worst case value of the metric N_A, since a participant aborts the interactive question session in the worst case. In this empirical study, we set N_A^{abort} to be equal to twice the maximum number of rule announcements we have observed so far (i.e. $N_A^{abort} = 2{*}N_A^{max}$).

Finally, written question set metrics can take values in the range $[0, 1]$. Hence, for each written question set metric M_{wr}, $M_{wr}^{max} = 1$ and $M_{wr}^{min} = 0$.

3.4 Deriving a Single Metric to Quantify Confirmation Bias Levels of Software Engineers

In order to quantify confirmation bias, we preferred to define a set of confirmation bias metrics. Our confirmation bias metrics set got its final form

in our previous research [5]. Our goal was to avoid under-representation of "confirmation bias" that is one of the threats to construct validity [33].

Confirmation bias can be represented in the form of a vector cb, where each component cb_i of the vector is one of the confirmation bias metrics that are listed in Tables 3 and 2. For instance, the first component of the vector cb corresponds to the confirmation bias metric N_A, the second component of the vector cb corresponds to the metric $Ind_{elim/enum}$ and the last component of the vector corresponds to the metric $R_{Matcher}$. As a result, the dimension of the vector cb is equal to the total number of confirmation bias metrics ($N = 17$).

However, it is much easier to understand and interpret scalars rather than a set of multiple parameters such as vectors [34]. Interpretation of the results of multi-way ANOVA is also much easier and hence less prone to incorrect interpretations compared to MANOVA which is preferred in the case of multiple response variables. Therefore, we decided to derive a single metric to quantify confirmation bias levels by using confirmation bias metrics. As a result, it was possible to perform multi-way ANOVA test for empirical analysis instead of multi-way MANOVA where multiple response variables are taken into account as well as multiple independent variables (i.e. factors).

We define confirmation bias level of an i^{th} software engineer (CB_i) as the deviation of confirmation metrics values of the i^{th} software engineer from the corresponding ideal metrics values. In order to quantify confirmation bias levels of software engineers in the form of a scalar value (i.e. a single derived metric), we formulated equation 1 as follows:

$$CB^i = D^i / maxDist \tag{1}$$

In equation 1, CB^i stands for the confirmation bias level of the i^{th} software engineer. D^i is used to measure the deviation of confirmation bias metrics values of the i^{th} software engineer (cb^i) from ideal values of the confirmation bias metrics (cb^{Ideal}). We calculate D^i as the Euclidean distance between two vectors cb^i and cb^{Ideal}.

$$D^i = \sqrt{(\sum_{j}^{N} (cb_j^i - cb_j^{Ideal})^2} \tag{2}$$

The members of both vectors cb^i and cb^{Ideal} are the confirmation bias metrics that are listed in Tables 2 and 3, respectively. Values for the components of the vector cb^i are the values of the confirmation bias metrics of the i^{th} software professional. As it can be deduced from Tables 2 and 3, cb^{Ideal} = [1 $Ind_{elim/enum}^{max}$ $F_{negative}^{max}$ 0 0 $UnqRs/Time^{max}$ 1 1 1 1 0 0 1 0 0 1 0 0]. Confirmation bias metrics, which are listed in Table 3 are in the range $[0, 1]$ (i.e., $0 \leq cb_j^i \leq 1$).

In order to guarantee the equivalent contribution of each metric to the calculation of the resulting confirmation bias levels, during the calculation, we also mapped the values for the metrics $Ind_{elim/enum}$, $F_{negative}$ and $UnqRs/Time$

to the range [0, 1], so that the following holds for all components of the vector cb^i: $0 \leq cb^i_j \leq 1$. For this purpose, we divided the metrics values by the corresponding maximum values (i.e. $Ind^{max}_{elim/enum}$, $F^{max}_{negative}$ and $UnqRs/Time^{max}$). As a result, the exact value of the vector cb^{Ideal} that we used during our empirical analysis is cb^{Ideal} = [1 1 1 0 0 1 1 1 1 1 0 0 1 0 0 1 0 0].

In order to map the estimated Euclidean distance D^i to the range [0,1] (i.e., $0 \leq D^i \leq 1$), we divided D^i by $maxDist$, which is the distance between the vectors cb^{worst} and cb^{Ideal}.

$$maxDist = \sqrt{(\sum_{j}^{N}(cb^{worst}_j - cb^{Ideal}_j)^2} \tag{3}$$

The components of the vector cb^{worst} are all the worst case values of the confirmation bias metrics. Hence, $maxDist$ is equal to the maximum possible deviation from the ideal case vector cb^{ideal}. As a result, we can rewrite our formulation in equation 1 as follows:

$$CB^i = \sqrt{\sum_{j}^{N}(cb^i_j - cb^{Ideal}_j)^2 / \sum_{j}^{N}(cb^{worst}_j - cb^{Ideal}_j)^2} \tag{4}$$

According to equation 4, high value of CB^i is an indication of high confirmation bias level, since it implies a high deviation from the ideal case. On the other hand, low value of the derived metric CB^i corresponds to low confirmation bias level which is the desirable case.

In N-dimensional space, confirmation bias value of the i^{th} software engineer is represented by a set of confirmation bias metrics in the form of the vector cb^i_j, while cb^{Ideal}_j stands for the ideal confirmation bias metrics values. By using equation 4, confirmation bias metrics values in N-dimensional space are transformed to a scalar value to measure the confirmation bias level. CB^i is the deviation of the confirmation bias metrics values of the i^{th} software engineer from the ideal confirmation bias metrics values. Therefore, N-dimensional vector cb^{Ideal}_j is transformed to $CB^{ideal} = 0$, while N-dimensional vector cb^{worst}_j is transformed to $CB^{worst} = 1$.

In order to make the interpretation of the physical meaning of the single derived metric CB^i easier, in Figure 1 the vectors cb^i, cb_{ideal} and cb^{worst} are illustrated as two-dimensional vectors instead of using the actual N-dimensional space. For illustrative purposes, let's assume that the first component of the vectors cb^i is F_{IR} and the second component is $Ind_{elim/enum}$ (i.e., $cb^i_1 : F_{IR}$ and $cb^i_2 : Ind_{elim/enum}$). In this two-dimensional case, the Euclidean distance between the ideal case confirmation bias vector (cb^{ideal}) and the worst case confirmation bias vector (cb^{worst}) is $\sqrt{2}$. As shown in Figure 1, if the i^{th} developer has the confirmation bias metric values $cb_1 : F_{IR} = 0.75$ and $cb_2 : Ind_{elim/enum} = 0.9$, then the value of the single derived metric for the i^{th} software engineer is 0.66. In the ideal case, the values of the confirmation bias metrics F_{IR} and $Ind_{Elim/Enum}$ for the i^{th} software engineer should be 0

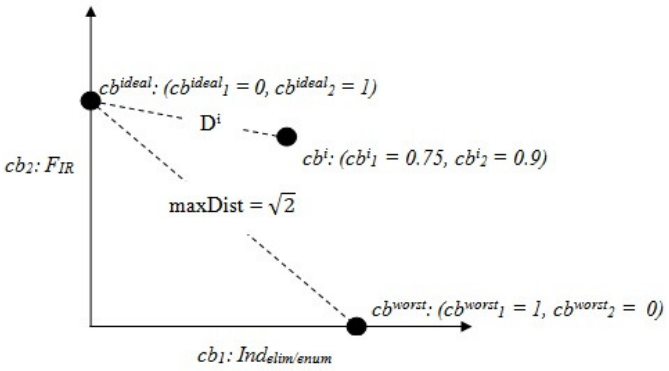

Fig. 1 Two-dimensional Illustration of the Single Derived Metric CB^i.

and 1, respectively, as it is shown in Table 2. Therefore, CB_i is the extent to which confirmation bias metrics values of the i^{th} software engineer deviates from the ideal values of these metrics. For convenience, we normalize the resulting Euclidean distance between cb^i and cb_{ideal} by the maximum possible distance $\sqrt{2}$, which is the Euclidean distance between cb^{ideal} and cb^{worst}. As a result, $0 \leq CB^i \leq 1$. Hence, the lower the value of the single derived metric CB^i, the lower the confirmation bias level, which is the desirable case.

4 Empirical Analysis

4.1 Data

In this research, 174 participants consisting of 18 Computer Engineering PhD candidates and 156 software engineers took part. Data of software engineers are collected from seven different software development companies which are listed in Table 4.

GSM-Company, which is the leading telecommunications (GSM) operator in Turkey, produces in-house software products. Our research covers four project groups within GSM-Company which are GSM-CRM, GSM-Billing and GSM-TDD, respectively. Project group GSM-CRM is responsible from the development of a Customer Relationship Management (CRM) software which serves to launch new and creative campaigns for company's customers. The project group GSM-Billing is responsible from the development of the software which provides billing, charging and revenue collection services. All project groups in GSM-Company use incremental software development methodology, except for the project group GSM-TDD which employs Test Driven Development (TDD) methodology.

Bank-IT is a large scale company, which develops an in-house software for online banking services by using agile development methodology.

Table 4 Characteristics of software development companies

Institution/ Company	Project Groups	ISV vs. In-House	Company Type	Country
GSM-Company	GSM-CRM GSM-Billing GSM-TDD	In-House	Large-Scale Company	Turkey
Bank-IT	Banking	In-House	Large-Scale Company	Turkey
Hi-Tech Corporation	DBMS-Group	ISV	Large-Scale Company	Canada
University	CMPE-Exp	various	various	Turkey
Finance Software Vendor	Finance	ISV	SME	Turkey
ERP-Vendor	ERP	ISV	Large-Scale Company	Turkey
CRM-Vendor	CRM	ISV	SME	Turkey
Business Solution Provider	Mobile	ISV	SME	Turkey

Table 5 Characteristics of software projects

Project Groups	Development Language	Development Methodology	Domain Expertise
GSM-CRM	Java	Incremental	Medium
GSM-Billing	Java	Incremental	Medium
GSM-TDD	Java	TDD	Medium
Banking	Java	Agile	Medium
DBMS	C++	TDD	High
CMPE-Exp	various	various	High/Medium
Finance	Java	Agile	Medium
ERP	Java	Incremental	Medium
CRM	Java	Agile	Medium
Mobile	C♯	Agile	Medium

Hi-Tech Corporation is also a large scale company and it is is located in Canada. The company develops a Database Management System (DBMS) which is highly demanded by industry for the storage and the management of large amount of data.

Participants of the group CMPE-Exp consist of Computer Engineering PhD students at Bogazici University in Turkey and each member of this group has minimum two years of development experience in software industry. Unlike the members of other groups, CMPE-Exp members are not active in the field

anymore, hence they can be defined as "researchers" rather than "software engineers". "Researchers" are the only PhD candidates among the participants of this empirical study and none of the other participants hold any PhD degrees.

Finance Software Vendor is a Small/Medium Enterprize (SME) which develops software for the finance sector such as portfolio, funds and asset management services by employing agile development methodology.

ERP-Vendor is Turkey's largest Independent Software Vendor (ISV) which provides business solutions in the form of Enterprize Resource Planning (ERP) products by using incremental development methodology.

CRM-Vendor and Business Solutions Provider are two medium scale independent software vendors. Project groups within both of these companies employ agile development methodology.

In the development of DBMS at the High-Tech Corporation, theoretical knowledge about the relational database model is essential. On the other hand, the set of functionalities provided by the rest of the software products can be represented in the form of rule-based systems. Therefore, compared to the software products which are developed by other companies in the dataset, high level of domain expertise is required in the development of DBMS.

Development language that is used for the implementation of DBMS is C++, while *C#* is preferred by the project group Mobile. The remaining project groups use Java as the development language.

4.2 Methodology for the Empirical Analysis

We performed a multi-way Analysis of Variance (ANOVA) test to analyze the effects of multiple factors on the mean values of confirmation bias levels. For this purpose, we used Statistics Toolbox of MATLAB.

In order to measure confirmation bias level of each software professional, we derived a single metric by using Equation 1. This single metric is derived from the confirmation bias metrics that are listed in Tables 2 and 3, respectively. Confirmation bias level is the dependent variable, while the independent variables are the factors which we classified under the following three main categories.

4.2.1 Characteristics of the Software Engineers

In the multi-way ANOVA test, *age*, *years of experience in development* and *years of experience in testing* are treated as continuous predictors, whereas *gender*, *education* and *job title*, are treated as categorical predictors. We defined two factors related to *education* which are *undergraduate degree* and *graduate degree*, respectively. The levels of the categorical factor *undergraduate degree* is based on the categorization scheme that is shown in Table 8. In order to form these categorization schemes, we examined curriculum of each field of study. We identified 7 subcategories 4 of which belong to the "computer

Table 6 Gender distribution and minimum, maximum, average age values for each project group and for the group CMPE-Exp

	Gender		Age		
Project Group	**Female**	**Male**	**Minimum**	**Maximum**	**Average**
GSM-CRM	30.00%	70.00%	23.00	39.00	28.37
GSM-Billing	14.29%	85.71%	22.00	34.00	29.00
GSM-TDD	0.00%	100.00%	32.00	35.00	33.67
Banking	48.28%	51.72%	24.00	43.00	32.00
DBMS	4.55%	95.45%	23.00	44.00	32.00
CMPE-Exp	15.38%	84.62%	26.00	32.00	29.00
Finance	7.69%	92.31%	21.00	33.00	27.00
ERP	0.00%	100.00%	28.00	39.00	31.00
CRM	0.00%	100.00%	22.00	27.00	24.00
Mobile	12.50%	87.50%	27.00	33.00	30.00

related" category, while the rest belongs to the "non-computer related" category. Categorical values for the factor *undergraduate degree* consist of these 7 subcategories (e.g. Computer Engineering/Computer Science/Software Engineering, Engineering, Math/Math Related,etc.). In order to analyze the effect of educational level on confirmation bias, we included the factor *graduate degree* which can take one of the categorical values "grad degree" and "no grad degree".

The factor *job title* can take one of the following categorical values: "researcher", "analyst", "developer" and "tester". Within each project group, the distribution of the participants according to their job titles is given in Table 9.

4.2.2 Characteristics of the Software Product

The categorical factor *software development methodology* can be assigned one of the following methodologies: incremental, Test Driven Development (TDD) or agile. The development methodologies employed in project groups are listed in Table 5 where the development methodology of the group *CMPE-Exp* is defined as "various", since among the members of this group, one member used to develop software according to the incremental methodology, while the rest of the members used to employ agile methodology. Development language and domain expertise are confounding factors and they can also be defined as categorical values. The values these two factors can take are given in Table 5.

4.2.3 Characteristics of the Software Company

Company characteristics are all categorical factors. The value range for these factors is given in Table 4. Country factor can take two values since data is collected from only two different countries, namely Turkey and Canada. These two countries were treated as if they are randomly selected from a large set of countries. Therefore, different from other factors, we applied *random effects* ANOVA to *country* factor, instead of *fixed effects* ANOVA.

Table 7 Categorization of the undergraduate degree fields of software engineers and CMPE-Exp members.

Computer Related		**133**
Comp. Eng./Comp. Sci./SE		**97**
Computer Engineering	76	
Computer Science	19	
Software Engineering	1	
Information System Engineering	1	
Engineering	**9**	
Electronics Engineering	7	
Telecommunication Engineering	1	
System Engineering	1	
Math and Computing	**11**	
Mathematical Engineering	10	
Statistics	1	
Others	**16**	
Computer Programming	5	
Computer Education	3	
Management Information Systems	1	
Information Systems and Technology	7	
Non-Computer Related		**41**
Math/Math Related	**19**	
Mathematics/Applied Math	11	
Mathematics Education	1	
Economics	7	
Engineering and Others	**17**	
Industrial Engineering	8	
Mechanical Engineering	2	
Electrical Engineering	2	
Physics Engineering	1	
Engineering Science	3	
Environmental Engineering	1	
Business and Arts	**5**	
Business Administration	4	
Arts	1	
TOTAL		**174**

4.3 Checking the Assumptions of Multi-Way ANOVA

In order to check the model (i.e., multi-way ANOVA) assumptions, we check the following assumptions on the random errors (i.e., residuals):

4.3.1 Normal Distribution of Residuals

One of the model assumptions for multi-way ANOVA is that residuals are normally distributed with zero mean ($\varepsilon_i \sim N(0, \sigma^2)$). In other words, we use the normal density as the working approximation for random errors. Figure 2 shows the Q-Q plots and histograms of the residuals for the multi-way ANOVA model, which we constructed for this study. As it can be seen from Figure 2, residuals conform to the normal distribution ($\varepsilon_i \sim N(0, \sigma^2)$). Moreover, Chi Square Goodness of Fit Test failed to reject the null hypothesis that residuals

Table 8 Categorization of the undergraduate degree fields of software engineers and CMPE-Exp members.

Computer Related			133
Comp. Eng./Comp. Sci./SE		97	
Computer Engineering	76		
Computer Science	19		
Software Engineering	1		
Information System Engineering	1		
Engineering		9	
Electronics Engineering	7		
Telecommunication Engineering	1		
System Engineering	1		
Math and Computing		11	
Mathematical Engineering	10		
Statistics	1		
Others		16	
Computer Programming	5		
Computer Education	3		
Management Information Systems	1		
Information Systems and Technology	7		
Non-Computer Related			41
Math/Math Related		19	
Mathematics/Applied Math	11		
Mathematics Education	1		
Economics	7		
Engineering and Others		17	
Industrial Engineering	8		
Mechanical Engineering	2		
Electrical Engineering	2		
Physics Engineering	1		
Engineering Science	3		
Environmental Engineering	1		
Business and Arts		5	
Business Administration	4		
Arts	1		
TOTAL			174

are a random sample from a normal distribution at 0.05 significance level (p = 0.4551).

4.3.2 Homoscedasticity

Since ANOVA assumes that variances are equal across groups, we need to check for "homoscedasticity". The homoscedasticity hypothesis implies constant variance σ^2. In other words, residuals must have a constant variance σ^2 for all settings of the independent variables. In order to check whether within-group variances of residuals for all groups are same, we applied Barlett's Test. The results of the Barlett's Test for all factors are given in Table 10. In Table 10, conditions for homoscedasticity are met for all categorical factors except for "Gender" factor. When all other assumptions for multi-way

Table 9 Total number of analysts, developers and testers in each project group

Project Group	Researcher	Analyst	Developer	Tester
GSM-CRM	–	–	13	16
GSM-Billing	–	–	14	–
GSM-TDD	–	–	3	–
Banking	–	29	27	–
DBMS	–	1	11	10
CMPE-Exp	18	–	–	–
Finance	–	–	12	–
ERP	–	–	6	–
CRM	–	–	6	–
Mobile	–	–	8	–
Subtotals	18	30	100	26
TOTAL	**174**			

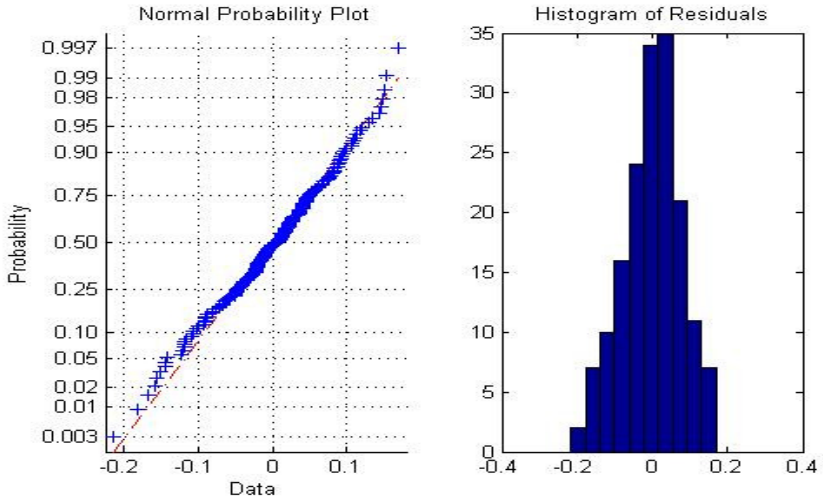

Fig. 2 Q-Q Plots and Histograms for the residuals of the multi-way ANOVA

ANOVA are met, violation of homoscedasticity (i.e., heteroscedasticity) does not result in biased parameter estimates. Heteroscedasticity has to be severe to cause bias in the estimates of standard errors [35]. In our case, the problem of heteroscedasticity is not severe according to the criteria of Allison [35]. Moreover, despite all of the simulation studies that have been done, there does not seem to be a consensus about when heteroscedasticity is a big enough problem that alternatives to ANOVA should be used [36], [37]. In order to check the validity of homoscedasticity for the continuous factors "Age", "Development Experiefnce" and "Testing Experience", we produced scatter plots of the standardized residuals against the observed values of these independent variables for the multi-way ANOVA model, as shown in Figures 3. As it can be seen from these Figures, there is not a significant increase or decrease in the values

Table 10 Bartlett's Test Results for the Multi-Way ANOVA Model.

Factor	T (Test Statistics)	d.f.	p-value
ISVVsInHouse	0.3614	1	0.0573
CompanyInstType	3.3912	2	0.1875
GradDegree	0.1955	1	0.6584
UnderGradDegree	11.7856	6	0.0669
Gender	4.7668	1	0.0290
Country	0.0259	1	0.8723
Language	0.3563	2	0.8368
DomainExpertise	0.0880	1	0.7668
JobTitle	0.2398	3	0.9709
Methodology	2.3754	2	0.3049

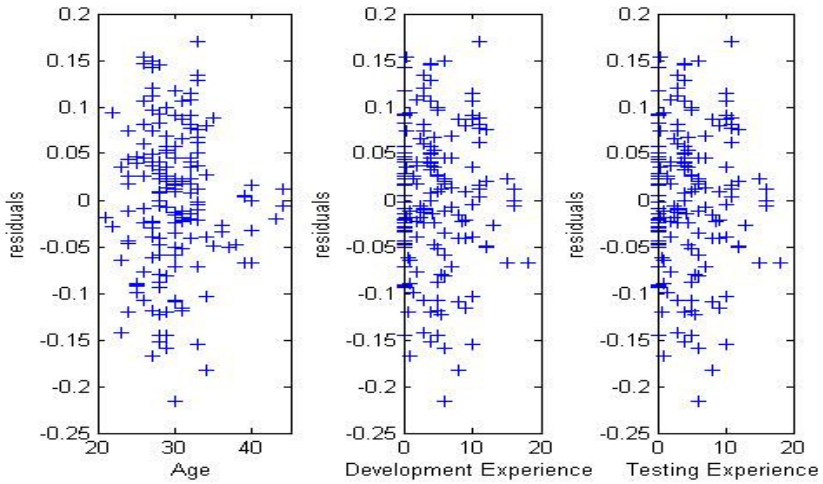

Fig. 3 Scatters plots of the standardized residuals against independent variables "Age", "Development Experience" and "Testing Experience" for the first multi-way ANOVA model.

of the residuals as the values of the independent variables increase. In other words, variability in the measurement error stays almost constant along the scale and there is not a severe homoscedasticity.

4.3.3 Multicollinearity

Multi-way ANOVA requires that the independent variables are not correlated and are independent from each other. Since the independent variables are all categorical (i.e. discrete) except for "Age", "Development Experience" and "Testing Experience", we estimated the "mutual information" among independent variables. In information theory, "mutual information" random variables is a quantity that measures the mutual dependence of the two random variables [38]. When any given two variables X and Y are discrete "mutual information" is one of the measures of correlation between these two variables.

Table 11 Mutual Information Values for Independent Variables.

	IV_1	IV_2	IV_3	IV_4	IV_5	IV_6	IV_7	IV_8	IV_9	IV_{10}	IV_{11}	IV_{12}	IV_{13}
IV_1	1.00	0.02	0.09	0.06	0.18	0.02	0.04	0.06	0.05	0.11	0.38	0.05	0.07
IV_2	—	1.00	0.36	0.00	0.08	0.10	0.20	0.23	0.09	0.09	0.01	0.02	0.15
IV_3	—	—	1.00	0.14	0.07	0.03	0.05	0.17	0.16	0.30	0.05	0.02	0.20
IV_4	—	—	—	1.00	0.08	0.02	0.03	0.02	0.03	0.29	0.02	0.03	0.01
IV_5	—	—	—	—	1.00	0.02	0.05	0.11	0.07	0.29	0.13	0.11	0.12
IV_6	—	—	—	—	—	1.00	0.05	0.09	0.04	0.09	0.06	0.03	0.07
IV_7	—	—	—	—	—	—	1.00	0.36	0.32	0.07	0.04	0.03	0.42
IV_8	—	—	—	—	—	—	—	1.00	0.48	0.14	0.07	0.04	0.33
IV_9	—	—	—	—	—	—	—	—	1.00	0.17	0.01	0.01	0.31
IV_{10}	—	—	—	—	—	—	—	—	—	1.00	0.12	0.09	0.27
IV_{11}	—	—	—	—	—	—	—	—	—	—	1.00	0.02	0.07
IV_{12}	—	—	—	—	—	—	—	—	—	—	—	1.00	0.04
IV_{13}	—	—	—	—	—	—	—	—	—	—	—	—	1.00

$$I(X;Y) = \sum_{y \epsilon Y} \sum_{x \epsilon X} p(x,y)/log_2(p(x,y)/(p(x) * p(y))) \quad (5)$$

In equation 5, $p(x,y)$ is the joint probability distribution function of X and Y, and $p(x)$ and $p(y)$ are the marginal probability distribution functions of X and Y, respectively. In order to find out the extent to which the continuous variables "Age", "Development Experience" and "Testing Experience" are correlated with the rest of the variables, which are categorical, we also discretized these three independent variables. The estimated "mutual information" values for the independent variables are given in Table 5. The information about which independent variable IV_i corresponds to which factor is given in Table 12. In Table 5, since $I(IV_i;IV_j) = I(IV_j;IV_i)$ we left the lower half of the resulting matrix blank. As it can be seen in Table 5, the highest three "mutual information" values are 0.48, 0.38 and 0.36, which stand for the extent of correlation between "IV_9: Domain Expertise" and "IV_8: Development Language", "IV_{13}: Methodology" and "IV_7: Country", and "IV_7: Country" and "IV_8: Language", respectively. The extent of correlation of any two independent variables is within the range $[0.00, 0.51]$ and it is 0.11 on average.

In order to check for multicolinearity of independent variables, we also calculated the Variance Inflation Factor (VIF) for the design matrix in the

Table 12 List of Independent Variables (IVs)

$IV\#$	IV name
IV_1	Age
IV_2	ISVVsInHouse
IV_3	Company
IV_4	GradDegree
IV_5	UnderGradDegree
IV_6	Gender
IV_7	Country
IV_8	Language
IV_9	DomainExpertise
IV_{10}	JobTitle
IV_{11}	DevelopmentExperience
IV_{12}	TestingExperience
IV_{13}	Methodology

regression model that is complementary to the ANOVA model. Multiple regression is closely related to ANOVA. In fact together with ANCOVA, ANOVA is an alternative but complementary way of presenting the same information. When ANOVA is carried out on the same data, it gives exactly same inferences as multiple regression [39]. Therefore, we formed the design matrix for the regression model with independent variables that are listed in Table 12 and with the dependent variable, which is confirmation bias level CB in order to calculate VIF. Table 13 shows the VIF values for the independent variables, which are all categorical except for *Age*, *DevelopmentExperience* and *TestingExperience*. In Table 13, the "Category" column for the continuous variables is left blank. In the design matrix of the regression model, we represented categorical variables in the form of dummy variables. For instance, we transformed the categorical variable Company into 3 dummy variables: *Large Scale Company*, *Research/Academics*, *SME*. We selected *Large Scale Company* as the reference category and omitted it. Then, we coded Large Scale Company 0 on the variables *Research/Academics* and *SME* each. Research/Academics was coded 1 on the variable *Research/Academics* and 0 on the variable *SME*. Finally, SME was coded 0 on the variable *Research/Academics* and 1 on the variable *SME*. The resulting VIF values for all independent variables are shown in Table 13.

The threshold value $VIF_{Th} = 10$ is exceeded for "Country" and "DomainExpertise" variables, hence multicolinearity is high for these two categorical variables [40]. Based on these results, we can conclude that in general multicollinearity is not a threat to the validity of our empirical analysis results.

5 Results and Discussions

According to the multi-way ANOVA test results, which are shown in Table 14, the confounding factors did not turn out to have any significant effect on confirmation bias.

Table 13 VIF values for the independent variables

Variable	Category	VIF
Age	—	2.8787
ISVVsInHouse	InHouse	6.4778
CompanyType	Research/Academics	5.7228
CompanyType	SME	4.5112
GradDegree	GradDegree	1.8650
UnderGradDegree	Business&Arts	1.3866
UnderGradDegree	Computer Related (Engineering)	1.2321
UnderGradDegree	Computer Related (Others)	1.3867
UnderGradDegree	Math/Math Related	1.4627
UnderGradDegree	Math&Computing	1.6292
UnderGradDegree	NonComputer Related (Engineering)	1.3942
Gender	Male	1.4580
Country	Canada	17.0653
Language	C#	1.6392
Language	C++	6.9667
DomainExpertise	High	12.4388
JobTitle	Developer	5.5580
JobTitle	Researcher	7.1231
JobTitle	Tester	3.3166
expDev	—	2.4151
expTest	—	2.0353
Methodology	Incremental	2.3673
Methodology	TDD	6.2105

Table 14 Results of the multi-way ANOVA test with all factors taken into account

Effects	**Sum Sq.**	**d.f.**	**Mean Sq.**	**F**	**Prob >F**	η^2	η_P^2
Age	0.0061	1	0.0061	0.76	0.3854	0.0039	0.0051
ISVsInHouse	0.0287	1	0.0287	3.55	0.0616	0.0183	0.0237
CompanyInstType	0.0226	2	0.0113	1.38	0.2558	0.0144	0.0188
GradDegree	0.0176	1	0.0176	2.17	0.1427	0.0110	0.0147
UnderGradDegree	0.0180	6	0.0030	0.37	0.8964	0.0115	0.0150
Gender	0.0029	1	0.0029	0.36	0.5507	0.0019	0.0024
Country	0.0169	1	0.0169	2.09	0.1503	0.0108	0.0141
Language	0.0093	2	0.0047	0.58	0.5634	0.0059	0.0078
DomainExpertise	0.0082	1	0.0082	1.02	0.3151	0.0052	0.0069
JobTitle	0.1099	3	0.0367	4.53	0.0045	0.0701	0.0851
DevelopmentExp	0.0035	1	0.0035	0.43	0.5114	0.0022	0.0030
TestingExp	0.0020	1	0.0020	0.25	0.6196	0.0013	0.0017
Methodology	0.0328	2	0.0164	2.03	0.1354	0.0209	0.0270
Error	1.1810	146	0.0081				
Total	1.5670	169					

As shown in Table 14, confirmation bias levels of the participants are not affected by their educational background or the degree of education. Therefore, we failed to reject the following hypotheses:

H_0^1 Confirmation bias levels of software engineers are not affected by their educational background (undergraduate level).

H_0^2 Confirmation bias levels of software engineers are not affected by their level of education (Bachelor's vs. Master's degree).

In a previous empirical study [7], we also investigated the effect of *educational background* on the confirmation biases of participants who belong to groups GSM-CRM, DBMS and CMPE-Exp. As an outcome of the said study, we could not find any supporting evidence either. In cognitive psychology literature, there are some studies which investigate how the field and degree of education affect the performance on variants of the Wason's Selection Task [41], [42], [43]. Hoch and Tschirgi [41] compared the performance of groups with three different levels of education attainment: high school, bachelor's degrees and masters degrees. The authors found a significant association with the degree of education and correct selections as well as having observed correct selection rates of 48% among groups of subjects with master's degree. Unlike the results obtained by Hoch and Tschirgi [41], Griggs and Ransdell [43] found correct selection rates on the standard abstract selection task of under 10% by doctoral scientists. In order to resolve this conflict, Jackson and Griggs [42] compared four areas of specialism which are social science, technical engineering, computer science and mathematics respectively. The authors also compared two levels of education (bachelor's and masters degree). The findings of Jackson and Griggs showed no effect of educational level, however a significant effect of area of expertise was observed: subjects with technical areas of specialism such as mathematics and engineering performed much better. In this research, our findings about the degree of education are inline with the findings of Jackson and Griggs. However, we did not observe any significant effect of the educational field, since areas of specialism of 97% of the participants who took part in our research are all technical.

Table 15 Results of the one-way ANOVA test with only the significant factor *JobTitle*

Effects	**Sum Sq.**	**d.f.**	**Mean Sq.**	**F**	**Prob >F**	η^2
JobTitle	0.1632	3	0.0544	6.43	0.0004	0.1041
Error	1.4038	166	0.0085			
Total	1.5670	169				

Factor *JobTitle* affects confirmation bias, significantly ($\alpha = 0.05$), according to multi-way ANOVA results, which are presented in Table 14. Since the effect is dispersed amongst many nonsignificant factors, we also built a simpler model with only *JobTitle* factor. The results of one-way ANOVA are shown in Table 15 and the box plot for confirmation bias levels of "researchers", "developers", "testers" and "analysts" is presented in Figure 4. There is a significant difference between "researchers" and others (i.e., "developers", "testers" and "analysts") as it can be seen from the box plot in Figure 6. However, according to the ANOVA results presented in Table 17, difference between confirmation bias levels of "developers", "testers" and "analysts" is not statistically significant ($p = 0.1588$). Figure 5 shows the box plot of the confirmation bias levels

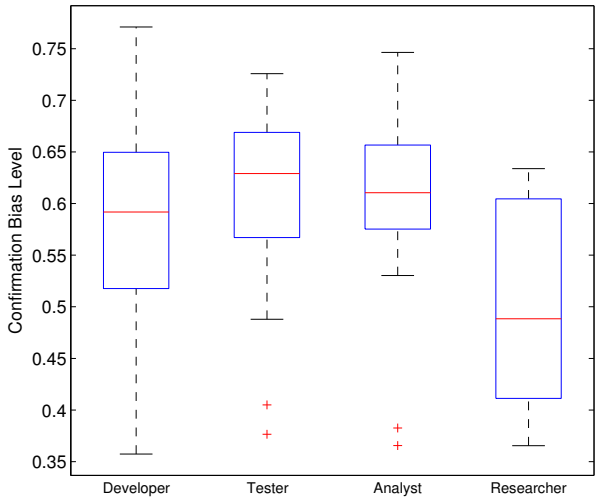

Fig. 4 Boxplot for the confirmation bias levels with respect to job titles

of software engineers with respect to their roles as "developers", "testers" and "analysts". Therefore, we failed to reject hypothesis H_0^4, which states that confirmation bias levels of a software engineer is not related to his/her role (e.g., analysts, developer, tester) in the software development process.

Table 16 Results of the one-way ANOVA test for software engineers with respect to their roles as "developers", "testers" and "analysts" in the software development process

Effects	**Sum Sq.**	**d.f.**	**Mean Sq.**	**F**	**Prob >F**	η^2
Developer-vs-Tester-vs-Analyst	0.0310	2	0.0155	1.86	0.1588	0.0250
Error	0.2407	149	0.0083			
Total	1.2717	152				

As mentioned in the previous paragraph, we found that confirmation bias levels of "researchers" are much lower compared to those of "developers", "testers" and "analysts", as shown in Figures 4 and 6. Researchers consist of Computer Engineering PhD candidates. It is highly probable that theoretical computer science courses have strengthened their reasoning skills and helped them to acquire an analytical and critical point of view. Moreover, researchers take part in research projects, which require strategic hypothesis testing skills. Therefore, we can conclude that confirmation bias is most probably affected by continuous usage of abstract reasoning and hypothesis testing skills. This result is inline with our previous findings [7], [8], [9]. Therefore, we rejected

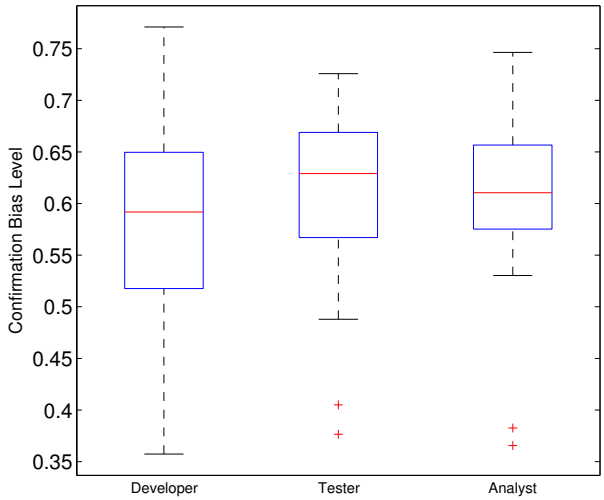

Fig. 5 Boxplot for the confirmation bias levels of software engineers with respect to their roles as "developer", "tester" and "analyst".

hypothesis H_0^3, which states that logical reasoning and strategic hypothesis testing skills are not differentiating factors in low confirmation bias levels.

Table 17 Results of the one-way ANOVA test to compare confirmation bias levels of "Researchers" with "Others" (i.e., developers, testers and analysts).

Effects	**Sum Sq.**	**d.f.**	**Mean Sq.**	**F**	**Prob >F**	η^2
Researchers-vs-		1				
Others	0.1322		0.1322	15.47	0.0001	0.8437
Error	1.4348	168	0.0085			
Total	1.5670	169				

Regarding the effect of *experience in development and testing* on confirmation bias, results of the ANOVA test are inline with our previous findings [7], [8], [9]. Since no significant effect of experience in development or testing was detected, we failed to reject the following hypotheses:

H_0^5 Confirmation bias level of a software engineer is not affected by his/her industrial software development experience (in years). .

H_0^6 Confirmation bias level of a software engineer is not affected by his/her industrial software testing experience (in years).

We could not detect any significant effect of development methodologies on confirmation bias levels. Therefore, we failed to reject hypothesis H_0^7, which

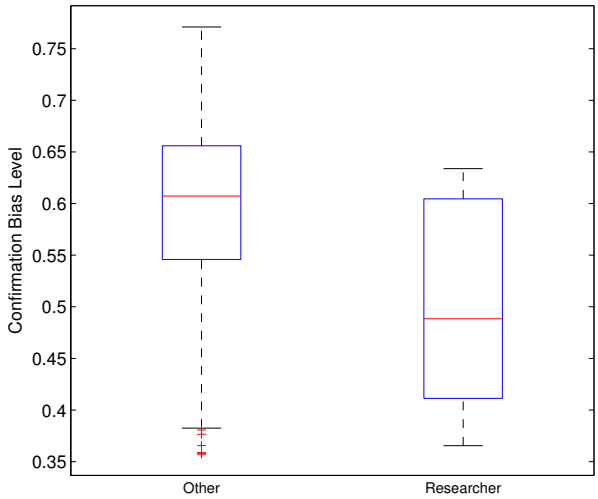

Fig. 6 Boxplot for the confirmation bias levels of "researchers" and "software engineers".

states that software development methodology (e.g. incremental, agile and TDD) does not affect confirmation bias levels of the software engineers.

Moreover, we did not observe any significant effect of *company size* on confirmation bias levels. This result is inline with our previous empirical findings [8]. In a previous study [8], we compared confirmation bias levels of software engineers working for the large-scale company GSM-Company with confirmation bias levels of software engineers who work for the following SMEs: Finance Software Vendor, CRM-Vendor and Business Solution Provider. In this research, having extended the dataset, we were able to find results supporting our previous findings. As a result, we failed to reject hypothesis H_0^8, which states that there is no significant difference between confirmation bias levels of software engineers who work for large-scale software development companies and confirmation bias levels of those who work for SMEs.

6 Threats to Validity

In order to avoid mono-method bias, which is one of the threats to construct validity, we defined a set of confirmation bias metrics. It was the second step to derive a single metric to quantify confirmation bias levels by using these metrics. In order to form our confirmation bias metrics set, we made an extensive survey in cognitive psychology literature covering significant studies that have been conducted since the first introduction of the term "confirmation bias" by Wason in 1960 [1]. Since our metric definition and extraction methodology

is iterative, we were able to improve the content of our metrics set through a pilot study as well as datasets collected during our related previous research [4], [7], [8]. Thus, we were able to demonstrate that multiple measures of key constructs behave as we theoretically expected them to.

Our metric extraction methodology consists of administration of a confirmation bias test that we prepared inheriting existing theories in cognitive psychology literature. In order to avoid the interaction of different treatments, we ensured that none of the participants were involved simultaneously in several other experiments designed to have similar effects, before the administration of confirmation bias tests to participant groups,.

Evaluation apprehension is a social threat to construct validity. Many people are anxious about being evaluated. Moreover, some people are even phobic about testing and measurement situations. Participants may perform poorly due to their apprehension, and they may feel psychologically pressured. In order to avoid such problems, we informed the participants before the confirmation bias tests started that the questions they are about to solve do not aim to measure IQ or any related capability. Participants were also told that the results would not be used in their performance evaluations and their identity would be kept anonymous. Moreover, participants were told that there was no time constraint for completing the questions.

Regarding administration of confirmation bias test, another social threat to construct validity is the expectancies of the researcher. There are many ways a researcher may bias the results of a study. Hence, the outcomes of test were independently evaluated by two researchers, one of whom was not actively involved in the study. The said researcher was given a tutorial about how to evaluate the confirmation bias metrics from the outcomes of the confirmation bias test. However, in order not to induce a bias, she was not told about what the desired answers to the questions were. The inter-rater reliability was found to be high for the evaluation of each confirmation bias metric. The average value for Cohen's kappa was 0.92. During the administration of the confirmation bias test, explanations given to the participants before they started solving the questions did not include any clue about the ideal responses.

To avoid external threats to validity, we collected data from seven different companies specialized in different software development domains. We also selected different projects within the GSM company. However our empirical study still has a limited sample comprising one company from Canada and all others from Turkey. Therefore, we cannot generalize the results to be necessarily representative of software companies worldwide.

In order to address statistical validity, we used multi-way ANOVA test taking confounding factors into account as well as the factors of interest. Moreover, before conducting the multi-way ANOVA test, we checked for the following assumptions that are required for multi-way ANOVA: normal distribution of the residuals, homoscedasticity, and multicolinearity. In order to check whether residuals of multi-way ANOVA model are normally distributed with zero mean, we plotted histogram of the residuals and Q-Q plot as well as performing Chi Square Goodness of Fit Test. We applied Barlett's test and produced scat-

tered plots of standardized residuals against independent continuous variables *Age*, *Development Experience* and *Testing Experience* in order to check homoscedasticity. We estimated mutual information values for the factors of the multi-way ANOVA model, in addition to calculating Variance Inflation Factor (VIF) values for the corresponding regression model. In the multi-way ANOVA model the actual effect of the significant factor *JobTitle* is dispersed among many non-significant factors. Therefore, we also build one-way ANOVA model with just the significant factor *JobTitle* in order to brain the actual effect of the significant factor.

Finally, for the credibility of our research, we got the ethics approvals for this research study from both the Ethics Board in Bogazici University in Istanbul, Turkey and the Research Ethics Board of Ryerson University in Toronto, Canada. We also took the guidance of an expert in cognitive psychology throughout the course of our study in order to have the credibility of other disciplines besides empirical software engineering.

7 Conclusions and Future Work

The overall aim of this research is to explore factors which affect confirmation biases of software engineers. Empirical investigation of factors affecting confirmation bias requires quantification/measurement of confirmation bias levels. For this purpose, we defined a methodology to define and extract confirmation bias metrics. During our previous studies [4], [7], [8], [9], we conducted the initial form of the methodology to measure/quantify confirmation bias and the content of confirmation bias metric set was at an immature level. Hence, we were unable to define a single derived metric to measure confirmation bias levels of software engineers. In this paper, in addition to performing our empirical investigation by using a more extensive dataset, we were able to perform multi-way ANOVA test where we managed to represent the dependent variable "confirmation bias level" as a single value. As a result of our empirical analyses, we were able to strengthen the following claims which originate from our previous studies:

- Confirmation bias levels of individuals who have been trained in logical reasoning and mathematical proof techniques are significantly lower. In other words, given a statement such individuals show tendency to refute that statement rather than immediately accepting its correctness.
- A significant effect of experience in software development/testing has not been observed. This implies that training in organizations is focused on tasks rather than personal skills. Considering that the percentage of people with low confirmation bias is very low in the population [1], [2], [44], an organization should find ways to improve basic logical reasoning and strategic hypothesis testing skills of their software engineers.

As future work, our field studies will also include observational techniques (e.g., think-aloud protocols, participant observation and observation synchronized shadowing. In this way, we aim to strengthen our hypothesis regarding

the connection between confirmation bias and defect rates through unit testing. We also aim to conduct laboratory experiments, where we can obtain detailed information about developers' unit testing activities in a controlled environment. Finally, we intend to investigate the relationship between levels of motivation and confirmation bias as well as the relationship between personality and confirmation bias.

References

1. Wason, P. C., On the failure to eliminate hypotheses in a conceptual task, Quarterly Journal of Experimental Psychology, 12, 129-140 (1960)
2. Wason, P. C., Reasoning about a rule, Quarterly Journal of Experimental Psychology, 20, 273281 (1968)
3. Stacy, W. and MacMillan, J., Cognitive bias in software engineering, Communication of the ACM, (38)6, (1995)
4. Calikli, G. and Bener, A., Preliminary analysis of the effects of confirmation bias on software defect density. Proceedings of the 4^{th} International Symposium on Empirical Software Engineering and Measurement, (2010)
5. Calikli, G. and Bener, A., Influence of confirmation biases of developers on software quality: an empirical study, Software Quality Journal, 21, 377-416 (2013).
6. Calikli, G, Bener, A., Aytac, T. and Bozcan, O., Towards a metric suite proposal to quantify confirmation biases of developers. Proceedings of the 7^{th} International Symposium on Empirical Software Engineering and Measurement, Industry Track, 2013.
7. Calikli, G., Bener, A., and Arslan, B., An analysis of the effects of company culture, education and experience on confirmation bias levels of software developers and testers. Proceedings of 32^{nd} International Conference on Software Engineering, (2010)
8. Calikli, G., Arslan, B., and Bener, A., Confirmation bias in software development and testing: an analysis of the effects of company size, experience and reasoning skills. Proceedings of the 22^{nd} Annual Psychology of Programming Interest Group Workshop, (2010)
9. Calikli, G. and Bener, A., Empirical analyses factors affecting confirmation bias and the effects of confirmation bias on software developer/tester performance. Proceedings of 5^{th} International Workshop on Predictor Models in Software Engineering, (2010)
10. Inglis, M. and Simpson, A., Mathematicians and the Selection Task. Proceedings of the International Group for the Psychology of Mathematics Education, Cape Town, South Africa, 2004.
11. Einhorn, H. J. and Hogarth, R. M., Confidence in judgment: Persistence of the illusion of validity. Psychological Review, 85, 395416, 1978.
12. Mahoney, M. J. and DeMonbreun, B. G., Confirmatory bias in scientists and nonscientists, Cognitive Therapy and Research, (1977).
13. Kahneman, D., Thinking, Fast and Slow, Doubleday, Canada, (2011).
14. Knauff M., Budeck C., Wolf A. G., Hamburger K., The Illogicality of Stock-Brokers: Psychological Experiments on the Effects of Prior Knowledge and Belief Biases on Logical Reasoning in Stock Trading, PLoS ONE Journal, 5(10): e13483. doi:10.1371/journal.pone.0013483, (2010).
15. Garavan, H., Doherty. M. E. and Mynatt, C. R., When Falsification Fails, The Irish Journal of Psychology, (13):3, (1997).
16. Murray, J. and Thompson, M. E., Applying Decision Making Theory to Clinical Judgement in Violence Risk Assessments, Europes Journal of Psychology, 2, 150-171, 2010.
17. Dawson, N. V., Physician Judgement in Clinical Settings: Methodological and Cognitive Performance, Clinical Chemistry, (39)7, 1468-1478, 1993.
18. Teasley, B., Leventhal, L. M., and Rohlman, S., Positive test bias in software engineering professionals: What is right and what's wrong. Proceedings of the 5^{th} Workshop on Empirical Studies of Programmers,(1993)
19. Teasley, B. E., Leventhal, L. M.,Mynatt, C. R., and Rohlman, D. S., Why software testing is sometimes ineffective: Two applied studies of positive test strategy, Journal of Aplied Psychology, (79)1, pp. 142-155, (1994)

20. Borkowski, J., Carr, M. and Pressley, M., Spontaneous strategy use: Perspectives from metacognitive theory, Intelligence, 11(1):6175, (1987).
21. Mair, C. and Shepperd, M., Human Judgement and Software Metrics: Vision for the Future, 33rd International Conference on Software Engineering, ICSE 11, May 2128, 2011, Waikiki, Honolulu, HI, USA, 2011.
22. Merkhofer, M. W., Assessment, Refinement, and Narrowing of Options, Chapter 8, Tools to Aid Environmental Decision Making, V. H. Dale, and M. R. English, eds., Springer, New york, 1998.
23. Erdogmus, H., Morisio, M. and Torchiano, M., On the effectiveness of test-first approach to programming. IEEE Transactions on Software Engineering, 31, pp. 1-14, (2005).
24. Cox, J. R. and Griggs, R. A., The effects of experience on performance in Wason's selection task, Memory and Cognition, 10, pp.496-502 (1982)
25. Griggs, R. A. and Cox, J. R., The elusive thematic materials effect in Wason's selection task, British Journal of Psychology, 73, pp.407-420 (1982)
26. Cheng, P. W. and Holyoak, K. J., Pragmatic reasoning schemas,Cognitive Psychology, 17, pp.391-416, (1985)
27. Cosmides, L., The logic of social exchange: Has natural selection shaped how humans reason? Studies with Wason's selection task, Cognition, 31, pp.187-276, (1989)
28. Manktelow, K. I. and Over, D. E., Inference and understanding: A philosophical and psychological perspective, (1990)
29. P. C. Wason and Shapiro, D., Natural and Contrived Experience in a Reasoning Problem, Quarterly Journal of Experimental Psychology, 23, pp.63-71, (1971)
30. Manktelow, K. I. and Evans, J. St. B. T., Facilitation of reasoning by realism: Effect or non-effect?, British Journal of Psychology, 70, pp.477-488, (1979)
31. Johnson-Laird, P.N. and Tridgell, J. M., When negation is easier than affirmation, Quarterly Journal of Experimental Psychology, 24, pp.87-91, (1972)
32. Griggs, R.A., The role of problem content in the selection task and in the THOG problem, Thinking and reasoning: psychological approaches. Routledge and Kegan Pauli, London, (1983).
33. Cook, T. D. and Campbell, D. T., Quasi- Experimentation: Design and Analysis Issues for Field Settings, Houghton-Mifflin Company, USA, (1979).
34. Hirschi, N. W. and Frey, D. D., Cognition and complexity: An experiment on the effect of coupling in parameter design, Research in Engineering Design, 13, pp. 123-131, (2002).
35. Allison, P. D., Multiple Regression: A Primer. Pine Forge Press, Thousand Oaks, CA, USA (1999).
36. Bradley, J. V., Robustness, British Journal of Mathematical and Statistical Psychology, 31:144-155 (1978).
37. Glass, G. V., Peckham, P. D. and Sanders, J. R., Consequences of Failure to Meet Assumptions Underlying Fixed Effects Analyses of Variance and Covariance, Review of Educational Research, 42, 237-288 (1972).
38. MacKay, D. J. C., Inference and learning algorithms, Cambridge University Press, Cambridge, UK (2003).
39. Tarling, R., Statistical modelling for social researchers: principles and practice, Routledge, New York, Ny, USA (2009)
40. Kutner M. H., Nachtsheim C. J. and Neter J., Applied linear regression models, 4th edition, McGraw-Hill Irwin, (2004)
41. Hoch, S. J. and Tschirgi, J. E., Logical reasoning and cue redundancy in deductive reasoning. Memory and Cognition, 13, pp. 453-462, (1985)
42. Jackson, S. L. and Griggs, R. A., Education and the selection task. Bulletin of the Psychometric Society, 26, pp. 327-330, (1988)
43. Griggs, R. A. and Ransdell, S. E., Scientists and the selection task. Social Studies of Science, 16, pp. 319-330, (1986).
44. Evans, J. St. B. T., Newstead, S. E. and Byrne, R. M., Human reasoning: the psychology of deduction. Lawrence Erlbaum Associates Ltd., East Sussex, U.K. (1993)
45. Caglayan, B., Tosun-Misirli, A., Calikli, G., Aytac, T., Bener, A. and Turhan, B., Dione: An Integrated Measurement and Defect Prediction Solution, Proceedings of the 20th International Symposium on Foundations of Software Engineering, Cary, North Carolina, USA, September, (2012).