Believing is Seeing: Confirmation Bias Studies in Software Engineering

Magne Jørgensen
Simula Research Laboratory & University of Oslo
Oslo, Norway
magnej@simula.no

Efi Papatheocharous
SICS Swedish ICT
Kista, Stockholm, Sweden
efi.papatheocharous@sics.se

***Abstract*—Confirmation bias is the human tendency to search for, collect, interpret, analyse and/or recall information in a way that confirms one's prior beliefs or wishes. In this paper, we conduct two studies on confirmation bias in software engineering contexts and demonstrate how it may lead to biased beliefs and conclusions. The first study involves 26 software managers from various companies. Initially, we collected the prior belief of the managers on how contract types of either fixed price or hourly payment would affect the best interest of the client and successful development of software projects. Then, the managers were presented with randomly generated project data that were neither in favour of fixed price nor hourly contracts. In our analysis for the presence of confirmation bias, we found that the prior belief of managers favouring a specific contract type (e.g., hourly contracts) affected how they interpreted the project data. The second study analyses the results of 35 published comparisons of regression and analogy-based cost estimation models. Twenty of the 35 comparisons evaluated the performance of a self-developed analogy-based estimation model, i.e., constitutes a situation where a bias towards confirming the desired outcome that one's own model is better than the alternative model, may be present. A statistical meta-analysis including all 35 studies gave that analogy-based models were more accurate than regression-based models. Removing the 20 cases where confirmation bias was more likely to be present, gave however the opposite result. The results of our two studies suggest that we should beware of the effect of confirmation bias when using software professionals' beliefs and conclusions drawn from researchers evaluating their own methods as supporting evidence. To strengthen evidence-based practices we need to use experience elicitation techniques and research methods that avoid conclusions heavily affected by confirmation bias.**

Keywords—confirmation bias, human judgment, evidence-based software engineering, empirical studies

I. Introduction

"*The human understanding when it has once adopted an opinion [...] draws all things else to support and agree with it. And though there be a greater number and weight of instances to be found on the other side, yet these it either neglects or despises, or else by some distinction sets aside or rejects.*" The above Francis Bacon quote from 1620 suggests that the human tendency of trying to confirm rather than disconfirm one's prior belief is far from a new phenomenon. Ideally, when people make a decision, they should select, interpret and use evidence in a neutral way and not in a way that unfairly favour an outcome that fits one's prior belief, interests or wishes. Unfortunately, that is frequently what happens. Prior works show that even in scientific contexts (e.g., clinical judgements), interpretation of facts are affected by the convictions of the scientists, data do not simply speak for themselves and conclusions are driven by bias or personal belief [1]. The tendency towards emphasising evidence supporting prior beliefs or what one wishes to be true is termed "confirmation bias". Confirmation bias poses a threat to the validity of decisions and conclusions made in software engineering contexts. Confirmation bias therefore needs to be addressed both in the context of software professionals' use of practise-based evidence, and in the context of software researchers' study design, data collection and data analysis. We believe that more knowledge about the nature of confirmation bias and better integration of this knowledge in approaches aiming at better judgment and decision-making, such as Evidence-Based Software Engineering (EBSE) [2], are needed.

In this paper, we summarise reported literature on the tendency towards confirmation bias in software engineering contexts (Section 2). We report how this bias leads to inefficient use of methods and processes, poor evidence collection and lower quality projects as some of the consequences. We report from two empirical studies where we demonstrate the relevance of confirmation bias in an industrial and a research context (Sections 3 and 4). We conclude that current evidence collection and analysis practices should be supported with effective ways to deal with confirmation bias and outline future research (Section 5).

II. Confirmation Bias In Software Engineering

The tendency towards confirmation bias is well documented in many domains; see for example [3]. To summarize the knowledge about confirmation bias in the domain of software engineering we conducted a literature search in *google scholar* using the term *(("confirmation bias" OR "confirmatory bias") AND ("software development" OR "software engineering"))* and supplemented it with backward and forward snowball search of identified relevant papers. Our inclusion criteria were that a paper included results related to confirmation bias in a software engineering context, was peer reviewed and written in English.

Below, we briefly summarize the main confirmation bias results from the identified papers:

- In [4] confirmation bias may explain why the students, when being taught to use evidence-based software engineering, tended to recommend the requirements management tools they had initially selected as intervention and not the one better supported by evidence.
- In [5] the tendency towards including only projects that confirms one's desired research outcome is discussed in the context of evaluation of cost estimation models.
- In [6] it is shown how software professional searching for knowledge in documentation can be affected by confirmation bias and how this leads to incomplete searches and misguided results.
- In [7] cognitive biases, including confirmation bias, exhibited by individuals and groups and their relevance to group behaviour in software engineering practice are described.
- In [8] confirmation bias in feature-bug-fix sample selection and defect prediction algorithms is reported in studies of several open source datasets and is suggested to be a serious problem in the testing domain. Similarly, confirmation bias and its effects on quality in software testing are reported in many other papers. Examples include positive testing, i.e., aiming at confirming rather than disconfirming that the code works [9][10], and compromised quality, i.e., delivering code with more defects [11][12]. Inefficient testing and inability to see a bug, due to confirmation bias prevents programmers from noticing facts that would lead to the bug's root cause.
- In [14] user acceptance reported being distorted by confirmation bias, leading to misguided expectations of an information system. Problems that are rare, but maybe more serious, are neglected and commonly after the initial requirements are stated, users may selectively detect information to support only the original statement and do not undertake a fully comprehensive investigation of possible features and requirements.
- In [15] it is reported that those who believed in the benefits of agile software development methods tended to interpret randomly generated project data to support agile methods. Those who were sceptic about the benefits of agile tended to see no such support.
- In [16] the presence of confirmation bias in market reactions to IT investments among people competent in finance and/or IT is shown. The degree of confirmation bias was especially large when analysts more competent in finance than IT received information about IT, i.e., about a topic in which they had less competence.
- In [17] confirmation bias is discussed as part of a set of theories to analyse software engineering behaviours and in particular how in a development team it can lead to dramatically reduce the solution space exploration.
- In [18] the authors find that the first impression of how much a software project may cost strongly affects the final cost estimate of experts, possible due to the effect of confirmation bias in selecting what is perceived as relevant project cost information.
- In [19] the results from a meta-analysis of 42 studies on defect prediction models suggest that the variable best predicting the accuracy of a type of defect prediction model is the research group conducting the study. It is argued that this may be due to researchers' tendency towards confirming their prior belief in a, frequently their own, defect prediction method.
- In [20] it is reported that the proportion of statistically significant results in software engineering experiments is much higher than what should be expected given the statistical power of the experiments. The authors argue that an important contributing reason is researcher bias, i.e., a tendency to bias the results to confirm the stated hypotheses.

The summarized research suggests that confirmation bias is a serious problem, with severe result and decision validity consequences, in various software engineering contexts. It is we argue, important to understand when we should expect confirmation bias to avoid being misled by it and, if possible, adjust for it. The two studies that follow, which analyse how confirmation bias may affect software managers' belief about different contract types (Section 3) and how software cost estimation researchers' vested interests leave them exposed to confirmation bias (Section 4), aim at contributing to this understanding.

III. Confirmation Bias in Contract Type Selection

A. Study Rationale and Hypothesis

The two most common types of contracts between software clients and providers are based on fixed price or hourly payment of the effort spent. There are different opinions and lack of consistent findings on which contract type is the most beneficial for the client, see for example the results reported in [21]-[25]. Software providers applying agile development methods for their projects may, for example, prefer contracts based on hourly payment and claim that fixed price contracts leads to lower client benefits and higher risk of project failures [26].

The study reported in this section examines how the prior belief in the benefits of a contract type affects the interpretation of evidence. We study this in a context where the evidence is intended to be neutral and not favouring any of the two contract types. To do this, we presented the participants of the study with a set of randomly created data about software projects and their use of contracts. If a substantial degree of confirmation bias existed in the interpretation of the projects information, we expected those in favour of fixed price would tend to find support for fixed price and those in favour of hourly payment would tend to find support for hourly payment in the same project data. That is, we test the following hypothesis:

Software managers' preference for a contract type (fixed price or hourly payment) influences them to interpret neutral project data as data confirming their preference.

We designed a study where we could argue that if we observe confirmation bias in that context, it would be likely that the degree of confirmation bias in a real-life situation would be even stronger.

B. Study Design

The participants of the study consisted of 26 experienced software managers present at a seminar. One of the topics presented and discussed at the seminar was the suitability of different contract types in software projects. The participants replied to a survey where the first part consisted of the following two questions:

Q1: What is usually your role in software projects?

Q2: Which contract type do you believe is, most of the time, better for the client of an IT-project? Assume medium or large IT-project and take into account how do you think the contract type will contribute to the IT-project meeting its planned goals (i.e,. delivers benefit to the client and contributes to work being conducted in a cost-efficient way).

For *Q2* the possible answers were:

2a) Fixed price payment is strongly preferred,

2b) Fixed price payment is preferred,

2c) No large difference,

2d) Hourly payment is preferred, or

2e) Hourly payment is strongly preferred.

Then, the participants were presented with randomly generated information about 16 previously completed software projects. The project information is shown in Table 1. An examination of the data[1] gave that were no systematic or apparent difference in terms of benefit for the client, cost-efficiency (productivity) of the project, or the combination of client benefit and cost-efficiency for any of the contract types. The project information was accompanied with the question:

Q3: Below you see a table showing how sixteen different fixed price and hourly paid software projects ended-up with respect to two variables: benefit for the client and cost-efficiency of the work conducted. The projects are similar with respect to size and complexity. Does the project data, in your opinion, give more support for fixed price or for hourly paid contracts?"

For *Q3* the possible answers were:

3a) Clear support for fixed price payment,

3b) Some support for fixed price payment,

3c) Neither support for fixed price or hourly payment,

3d) Some support for hourly payment, or

3e) Clear support for hourly payment.

TABLE I. RANDOMLY CREATED PROJECT DATA

Project	Contract Type	Client Benefit	Cost-Efficiency
1	Hourly	High	High
2	Hourly	Medium	Low
3	Hourly	Low	Medium
4	Hourly	High	Medium
5	Hourly	Medium	Medium
6	Hourly	High	Medium
7	Hourly	High	Medium
8	Fixed price	Low	Low
9	Fixed price	Medium	Medium
10	Fixed price	High	Medium
11	Fixed price	High	Low
12	Fixed price	Medium	High
13	Fixed price	High	Low
14	Fixed price	Medium	High
15	Fixed price	Medium	High
16	Fixed price	High	Low

C. Study Results

The responses on *Q1* showed that all participants could be categorized as either in the role of software client or provider. To simplify the analysis, we merged the prior preferences *2a)* and *2b)* into "*Prefer fixed price payment*", and *2d)* and *2e)* into "*Prefer hourly payment*". We also merged the data interpretation responses *3a)* and *3b)* into "*Fixed price support*" and *3d)* and *3e)* into "*Hourly payment support*".

Not part of our main analysis, but nevertheless noteworthy is the connection between the participants' typical role in the project and the contract type preference. Table II suggests that those in the role of software providers tended to prefer hourly contracts and those in the role of clients to prefer fixed price payment contracts. A *Chi-square test of independence* gave p=0.01, i.e., that it is very unlikely that a participants' typical role in a project and his/her contract preference are independent.

TABLE II. CONTRACT TYPE PREFERENCE PER ROLE

Role	*Fixed Price*	*Neutral*	*Hourly*
Client	11	1	1
Provider	4	1	8

[1] It is possible to argue that some (non-intended) patterns in Table I might exist supporting one of the contract types. The main focus of our study, however, is not to what extent it is reasonable to see a pattern or not, but to what extent prior belief about contract types determines the nature of that pattern, i.e., the degree of confirmation bias in the interpretation. For that purpose, the randomly generated project dataset is, we believe, meaningful.

Table III shows how the participants interpreted the project data in Table I with respect to prior contract type preference. Table III shows that those who preferred fixed price contracts found that the project information supported fixed price or they saw no support for any of the contract types. Those (only two) participants with no prior contract preference found support for neither fixed price nor hourly payment. Four of those who preferred hourly contracts found that the project information supported hourly payment, which was more than the two finding support for fixed price. Removing the two participants who had no prior contract preference and performing the *Chi-square test of independence* on the remaining observations gave p=0.02, i.e., that it is very unlikely that prior preference and interpretation of project data are independent. This suggests the presence of confirmation bias in the studied context.

On the positive side we find that the category with most (13) responses was "Neither". This demonstrates that even if confirmation bias is present in a software engineering context, this does not mean that *all* software professionals are strongly affected by it.

TABLE III. PROJECT DATA INTERPRETATION

	Project data provides support to		
Prior preference	***Fixed Price***	***Neither***	***Hourly***
Fixed Price	7	8	0
Neutral	0	2	0
Hourly	2	3	4

D. Discussion and Limitations

Although the managers participating in the study had extensive experience with software projects (only senior managers were invited to the seminar), the type of data they received about the projects was to some extent different from what they are used to receive. It is therefore not obvious to what degree the result can be generalized to real-world contexts. In spite of this potential problem with our study design, we believe that there are reasons to believe that our data do say something about the presence of confirmation bias in real-world contexts. Observing confirmation bias in our experimental situation with little room for selection bias and subjective data interpretations do, we argue, suggest that the effect of confirmation bias may be even higher in many real-life situations with more opportunities to select and interpret information to confirm prior beliefs.

IV. CONFIRMATION BIAS IN SOFTWARE COST ESTIMATION MODEL EVALUATION

A. Study Rationale and Hypothesis

This section examines the presence and effect of confirmation bias in research conducted to compare the accuracy of different types of effort estimation models. We analyse the presence of confirmation bias as a potential result of that researchers include their own estimation model in the comparison, and presumably wish that it outperforms the rest of the models. The analysis in this section is also motivated by the need to better understand the striking lack of conclusion stability and agreement in studies comparing the accuracy of different types of software cost estimation models, as reported in [27].

Our analysis focuses on the comparisons of two of the most published estimation models, i.e., the analogy- and regression-based estimation models. A search in the software estimation research database BEST-web (http://www.simula.no/BESTweb) gives that there are more than 160 journal papers describing the application of either a regression- or an analogy-based effort estimation model. Analogy-based estimations are relying on the selection of a, typically small, subset of previously completed projects similar (analogous) to the new project. The effort values of the selection are then used to predict the effort of a new project. Regression-based estimations are typically using linear or log-linear models of the development effort of previously completed projects, where software size usually is an important explanatory variable. The core assumption of analogy-based estimation models is that projects with similar characteristics will require similar amount of effort, while the core assumption of regression-based estimations is that there is a reasonably stable relationship between important variables, such as the functional size of the software to be developed, and the required effort. There are many ways of building and tuning both types of estimation models.

We based our selection of comparisons between analogy- and regression-based models on the review recently published as a Systematic Literature Review (SLR) in [28]. That SLR focused on the estimation accuracy of software development effort estimation methods using machine-learning models and includes 35 comparisons of analogy- (case-based reasoning) and regression-based effort estimation models. The authors of the SLR report that analogy-based methods had better mean estimation accuracy than regression-based methods in 66% of the published comparisons (23 out of 35) and that analogy-based models, in general, seem to outperform regression-based models.

We make use of these 35 comparisons for the purpose of our examination of confirmation bias in software effort estimation model research. Especially, we want to see how the inclusion of one's own estimation model affects the results of the summary of comparisons. In as many as 20 of the 35 studies the researchers included their own analogy-based estimation model in the accuracy comparison with other models. The remaining 15 comparisons did not include any self-developed estimation model. A tendency to find evidence confirming the presumably desired outcome that one's own model performs better than the rest may consequently lead to a bias towards better outcome for the analogy-based models.

We performed a meta-analysis of the comparisons made between the two types of models reported in the SRL [28] and test the following hypothesis:

Studies where authors compare their own analogy-based estimation model with regression-based models are more likely to find evidence confirming higher accuracy of analogy-based models than the other studies.

Notice that this hypothesis does not make any claim about deliberate manipulation of data. There are numerous ways that research on software cost estimation model may be exposed to confirmation bias that do not require deliberate data manipulation, such as:

- *Biased selection of project datasets.* A researcher may be more likely to use a dataset that favours one's own estimation model.
- *Biased factor selection*. A researcher may be more likely to include project factors that favour one's own estimation model.
- *Biased removal of outliers*. A researcher may be more likely to find arguments to show that a data point is an outlier (an extreme case, uninteresting, or a mistake in measurements) if that benefits one's own estimation model. Similarly, a researcher may be good at finding arguments to keep an unusual data point in the data set if it favours one's own model.
- *Biased effort on model tuning.* A researcher may be more likely to devote more effort on tailoring or improving one's own cost estimation model in relation to the effort spent on other models, e.g., as a result of one's higher expertise in the tailoring one's own model.
- *Biased data processing and analysis.* A researcher may be more likely to use statistical tests and accuracy measures that favour the results of one's own model. Similarly, a researcher may be more likely to apply transformation on the data values or select cross-validation methods that reach to results that favour one's own method.
- *Publication bias.* A research may chose only to publish the comparison where one's own model performs best.

B. Study Design

Our meta-analysis, displayed in Table IV, is using accuracy data obtained from the SLR reported in [28] and derives the *Confidence Intervals* (*CI*) [29] [30] for each comparison. The analysis is based on the following assumptions:

- The accuracy measure reported is the *Magnitude of Relative Error* (*MRE*), where *MRE* = |*Actual effort – Estimated effort*| /*Actual effort*.
- The *Mean Difference* (MD_{MMRE}) in accuracy reported is the difference of the *Mean MRE* (*MMRE*) between analogy-based and regression estimations of a comparison, i.e., $MD_{MMRE} = MMRE_{Analogy} - MMRE_{Regression}$. A negative value indicates that the analogy-based model on average is more accurate than the regression.
- We consider the comparison of estimation models as within-subject comparisons (re-sampling), i.e., the comparison of a project's estimation accuracy can be said to have a "pre-treatment" value (the analogy-based estimation accuracy) and a "post-treatment" value (the regression-based estimation accuracy).
- There is a "random effect" of changing from an analogy to a regression-based estimation model, i.e., we do not assume that the effect of replacing an analogy with a regression model will have the same ("fixed") effect in all comparisons.
- The *Standard Error* (*SE*) of the difference in accuracy between two estimation models ($SE(MD_{MMRE})$), which is needed to calculate the *CI* of the difference in *MMRE* for the meta-analysis, is given by: $SE(MD_{MMRE})=SD(MD_{MMRE}))/\sqrt{n}$, where n is the number of projects used to compare the two estimation models. $SD(MD_{MMRE})$ denotes the *Standard Deviation* (*SD*) of the difference between the accuracy values of the analogy- and regression-based estimation models. In most cases the value of $SD(MD_{MMRE})$ turned out not to be available or not possible to calculate directly from the published data. When not available we approximated its value by using values from other comparisons that were using the same dataset, or values that were from comparisons similar in estimation accuracy difference; see discussion on this in the limitation section.

We added information about whether the authors included their own estimation model or not and the number of projects for each comparison. We used the meta-analysis tool included in Review Manager 5.3 (http://tech.cochrane.org/revman/), which is supported by the Cochrane organisation.

C. Study Results

A statistical meta-analysis of the difference in estimation accuracy in *MMRE* of the analogy- and regression-based estimation methods of all 35 comparisons gave a 95% *CI* of [-0.30, 0.00], i.e., a statistically significant support for analogy-based estimation models. The mean MMRE of all 35 comparisons is -0.15, suggesting an average improvement of 15% when using analogy-based estimation models. The two-sided test of significance for an overall improvement in mean accuracy when applying analogy-based estimation models gave p=0.05. Our meta-analysis of all 35 comparisons, consequently, provides similar support to the benefits of analogy-based models as the SLR reported in [28], where 66% of the comparisons were in favour of analogy-based models.

If, however, we exclude the comparisons where the researchers' own analogy-based estimation methods are compared with regression-based methods, the conclusion reverses. The *CI* of the estimation accuracy difference is now in favour of regression-based estimation methods. The new 95% *CI* is [-0.02, 0.26], which rather gives support to the benefits of regression-based estimation methods (p=0.08). The mean difference in MMRE is now 12%, in favour of the regression-based models. Only 4 of the remaining 15

comparisons (27%) are now in favour of analogy-based estimation models.

TABLE IV. COMPARISONS OF REGRESSION AND ANALOGY-BASED MODELS

Id	Paper	Dataset[a]	Year	N[b]	Own Method[c]	MD_{MMRE}[d]	$SE(MD_{MMRE})$[e]	Cohen's *d*[f]
1	[31]	Desharnais	1997	50	None	-0.26	0.07	-0.56
2	[32]	Albrecht	1997	24	Analogy	-0.28	0.07	-0.82
3	[32]	Atkinson	1997	21	Analogy	-0.01	0.10	-0.02
4	[32]	Desharnais	1997	77	Analogy	-0.02	0.05	-0.04
5	[32]	Finnish	1997	38	Analogy	-0.60	0.12	-0.82
6	[32]	Kemerer	1997	15	Analogy	-0.45	0.12	-0.96
7	[32]	Mermaid	1997	28	Analogy	-1.48	0.09	-3.15
8	[32]	Telecom 1	1997	18	Analogy	-0.47	0.11	-1.00
9	[32]	Telecom 2	1997	33	Analogy	-0.35	0.08	-0.74
10	[33]	COTS	1998	48	None	0.10	0.07	0.21
11	[34]	Australian	1999	19	None	-0.13	0.11	-0.28
12	[35]	Finnish	1999	119	None	0.44	0.07	0.60
13	[36]	ESA	2000	131	None	0.46	0.06	0.63
14	[36]	Laturi	2000	29	None	0.13	0.14	0.18
15	[37]	Albrecht	2000	24	Analogy	-0.06	0.08	-0.16
16	[37]	CF	2000	21	Analogy	0.18	0.03	1.34
17	[38]	Desharnais	2000	30	None	-0.05	0.09	-0.11
18	[39]	ISBSG	2000	19	None	0.82	0.11	1.74
19	[40]	ISBSG	2001	12	None	0.05	0.14	0.11
20	[40]	ISBSG	2001	12	None	0.30	0.14	0.64
21	[41]	Laturi	2002	63	None	-0.04	0.09	-0.05
22	[42]	Medical	2003	52	None	0.10	0.07	0.21
23	[43]	TukuTuku	2004	67	None	0.01	0.06	0.02
24	[44]	ISBSG	2006	132	Analogy	-1.23	0.04	-2.62
25	[44]	Albrecht	2006	23	Analogy	-0.37	0.08	-0.97
26	[45]	Web applications	2006	15	None	0.04	0.12	0.09
27	[46]	Albrecht	2007	23	Analogy	-0.36	0.08	-0.95
28	[46]	CF	2007	21	Analogy	-0.39	0.10	-0.83
29	[47]	TukuTuku	2008	83	None	-0.09	0.05	-0.19
30	[48]	Desharnais	2009	77	Analogy	-0.26	0.05	-0.55
31	[48]	Maxwell	2009	62	Analogy	0.09	0.09	0.12
32	[49]	Albrecht	2009	24	Analogy	-0.53	0.07	-1.56
33	[49]	Desharnais	2009	77	Analogy	-0.21	0.05	-0.45
34	[49]	Maxwell	2009	62	Analogy	-0.29	0.09	0.40
35	[49]	ISBSG	2009	118	Analogy	-0.08	0.04	-0.17

[a]. Used to separate different estimation method comparisons within the same study; b. Number of projects in a dataset used in the model; c. Identifies what type of self-developed model is compared with a regression-based model; d. Mean Difference between the *MMRE* of the analogy-based and regression model (comparisons with IDs 13 and 14 use the median MRE) A negative value indicates that the analogy-based model on average was more accurate than the regression model; e. The *Standard Error* of d; f.. A measure of the effect size, defined as MD_{MMRE} divided by $SD(MD_{MMRE})$.

D. Discussions/Limitations

The results from the published comparisons of analogy- and regression-based models were heterogeneous and highly inconsistent (I^2 = 98%, see [50] on how to interpret this meta-analysis measure). This conclusion inconsistency makes it debatable whether a meta-analysis of the accuracy difference of these two types of estimation models is meaningful to be conducted or not. The lack of conclusion consistency may, for example, indicate that there are underlying contextual differences that account for nearly all the differences in the conclusions. One such context difference may be the degree of vested interest and, as a consequence, confirmation bias. Other variables, such as heterogeneity in the datasets characteristics, may also contribute to the lack of conclusion consistency.

Our calculation of the values of $SD(MD_{MMRE})$, the standard deviation of the MMRE differences, may be quite inaccurate for several of the data sets. We nevertheless believe that our results are sufficiently accurate for the purpose of the analysis, i.e., to study the effect of confirmation bias. Our approximations for $SD(MD_{MMRE})$ are, for example, not likely to give either analogy- or regression-based models any advantage, just to increase the inaccuracy of the effect size measurement. The decisions on which value of $SD(MD_{MMRE})$ to use, when not possible to derive directly from the study, were done "blindly", i.e., without knowing the outcome of the estimation method comparison, to avoid that we ourselves would be subject to confirmation bias.

The purpose of our meta-analysis is not to try to find out which is the best alternative of analogy- and regression-based estimation models or to investigate the meaningfulness of the comparisons made. Instead, we study to what degree researchers tend to report results confirming what they wish to be true, i.e., that their own model is the most accurate one when it is compared with others. We believe that confirmation bias is well demonstrated in our analysis.

V. Conclusions

We provide evidence on the presence of confirmation bias in two software engineering contexts. In the first context we observe that IT professionals find support for their preferred contract type in neutral, randomly created, project data. In the second context, we report a tendency of effort estimation model researchers to report evidence that favour their own model in comparison to others. Both results, we argue, provide evidence suggesting the importance of being aware of confirmation bias in software engineering contexts. While the existence of confirmation bias may come as no surprise, given its appearance in many other domains and contexts, we believe that our studies contribute with evidence of its existence in two important contexts where it has not been reported earlier.

Our results imply that the elicitation of practice-based software development experience, e.g., about the effect of software contracts, and research including one's own methods or models, need proper method and validation practises to avoid harmful effects of confirmation bias. Existing methods and validation practises, e.g., evidence-based software engineering [2], may not currently provide sufficient support to avoid confirmation bias. We plan to follow up the research with further research on approaches and support that reduce the risk of harmful effects of conformation bias.

References

[1] A.S. Elstein, Human factors in clinical judgment: Discussion of Scriven's clinical judgment. Engelhardt HT, Spicker SF, Towers B. Clinical judgment: a critical appraisal. Dordrecht: Reidel, 1979.

[2] T. Dybå,, B. Kitchenham, and M. Jørgensen, "Evidence-based software engineering for practitioners," IEEE Software, vol. 22(1), pp. 58-65, 2005.

[3] J. Klayman, "Varieties of confirmation bias," Psychology of learning and motivation, vol. 32, pp. 385-418, 1995.

[4] R. Austen, and S. Beecham, "A follow-up empirical evaluation of evidence based software engineering by undergraduate students," 12th International Conference on Evaluation and Assessment in Software Engineering, University of Bari, Italy, 2008.

[5] G. Stein, and M. Jørgensen, "A framework for the analysis of software cost estimation accuracy," Proceedings of the 2006 ACM/IEEE International Symposium on Empirical Software Engineering, ACM, pp. 58–65, 2006.

[6] K.A. De Graaf, P. Liang, A. Tang, and H. Van Vliet, "The impact of prior knowledge on searching in software documentation," Proceedings of the 2014 ACM Symposium on Document Engineering, ACM, pp. 189–198, 2014.

[7] F.P. Deek, and J.A.M. McHugh, "Cognitive and social psychology in collaboration," Computer-Supported Collaboration, Springer, pp. 7–26, 2003.

[8] C. Bird, A. Bachmann, E. Aune, J. Duffy, A. Bernstein, V. Filkov, and P. Devanbu, "Fair and balanced?: Bias in bug-fix datasets," Proceedings of the the 7th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering, ACM, pp. 121–130, 2009.

[9] B.E. Teasley, L.M. Leventhal, C.R. Mynatt, and D.S. Rohlman, "Why software testing is sometimes ineffective: Two applied studies of positive test strategy," Journal of Applied Psychology, vol. 79(1), pp. 142-155, 1994.

[10] L.M. Leventhal, B.E. Teasley, and D.S. Rohlman, "Analyses of factors related to positive test bias in software testing," International Journal of Human-Computer Studies, vol. 41(5), pp. 717-749, 1994.

[11] G. Çalikli, and A. Bener, "Empirical analyses of the factors affecting confirmation bias and the effects of confirmation bias on software developer/tester performance," Proceedings of the 6th International Conference on Predictive Models in Software Engineering, ACM, 2010.

[12] G. Çalikli, and A. Bener, "Preliminary analysis of the effects of confirmation bias on software defect density," Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement, vol. 1, New York, NY, USA: ACM, pp. 1–68, 2010.

[13] G. Çalıklı, and A.B. Bener, "Influence of confirmation biases of developers on software quality: An empirical study," Software Quality Journal, vol, 21(2), pp. 377-416, 2013.

[14] S.L. Linda, "Managing user expectations in information systems development," http://www.waset.org/publications/14658.

[15] M. Jørgensen, "Myths and Over-simplifications in Software Engineering," Lecture Notes on Software Engineering, vol. 1(1), 2013.

[16] R. Legoux, P.M. Leger, J. Robert, and M. Boyer, "Confirmation biases in the financial analysis of IT investments," Journal of the Association for Information Systems, vol. 15(1), pp. 33-52, 2014.

[17] R. Paul, "Possible core theories for software engineering." 2nd SEMAT Workshop on a General Theory of Software Engineering (GTSE), IEEE, pp. 35–38, 2013.

[18] M. Jørgensen, and E. Løhre, "First impressions in software development effort estimation: Easy to create and difficult to neutralize," 16th

International Conference on Evaluation & Assessment in Software Engineering, pp. 216-222, 2012.

[19] M. Shepperd, D. Bowes, and T. Hall, "Researcher bias: The use of machine learning in software defect prediction," IEEE Transactions on Software Engineering, vol. 40(6), pp. 603-616, 2014 .

[20] M. Jørgensen, T. Dybå, K. Liestøl, and D.I.K. Sjøberg, "Incorrect results in software engineering experiments: How to improve research practices," unpublished.

[21] U. Badenfelt, "Fixing the contract after the contract is fixed: a study of incomplete contracts in IT and construction projects," International Journal of Project Management, vol. 29(5), pp. 568-576, 2011.

[22] A. Arora, V.S. Arunachalam, J. Asundi, and R. Fernandes, "The Indian software services industry," Research policy, vol. 30(8), pp. 1267-1287, 2001.

[23] Y. Lichtenstein, "Puzzles in software development contracting," Communications of the ACM, vol. 47(2), pp. 61-65, 2004.

[24] A. Gopal, K. Sivaramakrishnan, M.S. Krishnan, and T. Mukhopadhyay, "Contracts in offshore software development: An empirical analysis," Management Science, vol. 49(12), pp. 1671-1683, 2003.

[25] M. Jørgensen, "The effects of the format of software project bidding processes," International Journal of Project Management, vol. 24(6), pp. 522-528, 2006.

[26] B. Eckfeldt, R. Madden, and J. Horowitz, "Selling agile: Target-cost contracts" Proceedings of Agile Conference, IEEE, 2005.

[27] C. Mair, and M. Shepperd, "The consistency of empirical comparisons of regression and analogy-based software project cost prediction," International Symposium on Empirical Software Engineering, IEEE, 2005.

[28] J. Wen, L. Shixian, L. Zhiyong, H. Yong, and H. Changqin, "Systematic literature review of machine learning based software development effort estimation models," Information and Software Technology, vol. 54(1), pp. 41-59.

[29] G.V. Glass, "Primary, secondary, and meta-analysis of research," Educational researcher, 1976, pp. 3-8.

[30] G. Cumming, "Understanding the new statistics: Effect sizes, confidence intervals, and meta-analysis," Routledge, 2012.

[31] G.R. Finnie, G.E. Wittig, and J.-M. Desharnais, "A comparison of software effort estimation techniques: Using function points with neural networks, case-based reasoning and regression models," Journal of Systems and Software, vol. 39(3), pp. 281-289, 1997 .

[32] M. Shepperd, and C. Schofield, "Estimating software project effort using analogies," IEEE Transactions on Software Engineering, vol. 23(11), pp. 736-743, 1997.

[33] E. Stensrud, and I. Myrtveit, "Human performance estimating with analogy and regression models: An empirical validation," International Software Metrics Symposium, Bethsda, MD: IEEE Comput. Soc, Los Alamitos, CA, USA, 1998.

[34] F. Walkerden, and R. Jeffery, "An empirical study of analogy-based software effort estimation," Empirical Software Engineering, vol. 4(2), pp. 135-158, 1999.

[35] L.C. Briand, K. El Emam, D. Surmann, I. Wieczorek, and K.D. Maxwell, "An assessment and comparison of common software cost estimation modeling techniques," Proceedings of the 1999 International Conference on Software Engineering, Los Angeles, USA: ACM, New York, pp. 313-323,1999.

[36] L.C. Briand, T. Langley, and I. Wieczorek, "A replicated assessment and comparison of common software cost modeling techniques," International Conference on Software Engineering., Limerick, Ireland: ACM, New York, 2000.

[37] L. Angelis, and I. Stamelos, "A simulation tool for efficient analogy based cost estimation," Empirical Software Engineering, vol. 5(1), pp. 35-68, 2000 .

[38] C. Mair, et al., "An investigation of machine learning based prediction systems", Journal of Systems and Software, vol. 53(1), pp. 23-29, 2000.

[39] D.R. Jeffery, M. Ruhe, and I. Wieczorek, "A comparative study of two software development cost modeling techniques using multi-organizational and company-specific data," Information and Software Technology, vol. 42(14), pp. 1009-1016, 2000.

[40] D.R. Jeffery, M. Ruhe, and I. Wieczorek, "Using public domain metrics to estimate software development effort," International Software Metrics Symposium., London, UK: IEEE Comput. Soc, Los Alamitos, CA, USA, 2001.

[41] I. Wieczorek, and M. Ruhe, "How valuable is company-specific data compared to multi-company data for software cost estimation?," International Software Metrics Symposium., Ottawa, Canada: Fraunhofer Inst. for Exp. Software Eng. Kaiserslautern Germany, 2002.

[42] S.G. MacDonell, and M.J. Shepperd, "Combining techniques to optimize effort predictions in software project management," Journal of Systems and Software, vol. 66(2), pp. 91-98, 2003.

[43] E. Mendes, and B. Kitchenham, "Further comparison of cross-company and within-company effort estimation models for web applications," Proceedings 10th International Symposium on Software Metrics, IEEE, 2004.

[44] S.-J. Huang, and N.-H. Chiu, "Optimization of analogy weights by genetic algorithm for software effort estimation," Information and Software Technology, vol. 48(11), pp. 1034-1045, 2006.

[45] G. Costagliola, S. Di Martino, F. Ferrucci, C. Gravino, G. Tortora, and G. Vitiello, "Effort estimation modeling techniques: A case study for web applications," Proceedings of the 6th International Conference on Web engineering, ACM, pp. 9-16, 2006.

[46] N.-H. Chiu, and S.-J. Huang, "The adjusted analogy-based software effort estimation based on similarity distances," Journal of Systems and Software, vol . 80(4), pp. 628-640, 2007.

[47] E. Mendes, , S. Di Martino, F. Ferrucci, C. Gravino, "Cross-company vs. single-company web effort models using the Tukutuku database: An extended study," Journal of Systems and Software, pp. 673-690, 2008.

[48] Y.F. Li, M. Xie, and T.N. Goh, "A study of mutual information based feature selection for case based reasoning in software cost estimation," Expert Systems with Applications, vol. 36(3) pp. 5921-5931, 2009.

[49] Y.F. Li, M. Xie, and T.N. Goh, "A study of the non-linear adjustment for analogy based software cost estimation," Empirical Software Engineering, vol. 14(6), pp. 603-643, 2009.

[50] J.P. Higgins, S.G. Thompson, J.J. Deeks, and D.G. Altman, "Measuring inconsistency in meta-analyses," BMJ, vol. 327(7414), pp. 557-560, 2003.