JOURNAL OF LEARNING ANALYTICS
SOLAR SOCIETY for LEARNING ANALYTICS RESEARCH
(2014) Curricular Design Analysis: A Data-Driven Perspective. *Journal of Learning Analytics*, *1*(3), 84–119.

Curricular Design Analysis: A Data-Driven Perspective

Gonzalo Méndez[(1)], Xavier Ochoa[(1)], Katherine Chiluiza[(1)], and Bram de Wever [(2)]
[(1)]Escuela Superior Politécnica del Litoral, Guayaquil, Ecuador
[(2)]Department of Education, Ghent University, Ghent, Belgium

ABSTRACT: Learning analytics has been as used a tool to improve the learning process mainly at the micro-level (courses and activities). However, another of the key promises of learning analytics research is to create tools that could help educational institutions at the meso- and macro-level to gain better insight into the inner workings of their programs in order to tune or correct them. This work presents a set of simple techniques that, if applied to readily available historical academic data, could provide such insights. The techniques described are real course difficulty estimation, course impact on the overall academic performance of students, curriculum coherence, dropout paths, and load/performance graph. The usefulness of these techniques is validated through their application to real academic data from a Computer Science program. The results of the analysis are used to obtain recommendations for curriculum redesign.

KEYWORDS: Statistical discourse analysis, informal cognition, social metacognition

1 INTRODUCTION

Analytics could be applied at very different levels within an educational institution. Siemens and Long (2011) seminally classified the scope of analysis into five levels: course, departmental, institutional, regional and national/international. The first two levels are the usual focus of learning analytics, while the remaining three are usually called academic analytics (Campbell, DeBlois, & Oblinger, 2007). Due to this distinction, most of previous research in learning analytics (Ferguson, 2012) deals only with the first level: the analysis of the behaviour and interaction of students and faculty as part of a course. Apart from studies on dropout (Wolff, Zdrahal, Nikolov, & Pantucek, 2013), there is very little research on how to analyze the learning process at the departmental or program level in order to guide the design or redesign of a curricular program.

To bootstrap the discussion of this macro level, this work proposes a simple set of learning analytics techniques that could help program coordinators or faculty groups gain a better insight on the nature and current performance of their programs and provide insight on possible problems or difficulties that the students may have with the program. These techniques are purposely designed to work with readily available academic performance data (final grades) to facilitate its adoption in a widely variety of institutions. While grades do not capture or explain the whole learning experience of a student, they provide the most common measurement applied to the learning process. To validate the usefulness of these simple techniques, they are applied to a case study involving at least twelve years of academic data of a particular Computer Science program that is being redesigned.

(2014) Curricular Design Analysis: A Data-Driven Perspective. *Journal of Learning Analytics*, *1*(3), 84–119.

The structure of the paper is as follows: Section 2 briefly reviews the related work on curriculum design techniques and the corresponding conceptual framework. Section 3 provides details on the context and data used in the case study. Section 4 describes a technique to establish the actual difficulty of the courses in the curriculum based on grades and their distribution. Section 5 provides a multivariate analysis to determine the negative impact of courses on the overall academic performance of students. Section 6 proposes the use of Exploratory Factor Analysis to find the underlying structure and coherence of the curriculum. Section 7 identifies how failing different courses affects the dropout rates of the program. Section 8 analyzes the performance of the students under different academic loads in order to recommend the optimal number of courses that should be taken in one academic period. Finally, Section 9 draws conclusions from the analyses performed and summarizes the recommendations for the redesign of the program in the case study. Additionally, this section presents a brief discussion about possible further work to improve the usefulness of the proposed techniques.

2 CONCEPTUAL FRAMEWORK AND RELATED WORK

Curriculum development comprehends planning, implementation, and evaluation of curriculum and the interactions of people, processes, and procedures with this curriculum (Ornstein & Hunkins, 2009). The literature broadly recognizes two models of curriculum development according to its orientation: product and process (See DaRosa & Bell, 2004; Neary, 2003). The former focuses more on the learning objectives, plans, and intentions with curriculum control, mostly on the side of the teacher; the latter emphasizes activities and effects with strong participation of students in decision making about the curriculum (O'Neill, 2010). The product model is rooted in Tyler's work (2013), which has greatly influenced curriculum development in both America and Europe (González & Wagenaar, 2003; National Committee of Inquiry in Higher Education, 1997). Despite this influence, currently, many curriculum designers all around the world, but especially in Latin America, are orienting curriculum development on learning outcomes or competences that students develop during their stay in higher education institutions (González, Wagenaar & Beneitone, 2004; Tobón, 2007; Villalobos, Gonzalez, Jimenez, & Rueda, 2011).

In this sense, learning outcomes are related to abilities developed or gained after being exposed to a learning experience (Watson, 2002). Curriculum development oriented to learning outcomes or competences is a process that generally follows this systematic approach: precise identification of the curricular design specifications and constraints (student outcomes, competences, learning goals), that are based on the needs of society; identification of a curricular conceptual model (research-based learning, inquiry learning, problem-based learning, case-based learning, etc.); developing and testing/evaluation of the curricular design; and refining the design with the feedback of students and stakeholders (Pukkila, DeCosmo, Swick, & Arnold, 2007). Due to the nature of this process, some of its components can be revisited after several iterations. A key component in this process is the establishment of the conceptual model related to instructional contexts that need emphasis,

teaching/learning methods, resources, etc. Moreover, the success of a learning-outcomes-centred curriculum is based on the effectiveness of reaching the required learning outcomes at a specified level (American Association for the Advancement of Science, 2001). International higher education accreditation agencies base their models of fulfilling specific criteria, among them the assessment and evaluation of learning outcomes (ABET, 2014). Therefore, assessment is an important component of feedback for the entire curriculum development process, generating opportunities to improve it. Nevertheless, many studies are devoted to the effects of certain components of a curricular conceptual model or the innovation of teaching strategies in the curriculum (Pukkila et al., 2007; Denton, Franke, & Surendra, 2005; Wolf, 2007), but few of them examine and redesign their curriculum using data that led them to a specific design decision in a related program. Precisely, Van den Akker (2003) underlines that in order to understand curriculum problems it is appropriate to analyze them with a broad perspective, for instance at the meso level (program level). This analysis should be performed after gathering the necessary information from inner curriculum levels (micro- and nano-levels) that have direct interaction with the outcomes of more external levels of the curriculum (Van den Akker et al., 2009).

Figure 1 depicts a typical learning-outcome-centred curriculum (Felder & Brent, 2004) based on the systematic approach described above. This figure also includes the proposed modification of including *data evidence* generated by the curriculum itself in the testing/evaluation component. This inclusion complements the learning outcome measurements of curriculum redesign inputs.

Studies that use a curriculum development model, as proposed in Figure 1 are not ubiquitous. Most of the available research regarding attempting to redesign a curriculum based on data follows a course level approach; in other words, researchers experiment with teaching strategies, learn from them, and later change their course curriculum. For instance Albert (2007) states, from the medical education point of view, most decisions related to curriculum and teaching are based on “opinions, intuitions, and personal preferences”; thus, she presents a curriculum for new rheumatology trainees based on trends that appeared after analyzing clinical problems with the greatest educational relevance in the rheumatology field. In this case, the decision making about the curriculum was more based on teaching experiences of relevant educational pieces. On the other hand, researchers such as Moretti, González-Brenes, and McKnight (2014), explored the use of data mining algorithms to discover patterns in learners’ feedback about their instructors and factors affecting Computer Science instruction, but, again, the information gained using these algorithms was applied at the course level. Nonetheless, despite these efforts, no studies yet present the analytical approach presented in this research.

(2014) Curricular Design Analysis: A Data-Driven Perspective. *Journal of Learning Analytics*, *1*(3), 84–119.

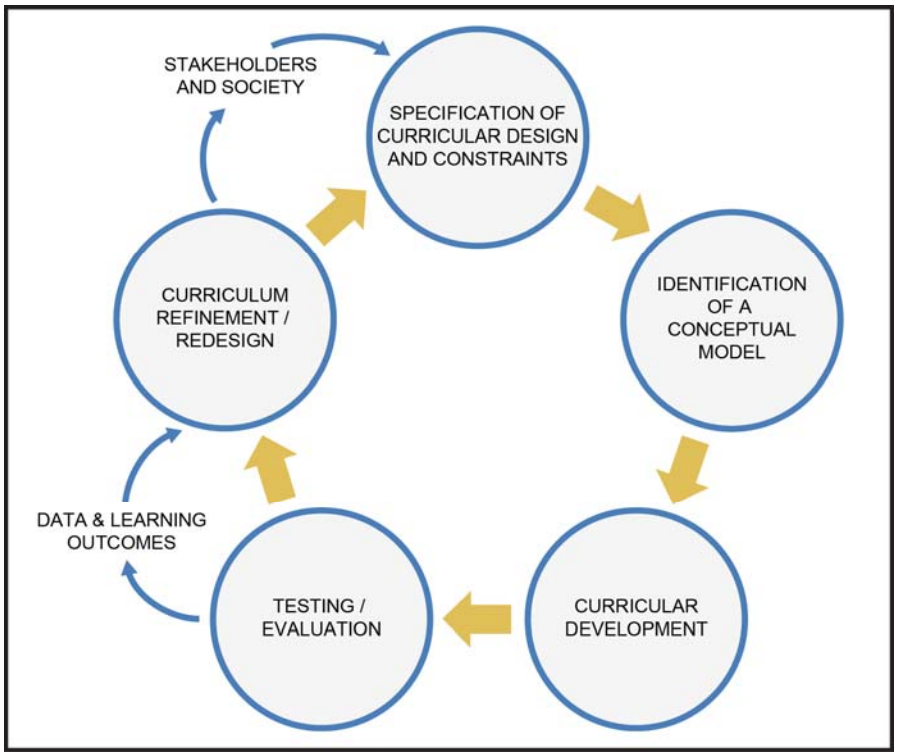


Figure 1: Proposed Data and Learning Outcomes Curriculum Development Model.

3 CASE STUDY

The data analyzed in this paper consists of the academic performance information of Computer Science (CS) undergraduate students from the Electrical and Computing Engineering Department of ESPOL University in Guayaquil, Ecuador. The studied dataset, composed of the grades of 2,543 individuals who eventually enrolled in the CS program of ESPOL, spans the academic history from 1978 up to 2012.

According to the performance scoring system of ESPOL, a class grade can take a real value from the interval [0.00, 10.00] and a student passes a given class by obtaining a grade equal to or greater than 6.00, otherwise he/she fails. The general CS curriculum categorizes its courses according to the topics and contents they cover: 1) Basic Sciences, 2) Humanities, and 3) Professional Training courses.

At ESPOL, students are allowed to adjust the number of courses they want to take within the selective-elective and free-selective categories, according to the number of credits they accumulate throughout their degree. Therefore, some students may complete the specified number of credits for these categories with more (or fewer) courses than others. Because of this, the entire academic history generated by the students considered in this analysis defined a huge set of unique courses. The heterogeneity of this set is not only explained by the diversification introduced by the selective-elective and free-selective courses of the program, but also by the curricular modifications performed on the programs during the time interval considered.

(2014) Curricular Design Analysis: A Data-Driven Perspective. *Journal of Learning Analytics*, *1*(3), 84–119.

In order to prepare such varied data for a proper analysis, a few processing steps were performed on the original dataset to *clean* it properly. First, we dealt with any missing information introduced by curricular modifications. Some of these changes include: 1) Courses removed from some versions of the curriculum, 2) Courses whose type was changed from compulsory to elective, 3) Courses split into more than one course, and 4) Courses whose names were modified. To cope with these events, a course-course matching process was conducted to determine the agreements among two or more distinct courses that can be considered equivalent across different versions of the curriculum. At the end of this course-matching process, the set of unique courses was slightly reduced but the dataset still contained a considerable amount of missing grading information.

To reduce the amount of missing data, we discarded in our analysis courses not highly frequently chosen by students and English courses that escape from the conceptual design of CS. With this step, we selected a common core of 26 courses mandatory for all CS students and for which grading information was readily available. The set of courses included in this last version of the dataset is illustrated in **Figure 2**. This set is composed of seven basic sciences courses, sixteen of professional instruction, and three from the humanities category.

4 DIFFICULTY ESTIMATION

Given the varied nature of the knowledge areas involved in the professional training process of computer scientists, as for many other academic disciplines, CS courses may exhibit distinct difficulty levels. The objective characterization of courses in terms of their difficulty is not a trivial task. Factors such as characteristics of the instructor, affinity of the student for the course material, or grading stringency standards applied are some variables that affect why the computation of a single value to represent the difficulty level of a course is not an easy-to-solve problem. In this regard, the number of credits contributed by a course within the curriculum is normally considered a difficulty indicator because it is intended to suggest the work and time loads that students are supposed to experience in a certain period of time (a week, for example) to pass a given course. However, the credit system overlooks other factors not necessarily related to time and effort. Special considerations such as extracurricular activities, other courses taken at the same time, and aspects not related to academic life, can make a course easier or harder independently of its number of credits.

$$HAG_j = \frac{\sum_i r_{ij}}{N_s^j} \quad (1)$$

(2014) Curricular Design Analysis: A Data-Driven Perspective. *Journal of Learning Analytics*, *1*(3), 84–119.

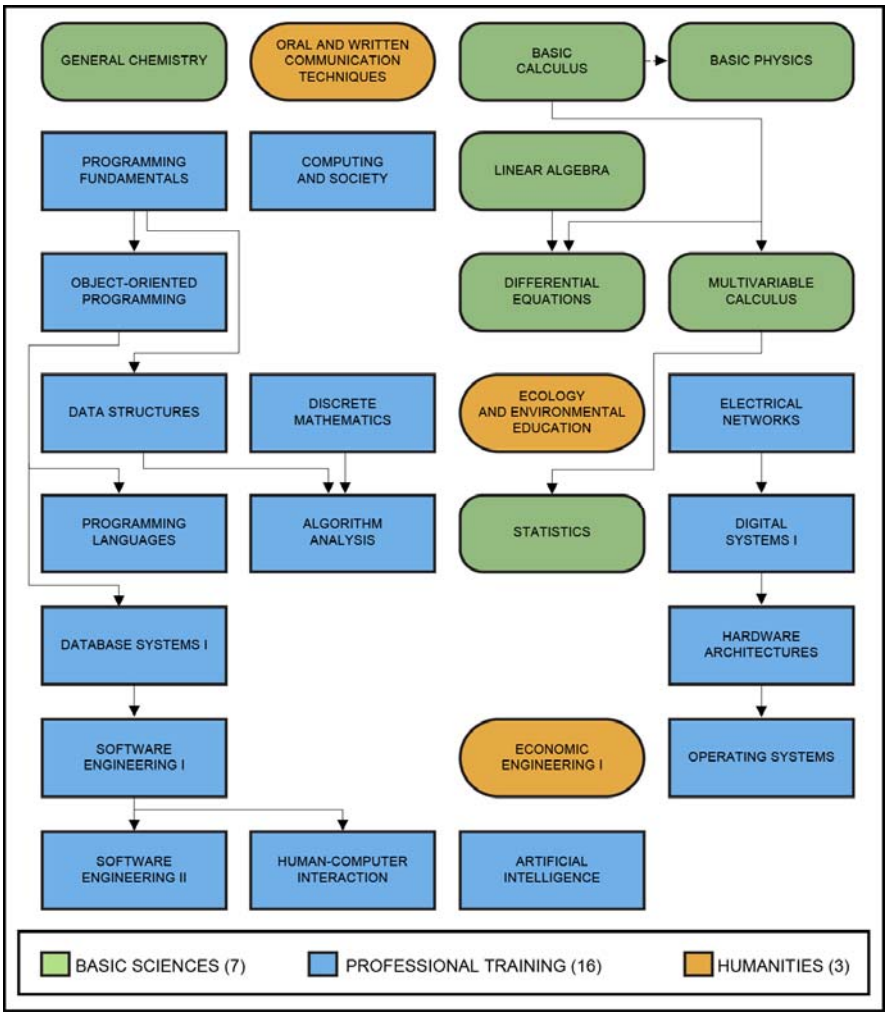


Figure 2: Courses included on the core curriculum of the ESPOL CS program.

The historical average grade of a course j, denoted here as HAG_j, and defined according to Equation 1, where r_{ij} is the grade obtained by the student i in course j and N_s^j is the total number of students who have taken the course j, might be the most basic data-inspired indicator that comes to mind when searching for objective estimators of course difficulty. However, the HAG may not only be biased by the factors mentioned above, but it also ignores the overall academic performance of the students who generated the grades involved in its computation. Thus, this value is not suitable to suggest, for example, whether a course classified as easy ended up being easy because it was always taken by high-performance students or because its contents were not challenging enough in such a way that even low-performance students got high grades on it.

In this section, the following research questions (RQs) are put forward:

RQ1: Which data-based indicators can support curricular designers or program coordinators to

decide which courses should be redesigned or revised due to their difficulty level?

RQ2: Do these indicators scale down to the level of the course teacher? More specifically, are these indicators related to the particular difficulty level associated with each teacher?

RQ3: Is there any substantial difference of difficulty in a given set of courses, considering that they are all taught by the same teacher?

The following subsections aim to answer these questions. First, several techniques are described and proposed to compute empirical, data-based, difficulty estimators for courses of an actual curriculum. These techniques are used to calculate difficulty estimators at the course level and are contrasted with the results of a perception study conducted to measure the subjective opinion of current CS students. Finally, the objective difficulty indicators are also calculated at the level of individual teachers to determine the effect that different courses have in the measured difficulty of the teacher.

4.1 Description

When estimating difficulty of a course, one indicator readily available to teachers and decision makers is the mean of the grades obtained by students in the course. For instance, if the mean of a given course is above the minimum passing grade, one may assume that the course was not difficult, but if the course mean is near or below that minimum then the opposite may be possible. However, the mean is an indicator of central tendency that is sensitive to outliers and is useful, foremost, in normal distribution. Grading systems where there is a minimum grade to pass a given course do not describe symmetric distributions. Thus, the mean as a general difficulty estimator of a course might not be such a good indicator due to the concentration of the mass of the distribution to either one side or the other. Caulkins, Larkey, & Wei (1996) propose other indicators that represent how difficult a course is. They propose that a course j has a grading stringency index β_j and a multiplicative magnitude α_j that represents its difficulty level. Both indicators consider the grade point average (GPA) in their computation and are computed according to Equations (2) and (3), respectively. In these equations, the GPA_i represents the overall academic performance of student i; r_{ij} and N_s^j are defined above.

$$\beta_j = \frac{\sum_i (GPA_i - r_{ij})}{N_s^j} \qquad (2)$$

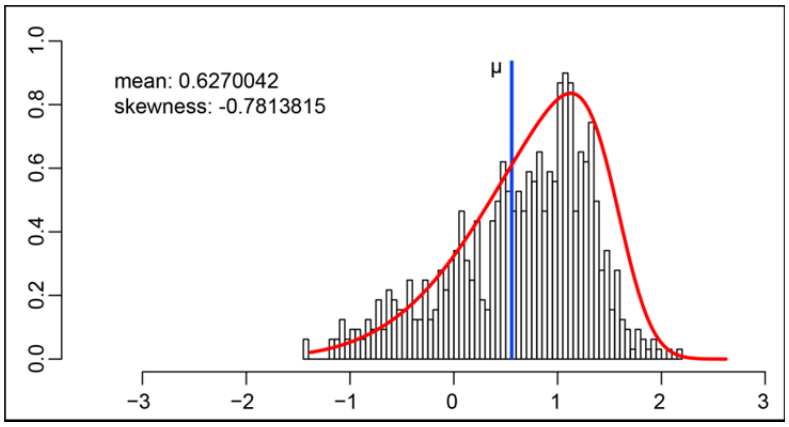
$$\alpha_j = \frac{\sum_i GPA_i{}^2}{\sum_i (r_{ij} * GPA_i)} \qquad (3)$$

The *influence* of a course on a student's academic history is related to the extent that the given course contributes either positively or negatively to the student's overall performance. If we consider the GPA as a general academic performance indicator, a course's contribution can be seen as positive or negative according to whether the grade obtained is greater or lower, respectively, than the given student's GPA. More specifically, the distance between the GPA and a particular course grade can be seen as a measure of how much the corresponding course shifted, up or down, the student's overall performance (GPA).

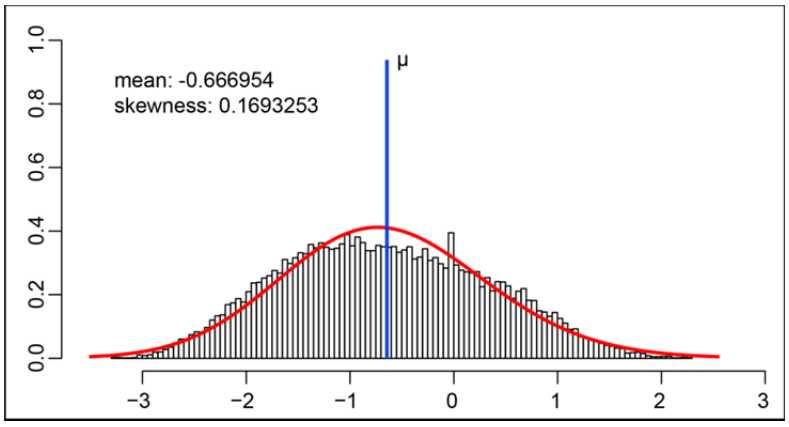
(2014) Curricular Design Analysis: A Data-Driven Perspective. *Journal of Learning Analytics*, *1*(3), 84–119.

When observed over time, on a set of several students, this distance might be considered an indicator of how much a given course usually moves students away from their GPA or brings them closer to it.

Note, however, that Equation (2) already takes into consideration the distance between the GPA and the course grade of students ($GPA_i - r_{ij}$). Thus, the grading stringency is an indicator of how much a course *moves* or *influences* the student's GPA and describes the mean of such influence. In this article, the authors argue that another way to characterize the difficulty of a course is the use of the skewness of the distribution of the distances between the GPAs of all the students who took a given course and the grades they obtained in this course. Examples of these distributions are shown in Figure 3 for two courses of the ESPOL CS curriculum: Algorithm Analysis (Figure 3a) and the Oral and Written Communication Techniques and Research course (Figure 3b).



(a) Algorithm Analysis



(b) Oral & Written Communication Techniques

Figure 3: GPA-course grade distances distribution for two courses from the ESPOL CS curriculum.

As can be observed, the shapes of these two distributions (shown in red) differ significantly in the longitude and leaning of their longer tail. In the case of Figure 3(a), Algorithm Analysis, the mass of the distribution is concentrated to the right of the figure, which means a negative skewness. On the

contrary, values for the Oral & Written Communication Techniques course are consolidated to the left side of their distribution mean, which leads to a positive skewness value. The skewness values for the distribution formed from the differences between the students' GPAs and their course grades are always related to whether the course grades have been greater or lower than the GPAs of the students involved in the computation. In that sense, the skewness of the distribution can be considered a difficulty indicator of a course. Thus, the more negative the skewness, the more difficult the course. In addition, the same distribution might be used to explore the probability of obtaining a grade that would increase or decrease a student's GPA. For instance, if a student taking Algorithm Analysis (negative skewness course) locates to the left side of this distance distribution, that student would have a lower probability of obtaining a grade above or near to his/her GPA. Conversely, someone who locates to the right side of the same distribution would have more chances of accomplishing a grade that negatively impacts his GPA. A possible interpretation for skewness is the bias of the teacher to assign "good" or "bad" grades to a selected group of students. For example, a positive skewness could be the result of a teacher who usually assigns higher grades to the best students than they would otherwise deserve. On the other hand, a negative skewness could represent a teacher who considers that no one should obtain the maximum grade and good students are graded lower than deserved.

In the following subsections, these difficulty indicators are analyzed both at the level of the course and at the level of the teacher.

4.2 Difficulty at the Course Level

The three approaches detailed in section 4.1 were applied over the courses that compose the curricular design of the CS program shown in **Figure 2**. The relationships among these three indicators for the twenty-six courses of the studied curriculum are depicted in Figure 4, where the courses appear ordered according to their β values.

As it can be observed, courses located to the left-hand side of the graph have lower α and β values and higher skewness (even with a positive value). The relative location of the courses within this graph is a general indicator of their level of difficulty for students. For instance, most of the courses taken by students during their last year of study appear on the left-hand side of the graph, whereas the ones on the right-hand side are mostly taken during the first two years (e.g. Basic Physics, Programming Fundamentals, etc.), which are also difficult. Therefore, if a student takes, for instance, Programming Fundamentals, she/he would face not only a difficult course, but would also face a very challenging situation because the skewness indicator is negative, which represents a grade that does not favour a higher GPA. Another characteristic to note in Figure 4 is that the α values are lower than their corresponding β values for those courses on the right-hand side and contrariwise for those taken in the last years.

(2014) Curricular Design Analysis: A Data-Driven Perspective. *Journal of Learning Analytics*, *1*(3), 84–119.

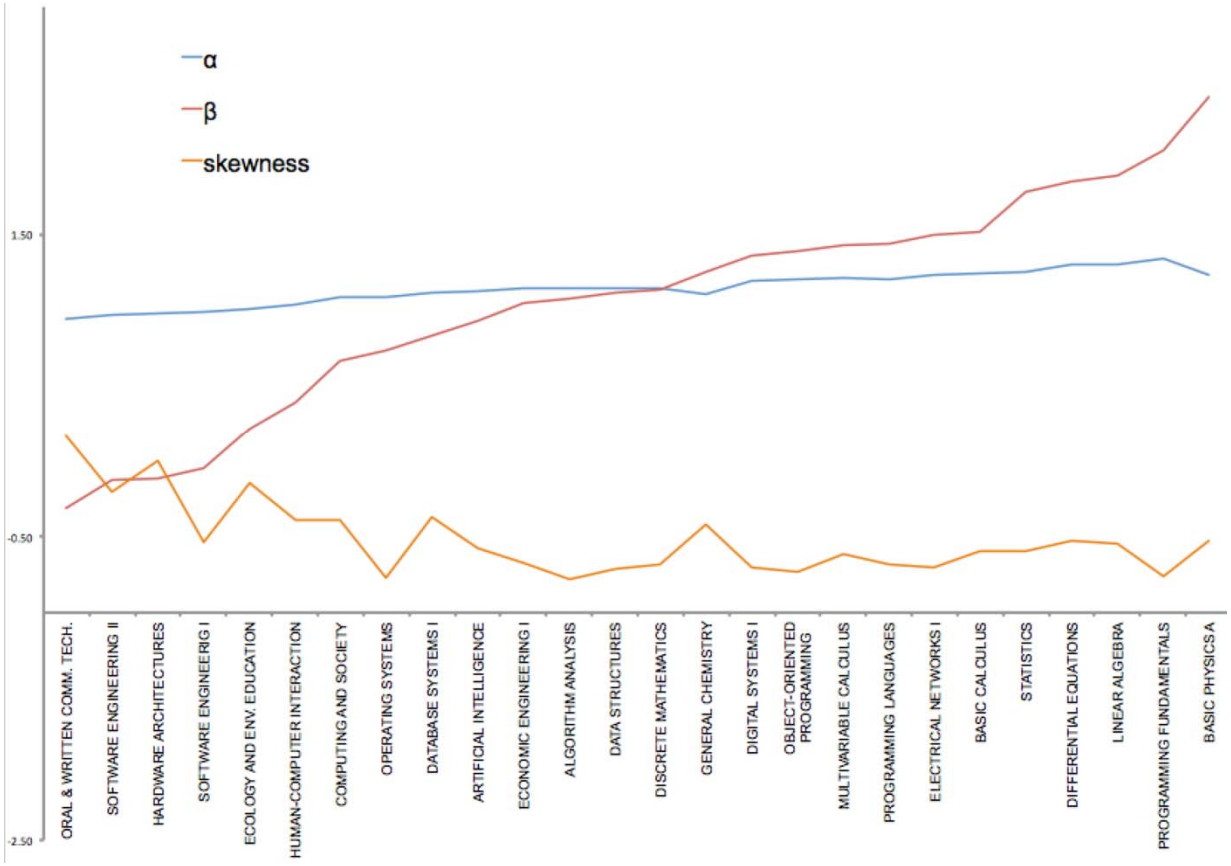


Figure 4: Difficulty Estimators for Twenty-Six Courses of the CS Program.

Parallel to these computations, a perception study was conducted with 80 out of the 317 current students of the CS program (25% response rate). The students were asked about their perception of courses they were currently taking or had taken previously. Perceptions about the following course characteristics were gathered using the survey: 1) difficulty level, 2) negative impact on GPA, and 3) importance within the program. The survey asked the student to identify a maximum of six courses that were considered 1) difficult, 2) had a high failure rate, and 3) important to continue the CS program. Descriptive results of difficulty level are shown in Figure 5. In subsequent sections, for contrasting purposes, other perception results will be presented.

As can be seen in Figure 5, courses like Linear Algebra, Differential Equations, Programming Fundamentals, Statistics, Object-Oriented Programming, and Data Structures, among others, are perceived as difficult, coinciding partially with the computations depicted in Figure 4. Nevertheless, other courses like Basic Physics, which is categorized in Figure 4 as one of the more difficult according to its α, β, and skewness values, seem to disappear in the memory of students as being difficult.

(2014) Curricular Design Analysis: A Data-Driven Perspective. *Journal of Learning Analytics*, *1*(3), 84–119.

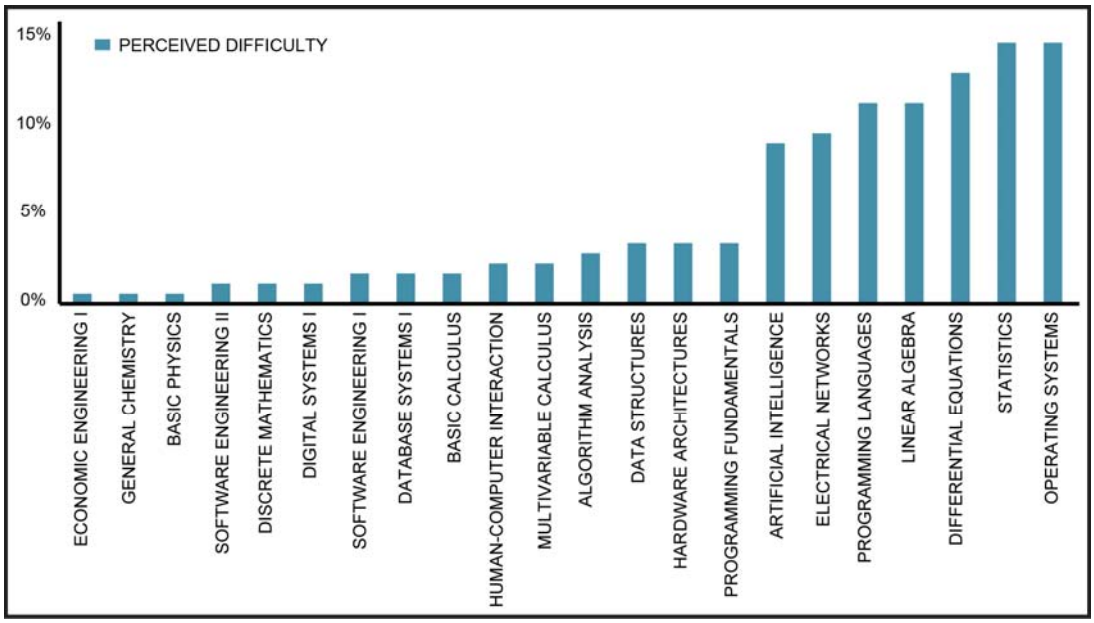


Figure 5: Student perceptions about difficult courses.

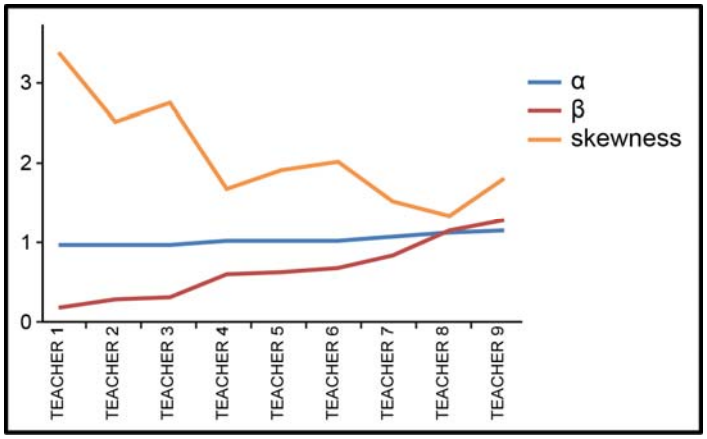
After contrasting the results obtained through the three approaches for calculating difficulty estimators with student perceptions about the level of difficulty and negative impact of courses to their GPAs, it was found that the majority of the courses that had a negative skewness, higher α and β values, are also perceived as courses with a high level of difficulty.

4.3 Difficulty at the Level of the Teacher

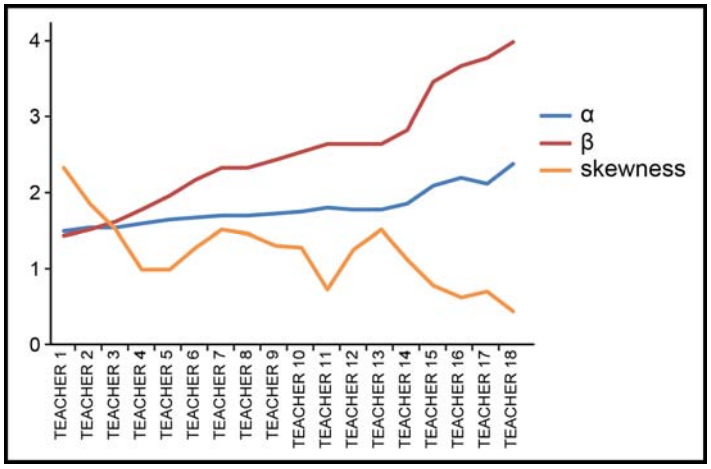
The dataset for this part of the study included all the students (N=531) who enrolled and passed the twenty-six analyzed courses in the last twelve years (first semester of 2000 to first semester of 2013). Only students who had finished the entire program were considered in order to have a stable final calculation of their GPAs. The three approaches to calculate difficulty indicators of a course were also used to find out whether what it is generally observed for a course differs or not when the teacher of the course is also considered in the difficulty estimation process. As such, this section aimed to answer the following research question: Do the α, β, and skewness indicators previously observed scale down to the level of the teacher? The first step in answering this question was to calculate the difficulty indicators for each course, but at the level of their corresponding teachers; that is, for each teacher who has taught a given course, we computed a triad of α, β, and skewness for that particular course. If, for example, 34 teachers taught Programming Fundamentals during the period under study, there would be 34 triads of difficulty indicators for the course.

The exploration of the triad of indicators revealed that those courses located in the "easy/ending zone" follow mostly the same pattern observed at the general course level. That is, the "easy" sections of a given course taught by a "less difficult" teacher, has an α indicator higher than its corresponding β value. In the same way, the more difficult a teacher is, the higher the α and β indicators are. The pattern

for the skewness is the same as in Figure 4; in other words, when analyzing a more difficult teacher, the skewness is lower when compared to an easier teacher. Figure 6 shows the calculated indicators per teacher, in β indicator ascending order, for two courses located at the opposite extremes of Figure 4.



(a) Operating Systems



(b) Basic Physics

Figure 6: GPA-Difficulty indicators per teacher for the Operating Systems and Basic Physics courses.

In Figure 6a, the Operating System course, located in the "easy/ending zone," mirrors the pattern explained above (see teachers 7 and 8). In Figure 6b, the difficulty indicators for the eighteen teachers of the Basic Physics course, which is part of the "difficult/initial zone," are shown. As can be seen, a slightly different pattern is observed: the α indicator is lower than the β indicator and the skewness diminishes when the difficulty increases. This last pattern is generally in line with the skewness indicator in Figure 4. Note, however, that the skewness, in general, is very dependent on the teacher; see, for example, teacher 13 in Figure 6b. The α and β estimators for this teacher indicate she/he is a difficult teacher; nevertheless, the impact on student GPA is positive when compared to other instructors proximate to

her/him in the "difficult/initial zone." This same pattern is observed in Figure 6a for teacher 9.

The patterns observed so far indicate the usefulness of the calculated difficulty indicators. The level of difficulty scales down to the level of a given course taught by different teachers and describes the level of difficulty each teacher represents to his/her students. The next question is whether there is a substantial difference of difficulty in a given set of courses, if they are taught by the same teacher.

Figure 7 presents the case of a teacher who has taught five different courses of the ESPOL CS curriculum. This case is especially interesting because on one hand, Discrete Mathematics, Operating Systems, and Algorithm Analysis are located in the "easy/ending zone"; on the other hand, Data Structures and Programming Fundamentals are from the "difficult/initial zone." Again, the patterns previously identified emerge here, specifically, the courses initially taken by students show a noticeably higher level of difficulty, even when a single teacher is analyzed. The skewness is also in line with that shown in Figure 4; note, however, that in this figure, the Operating Systems course has a lower skewness than the Discrete Mathematics course, but in **Figure 7**, we see the opposite. It seems that, despite the negative correlation of the skewness, in general, with the β estimator, the teacher may clearly modify this indicator in a given course.

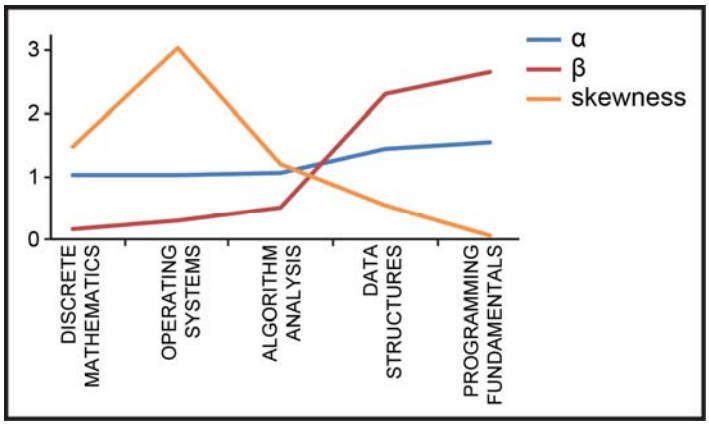


Figure 7: Difficulty indicators for courses taught by a single teacher.

Throughout this section, we examined the consistency between the calculated difficulty indicators at 1) the course level, 2) the teacher level for the same course, and 3) the teacher level when the same teacher has taught different courses. We found that the α and β estimators of Caulkins et al. (2007) and the skewness indicator proposed in this research might be useful when attempting to make sense of the observed grading. Grades per se are not unique indicators of a course's difficulty level. Teachers and content are very linked to the real difficulty a course imposes on student performance. On the other hand, the skewness indicator seems to be more sensitive, apparently linked to the content of a given course, but at the same time dependent on the style each teacher imprints on the course.

5 COURSE IMPACT ON GPA

Given the wide heterogeneity of classes that compose a curriculum, it must be expected that each of them influence a student's overall academic performance to a different extent. The identification of which courses have the most positive (or negative) impact on student GPA is of special importance for the decision making process that both faculty and students must face to best achieve their academic goals and objectives (Ansburg, 2001; Addison, Best, Warrington, 2006).

How much a course impacts the student GPA can be somewhat related to how difficult it is. However, as will be shown by the results of the perceptual study conducted for this research, students perceive course difficulty (section 4) and negative impact on their GPA differently. In this section, historical grading data is explored from the perspective of a Principal Component Regression (PCR) to answer the following research question:

> **RQ4:** What is the relative impact or influence that each course within a curricular design has on the overall students' academic performance?

The next sub-sections illustrate a strategy to answer this question. First, we will present the methodology used. Next, we apply this methodology to the CS curriculum to find objective evidence about the specific positive or negative impacts that a given course has over student GPA. These findings are also discussed and contrasted with student perceptions on the negative impact of their courses on their GPA.

5.1 Description

Principal Component Regression (PCR) is a multivariate exploratory technique in which an independent variable is regressed upon the principal components (PCs) resulting from a Principal Components Analysis (PCA) rather than upon the original measured variables (Jolliffe, 1982). These components do not have an evident meaning, but are used as a simplification technique to explain the effect of a variable upon others. In this technique, the variance of the original data is explained with as few factors as possible, retaining the fewest possible components. The retaining criterion used in PCR is based on the correlation among the dependent variables and the coefficients found on the linear regression performed in the analysis. In the context of this research, PCR is used as a *simplifying* technique to express the dependent variable (student GPA) in terms of some of the principal components found in the intermediate analysis, as proposed by Massy (1965). This analysis, leads to the regression shown in Equation 4, where PC_i refers to the i^{th} selected principal component with its corresponding regression coefficient β_i.

$$GPA = \beta_0 + \beta_1 PC_1 + \beta_2 PC_2 + \cdots + \beta_k PC_k \qquad (4)$$

Since each PC_i is a linear combination of the measured variables, these simplified results can then be

projected back into terms of the original course grades to determine to which extent each of the studied classes explains the response variable. This work proposes this explanatory power as an indicator of a course's relative importance when predicting overall student academic performance and, thus, as a measure of its impact, positive or negative, on student GPA.

5.2 Application

Data from 469 students (from first semester 2000 to second semester 2012) with complete grading information on the classes that compose the CS of ESPOL were analyzed in a PCR with student GPA as the response variable. **Table 1** shows a summary of the analysis for the first eight principal components found.

Table 1: Summary of the PCR for the first eight PCs.

	PC_1	PC_2	PC_3	PC_4	PC_5	PC_6	PC_7	PC_8
Standard Deviation	2.33	1.1689	1.07864	1.03627	0.99411	0.9442	0.9023	0.85644
Proportion Variance	0.26	0.06545	0.05573	0.05144	0.04734	0.0427	0.039	0.03514
Cumulative Variance	0.26	0.32546	0.38119	0.43263	0.47997	0.5227	0.5617	0.59681

As can be seen, up to the eighth component, the cumulative proportion of variance accounted for is 59%. In order to decide how many and which of the PCs found should be retained in the final regression model, their correlation with student GPA was analyzed. The correlation values computed for the previous eight PCs are shown in **Table 2**.

Table 2: PCs–GPA correlations for the first eight PCs.

	PC_1	PC_2	PC_3	PC_4	PC_5	PC_6	PC_7	PC_8
Correlation with GPA	0.948	-0.009	-0.039	0.039	-0.090	0.022	-0.003	-0.031

PC_1 and PC_5 have the strongest correlations with the response variable, with values of 0.948 and -0.090, respectively. From the coefficients for the linear combinations of these PCs, in terms of the original course grades (see **Table 3**), it can be observed that PC_1, the first principal component, has only positive loadings, with values between 0.1 and 0.3. On the other hand, PC_5 contains both positive and negative loadings. Therefore, this PC might be seen as representing a contrast between the positive and negative loadings associated with each course.

All the remaining PCs in this analysis correlate with the GPA to an extent that is always lower than the one found for PC_1 and PC_5. Thus, our further analysis will focus on predicting student GPA from the scores obtained in each of these two PCs. The fitted GPA values resulting from the linear model built from PC_1 and PC_5 are compared with the actual GPAs in Figure 8. The correlation value between these two variables is 0.952 and the final regression equation for the model is $GPA = 7.519 + 0.179\,PC_1 - 0.039\,PC_5$ (Residual standard error = 0.1337, Multiple R-squared = 0.9076, Adjusted R-squared =

0.9072).

Finally, as indicated previously, since both PC_1 and PC_5 are linear combinations of the original variables, this regression can also be expressed in terms of course grades. We computed the coefficients that each course might have to predict the GPA using the linear model. A plot of these coefficients illustrates the relative weight of each course grade for the prediction of GPA (Figure 9). In the plot shown, courses with negative (or low positive) influences appear in red, while the others fade to orange then yellow.

Table 3: Loadings associated with the linear combinations that form PC_1 and PC_5.

Course	PC_1	PC_5
Algorithm Analysis	0.205	-0.069
Artificial Intelligence	0.190	-0.069
Basic Calculus	0.157	-0.106
Basic Physics	0.120	-0.031
Computing and Society	0.187	-0.119
Data Structures	0.251	0.415
Database Systems I	0.167	0.086
Differential Equations	0.134	0.022
Digital Systems I	0.247	-0.022
Discrete Mathematics	0.251	-0.020
Ecology and Environmental Education	0.151	-0.384
Economic Engineering I	0.278	-0.048
Electrical Networks	0.120	-0.031
General Chemistry	0.199	-0.266
Hardware Architectures	0.234	-0.052
Human–Computer Interaction	0.196	-0.095
Linear Algebra	0.125	-0.040
Multivariable Calculus	0.158	-0.008
Object-Oriented Programming	0.273	0.230
Operating Systems	0.246	0.282
Oral & Written Communication Techniques	0.121	-0.521
Programming Languages	0.211	0.256
Programming Fundamentals	0.249	0.194
Software Engineering I	0.164	-0.119
Software Engineering II	0.157	-0.166
Statistics	0.141	-0.015

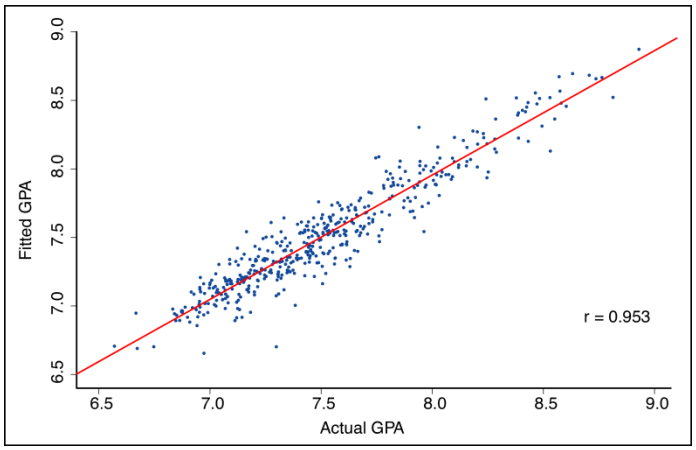


Figure 8: Actual and fitted GPA values.

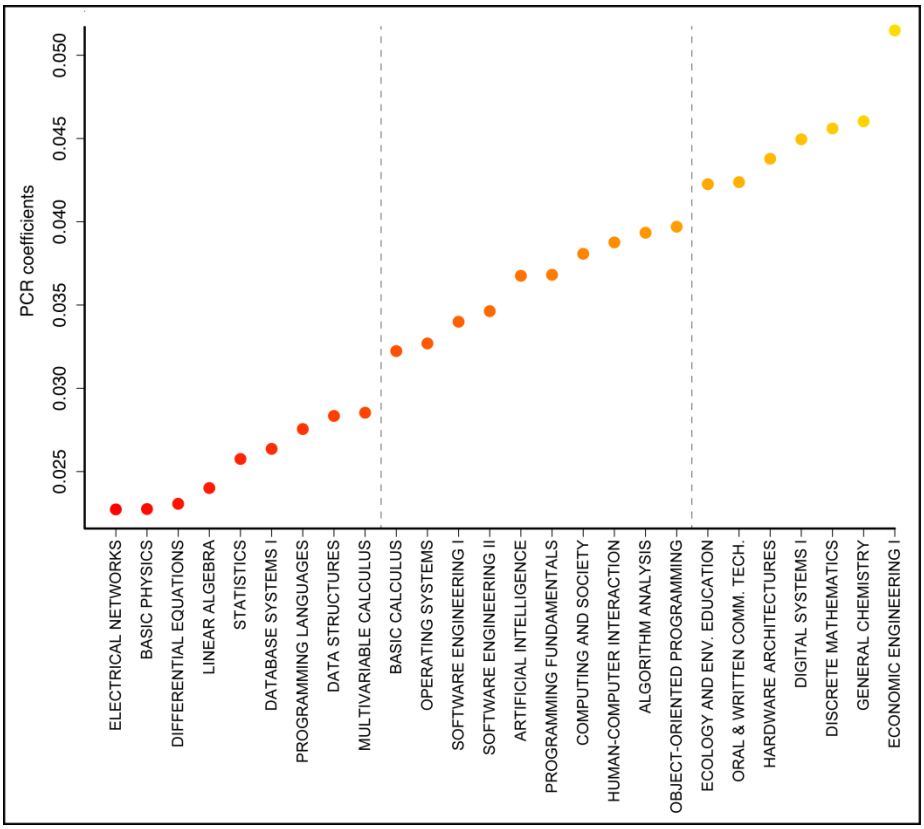


Figure 9: Ordered course coefficients for the prediction of student GPA.

As can be seen, the set of analyzed courses can be divided into three different groups, which appear in Figure 9 separated by the dashed lines. These three groups of classes can be considered as having a high, medium, and low negative impact on student GPA. Those clustered to the left-hand side of the plot may be considered those with higher negative impact on student GPA (note that their coefficients are low, which also represents a low positive contribution to the GPA). Classes with the higher PCR

coefficients appear clustered to the right-hand side of this plot, and can be considered to have a low negative impact on student GPA. Finally, those grouped in the middle region of the plot can be considered to have a medium negative (or neutral) impact on overall academic performance.

The results of this statistical analysis confirm most of those obtained in the perceptual study conducted with the students about the negative impact of their courses on their GPA, as depicted in Figure 10.

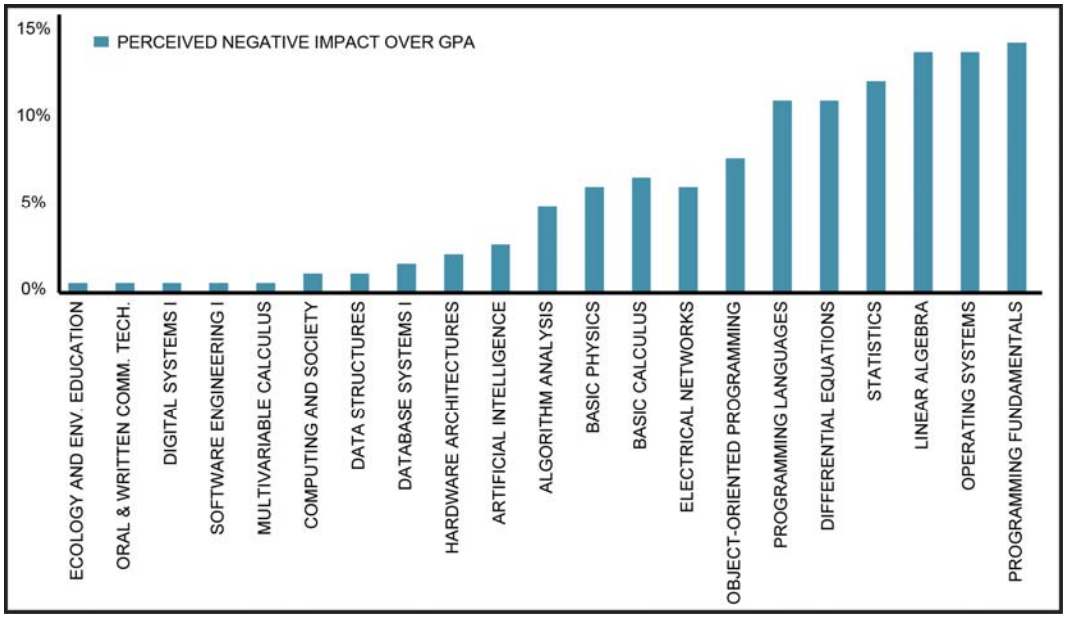


Figure 10: Student perceptions about courses with a negative impact on their GPA.

Three (Differential Equations, Statistics, and Linear Algebra) of the five courses identified by the students as those with the greatest negative impact on their GPA appear at the left-hand side of Figure 9. The other two (Programming Fundamentals and Operating Systems) appear in the mid cluster of the analysis.

Except for the Hardware Architectures course (contained by the middle cluster in Figure 9), all seven courses identified by the PCR analysis as those with the least negative impact either appear with the lowest percentages in the perception study (Ecology and Environmental Education, Oral & Written Communication Techniques, Digital Systems I) or were not even mentioned by the students (Discrete Mathematics, General Chemistry, Economic Engineering I). This suggests the usefulness of the PCR as a technique to measure the impact of courses on student GPA.

As expected, the intersection between student perceptions and PCR results for the middle impact zone is considerably large. The most notably misplaced course of this group would be the Programming Fundamentals course, identified by the students as the one with the highest negative impact in their GPA but categorized as a medium-impact course by the PCR technique. A similar misplacing occurs with the Electrical Networks course, which evidences a high negative impact in the PCR method, and a mid

impact according to student perceptions. The grouping of these two courses seems more related to the perceived difficulty reported by students rather than the perceived negative impact.

In general, the results of this analysis align to a very high extent with the student perception of the courses that commonly impact GPA in a negative way. In fact, most of the impact coefficients associated with the courses found by the PCR method can be naturally linked to their impact as perceived by the students. The identification of courses that will negatively impact the overall performance of students can lead to several concrete benefits:

1) Students can avoid taking concurrently two or more courses that have the potential to impact their GPAs negatively. The information about course difficulty can be used to better plan their academic progress in the program.
2) Additional pedagogical resources can be designed and provided to students for courses with a high negative impact in order to support the learning process so that overall performance is not highly affected.
3) A deep understanding of the nature of high-negative impacting courses can be achieved by further analysis and documentation.
4) Courses with too high positive GPA impact could be analyzed to determine if they are not challenging enough for students.

6 CURRICULUM COHERENCE

In the context of this paper, a coherent curriculum is defined as an organized set of courses that connects topics provoking students to achieve, by the completion of their degrees, several logically interrelated learning outcomes or competences (Stevens, Delgado, & Krajcik, 2010). The ACM and the IEEE Computer Society propose some curricular guidelines for undergraduate programs in computing, such as those defined in their Body of Knowledge (BoK) to cover the CS professional domain according to the desired competences of their future graduates. In such guidelines, a BoK is organized into several Knowledge Areas (KAs) further composed of various Knowledge Units (KUs) which ultimately constitute the building blocks of the courses included in the curriculum.

In this section, we will answer the following research questions:

RQ5: Which technique can be used to determine the coherence of a program?
RQ6: Is the current curriculum of the CS program in ESPOL coherently designed?

To answer these questions, an Exploratory Factor Analysis (EFA) is proposed as a technique to reveal the underlying structure of courses that should be grouped into sets that cover concepts associated with several interrelated professional competences. This set of courses should be coherently aligned with the

target learning outcomes of a given curricular design.

6.1 Description

The goal of EFA is to uncover the underlying structure of a relatively large set of variables by identifying interrelationships among them. This technique is especially useful to discover which variables in the set form coherent subsets that are relatively independent of one another (Tabachnick & Fidell, 2012). It outputs several unobserved latent variables, known as factors, which summarize patterns of correlation among the observed variables. These factors are thought to reflect the underlying processes or contents that have created the correlations among the observed input variables. In the context of this paper, the performance achieved by students in several courses that compose the CS curriculum are the individual variables that combine with the grades of some other courses to form factors that represent larger areas of the professional training process.

6.2 Application

The analysis described in this section used the academic performance of the same last six years of CS undergraduate students who successfully completed their degree at ESPOL between the first semester of 2000 and the second semester of 2012. A maximum-likelihood factor analysis was performed on the correlation matrix of the dataset composed by the performance achieved by 469 students on each of the 26 courses shown in Figure 2. The analysis was executed using SPSS (Version 20 for Mac). An oblique rotation (Promax) was applied on the first version of the loadings matrix resulting from the initial extraction of factors.

Twenty-six courses were included in the analysis. The Kaiser-Meyer-Olkin measure of sampling adequacy was 0.91, above the recommended value of 0.6, and Bartlett's test of sphericity was significant ($\chi^2(325) = 2{,}860.9, p < 0.05$). The Scree plot and cumulative variance explained by the eigenvalues were analyzed to decide how many factors were used in the further analysis. The first five factors had eigenvalues greater than one and a cumulative variance of 47.8%. Therefore, five factors were selected for subsequent analysis.

The groups of courses included in the five factors found in the analysis are shown in **Table 4**. This set of courses results after the application of a cut-off value of 0.3 on the loadings matrix of the rotated factors.

The groups of courses defined by the generated factors largely reflect the logic clusters defined in the curricular design of the analyzed CS program. Following, we provide the interpretations of each factor found according to the known characteristics of the analyzed curricular design.

Table 4: Loading factors for the five factors obtained from the EFA (cut-off > 0.3).

Course	Basic	Programming	Software	Advanced	Language

	Training	& CS Topics	Design	CS Topics	Paradigms
Differential Equations	0.578				
Electrical Networks	0.563				
Statistics	0.472				
Linear Algebra	0.455				
Multivariable Calculus	0.436				
Basic Calculus	0.435	0.312			
Basic Physics	0.372				
Digital Systems I	0.330				
Programming Fundamentals		0.708			
Object-Oriented Programming		0.409			
Discrete Mathematics		0.380			
Computing and Society		0.375			
Oral & Written Comm. Techniques		0.334			
Data Structures		0.325			
Hardware Architectures		0.311			
Software Engineering II			0.810		
Software Engineering I			0.560		
Human–Computer Interaction			0.510		
Operating Systems I				0.704	
Artificial Intelligence				0.674	
Programming Languages					0.823

6.3 Interpretation of Factors

6.3.1 Factor 1 (Basic Training)

Elements in this group can be classified as Mathematics, Science, and Basic Engineering courses. Mathematics related courses are included in this factor as well as Physics and related topics such as Electrical Networks (linked to applied physics).

6.3.2 Factor 2 (Programming and Basic CS Topics)

With the exception of Basic Calculus and Oral and Written Communication Techniques, all the courses in this group cover programming CS topics. This set of courses, called the programming sequence of the curriculum, covers abstract conceptualizations. The Communication Techniques course seems to be misplaced in this factor. The Basic Calculus course aims, among other goals, to support logical thinking in students; therefore, it is not unreasonable to see this course in Factor 2.

6.3.3 Factor 3 (Client Software Design)

This factor aggregates courses that seek to enhance and develop design skills and competences; moreover, the two software engineering courses are connected by a project developed throughout an

entire academic year. Additionally, the Human–Computer Interaction course develops skills related to the creation of new software/hardware solutions commonly linked to real-life projects. Several of the Human–Computer Interaction projects are also related to the Software Engineering projects.

6.3.4 Factor 4 (Advanced Topics in CS)

Two courses are grouped under this factor: Operating Systems and Artificial Intelligence. These courses, taken almost at the end of the program, require several prerequisites. The topics covered are the integrated application of techniques and concepts formerly learned by the students.

6.3.5 Factor 5 (Language Paradigms)

This factor includes only one course: Language programming; however, this was not unexpected. Several faculty members have noted that this course relates weakly related to other courses and no further deepening or connection is apparent in the curriculum.

Several courses included in the initial analysis were taken out of this 5-factor grouping: General Chemistry, Algorithm Analysis, Ecology, and Database Systems I. The lack of a latent variable that categorizes them into a "logical grouping" is an issue that should be considered when redesigning the curriculum. It is obvious that some courses are well connected and coherently aimed at developing a set of common competences; however, others unveil failures in the curriculum design. This problem is evident not only in these ungrouped courses that might be thought of as unnecessary in a given curriculum, but also in orphan courses or misplaced courses not necessarily related to the content or aims of their factors.

It can certainly be argued that curriculum design should not be data-driven. Despite the long path walked by curriculum designers, currently all around the world universities are involved in re-thinking their programs, based on what their alumni and their society require. Beyond the rich information a human being can provide for this design process, however, data can be a rich source of information as well. Nevertheless, analysis similar to that presented above requires subjective judgment. At the same time, analysis can only be done when the context in which the curriculum takes place is deeply understood. This implies that all the identified structures have to be explored further by human evaluators with expertise in the nature of the program and its particularities in order to see the logic behind the grouping structures.

7 DROPOUT AND ENROLMENT PATHS

In this study, an academic dropout is defined as a student who enrolled in an academic program and then abandoned it before completing the corresponding degree. This type of academic desertion has been widely investigated for several decades in works mainly oriented to explaining its causes. Factors such as race and gender (Jordan, Lara, & McPartland, 1996), socio-economic and family backgrounds (Rumberger, 1983), student engagement with school activities (Archambault, Janosz, Fallu, & Pagani,

2009), extracurricular load (McNeal, 1995), and the like, have been repeatedly explored, not only to explain dropout but also to construct predictive models for the early detection of potential desertion scenarios.

However, to the knowledge of the authors, no works have been conducted to explore the existence of relationships, if any, between dropouts and curriculum design. This section proposes using analytics to gain insight on courses that could be early indications of and even causes of dropout decisions in students. The goal of this analysis is to find answers to the following research questions:

RQ7: Is there a differential impact when passing or failing different courses on the probability of students dropping out from the program?
RQ8: Are there any frequent enrolment paths followed by students within a curriculum that lead to dropout settings?

To approach the first question, we will conduct a simple frequency analysis to establish the effect that different courses have on the outlook of students. For the second question, we will use a sequence analysis to find frequent sequences of courses that could lead students to abandon the program.

7.1 Course Impact on Dropout

It is common knowledge among faculty that students who fail courses are more disposed to dropout from a given program. This has been extensively explored as the “academic factor” in many dropout studies (Tinto, 1975). Due to their focus on the student and not on the program, these studies generally use the GPA of the student as a proxy of academic performance, overlooking the impact that individual subjects have on dropout. The following analysis focus on courses and not on students in order to gain insight on the impact that failing a course may have on the general probability of students completing their studies or not.

7.1.1 Description

The methodology to assess the impact that each course has on student dropout is based on simple frequency statistics. First, for each student in the study group, we calculate the dropout status. Because students can usually leave and re-enter the program at practically anytime, determination of the dropout status of a student is not a clear-cut calculation. A practical method to determine the dropout status is to determine the number of no-enrolment cycles. This number, however, is dependent on the characteristics of the enrolment in the program. For example, for programs with bi-annual (semester) enrolments, a student can be labelled as a dropout if there are two consecutive semesters with no enrolment in any course. For annual enrolments, the lack of one enrolment could be considered as a dropout. Students not labelled as dropouts are again divided into those who have finished the program (completed all the required courses) and those still taking courses. Students who have completed the required courses are given the status “Success”; those still pursuing the program are excluded from the analysis. Next, for each course, the numbers of students who have passed or failed at least once are

calculated. Finally created a cross table with the frequencies of the intersection of the Pass/Fail Course and the Success/Dropout Program information. **Table 5** shows an example of such an intersection.

Table 5: Cross-tab of Pass/Fail and Success/Dropout information for the Basic Calculus course.

	Pass Course	Fail Course
Status: Success	398	278
Status: Dropout	120	297

Finally, for each course, the percentage of Pass->Success and Fail->Success are calculated. For the example in Table 5, 398 out of 518 students who passed the course (77%) finished the program successfully (Pass->Success rate). On the other hand, only 278 out of the 575 who failed the course at least once (48%) were able to finish the program successfully (Fail->Success rate). This information, together with the fail rate of the course (students who failed the course divided by the total number of students who took the course) can be used to construct visualizations to determine the impact that different courses have on the dropout of students.

7.1.2 Application

This analysis was applied to the 26 core courses of the CS Program at ESPOL. All the students who entered the program since the first semester of 2000 until those who entered in the first semester of 2013 were considered in the study. A student was considered to have dropped out if there were at least 2 semesters without enrolment in the program. With this rule, from the 1,564 students initially considered, 677 (43%) successfully completed the program and 426 (27%) currently have a dropout status. Also, 461 (30%) are still taking courses regularly and were excluded from the analysis.

The calculations presented in the previous sub-section were applied to each one of the core courses of the CS program (see **Table 6**). Courses in the first levels (e.g., Computing and Society, Oral and Written Communications) have a lower Pass->Success rate (from 68% to 80%). Courses in the last years (e.g., Software Engineering I and II, Operating Systems) have a higher Pass->Success rate (from 85% to 90%). This distribution is expected, as it is more probable that a student will drop out from the program during the first semesters and less probable that they will abandon their studies with just few courses left to finish. Another interesting trend is that the Fail->Success rate does not strictly follow the chronological order of the courses. For example, Computing and Society and Oral and Written Communication Techniques are taken at similar times during the program; however, the first one has a Fail->Success rate of 43% while the second has a Fail->Success rate of 21%. Similarly, Operating Systems and Artificial Intelligence are both taken at the end of the program, with the first having a Fail->Success rate of 77% while the second rises to 86%.

Table 6: Pass->Success, Fail->Success and Fail-rate indicators for the 26 core CS courses order by Pass->Success.

Course	Pass->Success	Fail->Success	Fail-rate
Computing and Society	68%	43%	12%
Oral & Written Comm. Techniques	70%	21%	4%
Ecology and Environmental Education	74%	40%	5%
Discrete Mathematics	74%	40%	13%
General Chemistry	75%	41%	18%
Programming Fundamentals	76%	57%	39%
Basic Physics	76%	52%	40%
Basic Calculus	77%	48%	37%
Multivariate Calculus	78%	57%	23%
Linear Algebra	79%	52%	31%
Data Structures	81%	65%	22%
Economic Engineering I	82%	28%	5%
Algorithm Analysis	83%	68%	9%
Object-Oriented Programming	83%	55%	13%
Electrical Networks	84%	77%	34%
Statistics	84%	69%	31%
Differential Equations	84%	67%	35%
Database Systems I	85%	75%	15%
Programming Languages	85%	73%	17%
Software Engineering I	86%	75%	1%
Hardware Architectures	86%	69%	2%
Artificial Intelligence	87%	86%	13%
Software Engineering II	87%	64%	2%
Digital Systems I	88%	70%	15%
Human–Computer Interaction	88%	68%	3%
Operating Systems	90%	77%	11%

To help understand these numbers, they can be visualized in a bubble plot (Figure 11). In this figure, each circle represents a course. The x-coordinate of the circle is determined by the Pass->Success rate of the course and the y-coordinate by its Fail->Success rate. The Fail-rate of the course determines the size of the circle. The straight line marks the average difference between Pass->Fail and Pass->Success in this group of courses (a difference of 22 percent points). From this graph, three groups of courses can be easily identified based on the impact that they have on the dropout of students:

1) "Heavy-hitters" courses have a large Fail-rate and have a higher-than-average difference between the Pass->Success and Fail->Success rates. A representative of this group is Basic Calculus. If you fail this course at least once, your chances of completing the program successfully drop from 77% to 48%, and 37% of the students who take this course fail it at least

once. Other courses in this category are Basic Physics and Linear Algebra.

2) "Easy-but-deadly" courses are classified as easy by the analysis in Sections 3 and 4 and have a small Fail-rate (less than 6%); however, if you fail any of those courses, the chances of completing the program falls steeply. A representative of this group is Oral and Written Communication Techniques. If you fail the course at least once, your chances of completing the program successfully are only 21%. Other courses in this group are Ecology and Environmental Education and Economic Engineering.
3) "Fail-Friendly" courses have not much difference in dropout chances whether students fail them or not. A representative of this group is Artificial Intelligence. Failing this course only increases the chances of failing by one percent point. The other member of this group is Electrical Networks.

These results have many implications for the redesign of the CS curriculum. First, the "Heavy-hitters" are mainly Basic Science courses, which indicates, once again, the relative importance that these courses have in the curriculum regarding the success of the student. In addition, the impact of failing a course decreases with each passing semester, with most of final-year courses having a difference between Pass->Success and Fail->Success below the program average. This finding suggests that a better distribution of difficult courses among all the program years could lead a decreased dropout rate.

As part of the survey explained in section 4, students identified the level of importance of a course in order to continue and finish the CS program. Only one "Heavy-hitter" course, Linear Algebra (46%), was identified as of key importance to stay in the program. The other two courses in this category, Basic Physics (10%) and Basic Calculus (15%), were mentioned at a much lower proportion. This could be explained by the fact that the survey was applied to active students, not those with dropout status. On the other hand, Programming Fundamentals (50%) is perceived as the most important course in the program but does not show a large decrease in the success rate if the student fails it (from 76% to 57%). The explanation for this perception could be that Programming Fundamentals is the CS course with the highest Fail-rate (37%).

The students mentioned none of the "Easy-but-deadly" courses during the survey. The explanation for this could be that the perception is lost because those who struggle with this set of courses have already dropped out of the survey pool.

This analysis, being based only on grades, cannot account for all the reasons that lead students to fail. Other researchers (Tinto, 2012) have already pointed to several factors, apart from academic performance, as causes of dropout. However, curriculum designers can only affect dropout rates through curriculum design. Improving the curriculum cannot guarantee that a student will not dropout, but it could certainly facilitate the academic experience.

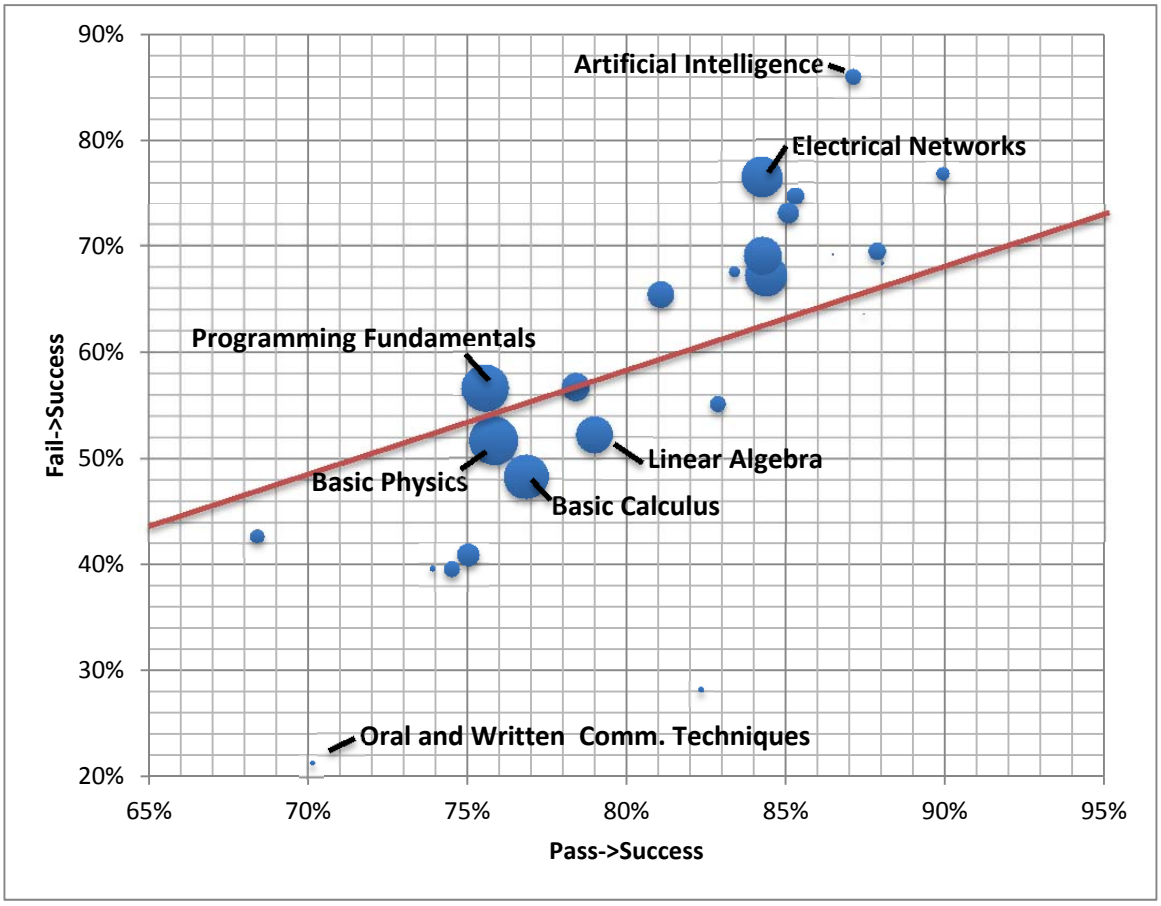


Figure 11: Bubble plot of the Pass->Success and Fail->Success rates of different courses. Size of the bubble represents the Fail-rate of the course. Straight line represents the average difference between Pass->Success and Fail->Success.

7.2 Enrolment Paths and Dropout

In order to discover the effect that failing several courses has on dropout, the sequences in which the students fail courses was analyzed. This analysis refers to an enrolment path as a specific sequence of courses taken (or failed) by a student along his/her academic history. The following section explains the theoretical motivations for the application of sequence mining and details how it can be used to track student enrolment patterns that lead them to quit their academic programs.

7.2.1 Description

The goal of sequence mining is to discover sequential patterns of frequently repeated events in a given database of temporal transactions (Parthasarathy, Zaki, Ogihara, & Dwarkadas, 1999). Among many others, scenarios where this type of mining task has been used include sales databases of items bought by customers (Ester, Kriegel, & Schubert, 2002), analysis of user activity in web searching and browsing (Pei, Han, Mortazavi-Asl, & Zhu, 2000), and disease identification based on sequences of symptoms (Turner, Clutterbuck, Semple, et al., 2003).

In sequence mining, frequent patterns are considered to emerge over time as the events that compose the sequences occur repeatedly. A sequence α is considered frequent if it is contained by a minimum number of input-sequences that form the database D. This minimum value is called the $support$ or $frequency$ of α. To illustrate the concept of $support$, consider the scenario where a bookstore keeps a historical record of the books bought by each customer. This information could be mined to find behavioural patterns in user buying habits to predict, for example, how likely it is that a given customer who bought book A will eventually come back to buy book B, given that 80% of previous buyers of book A actually did so.

Considering that student academic history takes places over a period with very well known course timestamps, sequence mining techniques can be applied to find frequent patterns in the way students enrol in the courses that compose their program curriculum. More specifically, frequent sequence analysis can be conducted over the failure segments of a student's academic history to investigate whether particular patterns of failing courses lead to more dropout scenarios than others.

7.2.2 Application

The failing academic events of 426 dropouts were investigated to find the most frequent courses and sequences of courses failed by students before abandoning their undergraduate studies. The analyzed subset was part of a greater cohort of 1,564 students who enrolled in the CS program of ESPOL from the first semester of 2000 to the first semester of 2013. Thus, the original CS population shows a dropout rate of 27%.

The SPADE algorithm (Zaki, 2001), an approach for the fast discovery of sequential patterns, was applied to the described subset of enrolment information using a minimum support of 0.3. The results of this algorithm are detailed in **Table 7**, which clearly shows that before deciding to quit, a large proportion of the investigated students failed courses in the basic training levels of their curriculum (that is, courses from Factor 1 of section 6.3) confirming the results presented in the previous sub-section (7.1). This result implies that most of these students decided to drop out immediately after dealing with fundamental concepts related to general Science and Engineering. Of all these courses, the most frequent event associated with dropout was failing the Basic Physics course, which was failed by about 61% of the students who eventually left the CS program. There is no evidence of any interaction between the failing of courses apart from failing Basic Physics and Basic Calculus in the same semester.

Since the investigated dropouts did not take place at advanced levels of their academic history, students who abandoned the CS program did it without having the opportunity to interact with actual CS topics. The Programming Fundamentals course is the first CS course that is part of the more frequent steps before dropout. It appears third on the list with a $support$ of 0.53.

Results of the sequence mining performed are consistent with those obtained from the perception study described in Sections 4 and 5. Interestingly, the Basic Physics course is perceived as an important course

that enables a student to continue the CS Program (10%) but not as important as the Programming Fundamentals course (50%). Therefore, failing such courses has a higher impact on students continuing their CS education. Again, 6 out of 11 perceived important courses are from the basic sciences strand and are mostly the same as those found on the failing enrolment path that leads to dropping out of the CS Program. This apparent relationship can be interpreted as a good indicator of the importance given to them by students who eventually graduate.

Table 7: Frequent failing enrolment paths that lead to dropout scenarios ($support$ = 0.3).

Sequence	*Support*
<{Basic Physics}, {Dropout}>	0.608
<{Basic Calculus}, {Dropout}>	0.570
<{Programming Fundamentals}, {Dropout}>	0.533
<{Basic Physics, Basic Calculus}, {Dropout}>	0.434
<{Linear Algebra}, {Dropout}>	0.433
<{General Chemistry}, {Dropout}>	0.320
<{Differential Equations}, {Dropout}>	0.312

This section proposed sequence mining for the identification of enrolment paths that frequently lead to dropout scenarios within a curriculum, as defined by students. The identification of these critical trajectories in curricular design is of special importance in deciding how to reduce or minimize dropout rates. Finding critical paths to avoid dropout and reduce the negative impact of certain courses on desertion rates is a key input for curricular design. Administrators, faculty members, and curricular designers can take data-driven decisions and plan accordingly. Moreover, this technique can also be applied to academic events of other types, such as successful scenarios to detect enrolment paths more likely to conduct students to graduation.

8 LOAD/PERFORMANCE GRAPH

The balance between academic load and academic performance is of vital importance for student success. The achievement of an optimal trade-off between these two factors is highly related to curricular design in the sense that it defines (or at least suggests) how students should enrol in the courses of their degree program. How courses are assigned to different levels, the established prerequisites, and the list of possible optional courses are some of the factors that influence the academic history of students at all levels. This implies that good curricular design should seek the maximization of student academic performance at the same time that it properly manages their workload. Given this context, mechanisms to visualize the load versus performance profile of a curriculum are highly desired. The research question to be approached in this section is:

RQ9: Which visualization techniques could be useful to represent the student load/performance

information to guide the redesign of the curriculum?

This section presents two visualizations of the student load/performance information useful to understanding the course taking and approval behaviour of students inside the program.

8.1 Description

Academic load can be associated with several factors: 1) number of courses taken in a particular academic period, 2) total difficulty pursued by the student, and 3) total grading stringency to which the student is subjected. On the other hand, performance can be seen as a function of other several factors: 1) number of courses a student must pass in a given academic period, 2) total course difficulty achieved, and 3) total grading stringency of the courses the student managed to pass. Constructing visualizations of the trajectories followed by students in the load/performance graph could provide curriculum designers with information about how well different students manage the recommended class load and what is the actual preferred load of different kinds of students.

8.2 Application

Figure 12 presents the density plot of the semester load and performance information for all students of the investigated dataset (from the first semester of 2000 to the first semester of 2013). Load is represented here by the difficulty of the courses taken by the student. On the other hand, performance is expressed in terms of the total difficulty of courses passed. This graph represents the relation between the total difficulty achieved and the total difficulty pursued in a semester. We calculate the total difficulty pursued by a student as the sum of the individual courses taken in a given academic period. Similarly, the total difficulty achieved is defined as the sum of the difficulties of the courses passed. Being a density plot, this graph shows how many times students had a specific value in the load/performance plane.

As can be easily seen in Figure 12, most students have a load around 4, 5, or 6 (darker tones). A large population usually achieves the pursued difficulty; that is, they pass all the courses they take. A noticeable population of students also fails 1 or 2 courses (dark spots under the diagonal), meaning that these students were taking more load than they could effectively manage. In the curriculum, the recommended load is six courses. However, the largest population of students is not taking that load, but only five or even four courses. Even with that load, a fair number of them are failing at least one course. It is clear that a more realistic recommended load of five courses (or four if a difficult course is taken) is more advisable in this program.

(2014) Curricular Design Analysis: A Data-Driven Perspective. *Journal of Learning Analytics*, *1*(3), 84–119.

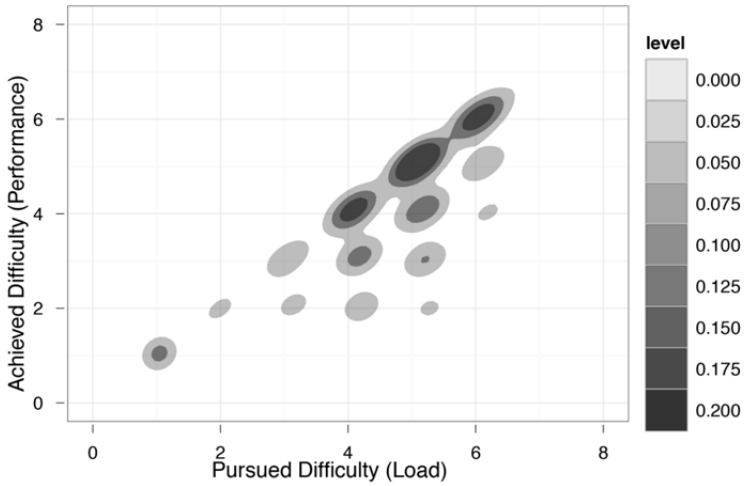


Figure 12: Semester load versus performance density plot.

Another way to visualize the load pursued and approved by students is to create a flow diagram that takes into consideration the sequence in which they took the courses. Such a visualization is presented in Figure 13. The rows in the diagram represent the number of Pursued and Approved (passed) courses. The columns represent the relative semester in which the student took those courses. Each node represents the number of courses taken and approved in a given semester. For example, students who took five courses in the first semester and approved all five will appear in the Pursued 5 and Approved 5 row and Semester 1 column. The size of each node represents the number of students who have taken and approved the same number of courses in a given semester. The width of the arrows represents the number of students who transit from one node to another. Only arrows with more than five students are represented. This graph allows a deeper analysis of the load/performance information of students in the program. It is again clear that most of students take mainly five, four, or six courses during the first ten semesters. A failure state — when the load passed is less that the load taken — occurs mainly during the first four semesters. The main path taken by students is taking five courses over ten semesters to complete the 50 courses required to finish the program. Some students need additional semesters to finish. This is depicted in the tail of the graph were it can be observed that students are taking and approving one course during semesters 11 to 18. The only semesters when a large proportion of students takes six courses appears to be semesters 6, 7, 8, and 9, namely the last planned semesters of the program.

All the results from these two analyses suggest that the recommended load should be reduced to five courses instead of the currently recommended six or seven courses, especially during the first semesters. In addition, the different paths followed show the importance of adequate counseling to novice students so that they can determine the load they can manage in order to finish the program successfully.

(2014) Curricular Design Analysis: A Data-Driven Perspective. *Journal of Learning Analytics*, *1*(3), 84–119.

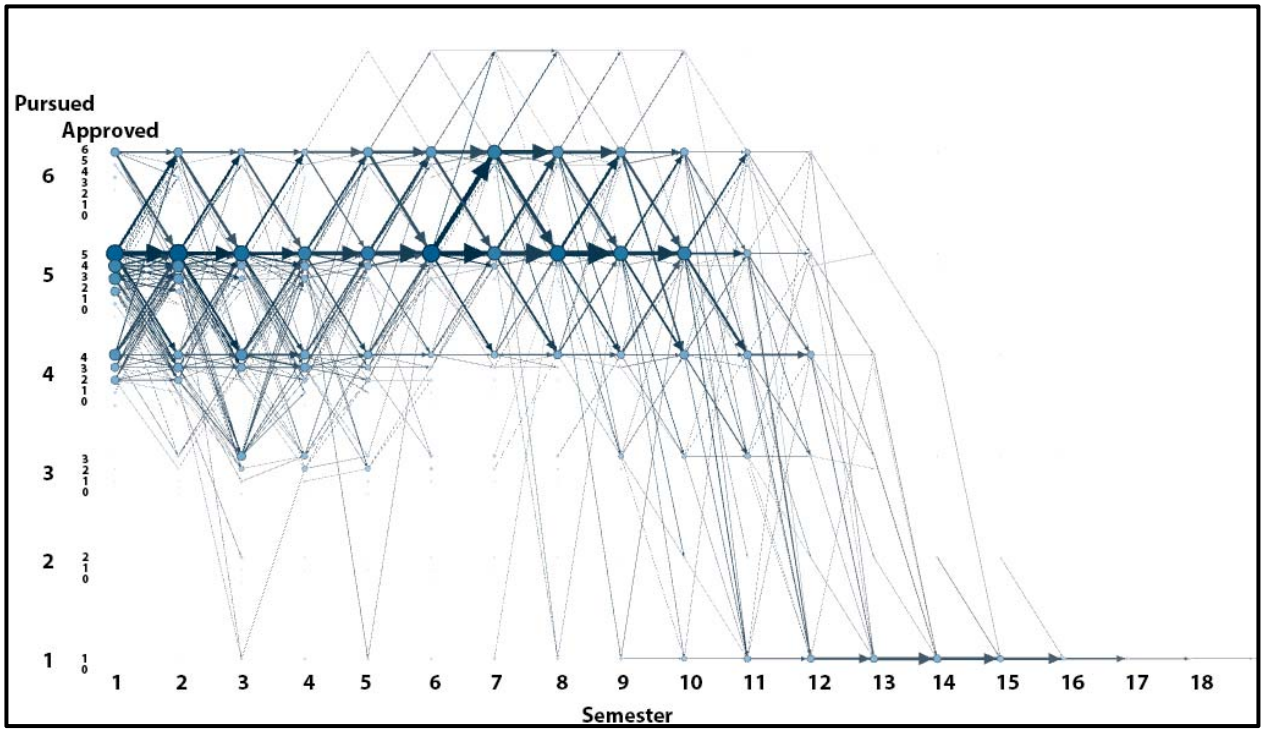


Figure 13: Flow diagram of student load/performance information

9 CONCLUSIONS AND FURTHER WORK

Historical academic performance data, while limited, provides an easy gateway to begin conducting learning analytics studies on the characteristics and relations of diverse courses inside a program curriculum. The information that can be obtained, even with very simple techniques, could help faculty and institutional administrators to understand their programs beyond a merely theoretical viewpoint. The techniques presented in this work have the potential to start data-informed discussions by faculty members that could lead to an improved learning process for the student, not by changing particular aspects of a course, but by redesigning the whole program curriculum. It is important to note that results presented by the proposed analyses should be interpreted in light of existing theory and common sense. At this stage, the techniques may only pinpoint interesting anomalies or (ir)regularities, but the sense-making process should be conducted by individuals with deep knowledge of the program been studied.

The application of these techniques to the case study has provided valuable information to be used in a planned redesign of the CS program at ESPOL. The main results are summarized in the following list:

- The group of Basic Science courses heavily influences the CS program. Those courses are among those with higher difficulty, meaning that they have a larger negative impact on student GPA and that failing them increases the probability of dropping out of the program. Given that the focus of the program is Computer Science and not Computer Engineering, the curriculum

designers should consider carefully which basic science concepts should be covered and where they should be positioned in the curricular map.

- The coherence analysis suggests the existence of orphan courses that do not effectively connect with the others (Programming Languages, General Chemistry). This disconnection could be a symptom of a lack of alignment with the program learning outcomes. Curriculum designers should pay special attention to these courses to determine their real usefulness for achieving program goals.
- The current curricular map presents unrealistic performance targets to students. A redesign of the curriculum should also introduce new ways that this information is presented to students to encourage a more personalized selection of load/performance balance.
- It is clear from the difficulty analysis that the teacher of a course has a large influence on the level of performance of the students. The goals and guides established by the curriculum designers should be clearly communicated and adopted by faculty. Preferably, they should be included in the redesign process in order to avoid a mismatch between the goals of the program and the individual course goals of each professor.

Far from presenting a comprehensive set of analytics tools for curriculum (re)design, this work discusses how very simple statistical and computational techniques can be applied to historic academic performance data to gain insight on possible problems or improvement opportunities in the entire program curriculum. Moreover, the real objective behind this research is to encourage other researchers to critique and improve these simple techniques in order to extend the impact of learning analytics beyond the scope of individual courses. As such, there are several ideas that could improve or extend this work:

- It could be useful to integrate the proposed techniques into a technological tool aimed at supporting curricular designers or program coordinators during the curricular redesign process. This tool (or set of tools) could facilitate the adoption of data-based decision making on curricular redesign even for expert designers who lack expertise in data processing.
- The usefulness of the techniques cannot be fully validated by just one case study. These techniques need to be applied to different contexts: programs in different subjects, programs with different structures, durations, and grading schemes, and programs that take place in different cultural settings. Only this kind of experimentation could lead to the development of valuable techniques worth transferring to curriculum design practitioners.
- As this work is far from exhaustive, further work by the learning analytics community should include the development and validation of new techniques that could better utilize grade data or more sophisticated data, such as curricular content and/or regular student and faculty surveys.

As a final remark, the results of this analysis should be presented to faculty in a way that avoids value judgment. Data analysis cannot determine if a program, course, or teacher is “good” or “bad,” only that

it approaches or deviates from certain patterns. It is important to stress that the role of learning analytics is to guide human reflection and cannot become an automatic assessment metric or replace good judgement.

ACKNOWLEDGEMENTS

The authors wish to acknowledge the support of the VLIR-UOS project ZEIN2010RIP09 and the SENESCYT Project "Andamios."

REFERENCES

ABET. (2014). *Accreditation policy and procedure manual (APPM) 2014–2015*. Baltimore. Retrieved from www.abet.org

Addison, W. E., Best, J., & Warrington, J. D. (2006). Students' perceptions of course difficulty and their ratings of the instructor. *College Student Journal*, *40*(2), 409–416.

Albert, L. (2007). Curriculum design: Finding a balance. *Journal of Rheumatology*, *34*, 458–459.

American Association for the Advancement of Science. (2001). *Designs for science literacy*. Oxford: Oxford University Press.

Ansburg, P. I. (2001). Students' expectations of workload and grade distribution by class difficulty. Paper presented at the annual meeting of the American Psychological Association, San Francisco, CA.

Archambault, I., Janosz, M., Fallu, J. S., & Pagani, L. S. (2009). Student engagement and its relationship with early high school dropout. *Journal of adolescence, 32*(3), 651–670.

Campbell, J. P., DeBlois, P. B., & Oblinger, D. G. (2007). Academic analytics: A new tool for a new era. *Educause Review*, *42*(4), 40.

Caulkins, J. P., Larkey, P. D., & Wei, J. (1996). Adjusting GPA to reflect course difficulty. Heinz College Research, Carnegie Mellon University.

DaRosa, D. A., & Bell, R. H. (2004). Graduate surgical education redesign: Reflections on curriculum theory and practice. *Surgery*, *136*(5), 996–974.

Denton, J. W., Franke, V., & Surendra, K. N. (2005). Curriculum and course design: A new approach using quality function deployment. *Journal of Education for Business*, *81*(2), 111–117.

Ester, M., Kriegel, H.-P., & Schubert, M. (2002). Web site mining: A new way to spot competitors, customers and suppliers in the World Wide Web . Presented at the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, July 2002, Edmonton, AB, Canada (pp. 249–258).New York: ACM.

Felder, R. M., & Brent, R. (2003). Designing and teaching courses to satisfy the ABET engineering criteria. *Journal of Engineering Education*, *92*(1), 7–25.

Ferguson, R. (2012). Learning analytics: Drivers, developments and challenges. *Technology Enhanced Learning*, *4*(5), 304–317.

González, J., & Wagenaar, R. (2003). *Tuning educational structures in Europe.* (Final – Phase One).

Bilbao: Universidad de Deusto.

González, J., Wagenaar, R., & Beneitone, P. (2004). Tuning-América Latina: Un proyecto de las universidades [Tuning-Latin America: A project of the universities]. *Revista Iberoamericana de Educación*, *35*, 151–164.

Jolliffe, I. T. (1982). A note on the use of principal components in regression. *Applied Statistics*, *9*(2), 300–303.

Jordan, W. J., Lara, J., & McPartland, J. M. (1996). Exploring the causes of early dropout among race-ethnic and gender groups. *Youth & Society*, *28*(1), 62–94.

Massy, W. F. (1965). Principal components regression in exploratory statistical research. *Journal of the American Statistical Association*, *60*(309), 234–256.

McNeal Jr., R. B. (1995). Extracurricular activities and high school dropouts. *Sociology of Education*, 62–80.

Moretti, A., González-Brenes, J. P., & McKnight, K. (2014). Data-driven curriculum design: Mining the Web to make better teaching decisions. *Proceedings of the 7th International Conference on Educational Data Mining.* July 2014. London, UK.

National Committee of Inquiry into Higher Education. (1997). *Higher education in the learning society*. Norwich, UK: HMSO (Her Majesty's Stationery Office).

Neary, M. (2003). Curriculum concepts and research. In *Curriculum studies in post-compulsory and adult education: A teacher's and student teacher's study guide*. Cheltenham: Nelson Thornes Ltd.

O'Neill, G. (2010). Initiating curriculum revision: Exploring the practices of educational developers. *International Journal for Academic Development*, *15*(1), 61–71.

Ornstein, A. C., & Hunkins, F. P. (2009). *Curriculum foundations, principles and issues* (5th ed.). Boston: Allyn & Bacon.

Parthasarathy, S., Zaki, M. J., Ogihara, M., & Dwarkadas, S. (1999). Incremental and interactive sequence mining. Presented at the Eight International Conference on Information and Knowledge Management, Kansas City, MO: ACM (pp. 251–258).

Pei, J., Han, J., Mortazavi-Asl, B., & Zhu, H. (2000). Mining access patterns efficiently from web logs. Proceedings *Knowledge Discovery and Data Mining: Current Issues and New Applications*, Kyoto, Japan, April 18–20, (pp. 396-407). Springer Berlin Heidelberg.

Pukkila, P. J., DeCosmo, J., Swick, D. C., & Arnold, M.S. (2007). How to engage in collaborative curriculum design to foster undergraduate inquiry and research in all disciplines. In K. K. Karukstis & T. Elgren (Ed.), *How to design, implement, and sustain a research-supportive undergraduate curriculum: A compendium of successful curricular practices for faculty and institutions engaged in undergraduate research*. Washington, D.C.: Council on Undergraduate Research.

Rumberger, R. W. (1983). Dropping out of high school: The influence of race, sex, and family background. *American Educational Research Journal*, *20*(2), 199–220.

Siemens, G. & Long, P. (2011). Penetrating the fog: Analytics in learning and education. *Educause Review*, *46*(5), 30–32.

Stevens, S. Y., Delgado, C., & Krajcik, J. S. (2010). Developing a theoretical learning progression for atomic structure and inter-atomic interactions. *Journal of Research in Science Teaching*, *47*, 687.

Tabachnick, B. G., & Fidell, L. (2012). *Using multivariate statistics* (International Edition). Boston, MA:

Pearson.

Tinto, V. (1975). Dropout from higher education: A theoretical synthesis of recent research. *Review of Educational Research*, *45*(1), 89–125.

Tinto, V. (2012). Completing college: Rethinking institutional action. Chicago: University of Chicago Press.

Tobón, S. (2007). El enfoque complejo de las competencias y el diseño curricular por ciclos propedéuticos [The complex approach of competences and curricular design for preparation cycles]. *Acción Pedagógica*, *16* (January–December), 14–28.

Turner, F. S., Clutterbuck, D. R., Semple, C. A., et al. (2003). Pocus: Mining genomic sequence annotation to predict disease genes. *Genome Biology*, *4*(11), R75.

Tyler, R. W. (2013). *Basic Principles of Curriculum and Instruction*. Chicago, IL: University of Chicago Press.

Van den Akker, J. (2003). Curriculum perspectives: An introduction. In J. van den Akker, W. Kuiper, & U. Hameyer (Eds.), *Curriculum landscapes and trends* (pp. 1–10). Dordrecht: Kluwer Academic Publishers.

Van den Akker, J., de Boer, W., Folmer, E., Kuiper, W., Letschert, J., Nieveen, N., & Thijs, A. (2009). *Curriculum in development*. Netherlands Institute for Curriculum Development (SLO). Enschede, Holland

Villalobos, J., Gonzalez, O., Jimenez, C., & Rueda, F. (2011). Curricula design model for designing and evaluating systems and computing engineering programs (pp. S4E–1, S4E–7). Presented at the Frontiers in Education Conference (FIE), Rapid City, South Dakota.

Watson, P. (2002). The role and integration of learning outcomes into the educational process. *Active Learning in Higher Education*, *3*, 205–219.

Wolf, P. (2007). A model for facilitating curriculum development in higher education: A faculty-driven, data-informed, and educational developer supported approach. *New Directions for Teaching and Learning*, *112*, 15–20.

Wolff, A., Zdrahal, Z., Nikolov, A., & Pantucek, M. (2013, April). Improving retention: Predicting at-risk students by analysing clicking behaviour in a virtual learning environment. *Proceedings of the 3rd International Conference on Learning Analytics and Knowledge* (pp. 145–149). New York: ACM.

Zaki, M. J. (2001). Spade: An efficient algorithm for mining frequent sequences. *Machine Learning*, *42*(1–2), 31–60.