To What Extent Mathematics Correlates With Programming: Statistical Analysis

Ayman Qahmash
The Department of Computer Science
University of Warwick
Coventry, UK
A.qahmash@warwick.ac.uk

Mike Joy
The Department of Computer Science
University of Warwick
Coventry, UK
M.S.Joy@warwick.ac.uk

Adam Boddison
Center for Profeesional Education
University of Warwick
Coventry, UK
A.boddison@warwick.ac.uk

***Abstract*— Generally, mathematical abilities have a certain degree of importance in computer science education and a correlation between mathematics and programming, derived from small samples, had been suggested. However, our study investigates whether the correlation is statistically significant or not by analysing a large data set spanning nineteen years for two programming modules and two mathematics modules with syllabi which had undergone minimal changes. In this paper, a statistical analysis based on students' performance has been used to determine the relationship between mathematics and programming in general and between two specific mathematics modules and two programming modules. Compared with small sample analyses derived from previous research, the large sample of 2,161 students' grades in the present study indicates a positive moderate correlation that is statistically significant. The results also suggest that discrete mathematics correlates with introductory programming more than calculus whereas data structures could relate positively to student performance in calculus more than discrete mathematics.**

Keywords-component; Programming; Mathematics; Statistics

I. INTRODUCTION

Recently, the number of computer programming vacancies has been increasing around the globe. In the U.S., there will be an increasing demand for computing skills. Similarly, the U.K. has placed more emphasis on computing at schools by replacing the Information and Communication Technology module with a computing curriculum in which students will be able to develop and acquire computational thinking abilities [13].

The role of mathematics in learning programming is still a debated issue among computer science educators. Students' mathematical ability might be considered to be an indicator of programming aptitude, as mathematics provides a variety of skills, such as reasoning and problem solving, that are required during computational thinking. Mathematics has been used as a significant factor when predicting student success in programming [6, 13] as well as designing programming aptitude tests based on mathematical ability. However, it has been claimed that most mathematics concepts have not been used effectively in most programming modules, and that the important relations between mathematical concepts and computer science fundamentals have not been made apparent to students [5]. For instance, it is essential that students must acquire basic algebra and logic to help them to learn programming fundamentals, whereas it might be confusing for students to understand the role of calculus in an introductory programming course.

However, introducing discrete mathematics basics may help first year computer science students to grasp the basics of programming and could be a much more valuable foundation for the third and fourth years of instruction, when advanced computing courses are introduced. A study conducted by Pioro [7] suggests a positive correlation between students' grades in a computer programming course and the averages of two grades obtained in Discrete Mathematics I and Calculus I. However, if the mathematics modules were instead Calculus I and Calculus II, the correlation between the mathematics grades and the programming grades was insignificant.

When such a question as "What is the correlation between mathematics and programming?" has been raised, the following concerns should also be addressed: "What types of mathematics are we referring to?" "How do we define programming, and which programming paradigms are we refer- ring to?" "In which context, whether it is a professional or an educational context, do we do programming?"

In this paper, the aim is to focus on the correlation be-tween first year student performance in computer programming modules and in mathematics modules. In addition, the purpose of the study is to determine whether different types of mathematics may relate to certain programming paradigms. Driven by a large sample of 2,161 students, two programming modules, and two mathematics modules, this study aims to provide statistical analyses to determine the relationship between mathematics and programming.

II. RELATED WORKS

While studying mathematics and its role in learning programming has been an interesting research area in

Study	Variable 1	Variable 2	N	Pearson Correlation
Pacheco [6]	Calculus	Programming with C	59	.49**
Pacheco [6]	Calculus	Programming with Python	36	.41*
Pacheco [6]	Math grades	Programming with Python	36	.37**
Harris [4]	Math SAT grades	Programming aptitude test	16	.54*
Tukiainen [12]	Pre-UNI math grades	CS1 programming	33	-.28
Bennedsen [1]	Pre-UNI math grades	CS1 programming	20	.39
Bergin [2]	Pre-UNI math grades	CS1 programming	30	.46
** Correlation is significant at the 0.01. * Correlation is significant at the 0.05.				

Table 1: Previous research correlation results for mathematics and programming.

computer science education, research has focused on how mathematics ability could affect student performance in programming [7, 14, 3]. Mathematical abilities, such as problem solving, reasoning and abstraction, have been vital predictor variables to determine student success in programming [11, 12, 8] and to be important factors for designing programming aptitude tests.

Several studies have focused on finding the relationship between students' mathematics background and their programming aptitude. A study conducted by Pacheco [6] aimed to verify an assumption that a lack of problem solving ability could lead to difficulties in learning programming. The study focused on two cohorts of freshmen studying in two different institutions. Both groups' grades were analysed to identify the correlation with different learner characteristics such as programming background, problem solving ability, motivation, and learning styles. The results shown in Table 1 indicated a positive correlation between programming and calculus ability, as well as a relationship between students' mathematics grades in secondary education and students' performance in programming.

Another study, which predicts a variety of success factors in programming, concluded that mathematics grades from high school positively correlated with programming exam grades [1]. An additional study applied the SAT standardised test in mathematics as a predictor variable to measure student performance in programming, finding the correlation to be positive [4]. Similarly, a mathematics background plays a positive role on programming performance [2]. However, research aimed to apply a programming aptitude test and other factors, including mathematics grades, from high school to predict students outcomes in programming [12] concluded with controversial findings indicating a statistically insignificant negative correlation as shown in Table 1.

Issues that may be of concern include sampling methodology and sample size. For instance, in [6], the two cohorts of the sample are from different institutions, so that multiple educational factors could affect their results. Thus, quality of teaching and curriculum could be reflected positively or negatively on students' performance. Second, a small sample size could affect the statistical significance of the correlation.

III. Research Design

While the work of [6] was based on determining whether students' previous mathematics background grades could relate to students' performance in programming, our study aims to find the correlation between mathematics and programming measured in the first year, providing an in-depth analysis of the variety of mathematics modules that can affect students programming abilities by performing a statistical analysis.

A. *Research Questions*

RQ 1: What is the correlation between mathematical ability and programming in general?

RQ 2: What is the correlation between discrete mathematics and an introductory programming module?

RQ 3: What is the correlation between discrete mathematics and a data structure and algorithms module?

RQ 4: What is the correlation between calculus and an introductory programming module?

RQ 5: What is the correlation between calculus and a data structure and algorithms module?

B. *Data*

This study was conducted in the Department of Computer Science at the University of Warwick. The raw data consist of 2,161 students' marks during the period from 1996 to 2014. The marks include two programming modules and two mathematics modules, all core modules for first year computer science students.

The first module, Programming for Computer Scientists CS118, has used Java since the year 2000 to introduce programming fundamentals; prior to this, Pascal had been taught. The second programming module, Design of Information Structures CS126, involves data structures and algorithms, requiring CS118 as a prerequisite. For both modules, marks consist of examined and assessed components of a large programming assignment contributing 40% of the total mark. Mathematics modules as conceived from 1996 to 2005 are Mathematics for Computer Scientists CS124 and Discrete Mathematics CS127. In 2006, the replacement modules, Mathematics for Computer Scientists I CS130 and Mathematics for Computer Scientists II CS131, were introduced; these modules syllabi overlap with those of the previous modules, CS124 and CS127, respectively.

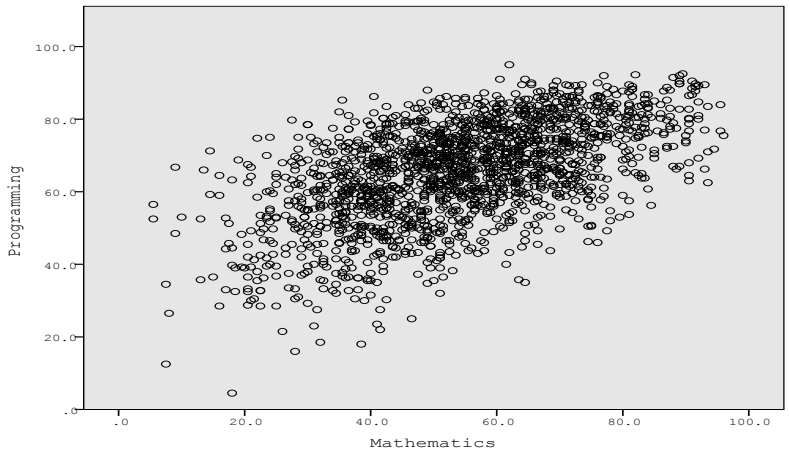

Figure 1: Linear relationship between programming and mathematics.

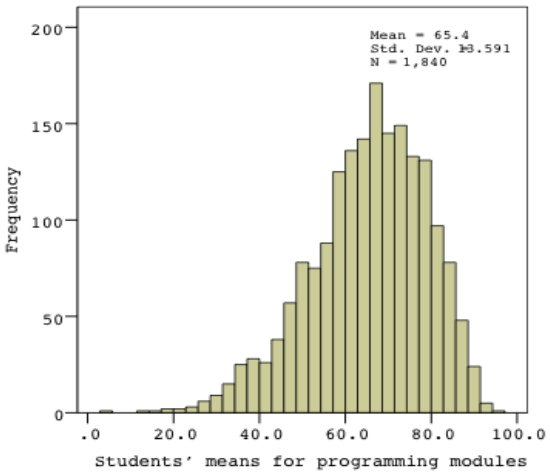

Figure 2: Distribution of students' means in programming modules.

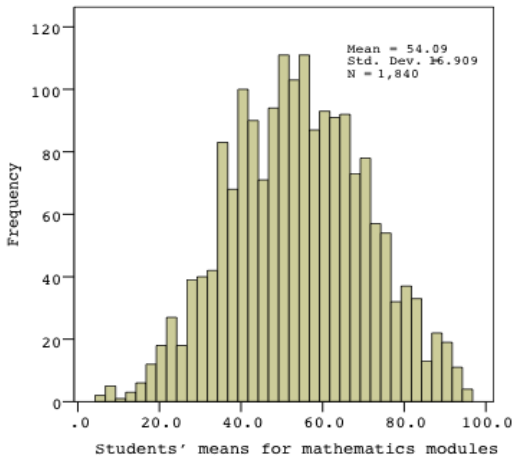

Figure 3: Distribution of students' means in mathmatics modules.

C. *Analyses*

We used the Pearson product-moment correlation coefficient which requires four assumptions to be fulfilled prior to the test. The first assumption is that variables must be continuous and can be measured either at the interval or ratio level. In our case, there exist two variables (students' marks) which are continuous and can be measured.

The second assumption is that the two variables must possess a linear relationship which can be proved by plotting a scatterplot and inspecting it visually. Figure 1 presents the scatterplot for the two variables and shows a positive weak to moderate linear relationship.

The third assumption is that there should be no significant outliers. Outliers could be the data which do not follow in the same pattern as other data points. In order to identify outliers, we use interquartile range to determine lower and upper bounds; data which are not within the boundaries could be outliers. Visual inspection of left tail of Figure 2 suggests the number of outliers with programming average below 31.

In our research context, it was realistic to have a number of students who were underachieving due to various factors; thus, we believe that those students' marks are not outliers and they will therefore not be removed from the data set.

The fourth assumption that needs to be addressed is to find whether variables are normally distributed, either by using numerical tests or graphic methods. Numerical tests, such as the Shapiro-Wilk and the Kolmogorov-Smirnov tests, indicate normality for small samples [10], whereas large sample sizes can be inspected visually for normal distribution using histogram. Figure 2 shows that the programming averages have imperfect normal distribution whereas Figure 3 indicates that averages for mathematics have normal bell-shaped distribution.

Programming	**Mathematics**	**N**	**Correlation**
		1840	**.56****
** Correlation is significant at the 0.01.			

Table 2: Correlation between mean of programming modules and mean of mathematics modules.

Intro Programming	Discrete Math	N	Correlation
CS118	CS127	1456	.52**
CS118	CS130	668	.57**
** Correlation is significant at the 0.01.			

Table 3: Correlation between introductory programming and discrete mathematics.

Data Structure	Discrete Math	N	Correlation
CS126	CS127	1433	.54**
CS126	CS130	643	.55**
** Correlation is significant at the 0.01.			

Table 4: Correlation between data structure and discrete mathematics.

IV. Results

A. Mathmeatics and programming (RQ 1)

Comparing the large sample with previous studies that aimed to find the relationship between mathematics and programming in the context of determining the success factors in programming, we aim to find the general correlation between the average of two programming modules, CS118 and CS126, and the average of two mathematics modules, CS124 or CS130 and CS127 or CS131. In order to calculate the correlation between the two averages, each student was required to have marks for all programming modules and for all mathematics modules. Thus, the total observations were reduced to 1,840. The result of the Pearson test indicated that there was a moderately positive correlation between mathematics and programming, as shown in Table 2.

B. Maintaining the Integrity of the Specifications and Programming (RQ 2 and 3)

Discrete mathematics plays a role in computer science education; most CS degree providers urge first-year students to enrol in discrete mathematics, and this discipline has been included in the ACM curriculum. However, the significance of discrete mathematics should be based on mutable factors, such as teaching methods, that help students to grasp, apply and evaluate mathematical concepts. Another element that needs to be considered by curriculum designers is how certain categories of mathematics could be appropriate to specific programming paradigms; for instance, teaching functional-driven language could be linked to function concepts in discrete mathematics [8]. In contrast, in object-oriented paradigms, the use of predefined set classes might not require students' understanding of mathematical set theory. In this stage of our analysis, we focus on object oriented programming module CS118. The results shown in Table 3 and 4 revealed that discrete mathematics correlated positively with introductory programming and data structures.

C. Calculus and Programming (RQ 4 and 5)

Teaching calculus to undergraduates in computer science is required in many institutions, but relevant questions include: how much calculus do students need, and how do they apply calculus in programming? It has been argued that requiring calculus as a prerequisite for the intermediate or upper-level mathematics courses that CS/SE students might take (e.g. combinatorics, graph theory, logic) is nonsensical because knowledge of calculus plays essentially no role in such

Intro Programming	Calculus	N	Correlation
CS118	CS131	642	.36**
** Correlation is significant at the 0.01.			

Table 5: Correlation between introductory programming and calculus.

Data Structure	Calculus	N	Correlation
CS126	CS131	624	.43**
** Correlation is significant at the 0.01.			

Table 6: Correlation between data structure and calculus.

courses and that, instead of calculus, discrete mathematics develops essential skills for computer science students [9]. Calculus could be seen to be less correlated with introductory programming than discrete mathematics is, as prior re- search suggests that students enrolled in calculus performed significantly less well in programming courses than others enrolled in discrete mathematics [7, 6], which explains that learning the basics of programming requires basic algebra and discrete mathematics. However, our results shown in Table 6 indicate that calculus has a more positive effect on student' performance in data structures compared with calculus role in introductory programming as shown in Table 5. Thus, calculus is a general mathematics concept that could play a role in learning programming by teaching specific problems that require the implementation of calculus. Further investigation into the role of calculus in different programming paradigms, such as functional, is still needed.

V. Conclusion

Generally, the statistical analysis of students' performance indicated that programming and mathematics had a positive, moderate correlation and there was no evidence in this study of causation. The roles of discrete mathematics and calculus in programming have been statistically analysed and resulted in a positive relationship. The analysis derived from a large sample of students' marks and a variety of computer science and mathematics modules in which the results could be more accurate than in previous, related work that had analysed small samples. In the educational context in which multiple factors are involved, it is difficult to draw a conclusion from statistical analysis; thus, other qualitative methods need to be considered in order to understand how other educational variables could affect the correlation between programming and mathematics. In addition, we need to investigate which programming paradigms, such as logical or functional programming, require different types of mathematics such as calculus.

References

[1] J. Bennedsen and M. E. Caspersen. An investigation of potential success factors for an introductory model-driven programming course. In Proceedings of the First International Workshop on Computing Education Research, pages 155–163. ACM, October 2005.

[2] S. Bergin and R. Reilly. Programming: Factors that influence success. SIGCSE Bull., 37(1):411–415, February 2005.

[3] A. Gomes and A. Mendes. A study on student's characteristics and programming learning. In Proceedings of World Conference on

Educational Multimedia, Hypermedia and Telecommunications 2008, pages 2895–2904. AACE, June 2008.

[4] J. Harris. Testing programming aptitude in introductory programming courses. J. Comput. Sci. Coll., 30(2):149–156, December 2014.

[5] P. B. Henderson and A. M. Stavely. Programming and mathematical thinking. ACM Inroads, 5(1):35–36, March 2014.

[6] A. Pacheco, A. Gomes, J. Henriques, A. M. de Almeida, and A. J. Mendes. Mathematics and programming: some studies. In Proceedings of the 9th International Conference on Computer Systems and Technologies and Workshop for PhD Students in Computing, pages 77:V.15–77:1. ACM, June 2008.

[7] B. T. Pioro. Introductory computer programming: Gender, major, discrete mathematics, and calculus. J. Comput. Sci. Coll., 21(5):123–129, May 2006.

[8] J. F. Power, T. Whelan, and S. Bergin. Teaching discrete structures: A systematic review of the literature. In Proceedings of the 42Nd ACM Technical Symposium on Computer Science Education, pages 275–280. ACM, March 2011.

[9] A. Ralston. Do we need any mathematics in computer science curricula? SIGCSE Bull., 37(2):6–9, June 2005.

[10] N. M. Razali and Y. B. Wah. Power comparisons of Shapiro-Wilk, Kolmogorov-Smirnov, Lilliefors and Anderson-Darling tests. Journal of Statistical Modeling and Analytics, 2(1):21–33, 2011.

[11] Simon, S. Fincher, A. Robins, B. Baker, I. Box, Q. Cutts, M. de Raadt, P. Haden, J. Hamer, M. Hamilton, R. Lister, M. Petre, K. Sutton, D. Tolhurst, and J. Tutty. Predictors of success in a first programming course. In Proceedings of the 8th Australasian Conference on Computing Education, pages 189–196. Australian Computer Society, Inc., October 2006.

[12] M. Tukiainen and E. Mönkkönen. Programming aptitude testing as a prediction of learning to program. In Proceedings of the 14th Workshop of the Psychology of Programming Interest Group, pages 45–57. Australian Computer Society, Inc., June 2002.

[13] C. Watson and F. W. Li. Failure rates in introductory programming revisited. In Proceedings of the 2014 Conference on Innovation and Technology in Computer Science Education, pages 39–44. ACM, June 2014.

[14] B. C. Wilson and S. Shrock. Contributing to success in an introductory computer science course: A study of twelve factors. SIGCSE Bull., 33(1):184–188, February 2001.