Integrated Software Effort Estimation: a Hybrid Approach

Prerna Singal[1*], Prabha Sharma[1], A Charan Kumari[2]
[1]The NorthCap University Gurugram, India
[2]Dayalbagh Educational Institute, Dayalbagh, Agra, India
E-mail: prernasingal@yahoo.com, prabhasharma@ncuindia.edu, charankumari@yahoo.co.in
*Corresponding author

Keywords: Software effort estimation, Risk exposure, Risk management, Evolutionary algorithms, ABC, PSO, GLBPSO

Received: November 14, 2022

Risks associated with delivery of a software project and the effort spent on managing these risks are well researched topics. Very few have included this extra effort termed as risk exposure of a project, in the software effort estimate of a project. This research proposes to improve the accuracy of software effort estimates by integrating the risk exposure with the initial effort estimate of the project. A function to calculate integrated effort estimates has been defined and evolutionary algorithms ABC, PSO and GLBPSO have been used to optimize the MMRE. The approach has been tested on two datasets collected from industry, one for waterfall projects, and another for agile projects. For both the datasets, integrated effort estimates were more accurate on account of MMRE, standardized accuracy, effect size and R^2, than the initial effort estimates. Evolutionary algorithms also gave the optimum weight values at which the MMRE was optimal for both the datasets. These weight values determine the contribution of risk associated with each project cost factor in the risk exposure of the project. Integrated effort estimates have been found to be more accurate, reliable, and comprehensive than the initial effort estimates. Application of evolutionary algorithms help in reducing any bias in the integrated effort estimates.

Povzetek: Raziskava predlaga integrirano oceno dela pri razvoju programske opreme z upoštevanjem izpostavljenosti tveganjem in z uporabo evolucijskih algoritmov, kar izboljša tončnost ocen.

1 Introduction

Software Effort estimation is the basis of software project management. But it is also one of the most challenging aspects of software project management. For a long time now, project management experts have been looking for estimation techniques which provide comprehensive effort estimates with high accuracy, which is required for delivering a project within schedule and within a budget. Despite the ongoing advancements and research in the field of software effort estimation, Standish group [1] and the International society of parametric analysis [2] report that two-thirds of the projects face budget overruns and schedule delays. The challenge lies in the accurate projection of project cost factors during the initial stages of the project delivery, and managing the uncertainties encountered during the development of the final product [3]. Uncertainties in the project cost factors lead to various risks in the software development process, which need to be identified, managed, and controlled for a successful software project delivery.

The risks associated with a software project are due to factors like volatility in project requirements, availability of experienced personnel, ever-changing technology and many other project cost factors [4]. These project cost factors play a significant role in the effort estimation process as well as in project risk identification and management process [5]. Most often, risk planning and its management are treated as a separate activity from the software effort estimation task [6]. Risks are identified, analyzed, mitigated, and controlled but their impact on the effort estimate of a project is not considered. This might result in over-optimism and over-confidence in the software effort estimates of projects [7]. Thus, there is a need to integrate the risk management process in the effort estimation process for a more accurate, comprehensive, and fair effort estimate.

Managing risks at the project level would require effort towards identification, analysis, mitigation, and control of the risks associated with the project. Projects with a pessimistic effort estimate can see a reduction in the overall effort required for the project if risk management is done along with effort estimation. Projects where the effort estimation has been done in an over optimistic manner will need extra effort to manage the risks [7]. This effort which is required for the risk management process is referred as the risk exposure of the project [6]. This research proposes to include the risk exposure of the project in the effort estimate of the project. The effort estimate would now include the effort required for development of the project along with the effort required in the risk management process [8].

In the proposed approach, the cost of risk exposure is calculated according to the procedure defined by Kitchenham and Linkman [9] in 1997. They suggested that the uncertainties in the software development process cause inaccuracies in the software effort estimate irrespective of the effort estimation technique being used. The effort estimation is done in the beginning of the project when not many details are available regarding the various project cost factors impacting the effort estimate of the project. They have categorized the sources of estimate uncertainties into three types: measurement error, model error and assumption error. Assumption errors occur in the evaluations of the input parameters due to the inherent uncertainties associated with these parameters. This assumption error is the risk which creeps into the project when project cost factor values do not meet the assumed level. Kitchenham and Linkman [9], have linked the risk exposure of the project to the error in assumption of the project cost factor values. They have suggested to collect alternative cost values of the project cost factors along with the probabilities of not meeting the initial cost factor values. For each cost factor, two values are determined, the initial assumed value, and the alternative value that the project cost factor might attain during the execution of the project. Kitchenham and Linkman compute the effort/ cost of managing risk as follows: For each factor difference in the initial and alternate cost is multiplied by the corresponding probability of not meeting the initial cost of that factor. This is the cost of managing risk related to that factor. Sum of all these costs define the total effort/ cost for managing the risk of the whole project or the risk exposure of the project.

The authors in an earlier paper [8] have added the cost of risk exposure to the initial effort estimate of the project to obtain an integrated effort estimate$(IE_{initial})$. Equation (5) gives the formula to calculate$(IE_{initial})$.

The probabilities and project cost factor values in the formula, proposed by Kitchenham and Linkman, are based on expert judgment, which may result in a biased effort estimate. In the proposed approach, these probabilities are treated as variables and these variables/ weights are then optimized using evolutionary algorithms like Artificial Bee Colony (ABC), Particle Swarm Optimization (PSO), and a hybrid global local binary particle swarm optimization algorithm (GLBPSO), to obtain more accurate and fair effort estimates. This reduces the dependency of the model on expert judgment and removes any bias in the integrated effort estimates due to the probabilities. This research determines the optimal value of the estimated effort and the weights which determine the impact of risks due to each cost factor on the estimated effort. The proposed approach has been tested for waterfall model delivery projects as well as projects using agile delivery.

The main contributions of this research are as follows:

- Risk exposure of a project is added to the effort estimate of the project which gives an integrated effort estimate for the project.
- A weighted function for calculating the integrated effort estimate has been defined.
- Two questionnaires have been prepared: one based on CoCoMo II project cost factors for waterfall delivery projects, and another based on project cost factors by Ziauddin for agile delivery projects. Based on the responses received for these questionnaires, two datasets have been collected: ‘Waterfall model dataset’ and ‘Agile model dataset’.
- Integrated effort estimate has been optimized using evolutionary algorithms ABC, PSO and GLBPSO.

2 Related work

The need for risk assessment and risk control in software development projects was highlighted by Boehm in 1989 [10]. His work proposed a framework for identification of software validation and verification activities and the effort which would go into the risk management process. They attributed 40%-50% of the software project costs to rework costs, and 80 % of that rework costs to the highest risk factor associated with the project. Fairley in 1994, used regression cost modelling of the effort estimates of the past projects in the organization to determine the risks associated with a software project [11]. Residual error due to the difference between the estimated effort and actual effort was used to determine the project cost factors which impacted the software effort estimates. He suggested controlling and mitigating these project cost factors to manage the project risk.

In 1997, Madachy proposed a heuristic to calculate the cost of risk exposure of the project using the project cost factors [12]. He identified risk rules based on the CoCoMo cost factors to assign levels to the risks identified in the project. The cost of risk exposure of the project was calculated based on the risk level and the contributing project cost factor values. Briand et al. introduced a hybrid model for project effort estimation, risk assessment and benchmarking without relying on the historical data for project effort estimate [13]. The proposed model based its effort estimate on productivity of the project and calculated the project cost overheads based on a questionnaire, which would be filled by the experts associated with the project.

In 2006, Jantzen estimated the impact of project risks on project effort estimate, project duration and project quality. His work emphasized on re-estimating and re-planning the software project during its execution, based on the various risks, their level and risk status at various stages of the project [14]. Huang et al. in 2006, proposed an effort estimation technique based on the fuzzy and uncertain nature of the project cost factors. The effort was estimated using fuzzy decision tree where along with the effort estimate of the project, the estimation error was used for risk analysis and management [15]. Manalif in 2013, proposed a fuzzy expert – CoCoMo model which added a contingency to the estimated effort based on the project cost factors of the CoCoMo model [16]. The contingency effort was kept separate from the total effort estimate of the project.

In 2017, Aslam et al. considered the risks associated with rich mobile application development projects developed using agile methodology [17]. Along with risk factors, they also included the quality aspect of the project in the effort estimate of the project which enabled the development effort estimation at multiple quality levels. Their work was limited to project on rich mobile application development.

In 2018, Koutbi & Idri proposed inclusion of the cost of risk management in the effort estimation process at the organization level instead of handling it at a specific project level [18]. They argued that risk is better handled and mitigated over a portfolio of projects which improves the effort estimation process of the organization. On the other hand, organizations can have projects of varied nature, for which the project cost factors contributing to risks may vary.

In 2019, Ramakrishnan et al. built a multilayer perceptron model to estimate the software development effort. The model included project risk score in the effort estimation process [19]. They used an enhanced gradient boosting technique which decreased the standard deviation of the residuals indicating better effort estimation results.

In 2020, Michael Kataev et al. [20], reiterated the importance of including the cost of risk management in the software development process for on time and in budget software project deliveries. They included the internal and external risks which impacted the overall financial health of the organization.

It is clear that researchers have focused on integrating risk management process with the software effort estimation process, but none have included the effort spent on risk management in the effort estimate of the project itself. Thus, this research tries to bridge this gap and analyses the impact of risk exposure on the effort estimate of the project. Also, the probability of risk occurrence is optimized using evolutionary algorithms ABC, PSO, GLB-PSO.

Rest of the paper is organized as follows: Section 3 details the proposed approach of integrating the impact of risk on the software effort estimate and optimizing it using evolutionary algorithms. It outlines the algorithms used in the proposed approach. Section 4 gives the details of the datasets collected for this research. Section 5 contains the details of calculating integrated effort estimates for waterfall and agile projects. Section 6 gives the details of experimental setup and algorithms for ABC, PSO and GLBPSO. It contains the details of the fitness function used in the algorithms and explains the evaluation criteria used for comparing the proposed approach with the baseline effort estimation techniques. Section 7 presents and compares the experimental results obtained for both the waterfall and agile delivery projects. Section 8 points out some threats to validity of the proposed approach. Section 9 draws the conclusion, and Section 10 describes the scope for future work.

3 Theoretical background

This section explains the risk exposure of a project, research questions which motivated the research and presents the proposed model for calculating integrated effort estimates.

3.1 Risk exposure

The risk exposure of the project is the total effort required to identify, mitigate, and control the risks in the project which occur due to various project cost factors. Risk exposure due to each cost factor of the project is calculated separately and then added together to give the total risk exposure of the project. These risks arise due to the uncertainties associated with the software projects. The initial effort estimate of the project is determined based on certain assumptions made regarding the project cost factors like reliable requirements, team communication, availability of hardware and software resources, and expertise & experience level of the team. Since very little information is available when these assumptions are made at the beginning of the project, quite often these assumptions are not met giving rise to risks which may increase or decrease the effort required to develop the project.

According to Kitchenham and Linkman [9], for a project with n cost factors, initial effort estimate $(E_{initial})$ required to develop a project is calculated using the assumed project cost factor values at the beginning of the project. To calculate the risk exposure of the project, risk exposure due to each project cost factor is added. Risk exposure of the i^{th} project cost factor is calculated by multiplying the probability of not meeting the initial level $(p_{i,alter})$ of the i^{th}project cost factor with difference between the effort estimated $(E_{i,alter})$ at the alternative level of the i^{th}project cost factor and the initial estimated effort$(E_{initial})$. The risk exposure (E_{risk}) of a project can be calculated using the equation given below:

$$E_{risk} = \sum_{i=1}^{n}(E_{i,alter} - E_{initial}) \times p_{i,alter} \quad (1)$$

This risk exposure is the extra effort that would be needed for risk management and planning of the project.

3.2 Proposed approach

In the proposed approach, the risk exposure of a project has been added to the initial effort estimated required to develop the project. Thus, according to the proposal in this research, the total effort estimate of a project must reflect the effort involved in managing the various risks encountered during the completion of the project. If $E_{initial}$ denotes the initial effort estimate of a project, E_{risk} is the risk exposure calculated using equation (1), then the integrated effort estimate $(IE_{initial})$ of the project can be determined using the equation given below:

$$IE_{initial} = E_{initial} + E_{risk} \quad (2)$$

Substituting the value of E_{risk} in equation (2) from equation (1) will give us the initial integrated software development effort estimate for a project.

$$IE_{inital} = E_{initial} + \sum_{i=1}^{n}(E_{i,alter} - E_{initial}) \times p_{i,alter} \quad (3)$$

This estimate will now include the effort that would be required for development as well as risk management, and planning of the project. Values of the probabilities $p_{i,alter}$ of not attaining the assumed initial level of project cost factors in equation (3) can be ascertained from the program manager or the expert responsible for effort estimation of the project. It is possible that the values of the probabilities $\{p_i\}_{i=1}^{n}$ may be biased, and hence may not accurately estimate the cost of risk management.

In this paper, p_i's are treated as variables between 0 and 1 and then their optimal values are computed using evolutionary algorithms such as ABC, PSO and GLBPSO, so that the estimated cost is as close as possible to the actual cost of a project. Therefore, in equation (1), the p_i's have been replaced by w_i's. The risk exposure of a project is expressed as the weighted mean of the risk exposures due to all the project cost factors using the formula given below:

$$E_{risk} = \sum_{i=1}^{n}(E_{i,alter} - E_{initial}) \times w_i \tag{4}$$

The formula in equation (3) for calculating integrated software development effort estimate of a project (IE) can now be expressed using the equation given below:

$$IE = E_{initial} + \sum_{i=1}^{n}(E_{i,alter} - E_{initial}) \times w_i \tag{5}$$

The w_i in equation (5) can now be optimized using evolutionary algorithms.

In this research three evolutionary algorithms have been experimented with: ABC, PSO and GLBPSO. Previous research on software effort estimation has demonstrated promising results with the application of evolutionary algorithms such as ABC, PSO, and GLBPSO [21-23]. These algorithms have been effective in optimizing complex, non-linear software project effort estimation problem.

3.3 Artificial bee colony optimization

Artificial bee colony (ABC) is a metaheuristic algorithm proposed by Karaboga et al. [24], which is based on the intelligent social behavior of the honeybee swarm. ABC algorithm employs collaborative trial and error approach to identify honeybee swarm. The ABC optimization algorithm is driven by peer-to-peer learning behavior of social colonies and reaches the optimal solution following an iterative process. There are four phases in the ABC algorithm: initialization phase, employed bee phase, scout phase and onlooker bee phase. In the initialization of the population, ABC generates a uniformly distributed population of solutions where each solution is a dimensional vector. In the proposed approach, the weight vector $(w_i's)$ described in Section 6.1 will represent the solution vector. Each w_i varies between 0 and 1. The employed bees update the current solution based on their own experience and fitness value of the new solution. If the new solution has a higher fitness value than the current solution, the bee selects the new solution and discards the current one. In this research, weight vector which gives the lowest MMRE will be selected.

3.4 Particle swarm optimization

Particle swarm optimization (PSO) algorithm also belongs to the family of swarm intelligence algorithms, and was first proposed by J. Kennedy and R. Eberhart in 1995 [25]. PSO algorithm models the social behavior of flocking of birds or school of fish to optimize nonlinear functions. Each particle/bird which represents the solution to the problem has a position and velocity associated with it. In the algorithm, particles change their position by adjusting their velocity either to seek food, avoid predators or to identity optimized environmental parameters. Also, each particle memorizes its best position during the process and communicates it to other particles in the swarm. So, the velocity of a particle is modified using the flying experience of the particle itself and the flying experience of the whole group, termed as global best PSO. For this research, the particles are the weight vectors which determine the risk exposure due to each cost factor. The objective is to minimize the MMRE of the project in the dataset.

3.5 Global local binary PSO optimization

An improved PSO algorithm was proposed by Rita Chhikara et al. [26] to overcome the disadvantages of the global best PSO. The algorithm Global local binary PSO (GLBPSO) integrated the global best PSO with local best PSO and dynamically changed the population size using (Hope/ Rehope). The algorithm begins by using the global best PSO and if the value of the fitness function does not change for two consecutive iterations, local best PSO is applied with a neighborhood size of 4. The particles move towards the best solution in the neighborhood by communicating with their four immediate neighbors. If the local best PSO does not improve the fitness function in three consecutive iterations, this indicates stagnation in the search process. To avoid this stagnation, hope/ rehope is applied on the population. For kth iteration, for particles having marginal distance (less than 0.01) among themselves, only the particle with higher fitness function value is retained in the population. However, it could lead to decrease in population size. To avoid such a situation, the population size is checked after each iteration, and if it reduces to less than 50% of its original size, then the population size is increased randomly by 30%. This eliminates the bad performing particles and at the same time revives the hope for a better solution. These steps are repeated until the stopping criteria is met or the set number of iterations have been executed.

3.6 Research questions

This paper aims to provide the experimental evidence to answer the research questions given below:

RQ1: *Does the accuracy of effort estimate of the project improve by adding the cost of risk exposure to the initial estimated effort of the project?*

This research integrates the effort that goes into risk management and control into the initial effort estimate of the project. A function $IE_{initial}$ as given in equation (3) is proposed, to determine the integrated effort estimates for software projects. The weighted cost of the risk exposure due to each project cost factor is added to the initial effort estimate of the project. The research aims to find out whether these integrated effort estimates are more accurate than the initial effort estimates.

RQ2: *What is the impact of bias on risk exposure of the project?*

Some biases in the cost of risk exposure might have been introduced due to $p_{i,alter}$ in the integrated effort estimates $(IE_{initial})$, as given in equation (3). The values of $p_{i,alter}$ are based on expert judgment and have been collected based on a questionnaire for the respective effort estimation models: waterfall and agile. These values might be biased as per the expert's understanding and knowledge. This research reduces these biases by obtaining optimal values of the $w_i's$ $(p_{i,alter})$ by using evolutionary algorithms ABC, PSO and GLBPSO. In the weighted function $IE_{initial}$ in equation (3), the probabilities are replaced with weights to obtain the weighted function IE as given in equation (5). IE is optimized using evolutionary algorithms, which obtain optimal values of the weights $w_i's$.

RQ3: *Can project cost factors be ranked with respect to their risk exposure?*

Using equation (5), optimal values of $w_i's$ are obtained. These weights determine the contribution of each project cost factor in the total cost of risk exposure of the project. Higher the value w_i implies that the associated project cost factor will contribution more to the total risk exposure of the project. This information can be used by project managers to identify the project cost factors which need better management of the risks associated with them.

4 Data collection

The proposed integrated effort estimation model was tested on data collected from an Indian IT firm involved in software development, maintenance, and consultancy. Two types of projects were considered for the research – projects with Waterfall delivery model [27] and projects with Agile delivery model [28]. Experts who handled the projects were interviewed over a span of 1 year and data was collected based on a questionnaire. Two separate questionnaires were prepared – one for each delivery model, Waterfall and Agile. Experts included project managers, technical architects, analysts, and developers. These experts were directly involved in the project effort estimation process. Experts from over 75 different projects were interviewed. 45 projects followed the Waterfall delivery model, and rest of the 30 projects were working on the agile principles. Projects were from varied domains covering banking, healthcare & pharmaceutical, and Insurance. Waterfall model questionnaire had 69 fields to be filled while the agile questionnaire had 45 fields. Table 1 shows a general format of the questionnaire.

Table 1: Questionnaire format

Questionnaire					
Project Id	KLOC / Story Points	Actual Effort (Man Months)	Cost Factor		
			Initial Level	Probability of not meeting the initial Level	Alternative Level

Questionnaire for the Waterfall model was based on CoCoMo II project cost factors. There were 5 scale factors and 17 cost factors identified in the CoCoMo II Model. All the scale and cost factors have been calibrated at five levels: very low, low, nominal, high, very high and extra high [29]. The questionnaire focused on lines of code in the project (measured in KLOC), actual effort spent (Man Months), scale factors and the cost factors. For the scale and cost factors three inputs were taken from the experts – their initial assumed level while estimating effort, probability of not meeting that assumed level and an expected alternative level. The dataset thus collected is referred to as the "Waterfall model" dataset.

Similarly, questionnaire for the Agile model was based on the frictional and dynamic forces suggested by Ziauddin [30] . In the Agile delivery model, the stories are delivered in sprints. This questionnaire collected responses for one sprint in each project covering story size, story complexity, actual velocity, sprint time, dynamic factors, and frictional factors related to the project. Size of the story was rated on a scale of 1-5 based on the effort required for the development of the story. Complexity was also rated on a scale of 1-5 depending upon the nature of the work and complexity of technical and non-technical requirements. There are 4 frictional factors and 9 dynamic factors identified in the model which impact the effort estimates of agile projects. Ziauddin has laid down guidelines to determine the size and complexity of the story on a scale of 1-5. The questionnaire focused on the sprint time, story size, actual velocity, story complexity and the variable forces (dynamic & frictional factors) – their initial assumed levels during effort estimation, probability of not meeting that assumed level and the expected alternative level. The dataset thus collected is referred to as the “Agile model” dataset.

5 Calculation of Integrated effort estimates

During the initial project planning phase, an estimate of the effort ($E_{initial}$) involved in development of a project is done with certain assumptions regarding the project cost factor values. With these assumptions the project cost factors are assigned certain values and then the initial effort is estimated. This effort can be estimated using any established effort estimation technique depending upon the nature of the project, its delivery model and local environment. The effort calculated is expressed in Man Months (MM), which is the average effort spent by one person for a month. Initial effort is calculated for both the datasets: waterfall and agile using the estimation techniques described below.

5.1 Waterfall model dataset

For the “Waterfall model” dataset, the project data was collected based on CoCoMo II project cost and scale factors [29]. So, the initial effort values ($E_{initial}$) for a project were calculated using the CoCoMo II effort estimation formula given below:

$$E_{initial} = A \times Size^{S} \times \prod_{i=1}^{n} EM_{i,initial} \quad (1)$$

where
$$S = B + 0.01 \times \sum_{k=1}^{5} SF_{k,initial} \quad (2)$$

A is a constant whose value can be calibrated according to the project’s local environment. It has been established that CoCoMo II estimates the software development effort more accurately when the constant A is calibrated according to the organization’s productivity and activity distributions [31]. This research uses the standard value of 2.94 proposed in the CoCoMo II Model. B is also a constant set at 0.90 in the CoCoMo II model. $EM_{i,initial}$ denotes the project effort multiplier for the i^{th} project cost actor which impacts the estimated effort of the project. There are 17 cost factors (n=17) in the CoCoMo II Model. Size of the project is determined in KLOC. $SF_{k,initial}$ are the five scale factors. From the expression for S, it can be observed that that SF_k’s make the effort grow exponentially. Substituting the values of $E_{initial}$ from equation (6) in equation (5), the integrated effort estimate (IE), for Waterfall model dataset can now be expressed as below:

$$IE = A \times Size^{S} \times \prod_{i=1}^{n} EM_{i,initial} + \sum_{i=1}^{n} \left\{ \left(\left(A \times Size^{S} \times \prod_{i=1}^{n} EM_{i,alter} \right) - \left(A \times Size^{S} \times \prod_{i=1}^{n} EM_{i,initial} \right) \right\} \times w_i \right. \quad (3)$$

where $EM_{i,initial}$ represent the project cost factor values at the initial assumed stage, and $EM_{i,alter}$ represents the project cost factor values at the alternative level.

5.2 Agile model dataset

For the “Agile model” dataset, initial effort value $E_{initial}$ for a project was calculated using the model proposed by Ziauddin [30]. The model estimates the effort for a sprint using the story size, complexity, dynamic and frictional factors. The software product to be developed, is described in the form of user stories creating a product backlog owned by the product owner, usually a representative of the customer for whom the product is being developed. The team delivers the selected user stories at completion of each sprint. As opposed to waterfall model where the manager is responsible for estimating the effort in the planning phase, in agile approach the team members decide on the effort that will go into the delivery of the user story at the beginning of each sprint. Team members estimate the required effort based on their experience, story size, complexity, and project cost factors. The effort is expressed in terms of story points, where one story point corresponds to a day’s work for the team member. The project cost factors might change during the sprint execution leading to the uncertainty in effort estimate by the team member. These project cost factors account for the risks associated with the project which impact the effort estimate of the sprint. Steps given below were followed to calculate the initial effort estimate.

a) **Effort for a story:** For each story, the effort dispensed towards the development of the story was calculated using the formula given below:

$$ES\ (Effort\ for\ a\ story) = story\ size \times story\ complexity \quad (4)$$

This effort estimate of the story is expressed in story points.

b) **Effort for the whole sprint:** The estimated effort for all the stories in the sprint is added to get the effort estimate of the sprint, using the equation given below:

$$E\ (Effort\ for\ whole\ sprint) = \sum_{i=1}^{n} ES_i \qquad (5)$$

where, n is the number of stories being delivered in the sprint. Now, the effort for the whole sprint is in story points.

c) **Variable factors**: From the agile project dataset, initial values of the frictional and dynamic factors were used to calculate the impact of variable factors on the initial effort estimate. The impact was calculated using the formula given below:

$$D\ (Variable\ Forces) = \prod_{k=1}^{4} Frictiona\ factors_i \times \prod_{m=1}^{9} Dynamic\ factors_j \qquad (6)$$

d) **Agile Velocity:** In this step, the velocity for each sprint in the project was determined based on the estimated sprint effort E, sprint time T and variable forces D in the sprint, using the formula given below:

$$V\ (Velocity) = \left(\frac{E}{T}\right)^{D} \qquad (7)$$

In Agile delivery, the velocity of the project is improved and stabilized over various sprints [32]. This stability in velocity will depend on the project cost factors, in this case dynamic and friction factors. These factors change often during the execution of the sprint thus leading to uncertainties in the estimated effort. These uncertainties are the risks associated with the project which need to be addressed during the project execution. The effort that goes into the control and mitigation of these risks has been accounted for in the estimated effort in the proposed approach. The formula for integrated effort ($IE_{initial}$) is given in equation (13).

e) **Effort estimate**: From equation (12), formula for $E_{initial}$ can be obtained as follows:

$$E_{initial} = E = (V)^{\frac{1}{D}} \times T \qquad (8)$$

Substituting the values of $E_{initial}$ from equation (13) in equation (5), the integrated effort estimate for the Agile model dataset can now be expressed as below:

$$IE = (V)^{\frac{1}{D}} \times T + \sum_{i=1}^{n} \left\{ \left((V)^{\frac{1}{D_{i,alter}}} \times T \right) - \left((V)^{\frac{1}{D}} \times T \right) \right\} \times w_i \qquad (9)$$

where $D_{i,alter}$ represents the variable forces or project cost factors at the alternative level.

6 Experimental setup

The estimated effort expressed in equations (8) and (14) for waterfall and agile delivery models respectively can now be optimised using evolutionary algorithms. In this work, ABC, PSO and GLBPSO have been used to optimise the effort estimates as described below:

6.1 Representation of the population

In this research, each individual particle/ bee in the population is represented as a vector of weight values as shown below:

Waterfall model

w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8	w_9	w_{10}	w_{11}	w_{12}	w_{13}	w_{14}	w_{15}	w_{16}	w_{17}	w_{18}	w_{19}	w_{20}	w_{21}

Agile model

w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8	w_9	w_{10}	w_{11}	w_{12}	w_{13}

The Waterfall model and the agile model datasets will have 21 and 13 weights respectively for associated project cost factors. In equations (8) and (14), the weights are optimized to calculate the integrated effort estimates for software project development. All the weights are in the range 0 to 1. The algorithms will return the vector with weight values at which the MMRE value is minimized. From equation (15), it follows that the optimization of weight values will bring the IE`s close to the actual effort.

6.2 Fitness function

In evolutionary algorithms, fitness function is used to evaluate the fitness of the individuals in the population. Mean magnitude of relative error (MMRE) is the most widely used fitness function for software effort estimation problem [33]. Tomas Urbanek [33] et al. have found MMRE to be an average fitness function for the effort estimation problem. Considering earlier performance of MMRE, we have used MMRE as the fitness function for ABC, PSO and GLBPSO algorithms.

6.2.1 Mean magnitude of relative error

For the algorithms used in this research, mean magnitude of relative error (MMRE) is used as the objective function to evaluate the fitness of the individuals in the population [34]. Magnitude of relative error (MRE) is the ratio of the absolute difference between the integrated effort (IE) and the actual effort spent on the project, and the actual effort spent on the project. Thus, the formula for MMRE will be:

$$MMRE = \frac{\sum_{i=1}^{N} MRE}{N} = \sum_{i=1}^{N} \left(\frac{IE_i - actual\ effort_i}{actual\ effort_i}\right) / N \qquad (10)$$

where N is the total number of projects in the dataset. All the three algorithms are used to obtain weight values (w_i) that minimize the MMRE.

6.3 Parameter values

The proposed approach is implemented on two datasets collected during the research described in section 4. The implementation for the considered evolutionary algorithms was executed on MATLAB. The fitness function values in ABC, PSO and GLBPSO showed little change after 20 iterations. So, the number of iterations is set to 20 in all the swarm optimization algorithms.

6.3.1 Parameter values for ABC

In ABC algorithm the population size is generally kept as the square of the number of employed bees [24]. The number of weights in the waterfall model dataset is small (21), an initial population of size 441 i.e., square of 21 was generated randomly. Similarly, the agile model dataset had 13 weights, so the randomly generated population size was kept at 169. The best fitness value after every iteration showed little change after around 17 iterations as shown in figures 2 and 6. Thus, the number of iterations in each run were set to 20. The number of trials for abandoning food source (Limit) was set to 50.

6.3.2 Parameter values for PSO and GLBPSO

The population size for PSO and GLBPSO were kept the same as for the ABC algorithm. So, the population size was set at 441 for Waterfall model dataset and 169 for agile model dataset. Other parameters values are given below:

1) Number of iterations: 20
2) $c_1 = c_2 = 1.5$
3) w = 0.8

For PSO and GLBPSO, the best fitness value after every iteration showed little change in waterfall model dataset after around 17 iterations and showed little change in agile model dataset after around 7 iterations as shown in figures 3, 4, 7 and 8 respectively.

6.4 Performance evaluation metrics

The integrated effort estimated with the proposed model is compared with the initial estimated effort using the benchmark model based on four performance evaluation metrics: mean magnitude of relative error (MMRE), standardized accuracy (SA), effect size (Δ) and coefficient of determination (R^2).

6.4.1 Standardized accuracy

The performance evaluation measures MRE and MMRE have been criticized for being biased towards effort estimation techniques resulting in underestimates [34-39]. Therefore, we compare integrated effort estimates from the proposed approach with the estimated effort of CoCoMo II and Ziauddin models using standardized accuracy (SA) also. In SA, an estimated effort value is randomly chosen and assigned as effort estimate of the remaining projects. This process is repeated 1000 times and then mean absolute error (MAR) is calculated every time. Standardized accuracy is calculated based on the formula given below:

$$SA = 1 - \frac{MAR}{MAR_{P0}} \times 100 \tag{11}$$

where MAR is the mean absolute error i.e., the mean of the absolute difference between the estimated and actual effort estimates of all the projects.

MAR_{P0} is the mean MAR of the 1000 random assignments [38]. For performance evaluation, a lower MMRE value or a higher SA value implies a better effort estimation approach.

6.4.2 Effect size

Effect size (Δ) is used to determine the reliability of the proposed approach [38, 40]. It can be calculated based on the formula given below:

$$\Delta = \frac{MAR - MAR_{P0}}{\sigma_{P0}} \tag{12}$$

where σ_{P0} refers to the standard deviation of MAR values of 1000 random assignments from MAR_{P0}. High value of Δ (>0.5) indicates that the results obtained by the proposed algorithm are more reliable than those obtained by random guessing.

6.4.3 Coefficient of determination

Coefficient of determination (R^2) is used to determine the correlation between the dependent and the independent variables [41]. It varies from 0 to 1. A value closer to 1 indicates a strong correlation between the variables. For this research, independent variables are the project cost factor values and the size of the project. Estimated effort will be the dependent variable.

7 Results & analysis

Each algorithm was run 25 times. We have reported the best results obtained for each algorithm.

7.1 Results for waterfall model dataset

Table 2 lists the weight values obtained for all the three algorithms: ABC, PSO and GLBPSO for the Waterfall dataset. These weights represent the optimal values of $p_{i,alter}$ defined in equation (5). A higher value of $p_{i,alter}$ (>0.5) indicates high level of risk exposure due to the corresponding project cost factor. Weight values 0, indicate that there was negligible risk due to that project cost factor and it did not impact the effort estimate of the project. All the projects considered are from the same organization, where these project cost factors such as main storage constraint, platform volatility, platform experience and execution time constraint, might already be controlled efficiently thus having no impact on the integrated effort estimate. Weight values of 1 indicate that the associated project cost factors highly impact the integrated effort of the project. These factors need to be monitored and controlled for successful project deliveries. Results indicate that the evolutionary algorithms give better results (lower MMRE) as compared to the MMRE values of initial integrated effort estimate IE_{intial}. From table 3 it can be observed that among the evolutionary algorithms, PSO outperformed the other two algorithms (ABC and GLBPSO) with the lowest MMRE value of 0.131 in the shortest time. MMRE for GLBPSO was also 0.131, but it took more execution time than PSO.

Table 2: Parameter values for ABC, PSO and GLBPSO algorithms: Waterfall model

Weights	ABC	PSO	GLBPSO
w_1	0	0	0
w_2	0.168	0.552	0.579
w_3	0.596	1	0.686
w_4	0.826	1	1
w_5	0.954	1	1
w_6	0.685	1	0.729
w_7	0.755	1	1
w_8	0.33	0.631	0.525
w_9	0.884	1	1
w_{10}	0.695	0.818	0.935
w_{11}	0.334	0	0
w_{12}	0.624	0	0
w_{13}	0.796	0.79	0
w_{14}	0.891	0	1
w_{15}	0.302	1	0
w_{16}	0.551	0	0.742
w_{17}	0.713	0	0
w_{18}	0.437	0.733	0
w_{19}	0.301	1	0
w_{20}	0.549	0.736	0.678
w_{21}	0.917	0.159	0

Table 3: Waterfall Model experimental results

Waterfall Model	MMRE	SA	(Δ)	R^2	Time (seconds)
CoCoMo II	0.215	0.829	0.521	0.581	456.15
IE(Initial)	0.183	0.845	0.596	0.729	453.23
ABC	0.147	0.843	0.507	0.773	5.56×10^3
PSO	0.131	0.843	0.575	0.767	3.52×10^3
GLBPSO	0.131	0.843	0.545	0.791	5.4×10^3

In table 4, the integrated effort estimates obtained by the five approaches: CoCoMo II, $IE_{initial}$, ABC, PSO and GLBPSO are depicted project wise. These effort estimates are calculated by substituting the weight values listed in table 2 in equation (8). It can be noted that for most of the projects, integrated effort estimates for evolutionary algorithms are lower than their corresponding $IE_{initial}$ value given in column 4 of table 7. This indicates that the experts might have overestimated the alternative cost of the project cost factors that increased the project risk exposure. For projects (P1, P3, P5, P8, P16, P19, P33, P36, P40 and P41), the integrated effort for the evolutionary algorithms is higher than their corresponding $IE_{initial}$ value. The MRE for these projects were in the range: CoCoMo II (0%, 39%), $IE_{initial}$ (3%, 38%), ABC (1%, 37%), PSO (0%, 41%) and GLBPSO (0%, 36%). This indicates under estimation of risk exposure due to the project cost factors by the experts. Figure 1 shows the variation in MRE of projects for all the considered five estimation approaches. MRE for projects (P4, P7, P10, P11, P15, P17, P19, P20, P23, P24, P25, P26, P27, P30, P31, P33, P37, P39, P40, P41, P44 and P45) has reduced considerably with the use of evolutionary algorithms as compared to the MRE values obtained when using CoCoMo II or initial integrated effort estimates $IE_{initial}$.

Table 4: Estimated effort for Waterfall Model

Project Id	Actual Effort	CoCoMo II	$IE_{initial}$	ABC	PSO	GLBPSO
P1	1634	1440.801	1728.743	1904.43	2036.967	2024.036
P2	700	666.9536	713.9921	594.992	589.1942	564.2396
P3	3987	3852.068	4123.872	4309.477	4125.641	4343.761
P4	450	404.2893	498.517	441.5914	449.4359	442.9401
P5	2608	2702.005	2535.76	2652.95	2430.574	2594.073
P6	3567	3304.395	3661.615	3085.545	2944.122	2957.799
P7	2256	2048.078	2473.87	2150.134	2320.11	2295.657
P8	912	965.8243	952.08	969.2639	918.2394	978.07
P9	2879	2518.215	2959.183	2329.926	2673.046	2492.919
P10	2435	1922.343	2318.947	2048.018	2440.157	2285.411
P11	1456	1297.093	2202.691	1957.832	1884.466	1590.199
P12	2234	1480.343	2414.968	2262.714	2347.047	2421.157
P13	3200	2020.047	3080.992	2835.488	3174.499	2578.99
P14	3567	2339.519	4248.97	3801.656	3581.139	3753.184
P15	3678	2044.514	3905.35	3523.734	3309.443	3461.998
P16	2759	2756.707	2585.53	2705.774	2478.366	2645.365
P17	3015	4621.11	4901.983	4300.253	4093.611	4115.907
P18	3459	2897.416	3497.342	3053.948	3288.689	3253.642
P19	2435	1490.16	1981.087	2114.641	2201.621	2278.578
P20	859	1412.506	1496.216	1236.099	1208.915	1163.307
P21	3147	1821.516	2934.557	2384.43	1944.12	2307.894
P22	1987	1304.258	2239.509	1765.508	1465.091	1709.601
P23	4567	4632.826	5968.334	4640.557	4502.962	4047.402
P24	3629	3275.441	4936.89	4158.498	3676.23	4085.744
P25	2897	2937.634	4906.66	3734.724	3305.251	3671.437
P26	2453	1247.939	1496.43	1498.683	1644.663	1492.266
P27	3786	6163.266	5581.173	4626.056	4513.531	3955.972
P28	2687	2577.099	2510.667	2574.539	2437.885	2385.794

Project Id	Actual Effort	CoCoMo II	$IE_{initial}$	ABC	PSO	GLBPSO
P29	2937	2395.47	2887.848	2622.176	2534.778	2455.068
P30	2874	3730.238	3463.449	3229.824	2879.849	3033.284
P31	3384	3851.66	3346.509	3377.767	3371.97	3199.105
P32	3287	3480.278	3906.636	3347.397	3729.617	3474.419
P33	2845	2376.817	2534.226	2807.818	2936.245	2840.376
P34	3504	3542.165	3598.086	3204.293	3514.348	3175.856
P35	2134	2005.142	2034.214	1882.425	1901.643	1827.661
P36	2739	1746.16	1710.686	1734.193	1628.757	1740.316
P37	2469	2391.855	3657.088	3391.128	3802.217	3087.405

Project Id	Actual Effort	CoCoMo II	$IE_{initial}$	ABC	PSO	GLBPSO
P38	3200	3963.386	3911.636	4026.358	3805.979	3724.716
P39	2489	2562.534	2590.451	2391.678	2407.534	2317.195
P40	1999	1411.736	1694.448	1865.935	1995.9	1983.034
P41	2598	1704.9	2264.622	2421.514	2521.187	2610.084
P42	2876	1369.664	2605.072	2319.319	2176.62	2281.717
P43	2309	2441.109	2374.307	2433.534	2304.895	2255.642
P44	1784	905.3366	1072.126	937.9648	1126.777	1054.362
P45	2792	3159.18	3502.639	2952.263	2818.442	2831.023

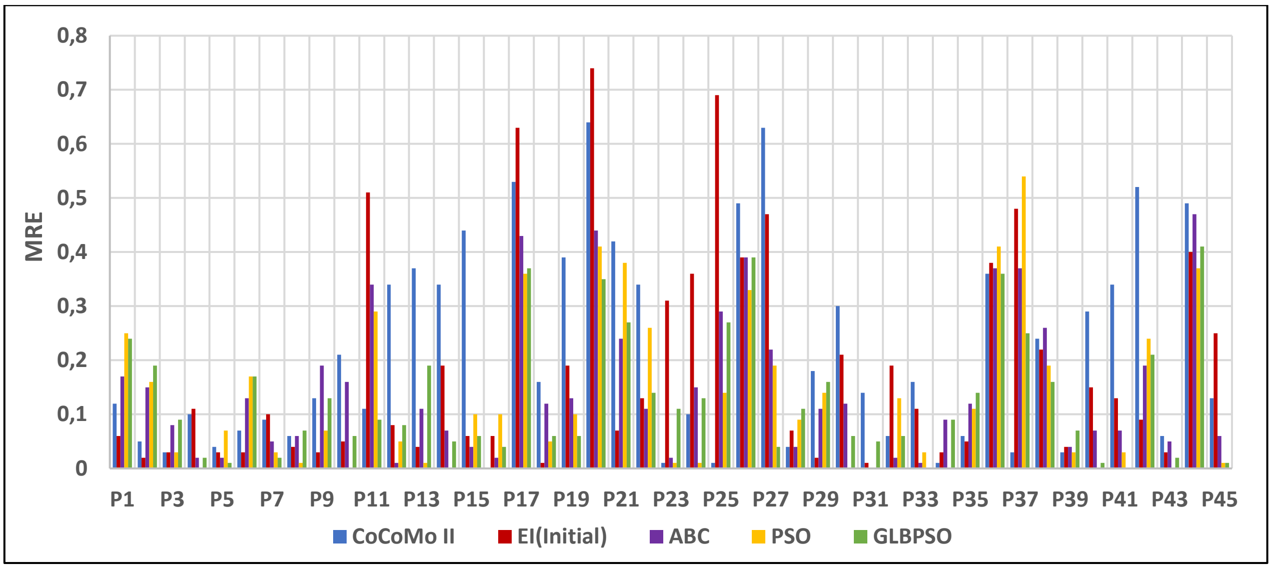

Figure 1: Variation of MRE in waterfall dataset

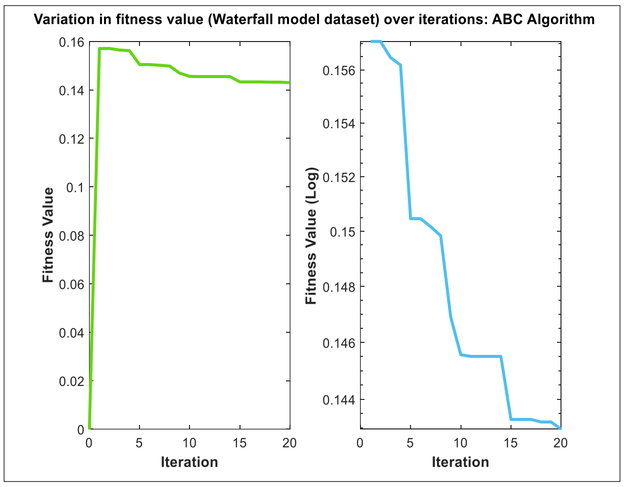

Figure 2: Variation in fitness value: ABC algorithm

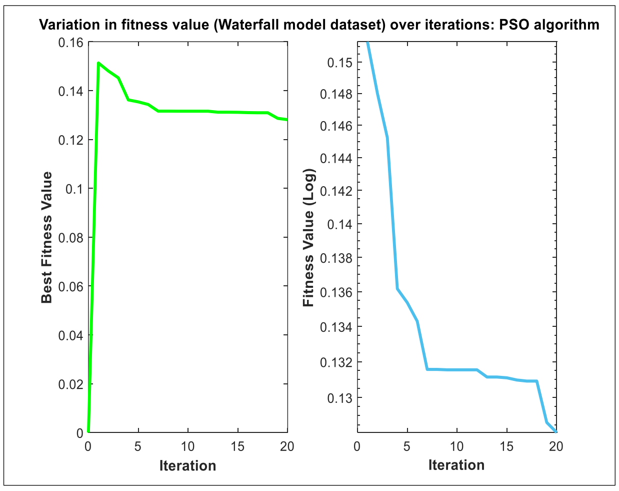

Figure 3: Variation in fitness value: PSO algorithm

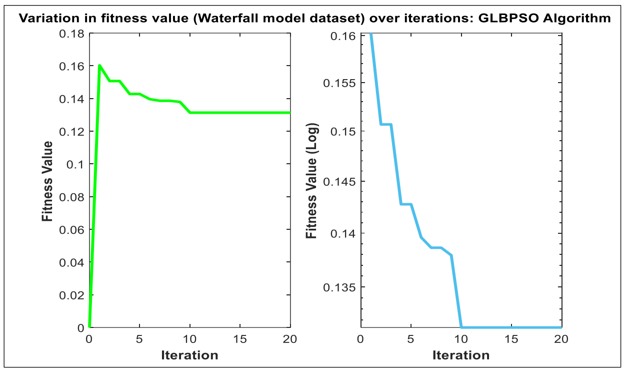

Figure 4: Variation in fitness value: GLBPSO algorithm

7.2 Results for agile model dataset

For the Agile dataset, the optimal weight values obtained for all the three algorithms: ABC, PSO and GLBPSO are listed in table 5. Results shown in table 6, indicate that the evolutionary algorithms gave better results (lower MMRE) as compared to the MMRE obtained for the initial integrated effort estimate (equation 3). It can also be observed that among the evolutionary algorithms, PSO outperformed the other two algorithms (ABC and GLBPSO) with the lowest MMRE value of 0.151 in the shortest time. MMRE for GLBPSO was also 0.151, but it took more execution time than the PSO.

Table 5: Parameter values for ABC, PSO and GLBPSO Algorithms for Agile model

Weights	ABC	PSO	GLBPSO
w1	0.9183	1	1
w2	0.0223	0	0
w3	0.603	0.783	0.781
w4	0	0	0
w5	0.0733	0	0
w6	1	1	1
w7	0.4551	0.3888	0.3936
w8	0.8715	1	1
w9	0.7779	1	1
w10	0.0417	0	0
w11	0.2197	0	0
w12	0.0697	0	0
w13	0.0396	0	0

The integrated effort estimates for the various approaches: Ziauddin, IE(Initial), ABC, PSO and GLBPSO are depicted in table 7. These effort estimates are calculated by substituting the weight values listed in table 5 in equation (11). For the Agile dataset, the MMRE (0.282) for IE(Initial) is close to the MMRE (0.288) of Ziauddin approach. The initial integrated effort estimates IE(Initial) for most of the projects are lower than the effort estimates using Ziauddin approach. The integrated effort estimates IE for all the evolutionary algorithms are higher than their IE(Initial) estimates. This indicates that the experts had assumed the cost of project cost factors optimistically. Use of evolutionary algorithms has considerably reduced this over optimism, as indicated by the MMRE values (ABC: 0.155, PSO, GLBPSO: 0.151). Figure 5 shows the variation of MRE obtained for projects for all the considered estimation approaches. MRE for projects (P3, P4, P5, P6, P9, P12, P15, P16, P17, P20, P21, P24, P26 and P30) has reduced considerably with the use of evolutionary algorithms as compared to the MRE values obtained when using Ziauddin approach or the initial integrated effort estimates.

Table 6: Agile model experimental results

Agile Model	MMRE	SA	Effect Size (Δ)	R^2	Time (seconds)
Ziauddin	0.288	1.85	0.603	0.018	356.36
IE(Initial)	0.282	2.14	0.713	0.102	347.56
ABC	0.155	2.34	0.756	0.113	9.32×10^3
PSO	**0.151**	**2.43**	**0.766**	**0.114**	$\mathbf{6.17 \times 10^3}$
GLBPSO	**0.151**	**2.44**	**0.765**	**0.113**	$\mathbf{1.19 \times 10^4}$

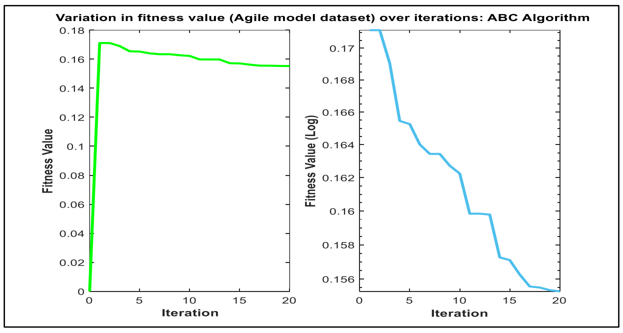

Figure 6: Variation in fitness value: ABC algorithm

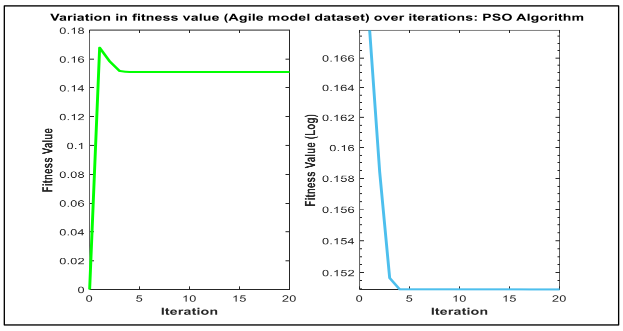

Figure 7: Variation in fitness value: PSO algorithm

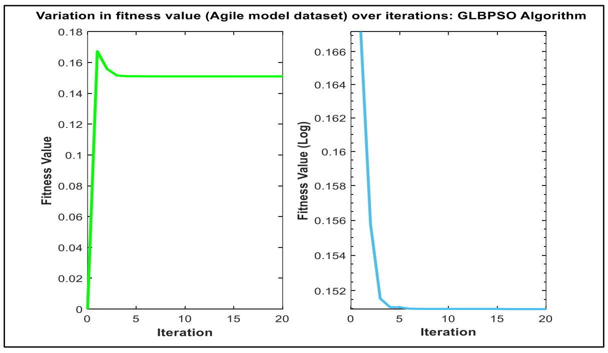

Figure 8: Variation in fitness value: GLBPSO algorithm

Table 7: Estimated effort for agile model

Project Id	Actual Effort	Ziauddin	IE (Initial)	ABC	PSO	GLBPSO
1	64	86.49231	83.60254	81.18652	82.4856	82.48179
2	76	117.1165	109.1453	109.792	110.9355	110.9644
3	81	145.9146	132.0753	125.3142	125.1441	125.1514
4	68	119.1528	107.4205	92.31646	93.95784	93.87866
5	66	57.66671	56.37771	62.58207	62.75219	62.77239
6	67	63.59279	60.88424	64.93346	65.75203	65.7274
7	67	43.43548	43.27209	45.72152	46.55416	46.55146
8	68	67.05171	52.21071	57.06398	58.15929	58.10352
9	56	56.27627	44.25048	50.32494	50.82362	50.8004
10	60	47.195	44.33398	46.6364	47.29976	47.30984
11	58	72.43631	59.08681	67.43594	68.02355	67.97869
12	62	48.89734	45.28741	53.88648	54.93635	54.95424
13	65	48.81696	47.96385	53.45084	53.79691	53.81199
14	63	60.65094	52.03628	50.69616	51.91528	51.87922
15	65	56.06237	55.76618	63.89089	65.00141	65.01872
16	66	77.1008	45.07614	61.89129	64.40236	64.34528
17	65	70.89737	58.16727	67.64816	68.42956	68.44052
18	62	57.01432	55.20099	56.30309	56.1034	56.12013
19	76	57.59758	57.12901	61.91986	62.78898	62.80839
20	75	75.05299	65.01085	69.15184	70.20775	70.20775
21	73	70.55185	64.72608	68.51258	69.79861	69.78006
22	72	61.62484	53.08437	58.69321	58.78924	58.80578
23	73	70.14982	67.56121	81.80111	83.00832	83.00832
24	74	89.07373	76.97409	74.24416	75.22673	75.20611
25	60	49.52664	42.81767	49.61795	50.55862	50.54135
26	65	77.78041	59.7399	65.04337	65.00297	64.99734
27	62	58.08273	42.89178	49.42383	49.86158	49.86158
28	55	47.45795	46.40748	47.02344	47.1733	47.17914
29	60	66.81642	61.06547	63.36946	63.95546	63.93309
30	59	64.98048	44.34687	55.10337	56.39194	56.37387

7.3 Revisiting the research questions

RQ1: *Does the accuracy of effort estimate of the project improve by adding the cost of risk exposure to the initial estimated effort of the project?*

Results as discussed in sections 7.1 and 7.2, show that the integrated effort estimates have lower values of MMRE and higher values of SA, effect size and R^2 than the corresponding initial effort estimates, for both the datasets. Thus, it can be concluded that the integrated effort estimates are more accurate, reliable, and comprehensive than the initial effort estimates.

RQ2: *What is the impact of bias on risk exposure of the project?*

Results in sections 7.1 and 7.2, show that the MMRE of the software effort estimate is reduced by using evolutionary algorithms ABC, PSO and GLBPSO for both the datasets. In the process, we also obtain the optimum weight value, w_i, corresponding to the i^{th} project cost factor, which is the optimum value of $p_{i,alter}$. These optimal values of the $w_i's$ reflect the unbiased values of $(p_{i,alter})'s$.

RQ3: *Can project cost factors be ranked with respect to their risk exposure?*

High value of w_i implies that the contribution of the i^{th}project cost factor is also high in the risk exposure of the project. So, the values of w_i's are good indicators of project cost factors which have high risk exposure.

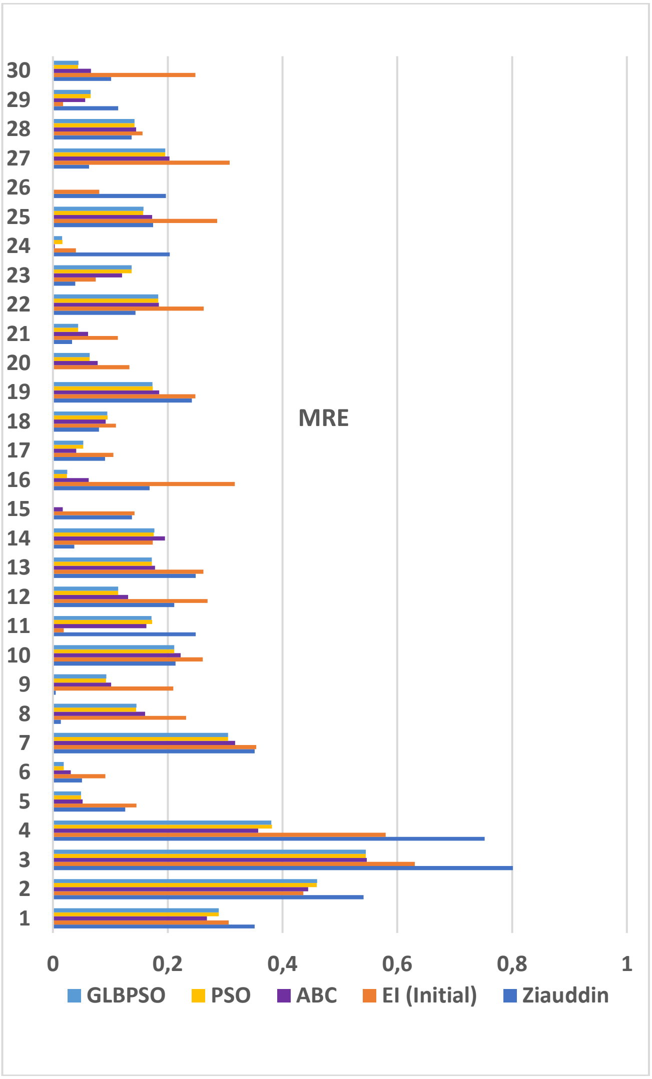

Figure 5: Variation of MRE in agile dataset

8 Threats to validity

This section discusses threats to validity of the proposed model.

External validity

External validity [42] is concerned with generalization of the results obtained. Threats to external validity are conditions that limit the ability to generalize the results of the proposed experiment to other effort estimation models [43-46].

For this research, the datasets have been collected from a single organization. Although the organization is a large IT consultancy firm working on projects in varied domains, the proposed approach could be further validated by experimenting on data from different organizations. To test the efficacy of evolutionary algorithms, large datasets are ideal. The results in this paper are based on datasets with 30 to 45 projects.

9 Conclusion

This paper has introduced a novel approach for integrating the impact of risk exposure into the effort estimate of a software project. This impact is determined using the weights associated with the risk exposure due to each project cost factor. These weights are then optimized using evolutionary approaches like ABC, PSO and GLBPSO. Experimental results show that the PSO and GLBPSO algorithms gave more accurate effort estimates for both waterfall and agile projects, but GLBPSO took more time. The approach essentially reduces the bias due to the probabilities which were associated with the impact of risk exposure on the effort estimates of the projects. Software effort estimation for projects now, will, not have to rely solely on the expert judgment for assessing the probable impact of the risk exposure due to project cost factors.

The project factors can be ranked based upon the associated optimal weight values. Software Project managers can prepare and plan for risk management and development of the project effectively using the ranking obtained. Cost factors with higher weight values will need to be mitigated and controlled earlier than the cost factors with lower weight values.

10 Future directions

In the manuscript, tables 4 & 7 list the effort estimates calculated by using the optimum weight values obtained by applying ABC, PSO and GLBPSO on waterfall and agile model datasets respectively. To validate the obtained results, tables 5 and 6, then compare the calculated effort estimates based on MMRE, SA, Effect size and R^2. The results obtained confirm that the risk integrated effort estimation accuracy improves with the application of evolutionary algorithms such as ABC, PSO and GLBPSO. The proposed risk integrated approach can further be validated through additional case study / company data.

The proposed risk integrated effort estimation approach can be applied to other benchmark effort estimation models such as Use Case Point [47], Function Point [48], and Analogy based estimation [49] for Waterfall projects. Poker [50], T-shirt sizing and Three point estimation [51] for Agile projects. To enable the comparison, cost factor data for the suggested benchmark models will have to be collected / generated.

To further investigate the impact of evolutionary algorithms on weight values associated with cost factors, other available evolutionary algorithms such as firefly, ant colony optimization, cuckoo search and whale optimization could be used, and results compared with the results obtained in this research. The weight values can also be optimized using artificial intelligence techniques like neural networks, convolutional neural networks, and deep learning techniques.

11 Declarations

Funding
Not Applicable

Availability of data and material
The datasets generated during and/or analyzed during the current study are available from the corresponding author on reasonable request.

Code availability
Not Applicable

References

[1] S. Hastie and S. Wojewoda, "Standish group 2015 chaos report-q&a with jennifer lynch," *Retrieved,* vol. 1, p. 2016, 2015.

[2] D. Eck, B. Brundick, T. Fettig, J. Dechoretz, and J. Ugljesa, "Parametric estimating handbook," *The International Society of Parametric Analysis (ISPA),* 2009.

[3] P. Pospieszny, B. Czarnacka-Chrobot, and A. Kobylinski, "An effective approach for software project effort and duration estimation with machine learning algorithms," *Journal of Systems and Software,* vol. 137, pp. 184-196, 2018. https://doi.org/10.1016/j.jss.2017.11.066

[4] O. Morgenshtern, T. Raz, and D. Dvir, "Factors affecting duration and effort estimation errors in software development projects," *Information and Software Technology,* vol. 49, pp. 827-837, 2007. https://doi.org/10.1016/j.infsof.2006.09.006

[5] K. Kansala, "Integrating risk assessment with cost estimation," *IEEE software,* vol. 14, pp. 61-67, 1997. https://doi.org/10.1109/52.589236

[6] B. W. Boehm, "Software risk management: principles and practices," *IEEE software,* vol. 8, pp. 32-41, 1991.
https://doi.org/10.1109/52.62930

[7] M. Jørgensen, "Identification of more risks can lead to increased over-optimism of and over-confidence in software development effort estimates," *Information and Software Technology*, vol. 52, pp. 506-516, 2010. https://doi.org/10.1016/j.infsof.2009.12.002

[8] P. Singal, P. Sharma, and A. C. Kumari, "Integrating software effort estimation with risk management,"

International Journal of System Assurance Engineering and Management, pp. 1-16, 2022. https://doi.org/10.1007/s13198-022-01652-y

[9] B. Kitchenham and S. Linkman, "Estimates, uncertainty, and risk," *IEEE Software*, vol. 14, pp. 69-74, 1997. https://doi.org/10.1109/52.589239

[10] B. Boehm, "Software risk management," in *European Software Engineering Conference*, 1989, pp. 1-19. https://doi.org/10.1007/3-540-51635-2_29

[11] R. Fairley, "Risk management for software projects," *IEEE software*, vol. 11, pp. 57-67, 1994. https://doi.org/10.1109/52.281716

[12] R. J. Madachy, "Heuristic risk assessment using cost factors," *IEEE software*, vol. 14, pp. 51-59, 1997. https://doi.org/10.1109/52.589234

[13] L. C. Briand, K. El Emam, and F. Bomarius, "COBRA: a hybrid method for software cost estimation, benchmarking, and risk assessment," in *Proceedings of the 20th international conference on Software engineering*, 1998, pp. 390-399. https://doi.org/10.1109/ICSE.1998.671392

[14] K. Jantzen, "Estimating the effects of project risks in software development projects," 2006.

[15] S.-J. Huang, C.-Y. Lin, and N.-H. Chiu, "Fuzzy decision tree approach for embedding risk assessment information into software cost estimation model," *Journal of information science and engineering*, vol. 22, pp. 297-313, 2006.

[16] E. Manalif, "Fuzzy Expert-COCOMO risk assessment and effort contingency model in software project management," 2013.

[17] W. Aslam, F. Ijaz, M. I. U. Lali, and W. Mehmood, "Risk Aware and Quality Enriched Effort Estimation for Mobile Applications in Distributed Agile Software Development," *J. Inf. Sci. Eng.*, vol. 33, pp. 1481-1500, 2017. https://doi.org/10.6688/JISE.2017.33.6.6

[18] S. El Koutbi and A. Idri, "Software Effort Estimation Risk Management over Projects Portfolio," *Comput. Inf. Sci.*, vol. 11, pp. 45-76, 2018. https://doi.org/10.5539/cis.v11n4p45

[19] N. Ramakrishnan, H. Girijamma, and K. Balachandran, "Enhanced Process Model and Analysis of Risk Integration in Software effort estimation," in *2019 International Conference on Smart Systems and Inventive Technology (ICSSIT)*, 2019, pp. 419-422. https://doi.org/10.1109/ICSSIT46314.2019.8987841

[20] M. Kataev, L. Bulysheva, L. Xu, Y. Ekhlakov, N. Permyakova, and V. Jovanovic, "Fuzzy model estimation of the risk factors impact on the target of promotion of the software product," *Enterprise Information Systems*, vol. 14, pp. 797-811, 2020. https://doi.org/10.1080/17517575.2020.1713407

[21] M. Azzeh, A. B. Nassif, and S. Banitaan, "Comparative analysis of soft computing techniques for predicting software effort based use case points," *IET Software*, vol. 12, pp. 19-29, 2018. https://doi.org/10.1049/iet-sen.2016.0322

[22] T. T. Khuat and M. H. Le, "A novel hybrid abc-pso algorithm for effort estimation of software projects using agile methodologies," *Journal of Intelligent Systems*, vol. 27, pp. 489-506, 2018. https://doi.org/10.1515/jisys-2016-0294

[23] D. Novitasari, I. Cholissodin, and W. F. Mahmudy, "Hybridizing PSO with SA for optimizing SVR applied to software effort estimation," *Telkomnika (Telecommunication Computing Electronics and Control)*, vol. 14, pp. 245-253, 2016. http://doi.org/10.12928/telkomnika.v14i1.2812

[24] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm," *Journal of global optimization*, vol. 39, pp. 459-471, 2007. http://doi.org/0.1007/s10898-007-9149-x

[25] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95-International Conference on Neural Networks*, 1995, pp. 1942-1948. https://doi.org/10.1109/ICNN.1995.488968

[26] R. R. Chhikara, P. Sharma, and L. Singh, "A hybrid feature selection approach based on improved PSO and filter approaches for image steganalysis," *International Journal of Machine Learning and Cybernetics*, vol. 7, pp. 1195-1206, 2016. https://doi.org/10.1007/s13042-015-0448-0

[27] T. Gilb, "Evolutionary Delivery versus the" waterfall model"," *ACM sigsoft software engineering notes*, vol. 10, pp. 49-61, 1985. https://doi.org/10.1145/1012483.1012490

[28] R. C. Martin, M. Martin, and M. Martin, *Agile principles, patterns, and practices in C#*: Prentice Hall, 2007.

[29] B. Boehm, C. Abts, B. Clark, and S. Devnani-Chulani, "COCOMO II model definition manual," *The University of Southern California*, 1997.

[30] S. K. T. Ziauddin and S. Zia, "An effort estimation model for agile software development," *Advances in computer science and its applications (ACSA)*, vol. 2, pp. 314-324, 2012.

[31] S. Dalal, N. Dahiya, and V. Jaglan, "Efficient Tuning of COCOMO Model Cost Drivers Through Generalized Reduced Gradient (GRG) Nonlinear Optimization with Best-Fit Analysis," in *Progress in Advanced Computing and Intelligent Engineering*, ed: Springer, 2018, pp. 347-354. https://doi.org/10.1007/978-981-10-6872-0_32

[32] A. W. M. M. Parvez, "Efficiency factor and risk factor based user case point test effort estimation model compatible with agile software development," in *Information Technology and Electrical Engineering (ICITEE), 2013 International Conference on*, 2013, pp. 113-118. https://doi.org/10.1109/ICITEED.2013.6676222

[33] T. Urbanek, Z. Prokopova, R. Silhavy, and V. Vesela, "Prediction accuracy measurements as a fitness function for software effort estimation," *SpringerPlus*, vol. 4, pp. 1-17, 2015. https://doi.org/10.1186/s40064-015-1555-9

[34] T. Foss, E. Stensrud, B. Kitchenham, and I. Myrtveit, "A simulation study of the model evaluation criterion MMRE," *IEEE transactions on software engineering,* vol. 29, pp. 985-995, 2003. https://doi.org/10.1109/TSE.2003.1245300

[35] B. A. Kitchenham, L. M. Pickard, S. G. MacDonell, and M. J. Shepperd, "What accuracy statistics really measure," *IEE Proceedings-Software,* vol. 148, pp. 81-85, 2001. https://doi.org/10.1049/ip-sen:20010506

[36] M. Korte and D. Port, "Confidence in software cost estimation results based on MMRE and PRED," in *Proceedings of the 4th international workshop on Predictor models in software engineering*, 2008, pp. 63-70. https://doi.org/10.1145/1370788.1370804

[37] D. Port and M. Korte, "Comparative studies of the model evaluation criterions mmre and pred in software cost estimation research," in *Proceedings of the Second ACM-IEEE international symposium on Empirical software engineering and measurement*, 2008, pp. 51-60. https://doi.org/10.1145/1414004.1414015

[38] A. Idri, I. Abnane, and A. Abran, "Evaluating Pred (p) and standardized accuracy criteria in software development effort estimation," *Journal of Software: Evolution and Process,* vol. 30, p. e1925, 2018. https://doi.org/10.1002/smr.1925

[39] E. Stensrud, T. Foss, B. Kitchenham, and I. Myrtveit, "A further empirical investigation of the relationship between MRE and project size," *Empirical software engineering,* vol. 8, pp. 139-161, 2003. https://doi.org/10.1023/A:1023010612345

[40] A. B. Nassif, M. Azzeh, A. Idri, and A. Abran, "Software development effort estimation using regression fuzzy models," *Computational Intelligence and neuroscience,* vol. 2019, 2019. https://doi.org/10.1155/2019/8367214

[41] N. J. Nagelkerke, "A note on a general definition of the coefficient of determination," *Biometrika,* vol. 78, pp. 691-692, 1991.

[42] C. Wohlin, P. Runeson, M. Höst, M. C. Ohlsson, B. Regnell, and A. Wesslén, *Experimentation in software engineering*: Springer Science & Business Media, 2012.

[43] A. Idri, F. azzahra Amazal, and A. Abran, "Analogy-based software development effort estimation: A systematic mapping and review," *Information and Software Technology,* vol. 58, pp. 206-230, 2015. https://doi.org/10.1016/j.infsof.2014.07.013

[44] J. Popovic, D. Bojic, and N. Korolija, "Analysis of task effort estimation accuracy based on use case point size," *IET Software,* vol. 9, pp. 166-173, 2015. https://doi.org/10.1049/iet-sen.2014.0254

[45] I. Hussain, L. Kosseim, and O. Ormandjieva, "Approximation of COSMIC functional size to support early effort estimation in Agile," *Data & Knowledge Engineering,* vol. 85, pp. 2-14, 2013. https://doi.org/10.1016/j.datak.2012.06.005

[46] M. Usman, "Improving Expert Estimation of Software Development Effort in Agile Contexts," Blekinge Tekniska Högskola, 2018.

[47] M. R. Braz and S. R. Vergilio, "Software effort estimation based on use cases," in *30th Annual International Computer Software and Applications Conference (COMPSAC'06)*, 2006, pp. 221-228. https://doi.org/10.1109/COMPSAC.2006.77

[48] A. Hira and B. Boehm, "COSMIC Function Points Evaluation for Software Maintenance," in *Proceedings of the 11th Innovations in Software Engineering Conference*, 2018, p. 4. https://doi.org/10.1145/3172871.3172874

[49] M. Azzeh and A. B. Nassif, "Analogy-based effort estimation: a new method to discover set of analogies from dataset characteristics," *IET Software*, vol. 9, pp. 39-50, 2015. https://doi.org/10.1049/iet-sen.2013.0165

[50] V. Mahnič and T. Hovelja, "On using planning poker for estimating user stories," *Journal of Systems and Software,* vol. 85, pp. 2086-2095, 2012. https://doi.org/10.1016/j.jss.2012.04.005

[51] R. K. Mallidi and M. Sharma, "Study on agile story point estimation techniques and challenges," *Int. J. Comput. Appl,* vol. 174, pp. 9-14, 2021.