QUANTIFYING THE THEORY VS. PROGRAMMING DISPARITY USING SPECTRAL ANALYSIS

Natarajan Meghanathan

Department of Electrical & Computer Engineering and Computer Science,
Jackson State University, Jackson, MS, USA

ABSTRACT

Some students in the Computer Science and related majors excel very well in programming-related assignments, but not equally well in the theoretical assignments (that are not programming-based) and vice-versa. We refer to this as the "Theory vs. Programming Disparity (TPD)". In this paper, we propose a spectral analysis-based approach to quantify the TPD metric for any student in a course based on the percentage scores (considered as decimal values in the range of 0 to 1) of the student in the course assignments (that involves both theoretical and programming-based assignments). For the student whose TPD metric is to be determined: we compute a Difference Matrix of the scores in the assignments, wherein an entry (u, v) in the matrix is the absolute difference in the decimal percentage scores of the student in assignments u and v. We subject the Difference Matrix to spectral analysis and observe that the assignments could be partitioned to two disjoint sets wherein the assignments within each set have the decimal percentage scores closer to each other, and the assignments across the two sets have the decimal percentage scores relatively more different from each other. The TPD metric is computed based on the Euclidean distance between the tuples representing the actual numbers of theoretical and programming assignments vis-a-vis the number of theoretical and programming assignments in each of the two disjoint sets. The larger the TPD score (in a scale of 0 to 1), the greater the disparity and vice-versa.

KEYWORDS

Spectral Analysis, Theory vs. Programming Disparity, Eigenvector, Bipartivity.

1. INTRODUCTION

Spectral analysis of a complex network has been observed to reveal significant structural details that would have been hitherto unknown in the scientific community [1]. Spectral analysis of a network graph typically involves the computation of the Eigenvectors and their corresponding Eigenvalues using one of the symmetric matrices such as the adjacency matrix, Laplacian matrix [7] and etc that reflect the topology of the network [2]. The indexes of the entries in an Eigenvector correspond to the sorted order (increasing order) of the node ids. Spectral analysis of an *nxn* matrix results in *'n'* Eigenvalues and the corresponding 'n' Eigenvectors (one Eigenvector per Eigenvalue). Any two Eigenvectors are orthogonal to each other (i.e., the dot product of any two Eigenvectors is zero). The largest Eigenvalue is called the "Principal Eigenvalue" and the Eigenvector corresponding to the principal Eigenvalue is called the "Principal Eigenvector" [3]. If all the entries in the underlying matrix used for spectral analysis are positive (≥ 0), then the principal Eigenvalue is guaranteed to be positive and all the entries in the principal Eigenvector are also positive. Hence, when computed based on matrices with positive entries, in order to still

David C. Wyld et al. (Eds): AIFZ, SIGPRO, NWCOM, CLSB, MLNLP, DTMN, BDIOT, CSITY - 2022
pp. 135-145, 2022. CS & IT - CSCP 2022 DOI: 10.5121/csit.2022.121611

satisfy the *mutually orthogonal property*, (unless all the entries in an Eigenvector are zero) at least one non-zero entry in every Eigenvector other than the Principal Eigenvector is guaranteed to be negative so that the dot product of any two Eigenvectors evaluates to zero.

In a classical work [4], Estrada et al proposed that the extent of bipartivity (in the form of a bipartivity index) among the vertices in a network could be quantified using the Eigenvalues of the adjacency matrix of the network graph. A graph is said to be bipartite if the vertices of the graph could be grouped into two disjoint sets such that the end vertices of any edge are in the two different partitions and not in the same partition. Estrada et al observed that if the underlying graph is bipartite, the vertices with positive and negative entries in the Eigenvector (hereafter referred to as the 'bipartite Eigenvector') corresponding to the smallest Eigenvalue represent the two disjoint partitions of vertices in the graph. Estrada et al also observed that if the underlying graph is not bipartite, the vertices with positive and negative entries in the bipartite Eigenvector could still be construed to form the two disjoint partitions of the vertices of the graph such that there are exist a minimal number of edges (referred to as the 'frustrated edges') between vertices in the same partition and a majority of the edges are between vertices across the two partitions.

In this paper, we conduct spectral bipartivity analysis of the scores earned by a student in theoretical and programming assignments of a Computer Science course and seek to quantify the extent of disparity in the scores earned by the student in the theoretical assignments vs. programming assignments. Some students in the Computer Science and related majors excel very well in programming-related assignments, but not equally well in the theoretical assignments (that are not programming-based) and vice-versa. We refer to this as the "Theory vs. Programming Disparity (*TPD*)". Our methodology is briefly described here (more details are in Section 2): The student score in each assignment is considered in a decimal percentage scale of 0 to 1 (i.e., each assignment is evaluated for 100% and the decimal percentage score for a student in the assignment is the percentage score divided by 100: for example, if an assignment score is 81%, the decimal percentage score is 81/100 = 0.81). We first determine the Difference Matrix (*DM*) of the student scores in the assignments wherein an entry DM_{uv} is the absolute difference in the decimal percentage scores of the two assignments u and v. We then determine the bipartite Eigenvector of the *DM* by subjecting it to spectral analysis. We identify the index entries with positive signs and negative signs, and the corresponding assignment IDs are grouped into two separate (disjoint) sets. We observe that any two assignments with similar (closer) values for the decimal percentage scores are more likely to be grouped into the same set and two assignments with appreciably different decimal percentage scores are more likely to be grouped in separate sets. That is any two vertices u and v whose DM_{uv} entry is closer to 0 are more likely to end up in the same set of vertices and vertices u and v whose DM_{uv} entry is much greater than 0 are more likely to end up in different sets of vertices. Such an observation is consistent with the observations made by Estrada et al for bipartivity analysis using an adjacency matrix A (i.e., vertices u and v whose A_{uv} entries were 0 are more likely to be in the same partition and vertices u and v whose A_{uv} entries were 1 are more likely in different partitions). We quantify the *TPD* on the basis of the Euclidean distance between the actual number of theoretical and programming assignments vs. the number of theoretical and programming assignments in the two sets of disjoint assignments identified through spectral bipartivity analysis.

The rest of the paper is organized as follows: In Section 2, we present our proposed methodology to quantify the *TPD* metric using a running example. Section 3 evaluates the effectiveness of the proposed *TPD* approach with two of the well-known metrics (Bipartivity index: *BPI* [4] and Hausdorff Distance: *HD* [5]) that exist in the literature to study the effectiveness of partitioning of a data set to two clusters. Section 3 also highlights the uniqueness of the proposed *TPD* approach. Section 4 reviews related work and Section 5 concludes the paper. Throughout the paper, the terms 'set' and 'partition', 'network' and 'graph', 'edge' and 'link' are used interchangeably. They mean the same.

2. Spectral Bipartivity Analysis to Quantify Theory vs. Programming Disparity

Let *P* and *T* be respectively the set of scores (represented in decimal percentage format) earned by a student in programming and theoretical assignments. The indexes for the assignments in the set *P* range from 0 to |*P*| - 1, where |*P*| is the cardinality of the set *P* (i.e., the number of programming assignments). The indexes for the assignments in the set *T* range from |*P*| to |*P*| + |*T*| - 1, where |*T*| is the cardinality of the set *T* (i.e., the number of theoretical assignments). Let S be the union of the two sets *P* and *T*. That is, the set S comprises of the scores earned by the student in the programming assignments, followed by the theoretical assignments. The indexes for the assignments in *S* are the same as their indexes in the sets *P* or *T*, whichever they come from. Let *DM* be a symmetric/square matrix whose dimensions correspond to the cardinality of the set *S*. An entry DM_{ij} for row index *i* and column index *j* represents the absolute difference in the decimal percentage scores for the i^{th} and j^{th} element/assignment in the set *S*. Figure 1 presents the sets *P*, *T* and *S* (of size 8, 4 and 12 respectively) as well as the *DM* matrix (of dimensions 12 x 12) for a sample data set that is used as a running example to explain the proposed methodology in this section. The indexes for the assignments in the sets *P* and *T* range from 0...7 and 8...11 respectively; the indexes of these assignments are retained in the set *S* that is the amalgamation of the assignments in the sets *P* and *T* (in the same order). Likewise, the indexes in *DM* correspond to the indexes in the set *S*.

Index 0 1 2 3 4 5 6 7
P = {1.00, 0.94, 0.90, 1.00, 1.00, 0.60, 1.00, 0.49}

Index 8 9 10 11
T = {0.79, 0.72, 0.61, 0.15}

Index 0 1 2 3 4 5 6 7 8 9 10 11
S = P ⋃ T = {1.00, 0.94, 0.90, 1.00, 1.00, 0.60, 1.00, 0.49, 0.79, 0.72, 0.61, 0.15}

Difference Matrix (DM)

		Index v											
Index u		0	1	2	3	4	5	6	7	8	9	10	11
		1.00	0.94	0.90	1.00	1.00	0.60	1.00	0.49	0.79	0.72	0.61	0.15
0	1.00	0.00	0.06	0.10	0.00	0.00	0.40	0.00	0.51	0.21	0.28	0.39	0.85
1	0.94	0.06	0.00	0.04	0.06	0.06	0.34	0.06	0.45	0.15	0.22	0.33	0.79
2	0.90	0.10	0.04	0.00	0.10	0.10	0.30	0.10	0.41	0.11	0.18	0.29	0.75
3	1.00	0.00	0.06	0.10	0.00	0.00	0.40	0.00	0.51	0.21	0.28	0.39	0.85
4	1.00	0.00	0.06	0.10	0.00	0.00	0.40	0.00	0.51	0.21	0.28	0.39	0.85
5	0.60	0.40	0.34	0.30	0.40	0.40	0.00	0.40	0.11	0.19	0.12	0.01	0.45
6	1.00	0.00	0.06	0.10	0.00	0.00	0.40	0.00	0.51	0.21	0.28	0.39	0.85
7	0.49	0.51	0.45	0.41	0.51	0.51	0.11	0.51	0.00	0.30	0.23	0.12	0.34
8	0.79	0.21	0.15	0.11	0.21	0.21	0.19	0.21	0.30	0.00	0.07	0.18	0.64
9	0.72	0.28	0.22	0.18	0.28	0.28	0.12	0.28	0.23	0.07	0.00	0.11	0.57
10	0.61	0.39	0.33	0.29	0.39	0.39	0.01	0.39	0.12	0.18	0.11	0.00	0.46
11	0.15	0.85	0.79	0.75	0.85	0.85	0.45	0.85	0.34	0.64	0.57	0.46	0.00

Figure 1. Sample Data set to Illustrate the Spectral Bipartivity-based Analysis for Theory vs. Programming Disparity

Figure 2 presents the 12 Eigenvalues and the entries of the Bipartite Eigenvector, which is the Eigenvector corresponding to the smallest Eigenvalue of -2.2285, of the Difference Matrix (*DM*). The indexes (colored in blue) whose entries are positive (≥ 0) are grouped to partition (set) *X* and the indexes (colored in red) whose entries are negative (< 0) are grouped to partition (set) *Y*. These are the two disjoint partitions of the assignments in the set *S* predicted based on spectral bipartivity analysis. In Figure 2, we also display the submatrices of the Difference Matrix that show the difference in the decimal percentage scores for any two assignments within the sets *X* and *Y* as well as between the two sets *X* and *Y*. We notice the entries in the submatrices corresponding to the differences in the assignment scores within the sets *X* or *Y* are relatively much smaller (closer to 0) compared to the entries in the submatrix corresponding to the differences in the assignment scores between an assignment in set *X* and an assignment in set *Y*.

Index / Assignment, i	Bipartite Eigenvector
0	0.3199
1	0.2681
2	0.2240
3	0.3199
4	0.3199
5	-0.1779
6	0.3199
7	-0.2938
8	0.0805
9	-0.0159
10	-0.1657
11	-0.5625

Index u \ Index v		0	1	2	3	4	6	8
		1.00	0.94	0.90	1.00	1.00	1.00	0.79
0	1.00	0.00	0.06	0.10	0.00	0.00	0.00	0.21
1	0.94	0.06	0.00	0.04	0.06	0.06	0.06	0.15
2	0.90	0.10	0.04	0.00	0.10	0.10	0.10	0.11
3	1.00	0.00	0.06	0.10	0.00	0.00	0.00	0.21
4	1.00	0.00	0.06	0.10	0.00	0.00	0.00	0.21
6	1.00	0.00	0.06	0.10	0.00	0.00	0.00	0.21
8	0.79	0.21	0.15	0.11	0.21	0.21	0.21	0.00

Difference in the Scores between any Two Assignments within Partition X

Partition X	Partition Y
0	
1	5
2	7
3	9
4	10
6	11
8	

Index u \ Index v		5	7	9	10	11
		0.60	0.49	0.72	0.61	0.15
5	0.60	0.00	0.11	0.12	0.01	0.45
7	0.49	0.11	0.00	0.23	0.12	0.34
9	0.72	0.12	0.23	0.00	0.11	0.57
10	0.61	0.01	0.12	0.11	0.00	0.46
11	0.15	0.45	0.34	0.57	0.46	0.00

Difference in the Scores between any Two Assignments within Partition Y

Index u \ Index v		5	7	9	10	11
		0.60	0.49	0.72	0.61	0.15
0	1.00	0.40	0.51	0.28	0.39	0.85
1	0.94	0.34	0.45	0.22	0.33	0.79
2	0.90	0.30	0.41	0.18	0.29	0.75
3	1.00	0.40	0.51	0.28	0.39	0.85
4	1.00	0.40	0.51	0.28	0.39	0.85
6	1.00	0.40	0.51	0.28	0.39	0.85
8	0.79	0.19	0.30	0.07	0.18	0.64

Difference in the Scores between an Assignment in Partition X and an Assignment in Partition Y

Figure 2. Bipartite Eigenvector for the Difference Matrix of Figure 1 as well as the Submatrices Representing the Difference Values between Assignments within the two Partitions and across the Two Partitions

We now seek to quantify the extent to which the grouping of the assignments in sets *X* and *Y* are closer to the grouping of the programming and theoretical assignments in the sets *T* and *P*. In order for our hypothesis (that there exists a disparity in the scores earned by the students in the programming vs. theoretical assignments) to be true, we would like the two sets *X* and *Y* to be the same as the two sets *T* and *P* (or *P* and *T*); that is, we would prefer the set *X* to be all theoretical assignments and the set *Y* to be all programming assignments or the set *X* to be all programming assignments and the set *Y* to be all theoretical assignments. In this pursuit, we first determine the number of theoretical assignments and the number of programming assignments in each of the sets *X* and *Y* and let these be indicated using symbols $|X_T|$, $|X_P|$, $|Y_T|$ and $|Y_P|$. We then determine the Euclidean distance between the tuples ($|X_T|$, $|Y_P|$) and ($|T|$, $|P|$) as well as between the tuples ($|Y_T|$, $|X_P|$) and ($|T|$, $|P|$) and refer to the minimum of these two Euclidean distances as the Theory vs. Programming Tuple Proximity (*TPTP*) distance for the given data set.

The maximum value for the *TPTP* distance is $\sqrt{|T|^2 + |P|^2}$ and we will incur it when all the four values $|X_T|$, $|X_P|$, $|Y_T|$ and $|Y_P|$ are zero each (i.e., the assignment IDs in the partitions *X* and *Y* identified through spectral bipartivity analysis have no overlap with the assignment IDs in the partitions *P* and *T*). Such a scenario occurs when there is minimal or no disparity among the scores in the programming vs. theoretical assignments and the distribution of the assignment IDs in the partitions *X* and *Y* are random. On the other hand, if there is maximum disparity, the assignment IDs in partitions *X* and *Y* will overlap with those of *P* and *T*, and either $|X_P| \approx |P|$ and $|Y_T| \approx |T|$ or $|Y_P| \approx |P|$ and $|X_T| \approx |T|$. This would make the *TPTP* distance much smaller than the maximum value of $\sqrt{|T|^2 + |P|^2}$. Considering the above interpretation of the *TPTP* distance, we formulate the *TPD* (Theory vs. Programming Disparity) metric as follows:

$$TPD = 1 - \frac{TPTP}{\sqrt{|T|^2 + |P|^2}}$$

If there is maximum disparity, then the *TPTP* distance will be either closer to 0 or the ratio *TPTP* / $\sqrt{|T|^2 + |P|^2}$ be closer to 0, making the *TPD* metric score to be closer to 1. On the other hand, if there is no disparity, the *TPTP* distance will be closer to the maximum value of $\sqrt{|T|^2 + |P|^2}$ and as a result the ratio *TPTP* / $\sqrt{|T|^2 + |P|^2}$ be closer to 1, making the *TPD* metric score to be closer to 0.

Figure 3 presents the computation of the $|X_T|$, $|X_P|$, $|Y_T|$ and $|Y_P|$ numbers for the running example of Figures 1-2 and the computation of the *TPD* metric for the sample data set. The *TPD* metric value for this data set is 0.75, indicating that there is an appreciable disparity in the theoretical vs. programming scores in this data set. Such a conclusion could also be justified by visually looking at the proximity of the tuple ($|Y_T| = 3$, $|X_P| = 6$) corresponding to the *TPTP* distance to the tuple ($|T| = 4$, $|P| = 8$) in Figure 3 as well as the raw data set values {0.79, 0.72, 0.61, 0.15} and {1.00, 0.94, 0.90, 1.00, 1.00, 0.60, 1.00, 0.49} for the sets *T* and *P* respectively.

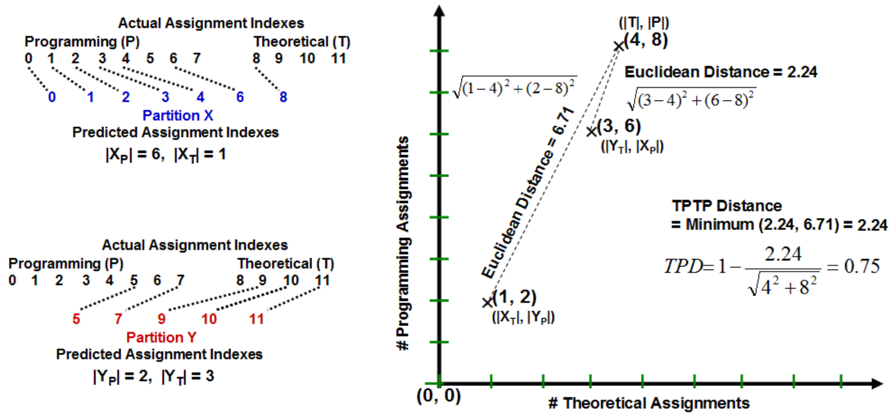

Figure 3. Computation of the *TPD* Metric for the Sample Data Set of Figure 1

3. Evaluation of the Proposed Approach

In this section, we apply the proposed spectral bipartivity analysis-based approach to quantify the theory vs. programming disparity per student in the CSC 228 Data Structures and Algorithms course taught in Spring 2020 at Jackson State University, MS, USA. In addition to the *TPD* metric, we consider two other metrics that appear to be potentially applicable to quantify the extent of disparity in a data set with respect to two different categories (in this case, theoretical vs. programming assignments). These are:

(i) Bipartivity Index (*BPI*): The *BPI* was originally proposed by Estrada et al [4] to quantify the extent of bipartivity between the two partitions of vertices identified using the Eigenvector (referred to as the Bipartite Eigenvector) corresponding to the smallest Eigenvalue. The input matrix for Estrada et al's spectral bipartivity analysis is a 0-1 adjacency matrix. If the underlying graph is not bipartite, the two partitions of vertices identified using the Bipartite Eigenvector have as few edges as possible between vertices within the same partition and a majority of the edges are between vertices across the two partitions. If the underlying graph is indeed bipartite, the two partitions of vertices identified using the Bipartite Eigenvector will have no edges between vertices within the same partition and all the edges in the graph will be between vertices across the two partitions. The *BPI* of a graph of *'n'* vertices is computed using the following formulation based on the 'n' Eigenvalues (λ) of its 0-1 adjacency matrix. If the underlying graph is bipartite, the sum of the sinh values of the '*n*' Eigenvalues will be zero and the *BPI* of the graph will be 1.0. If the underlying graph is not bipartite, then the sum of the sinh values of the '*n*' Eigenvalues will be greater than 0, and the *BPI* will be less than 1.0.

$$BPI = \frac{\sum_{i=0}^{n-1} \cosh(\lambda_i)}{\sum_{i=0}^{n-1} \cosh(\lambda_i) + \sum_{i=0}^{n-1} \sinh(\lambda_i)}$$

In this section, we will explore how well the Eigenvalues of the Difference Matrix (*DM*) of an assignment score data set capture the Theory vs. Programming Disparity such that the *BPI* values are closer to 1.0 for data sets with larger values for the *TPD* metric and vice-versa. Figure 4 displays the 12 Eigenvalues of the running example assignment scores data set of Section 2: the sums of the cosh and sinh functions of the Eigenvalues are 33.4068 and 12.2509 respectively, leading to a *BPI* of 33.4068 / (33.4068 + 12.2509) = 0.73.

Index, j	Eigenvalue, λj	cosh(λj)	sinh(λj)
0	-2.2285	4.6970	-4.5893
1	-0.7947	1.3327	-0.8810
2	-0.2514	1.0318	-0.2541
3	-0.1301	1.0085	-0.1305
4	-0.1039	1.0054	-0.1041
5	-0.0518	1.0013	-0.0518
6	-0.0309	1.0005	-0.0309
7	-0.0095	1.0000	-0.0095
8	0.0000	1.0000	0.0000
9	0.0000	1.0000	0.0000
10	0.0000	1.0000	0.0000
11	3.6009	18.3295	18.3022
	Sum	33.4068	12.2509

$$BPI(T,P) = \frac{33.4068}{33.4068 + 12.2509} = 0.73$$

Set T

Set P		8	9	10	11	Min
		0.79	0.72	0.61	0.15	
0	1.00	0.21	0.28	0.39	0.85	0.21
1	0.94	0.15	0.22	0.33	0.79	0.15
2	0.90	0.11	0.18	0.29	0.75	0.11
3	1.00	0.21	0.28	0.39	0.85	0.21
4	1.00	0.21	0.28	0.39	0.85	0.21
5	0.60	0.19	0.12	0.01	0.45	0.01
6	1.00	0.21	0.28	0.39	0.85	0.21
7	0.49	0.30	0.23	0.12	0.34	0.12
	Min	0.11	0.12	0.01	0.34	

Hasudorff Distance ***HD*(*T*, *P*)**

Figure 4. Computation of the Bipartivity Index (*BPI*) and Hausdorff Distance (*HD*) Metric Values for the Sample Data Set of Figure 1

(ii) Hausdorff Distance: The Hausdorff Distance (*HD*) metric [5] has been traditionally used to quantify how far are two data sets in a particular metric space. In the context of quantifying the Theoretical vs. Programming Disparity, we propose to compute the Hausdorff Distance (see below for the formulation) between the decimal percentage scores (referred to as data points) in the sets of theoretical assignments (*T*) vs. programming assignments (*P*). For every data point in data set *T* (and likewise, *P*), we determine the closest distance (in our case, the absolute difference) to a data point in the other data set *P* (*T*). The Hausdorff Distance for the *P* vs. *T* scores for a student data set is the maximum of the closest distances determined as mentioned above. The Hausdorff Distance will be thus smaller if every data point in one data set is closer to some data point in the other data set.

$$HD(T,P) = Max\left\{ \underset{i \in T}{Min}\left[\underset{j \in P}{\left| T(i) - P(j) \right|} \right], \underset{j \in P}{Min}\left[\underset{i \in T}{\left| P(j) - T(i) \right|} \right] \right\}$$

For *T* vs. *P* data sets that exhibit larger disparity, we expect several data points (assignment scores) in one data set to be appreciably different from those of the other data set. However, even if there exists one outlier data point in either of the two data sets (that is farther away from the other data points in the two data sets), the Hausdorff Distance metric has the vulnerability to get larger and not be an accurate reflection of the closeness or the extent of disparity among the assignment scores in the two sets *T* and *P*. Figure 4 displays the computation of the Hausdorff Distance metric values for the running example *P* vs. *T* data set of Section 2: we observe the presence of a lower theoretical assignment score (0.15 corresponding to index 11 in set *T*) contributes to a relatively larger *HD*(*T*, *P*) value of 0.34 (note that the next largest value among all the minimum values computed across the two data sets is 0.21).

Figure 5 presents a real-time data set comprising of the assignment scores (13 programming assignments and 4 theoretical assignments) for 17 students of the CSC 228 Data Structures and Algorithms course taught in Spring 2020 at Jackson State University, MS. Figure 6 presents the values for the *TPD*, *BPI* and *HD* metrics (also visually compared using a heat map [6]) obtained for the data set of 17 students as well as plots the *TPD* vs. *BPI* and *TPD* vs. *HD* distributions. In the heat map shown in Figure 6, the red, yellow/orange and green colors are respectively indicators of high, moderate and lower values for the *TPD*, *BPI* and *HD* metrics (all of which can be represented in a scale of 0 to 1). We observe the *BPI* and *HD* metrics to have a tendency of over rating (too much red cells for the *BPI* metric) and under rating (too much green cells for the *HD* metric) the theoretical vs. programming disparity. On the other hand, the values for the *TPD* metric are within a moderate range of 0.54 to 0.77 that is sufficient enough to distinguish students with respect to the theoretical vs. programming disparity.

	Programming Assignments													Theoretical Assignments			
Student / Index	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	0.00	0.00	0.00	0.00	0.80	0.65	0.00	0.00	0.30	0.00	1.00	0.00	0.25	0.00	0.25	0.00	0.56
1	1.00	1.00	0.80	0.95	1.00	1.00	1.00	1.00	1.00	0.98	1.00	0.92	1.00	0.96	1.00	0.96	1.00
2	0.00	0.00	0.00	1.00	1.00	1.00	0.00	0.00	0.00	0.00	1.00	0.80	0.50	0.75	0.00	0.82	0.78
3	0.25	0.30	0.25	1.00	0.75	0.30	0.25	0.35	0.55	0.00	0.00	0.00	0.00	0.30	0.00	0.00	0.00
4	0.00	0.00	0.00	0.45	0.35	0.60	0.00	0.90	0.70	0.00	0.25	0.00	0.00	0.00	0.25	0.00	0.71
5	0.00	0.20	1.00	0.80	0.30	0.65	0.00	0.00	0.00	0.90	0.85	0.50	0.00	0.00	0.00	0.48	0.59
6	0.75	0.00	0.60	0.82	0.75	0.75	0.78	0.90	0.55	0.00	1.00	1.00	0.30	0.93	0.35	0.43	0.90
7	0.00	0.85	0.60	0.85	1.00	0.90	1.00	0.90	0.85	0.65	1.00	1.00	0.40	0.62	0.50	0.94	0.98
8	0.00	0.00	0.40	0.80	0.00	0.10	0.10	0.25	0.70	0.10	0.30	0.50	0.30	0.28	0.20	0.33	0.85
9	0.60	0.00	0.15	0.85	0.50	0.50	1.00	0.90	0.30	0.10	0.85	1.00	0.50	0.76	0.28	0.42	0.65
10	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.30	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
11	0.00	0.00	0.00	0.80	0.00	0.00	0.75	0.90	0.50	0.60	0.00	1.00	0.00	0.00	0.00	0.41	0.51
12	0.00	0.00	0.00	0.00	0.00	0.00	0.75	0.00	0.00	0.00	0.00	0.00	0.00	0.05	0.00	0.00	0.00
13	0.00	0.00	0.00	0.40	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.54	0.00	0.00	0.04
14	0.25	0.15	0.00	0.20	0.00	0.10	0.00	0.90	0.30	0.10	0.00	1.00	0.00	0.25	0.30	0.19	0.29
15	1.00	0.50	0.60	0.90	1.00	1.00	1.00	0.90	0.85	0.85	1.00	1.00	0.55	0.52	0.70	1.00	0.97
16	0.25	1.00	1.00	0.95	1.00	0.78	0.60	1.00	1.00	0.65	1.00	1.00	0.75	0.98	1.00	0.82	0.74

Figure 5. Real-time Data Set used to Apply and Evaluate the Proposed Approach

The tendency of the *BPI* and *HD* metrics to respectively over rate and under rate the theoretical vs. programming disparity can be observed in the case of students 10, 12 and 13, who all have just one or two submissions in one of the two categories (programming or theoretical). For such students, the majority of the entries are 0.0. The *BPI* metric tends to cluster the assignment scores for each of these students to two sets: a set of assignments for which submission has been made and a set for which no submission has been made, with the edge weights capturing the difference between these two sets. On the other hand, the *HD* metric tends to give more weight to the minimal difference in the scores between any two assignments. Both the *BPI* and *HD* metrics tend to treat all the 17 assignments as one set (i.e., one dimension) and tends to partition to two clusters (*BPI*) or find the minimum score between any two assignments; whereas, the *TPD* approach considers the problem in a two-dimensional perspective of theoretical vs. programming assignments; the clustering in the two-dimensional space gives leverage to consider four possible combinations for two clusters: the number of theoretical assignments in the sets *X* and *Y* as well as the number of programming assignments in the sets *X* and *Y* identified using spectral analysis as well as makes use of the actual number of theoretical and programming assignments to compute the Euclidean distances as formulated in Section 2. Due to the different approaches taken, we observe only a weak-moderate correlation between the *TPD* vs. *BPI* scores and the *TPD* vs. *HD* scores for the data set analyzed in Figures 5 and 6.

Student	TPD	BPI	HD
0	0.63	0.62	0.44
1	0.63	0.99	0.16
2	0.55	0.64	0.25
3	0.58	0.58	0.7
4	0.57	0.63	0.2
5	0.54	0.57	0.41
6	0.67	0.56	0.35
7	0.67	0.59	0.5
8	0.63	0.59	0.2
9	0.54	0.55	0.28
10	0.69	1	0.3
11	0.54	0.61	0.49
12	0.69	0.98	0.7
13	0.77	0.96	0.14
14	0.67	0.59	0.7
15	0.73	0.81	0.12
16	0.57	0.73	0.49

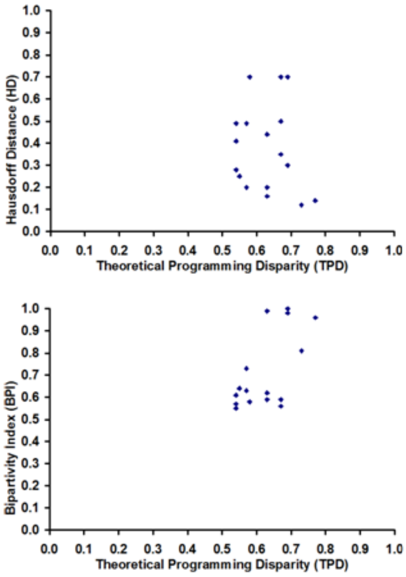

Figure 6. Comparison the Quantitative Assessments of Theory vs. Programming Disparity using the Proposed Theoretical Programming Disparity (*TPD*) Metric vs. the Bipartivity Index (*BPI*) and Hausdorff Distance (*HD*) Metrics

4. Related Work

To the best of our knowledge, we have not come across any work in the literature that focuses on assessing and quantifying the disparity found between two different categories of assignments on a per-student basis. Disparity studies in academic settings have been so far mainly focused on gender [12] and race [13] as well as on the class, as a whole (e.g., [9-11]), and not on a per-student basis. The closest work we have come across related to our topic is the work of [14] wherein the authors apply principles from phenomenography [15] and variation theory [16] to explore the practices that are needed to bridge the gap in learning Computer Science theory and learning Computer Science practice (programming). But, there are no efforts to quantify the extent of the gap (or the disparity), as is done in our paper.

Below is a review of the works that we came across in the literature that focus on studies conducted to assess the contributing factors to the success or hardship for students majoring in Computer Science and the programming component of it. In [17], the authors did a survey to find out that students think Computer Science-ability is something both innate as well as extensible through effort. In [18], the authors surveyed Computer Science (CS) student performance data for 10 years and conclude that a successful CS student needs to be strong both in critical thinking skills and the core CS skills. They observed the critical thinking skills for a CS student typically come from the Math and Physics courses and concluded that these courses need to be enforced as pre-requisites for CS courses early on in the curriculum instead of being taken along with CS courses. However, no analysis has been reported in [18] regarding the skills that influence the theory vs. programming disparity found among CS students. In [19], the authors report there is no statistically significant influence of the assessment mode (programming in a computer vs. writing a program in pen by hand) on the performance of students in a programming course. In [20], the authors observed that novice programmers tend to program using problem-solving skills obtained from domains familiar to them. In [21], the authors used reflective essays-based Attribution Theory to elucidate the internal and external causes that influence the performance of students in their first programming course.

The following works focus on analyzing the impact of one examination format on another. In [9], the authors build a model to predict the performance of students on the basis of examination formats: whether or not the performance of students in practical examinations can be predicted using their performance in the standardized examinations? The answer reported in [9] is No, as the two examination formats are observed to test different skill sets. Likewise, Haberyan [10] found no correlation between performance in weekly quizzes and examinations among students majoring in Biology. However, in [11], the authors observed that psychology undergraduates performed well in the examinations when they were also given weekly reading assignment-based quizzes throughout the course.

5. CONCLUSIONS

The high-level contribution of this paper is a spectral analysis-based approach to quantify the disparity (the proposed metric is referred to as Theoretical vs. Programming Disparity: *TPD* metric) in the scores earned by students in two categories of assignments: theoretical and programming. The uniqueness of the proposed approach is that *TPD* quantifies the disparity on a per-student basis; the typical approach in the academic community so far is to use quantitative metrics that capture the disparity for an entire class as a whole [8]. Also, traditionally for such problems, the correlation measures [3, 8, 9] are used to quantify the extent of influence on one category of assignments over another. But, to use correlation measures, we need to have the same number of assignments under both the categories. With our proposed approach, there can be a different number of assignments for the two categories. The pre-requisites are just the need to input the classification of the assignments ids (as either theoretical or programming) and the actual number of assignments under the two categories. We have demonstrated the characteristics and the uniqueness of the *TPD* metric and the spectral analysis through an exhaustive evaluation with a real-time dataset as well as through comparison with quantitative metrics that appear to be compelling enough to be suitable to capture the disparity in the student scores in the assignment categories. The disparity problem on a per-student basis among two data sets of uneven size, especially in an academic setting, has not been so far considered in the literature for quantitative evaluation; we expect the proposed *TPD* metric and its computation approach would be valuable to both academicians and researchers.

ACKNOWLEDGEMENTS

The author would like to thank everyone, just everyone!

REFERENCES

[1] Estrada, E, (2010) "Structural Patterns in Complex Networks through Spectral Analysis," *Proceedings of the 2010 Joint IAPR International Conference on Structural, Syntactic, and Statistical Pattern Recognition*, pp. 45-59. Springer-Verlag, Cesme Izmir, Turkey.

[2] Sarkar, C., and Jalan, (2018) "Spectral Properties of Complex Networks," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 28, no. 10, 102101.
[3] Strang, G (2019) *Linear Algebra and Learning from Data*, 1st edition, Wellesley-Cambridge Press, Wellesley, MA, USA.
[4] Ernada, E., and Rodriguez-Velazquez, J. A (2005) "Spectral Measures of Bipartivity in Complex Network," *Physical Review* E, vol. 72, no. 4, 2, 046105.
[5] Birsan, T., and Tiba, D (2006) "One Hundred Years since the Introduction of the Set Distance by Dimitrie Pompeiu," *Proceedings of the IFIP Conference on System Modeling and Optimization*, vol. 199, pp. 35-39. Springer, Turin, Italy.
[6] Wilkinson, L., and Friendly, M (2009) "The History of the Cluster Heat Map," *The American Statistician*, vol. 63, no. 2, pp. 179-184.
[7] Godsil, C., and Royle, G. F (2013) *Algebraic Graph Theory*, 1st edition, Springer, Berlin, Germany.
[8] Caven, M (2019) "Quantification, Inequality, and the Contestation of School Closures in Philadelphia," *Sociology of Education*, vol. 92, no. 1, pp. 21-40.
[9] Davison, C. B., and Dustova, G (2017) "A Quantitative Assessment of Student Performance and Examination Format," *Journal of Instructional Pedagogies*, vol. 18, pp. 1-10.
[10] Haberyan, K. (2003) "Do Weekly Quizzes Improve Student Performance on General Biology Exams?" *The American Biology Teacher*, vol. 65, pp. 110-114.
[11] Johnson, B. C., and Kiviniemi, M. T (2009) "The Effect of Online Chapter Quizzes on Exam Performance in an Undergraduate Social Psychology Course," *Teaching of Psychology*, vol. 36, no. 1, pp. 33-37.
[12] Master, A., Meltzoff, A. N., and Cheryan, S (2021) "Gender Stereotypes about Interests Start Early and Cause Gender Disparities in Computer Science and Engineering," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 118, no. 48, e2100030118.
[13] Kozlowski, D., Lariviere, V., Sugimoto, C. R., and Monroe-White, T (2002) "Intersectional Inequalities in Science," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 119, no. 2, e2113067119.
[14] Thune, M., and Eckerdal, A (2019) "Analysis of Students' Learning of Computer Programming in a Computer Laboratory Context," *European Journal of Engineering Education*, vol. 44, no. 5, pp. 769-786.
[15] Farton, M (1986) "Phenomenography - A Research Approach Investigating Different Understandings of Reality," *Journal of Thought*, vol. 21, no. 2, pp. 28-49.
[16] Bussey, T. J., Orgill, M., and Crippen, K. J (2013) "Variation Theory: A Theory of Learning and a Useful Theoretical Framework for Chemical Education Research," *Chemical Education Research Practice*, vol. 14, pp. 9-22.
[17] Lewis, C. M., Yasuhara, K., and Anderson, R. E (2011). "Deciding to Major in Computer Science: A Grounded Theory of Students' Self-Assessment of Ability," *Proceedings of the 7th International Workshop on Computing Education Research*, pp. 3-10, ACM, Providence, RI, USA.
[18] Yang, H., Olson, T. W., and Puder, A (2021) "Analyzing Computer Science Students' Performance Data to Identify Impactful Curricular Changes," *Proceedings of the IEEE Frontiers in Education Conference*, pp. 1-9, IEEE, Lincoln, NE, USA.
[19] Oqvist, M., and Nouri, J (2018) "Coding by Hand or on the Computer? Evaluating the Effect of Assessment Mode on Performance of Students Learning Programming," *Journal of Computers in Education*, vol. 5, pp. 199-219.
[20] Wellons, J., and Johnson, J (2011) "A Grounded Theory Analysis of Introductory Computer Science Pedagogy," *Systemics, Cybernetics and Informatics*, vol. 9, no. 6, pp. 9-14.
[21] Vivian, R., Falkner, K., and Falkner, N (2013) "Computer Science Students' Casual Attributions for Successful and Unsuccessful Outcomes in Programming Assignments," *Proceedings of the 13th Koli Calling International Conference on Computing Education Research*, pp. 125-134, ACM, Koli, Finland.

AUTHOR

Natarajan Meghanathan is a tenured Full Professor of Computer Science at Jackson State University, MS, USA. He graduated with a PhD in Computer Science from The University of Texas at Dallas in 2005. His primary areas of research interests are Network Science, Machine Learning and Cyber Security. He has authored more than 175 peer-reviewed research publications in these areas.