MIC Check: A Correlation Tactic for ESE Data

Daryl Posnett

Department of Computer Science
University of California, Davis
Davis, CA
dpposnett@ucdavis.edu

Prem Devanbu

Department of Computer Science
University of California, Davis
Davis, CA
devanbu@ucdavis.edu

Vladimir Filkov

Department of Computer Science
University of California, Davis
Davis, CA
filkov@cs.ucdavis.edu

Abstract—**Empirical software engineering researchers are concerned with understanding the relationships between outcomes of interest, *e.g.* defects, and process and product measures. The use of correlations to uncover strong relationships is a natural precursor to multivariate modeling. Unfortunately, correlation coefficients can be difficult and/or misleading to interpret. For example, a strong correlation occurs between variables that stand in a polynomial relationship; this may lead one mistakenly, and eventually misleadingly, to model a polynomially related variable in a linear regression. Likewise, a non-monotonic functional, or even non-functional relationship might be entirely missed by a correlation coefficient. Outliers can influence standard correlation measures, tied values can unduly influence even robust non-parametric rank correlation,measures, and smaller sample sizes can cause instability in correlation measures. A new bivariate measure of association, Maximal Information Coefficient (MIC) [1], promises to simultaneously discover if two variables have: *a)* any association, *b)* a functional relationship, and *c)* a non-linear relationship. The MIC is a very useful complement to standard and rank correlation measures. It separately characterizes the *existence* of a relationship and *its precise nature*; thus, it enables more informed choices in modeling non-functional and non-linear relationships, *and* a more nuanced indicator of potential problems with the values reported by standard and rank correlation measures. We illustrate the use of MIC using a variety of software engineering metrics. We study and explain the distributional properties of MIC and related measures in software engineering data, and illustrate the value of these measures for the empirical software engineering researcher.**

I. Introduction

Empirical software engineering seeks to uncover relationships between measurable properties of source code and the development process, *i.e.* product and process metrics, and outcomes of interest such as source code quality or developer productivity. Ideally, these identified relationships drive new insights that may lead to improved productivity, quality, and lower cost. Finding these relationships can be challenging and researchers have used a variety of methods.

During the exploratory phase of research, there might be many variables of potential interest; it is often the case that the specific type of relationship, *i.e.* its functional or non-functional form, between them is unknown. A measure that can detect the *presence* of strong relationships, independent of the *type* of relationship, can help a researcher focus their explorations. One important tool often used by researchers is the *correlation coefficient*, which measures the strength of the relationship between two variables of interest.

The correlation coefficient is not uncontroversial; there are many types of relationships that it may fail to identify, and, in some cases, it may incorrectly indicate a relationship where nothing significant exists.

Reshef *et al.* [1]. recently introduced an alternative measure of association, the *maximal information coefficient*, or MIC, that builds on half a century of work on *entropy* and *mutual information* of random variables. Measures of association based on mutual information are non-parametric, *viz.*, they do not rely on distributional assumptions of the measured data; consequently they are able to identify a much broader class of relationships than can be accurately identified with the correlation coefficient. Information theoretic measures of association are not, however, without limitation and MIC addresses some computational and interpretation issues that have hampered adoption of these measures to date.

This measure is potentially interesting to software engineering researchers. In addition to indicating non-linear relationships, Reshef *et al.* also introduce new related measures of the degree of non-linearity, monotonicity, asymmetry, and functionality present in data. They refer to the collection of measures provided by their tool[1] as the MINE measures. Our interest here is the application of these measures to assist empirical software engineering research.

This paper will explore the applicability of MIC to software engineering data. Our motivation here is whether MIC and its associated measures are applicable to empirical software engineering studies.

Motivation: ***To what extent are the* MINE *measures of association, non-linearity, monotonicity, asymmetry, and non-functionality useful in uncovering and exploring relationships in software engineering?***

In addition to evaluating the value of the MINE measures with respect to ESE data, we are also interested in developing intuition for the measures and understanding how

[1] http://www.exploredata.net/Downloads

their values can be understood and interpreted with actual software engineering data.

Research Question 1: What are the distributional properties of the MINE measures of association, non-linearity, monotonicity, asymmetry, and non-functionality in software engineering data and how should their values be interpreted?

Even though MIC is a non-parametric measure we expect that some attributes of the data may impact the stability of the MINE measures.[2] Reshef has specified that MIC should be interpreted as r^2, *i.e. coefficient of determination for a linear regression*[3], but this property was evaluated through simulation [1]. How reasonable is this for ESE data and what about the other MINE measures? ESE data, in particular, can sometimes suffer from restricted sample sizes. Our next question addresses that.

Research Question 2: Are the MINE measures stable given the sample size limitations of ESE data?

Next, we ask whether in our data, the MINE measures lead us to observations of relationships that are in general not readily found by correlation measures. This can help generate new hypotheses.

Research Question 3: How do the MINE measures help the empirical software engineering researcher identify interesting new relationships?

II. Measures of Dependence Between Two Variables

Before we discuss the MINE measures in more detail, we motivate their use by introducing other widely-used approaches to study bivariate relationships. Pearson's coefficient (based on ideas from Galton [2]) is the most ubiquitous.

1) Pearson Correlation: The Pearson population correlation coefficient of two variables X and Y is defined as the ratio of the covariance of X and Y and the product of their standard deviation $\rho_{X,Y} = \frac{cov(X,Y)}{\sigma_X \sigma_Y}$. In practice we use the sample variant r which is computed as follows: *(Where s_x, s_y are the sample standard deviations.)*

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{(n-1)s_x s_y}$$

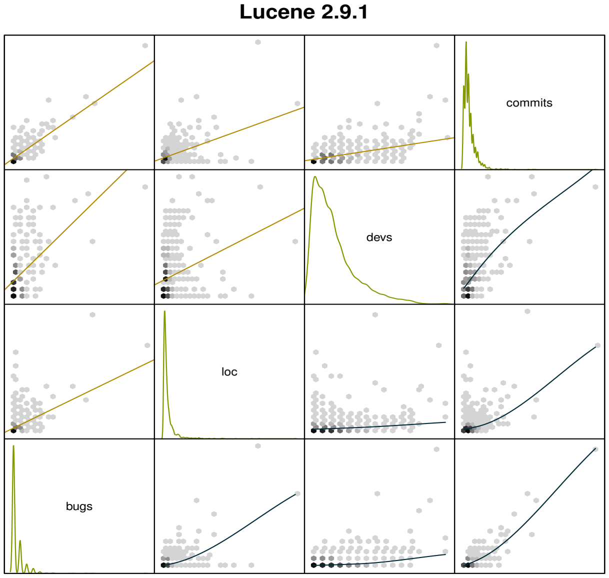

Figure 1: Hexbin scatter plot of a subset of the variables from Lucene 2.9.1. Hexbin plots aggreate points and assign color to density for easier interpretion of noisy data or data with repeated values. A darker color means higher density. Lines in upper diagonal are regression lines, lower diagonal are loess locally fitted lines. The diagonal shows density plot of univariate variable. (*bugs = count of defects in file, loc = lines of code, devs = number of developers who have worked on the file, commits = number of distinct commits to the file*)

This value is undefined when the variance of either variable is zero. It is robust under linear transformations of the data, which is assumed to be on an interval scale with finite variances and finite covariances. If the data is drawn from a population with a bivariate normal distribution then r provides a complete picture of the association [3]. When the data is non-linear or non-functional the correlation coefficient can be misleading. Non-linear relationships increase distances of the total sum of squares from the regression line induced from the assumed linear relationship. Alternatives to Pearson's measure, discussed below, relax the distributional requirements, are better able to handle non-linear relationships, and can work even with ordinal-scale data.

2) Rank Correlation: Rank correlation measures are less sensitive to the nature of the relationships between data, and capture the extent to which one variable increases (or decreases) as another increases. Spearman rank correlation is a method that is often used when data does not meet requirements of Pearson's r; it is used extensively in ESE research. It is defined as the Pearson correlation coefficient on the ranked data where tied values are assigned a rank equal to the mean of their positions in the ascending order of their values [3]. Spearman's rho is far less sensitive to nonlinearities; it doesn't, however, yields a reasonable measure of association in the case of non-monotonic or non-

[2] By stability of a measure, we mean that the measure is relatively independent of factors that are not always easy to control, *e.g.*, sample size.

[3] R^2 is the standard notation, but we use r^2 here to make clear the relationship with r

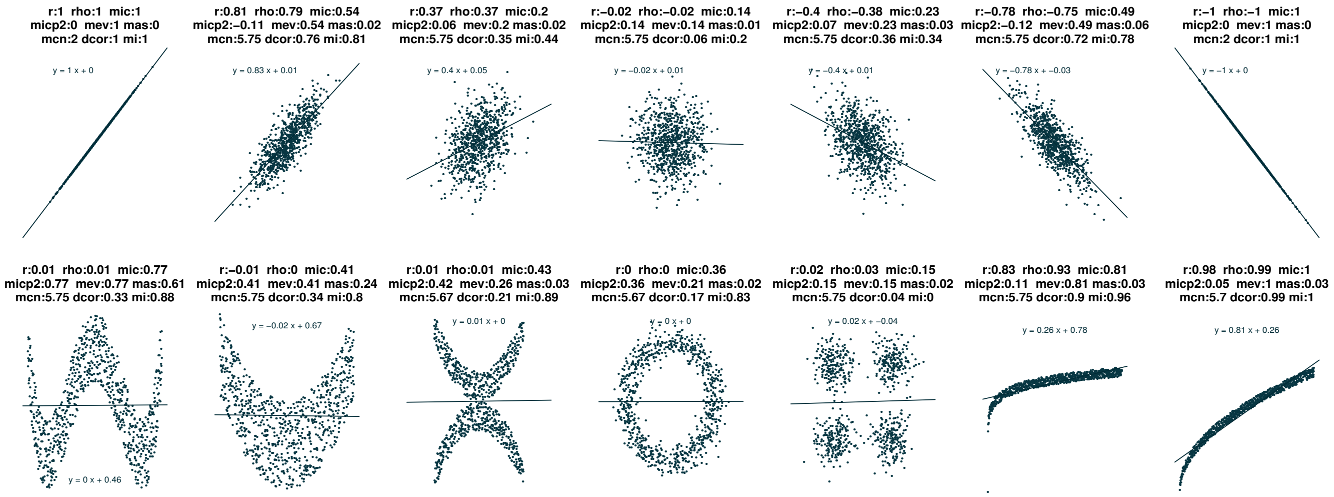

Figure 2: Examples of Pearson's correlation coefficient between two variables that highlight some of the limitations of the measure. While it captures the degree of non-linearity for linear relationships, as can be seen in the top row, and it captures the direction, it fails to capture the slope of the relationship. Finally, for many types of nonlinear relationships, the bottom row demonstrates that Pearson's R fails to show any relationship where one clearly exists. Blue line and equation is the least squares regression line for each data set. (Second row from left: $y = 4(x^2 - \frac{1}{2})^2$, $y = 2x^2$, $y = + - x^2$, $y = \cos(\pi x), x = \sin(\pi x)$, uniforn noise centerd in each quadrant, $y = x^{0.1}$, $\sqrt{x}$) *(Based on original image placed in the public domain by Denis Boigelot and available from the Wikipedia Commons)*

functional relationships. Even when the data is otherwise suitable, they may still yield results that are not straightforward to interpret, for example in the presence of ties.

3) Problems with Correlation: Fig. 2 vividly illustrates the problems that can arise with correlation. Each figure is a particular bivariate dataset, shown as a scatterplot; the two lines above each plot show the standard correlations, the *r* (Pearson's) and *r*ho (Spearman's) correlations. The second line contains MINE measures. We'll return to those later; for now, we focus on the values in the first line, and the tendency to mislead, misrepresent, or obscure relationships.

The first row of the figure illustrates the well behaved nature of *r* and *r*ho in the ideal setting: linear relationships in the presence of varying degrees of noise. The second row illustrates the varying types of problems that occur with these measures. The first five of these figures are are from a standard set of figures used to demonstrate limitations of *r*ho and *r*. The oscillating, and crescent data sets (Second row, first and second) are clearly functional, but non-linear; both *r* and *r*ho completely miss this relationship. Second, they *miss non-functional relationships* for example, the double crescent and the circle are non-functional (Second row, third and fourth) but *r* and *r*ho cannot detect this. Finally, they can *mislead based on the degree of non-linearity*. The two figures (second row, rightmost) indicate monotonic functional relationships (with added noise) with varying degrees of linearity; the relationships are $y = x^{0.1}$, and $y = x^{0.5}$. For these non-linear relationships Pearson's *r* captures the most linear portion of the curve as indicated by the equivalent regression line. The MINE measures also show that a strong relationship exists and yields quantitative evidence that the relationship is non-linear. With greater non-linearity, the greater is the tendency of *r* and *r*ho to be influenced by outliers. The MINE measures, together with *r* provide key insight into the degree of non-linearity, that can be very useful for properly modeling the relationship.

An examination of scatter plots (Fig. 1 of several variables of interest from Lucene 2.9.1 illustrates some of the typical issues in software engineering data. It is well known that data is very often right-skewed (*e.g.*, most files are quite small) and long-tailed (*e.g.*, a few large files, a few highly productive developers). Data is often discrete, with few distinct values (*e.g.* active developers, bugs). Turning to bivariate relationships, very few relationships show a discernible linear relationship, and many suggest non-linearity or even non-functionality. Quite a few plots show outliers, which can push Pearson's *r* unduly high. A number of them also show heavy origin-weighting, *e.g.*, many files have no commits and no defects, which can lead to high Spearman's *r*ho, which requires careful interpretation, as we discuss below.

4) Choosing Measures of Relationships: The problematic over-interpretation of correlation coefficients, and in particular, *r*, has been well discussed. Carroll cautioned that although no assumptions are necessary to compute *r*, its interpretation is dependent on the extent to which data conforms to an appropriate statistical model [4]. Courtney and Gustafson observed mean correlation values of 0.7 in simulated random data demonstrating that the probability of finding high correlation in small software datasets is quite

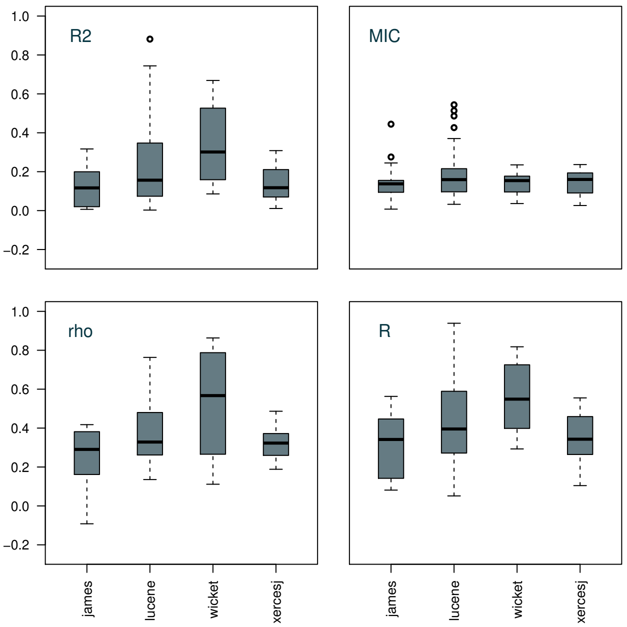

Figure 3: Dependence measures between defect and all other variables broken down by project. r and rho are the Pearson and Spearman correlations, resp., and $R2 = R^2$ is the coefficient of determination.

high [5]. The authors suggest the use of non-parametric measures such as *r*ho. Briand *et al.* cautions that even non-parametric measures require understanding of scale, and notes that in software engineering data, it is often quite difficult to determine what scale is being used [6]. Thus, measures that can work independently of distribution, functional form, or scale should have value to software engineers.

Rank correlation can be viewed as a generalization of correlation discarding the magnitude of changes through a non-linear transformation to ranks. Linfoot argued that the natural extension of the correlation coefficient is to consider only the amount of shared information between the two data sets, *viz.* their mutual information [7].

5) Mutual Information: Mutual Information (MI) can be traced to Shannon, who defined the entropy of a single random variable as $H(X) = -\sum_{i=1}^{n} p(x_i) log_2 p(x_i)$. MI is a measure of the amount of information shared by two random variables. The mutual information of two discrete random variables X and Y was first proposed as a measure of independence by Linfoot in 1957 [7]. The mutual information I, measured in bits, is defined as:

$$I(X;Y) = \sum_{y \in Y} \sum_{x \in X} p(x,y)\ log_2 \left(\frac{p(x,y)}{p(x)p(y)} \right)$$

If X and Y are independent, then we learn nothing about the value of Y from knowing the value of X; their mutual information is zero. If, on the other hand, the two variables are identical, or functionally related, then knowledge of X tells us everything about Y and the mutual information is the same as the entropy of either random variable, $I(X;Y) = H(X) = H(Y)$. Intuitively, mutual information quantifies the reduction in uncertainty in what we know about Y given that we know X.

While Linfoot's original 1957 paper includes a normalization, $r_{mi} = \sqrt{1 - e^{2r_{0_{mi}}}}$, that constrains the range of values between 0 and 1, there are other issues with using MI as an effective replacement for the correlation coefficient; MI is difficult to estimate well for small amounts of data and does not satisfy the criterion of equitability, *viz.*, it doesn't yield similar values of association for similarly noisy relationships, independent of the nature of the association [8]. MIC addresses these problems and yields values that can provide new insight for the examples shown in Fig. 2.

III. MINE THEORY

A. Maximum Information Coefficient

The MIC coefficient uses mutual information at its core and in a widely reported article has been celebrated as a "Correlation Coefficient for the 21st century" [8]. Like *r* and *r*ho, MIC is bound between 0 and 1; unlike *r* and *r*ho, it is able to detect non-linear and non-functional relationships and it is equitable across similar strength relationships of different functional forms. It should be interpreted like r^2 and is equivalent to r^2 for linear relationships.

The intuition behind this measure is that if there is a relationship between two variables then the scatter plot of the data can be partitioned by grids of successively finer granularity in order to isolate and identify any relationship. Reshef's approach builds on prior work [9], [10]. In some sense, it is a mechanical realization of how humans visualize data in a graph to understand the relationship between the variables. We present a brief description of the algorithm here; a formal description of the procedure can be found in Reshef *et al.* with detailed proofs of many of its properties in the associated supplementary materials [1].

MIC subdivides the data in many ways searching for the optimal MI value by superimposing many grids over the data and computing MI for each grid. For each x, y pair of grid dimensions (i.e. number of rows and columns, respectively), the MIC algorithm constructs a grid $G_{x,y}$ subdividing the data into xy cells. For example, if $x = 2$, then some portion of the points will fall in the first row, and some into the second. These two proportions form the probability distribution P_x for the x dimension. The y dimension will similarly induce a probability distribution P_y. The mutual information $I_G(P_x, P_y)$ is then estimated and normalized by the log of the smaller dimension, yielding a normalized measure, $M_{x,y}$. This binning process is repeated for all x, y pairs for which the product xy is less than the sample size B. Thus, for small sample sizes, the number of explored bin configurations could be quite small. MIC is then the maximum of the normalized mutual information value $M_{x,y}$

over all such grids. In addition to determining the number of partitions of each dimension, it is also necessary to choose the grid coordinates for each cell. The constructed grids are not simply equal partitions of the data as this approach would yield sub-optimal values for MIC and exploring all possible x, y partitions would be prohibitively expensive, so the algorithm fixes a y axis and uses dynamic programing to determine a corresponding optimal x partition. The procedure to calculate MIC also results in the calculation of the related MINE metrics; these metrics together convey more information about the underlying relationship than does a single correlation measure. They are:

- MIC $-\rho^2$, which measures the non-linearity of the relationship. MIC $-\rho^2$ is the difference between MIC and the ρ^2 where $\rho^2 = r^2$. Since MIC is constructed to be approximately equal r^2, MIC $-\rho^2$ should be zero for linear relationships.
- MEV, *Maximim Edge Value*, measures the degree to which relationships are non-functional. A functional relationship would pass a vertical line test. In such a case MIC should be close to optimal for $x = 2$. No larger grid would be necessary to optimally partition the unique y values for each x. When there are multiple y values for each x the optimal grid might include a cell that contains both y values for a particular x. Similarly, $y = 2$ is close to optimal for a horizontal line test. This observation yields a simple measure for MEV, it is the maximum normalized mutual information value, $M_{x,y}$, with $x = 2$ or $y = 2$, and it is always less than or equal to MIC. If MIC is much higher than MEV then we have evidence that a strong non-functional relationship exists.
- MCN, *Minimum Cell Number* relates to the minimum number of cells required to compute the MIC value. Simple well defined functions require fewer cells for MIC than more complex functions. The value is taken as an adjusted log of the actual count adjusted for noise.
- MAS, *Maximum Asymmetry Score*, measures the monotonicity of the relationship by computing the maximum absolute difference of $M_{x,y}$ and $M_{y,x}$. As with MEV, this value is always less than or equal to MIC. Relationships that are monotonic will tend to yield similar values when viewed from either the x or y perspective, in which case, MAS will be very close to MIC.

B. *Interpreting* MINE *Measures*

For each of the data presented in Fig. 2 we have computed the MINE metrics. The simplest relationship at top left of Fig. 2 yields 1 for all measures of correlation as well as MIC. The remainder of the MINE metrics reflect the linear relationship (MIC $-\rho^2 = 0$). It has low complexity (MCN $= 2$), perfect monotonicity (MAS $= 1$), and perfect functionality (MEV $= 1$). The plots to the right reveal the effects of noise on this relationship. Adding even a moderate amount of noise, the complexity, MCN, increases to 5.75, and MIC drops considerably compared to the correlation measures. The difference between MIC and r^2, however, is less than 0.15 for the linear examples. This is the appropriate comparison for MIC. We see only a small change in monotonicity, and that as we move to the right, this value does not continue to increase and becomes difficult to interpret in the presence of noise. MEV remains equal to MIC across all of the figures indicating that these are not non-functional relationships. Interestingly, the center plot, which shows almost no correlation dependence, has a MIC score of 0.14. There shouldn't be any relationship here as the data is generated randomly from a multivariate normal distribution with a covariance matrix equal to the identity matrix; theoretically, the data should be independent. MIC, however, measures some mutual information between the two samples. We discuss this further in Section V.

In the second row the first figure is non-monotonic and has an MAS value of 0.61 but the MIC score is quite high at 0.78, whereas the non-monotonic figure immediately to the right has a much lower MIC score and a correspondingly lower MAS score. Because MAS and MEV are bound by MIC, *viz.* they are in fact MIC values for a specific set of grids, neither can be interpreted independently of MIC; The next two figures are non-functional and we see that their MEV scores are markedly lower than their MIC scores, but, notice that MAS is now very low. These figures are not monotonic but exhibit similar mutual information when viewed from either axis. From these figures it appears that MAS is more a measure of periodicity than strict monotonicity. The final two figures on the right bottom show that for some typical power law relationships MIC is similar to r^2 as evidenced by the low value of MIC $-\rho^2$, but we can also see, despite, the strong linear component, that MIC $-\rho^2$ is greater for the first of the two power plots which less linear than the second. We generated this particular dataset 30 times and compared the two populations of MIC $-\rho^2$ with a Wilcoxon signed ranks test which returned a p-value $<< 0$ indicating that on this simulated data MIC $-\rho^2$ is able to distinguish grades of linearity. Finally,with the exception of the first and last figure in the top row, all of the complexity scores, MCN, are between 5 and 6. MCN holds little discriminatory power in noisy datasets.

C. *Alternatives*

In this work we are primarily interested in the practical application of the MINE measures; however, we discuss briefly two alternatives. First, we include the raw normalized mutual information score as originally proposed by Linfoot and using methods due to Kraskov *et al.* [7], [11]. Reshef *et al.* attempted to construct a non-linear detection protocol using Kraskov's MI but found that the lack of equitability made the measure unreliable. We note here that MI returns much higher values than MIC for some strong non-functional relationships such as the third and fourth figures in the second row. In addition, it also sometimes returns lower

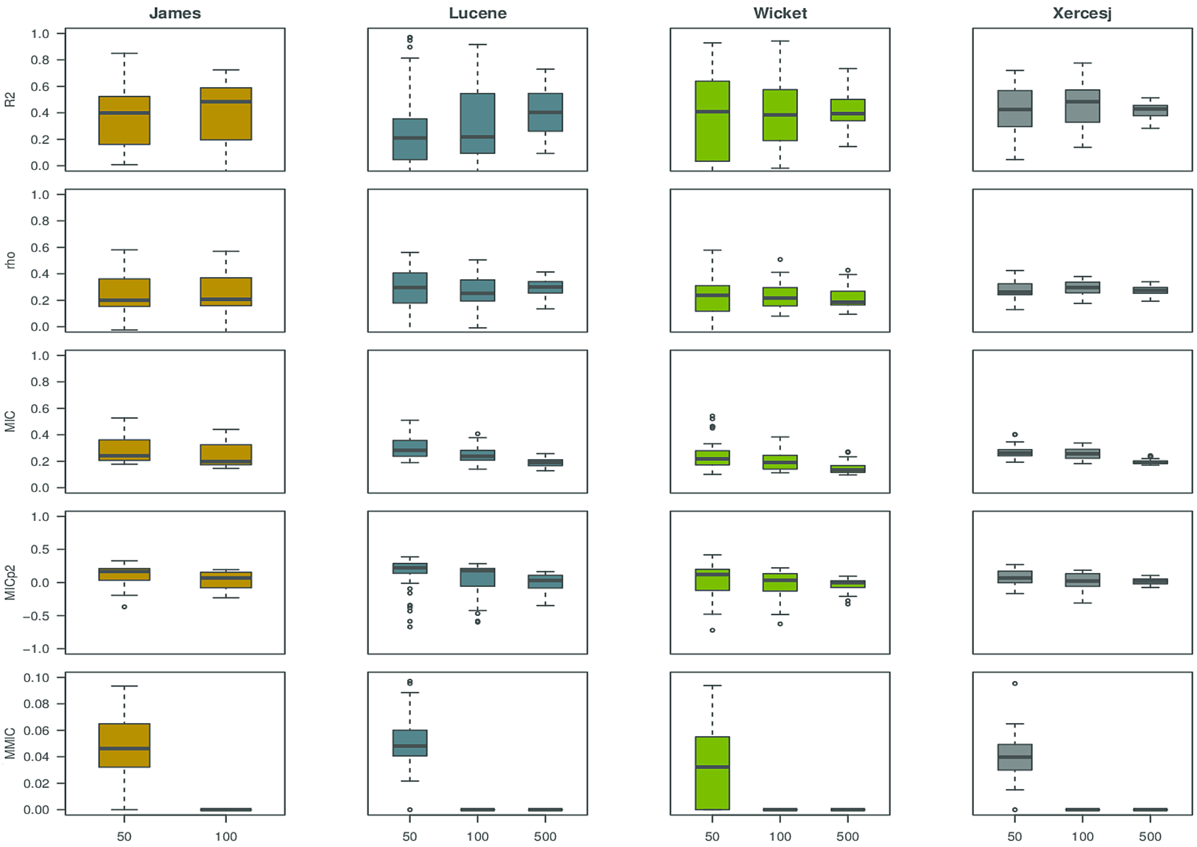

Figure 4: The distribution of correlation measures over increasing sample sizes for all releases of each project.

values when there is no relationship, such as the fifth figure in the second row. However, this is not consistently true as the center figure in the top row of Fig. 2 indicates.

Second, a measure called Distance Correlation, or DCOR, was recommended by Simon and Tibshirani in their recent comment on the low statistical power of MIC [12]. In the second row, the first two figures yield approximately the same value for DCOR, yet MIC, is substantially different. Although the MCN values are identical, most likely owing to noise, the first figure has a much higher MAS indicating greater non-monotonicity (or periodicity). Knowing X here tells us more about Y than in the second plot perhaps, again, owing to greater apparent noise in the second plot. With our limited evaluation here, DCOR appears to be less sensitive to non-functional relationships, consistently lower when no relationship exists, but lacks many features of the MINE suite.

IV. Methodology

In this section, we describe our methods for data extraction in order to evaluate MINE on ESE data. We used the code[4] provided by Reshef *et al.*, however, we extended it[5] so that it could be used more easily within the R programming environment [13].

Metric	description
Defects (Bugs)	# defects associated with each file
LOC	Source lines of code
# Developers	# devs who have ever touched a file
# Active Developers	# devs actively editing a file
Added	# added changed lines
Removed	# deleted lines
Commits	Count of commits to file

Table I: Apache Data metrics gathered and their description.

Project	Releases	Files	Commits	Defects
James	2.3.0 - 3.0-M2	375-477	4,068	265
Lucene	1.9.1 - 3.0.3	541-1368	12,384	2,093
Wicket	1.3.0 - 1.3.5	1,894-1,940	7,157	781
XercesJ	2.8.1 - 2.10.0	740-824	2,511	199

Table II: Apache projects used in this study.

[4] http://www.exploredata.net/Downloads

[5] http://miccheck.tcfacs.org

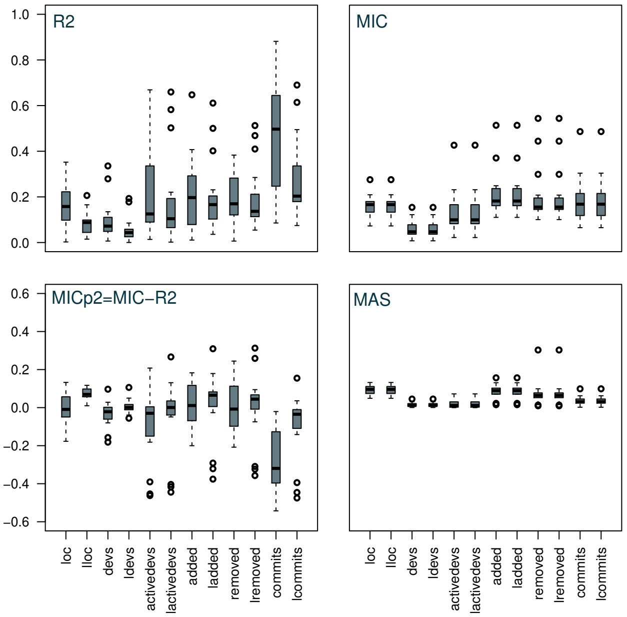

Figure 5: MIC and associated nonlinearity and asymmetry measures between defect and all other variables broken down by variables across all projects.

A. Data

We extracted a selection of metrics often used for defect prediction from four projects maintained by the Apache Software Foundation as listed in Tbl. II. For each of these projects we used data from the source code repository and the Jira issue tracking system to extract process and code metrics; we discuss Jira in more detail in the next subsection. The metrics are listed in Tbl. I and described in the sections below. Both process and code metrics have previously been associated with defects [14], [15], [16]. We choose just a few representatives of each to study here.

1) Jira: Jira is an issue and bug tracking system that manages a database of issue reports submitted by developers and users. In addition to maintaining this database of issue (defect) reports, Jira can enforce a basic development process by mapping issue reports to version control commit messages. It does this by cross linking Jira issue ideas extracted from version control system commit log messages with the associated report in the Jira database. For this study we extracted the Jira issues from the XML report available on the Apache Software Foundation's project website for each of the projects. The detailed data for each issue is then extracted by crawling the issue's associated web page and gathering the commits related to the issue in question. Finally, we group the commits by their commit ID to determine which Jira issues are associated with each commit.

2) Version Control: Version control systems, *e.g.* Git, SVN, and CVS, facilitate collaboration among developers by maintaining a history of changes and an associated log entry for each change. To obtain the number of commits and developers associated with each file we parse the log data retrieved from the version control system. We draw a distinction between active developers, *viz.*, the count of those actively working on the file, and the number of developers who have ever touched the file.

With both the Jira issue IDs and the Git version control log file, we link issues associated with each commit to the files involved in the commit, associating a defect count for each file within the release. To capture the size of each file we extracted LOC *(lines of code)* using the *SLOCcount* tool [17].

V. Results and Discussion

A. **RQ1: Distribution and Interpretation of the MINE Measures**

To frame our expectation of the distribution of values that can be expected for the MINE measures with typical ESE data, in Fig. 3, we look at the various correlation properties across all of the variables listed in Tbl. I, broken down by project. We observe several phenomena in this plot.

The MIC scores are, on average, markedly lower than either of the traditional correlation measures but when MIC is more appropriately compared to, r^2 [1], the median r^2 values in three of the four projects look reasonably similar to the MIC values (see Fig. 3). A correlation value of 0.70 is equivalent to an R^2 of 0.49. A paired Wilcoxon signed ranks test shows that for Lucene, Xercesj, and James, that we cannot reject the null hypothesis that the samples were drawn from the same distribution. For Wicket we can reject the null with $p - value = 0.0002$. MIC yields values quite similar to existing correlation measures so long as we think in terms of how r maps to r^2.

The variance in MIC scores is lower across all projects than the other measures of correlation as well as r^2. Of the four projects, Wicket has the largest variance across the correlation measures; we consider release *wicket 1.3.4*. The smallest *r*ho $= 0.11$ is attributed to the relationship between bugs and developers. There are at most 13 developers that contribute to a single file but virtually all files have either $0, 1, \text{or}, 2$ bugs so the relationship is largely non-functional. MIC $= 0.04$ is also small for this relationship indicating that the two variables share little information. Interestingly, the r is moderate at 0.30 owing to the few files with many developers and more than 2 bugs. At the other extreme, the relationship between commits and bugs has *r*ho $= 0.86$ with MIC $= 0.218$. For this small release there are only 6 unique commit counts and 4 unique bug counts. If we add a small amount of noise to both variables to break ties then *r*ho $= 0.11$ whereas MIC $= 0.215$. Low sample set cardinality leads to many ties which can inflate *r*ho leading to high variance.

Next we take a look at the non-linearity score MIC $-\rho^2$ and the asymmetry score MAS. Fig. 5 shows several of the MINE measures broken down by variable across all of the

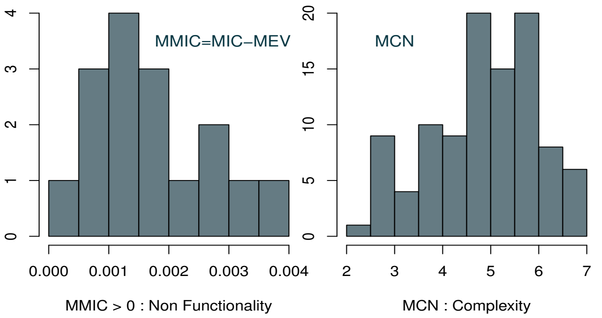

Figure 6: Non-functionality and complexity measures between defects and all other variables.

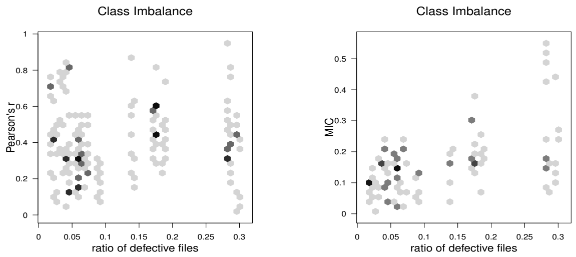

Figure 7: How the ratio of defective files affects MIC and **r**. In the figure on the right it can be seem that as the number of defective files decreases as a proportion of the total, that MIC also decreases.

projects. In addition to the raw variables, we also computed the log of each variable using $lvar = log(var + 0.5)$. This allows us to see how a typical transformation often used to linearize relationships affects MINE scores.

1) MIC $-\rho^2$*:* Because MICp2 = MIC $-\rho^2$ (where rho refers to Pearson's population correlation coefficient but, in practice, means the sample correlation *r*) the degree to which MIC $-\rho^2$ measures non-linearity is precisely the degree to which r^2 and MIC disagree on the dependence of a relationship; consequently, MIC $-\rho^2$ demonstrates a wide range of values across different variables. In some cases, *e.g. LOC* and *commits*, we can see a fairly dramatic change in MIC $-\rho^2$ after taking the logarithm. Taking the log of a skewed variable often has a linearizing effect and is done to to stabilize the variance [3]. In most cases we can see, not surprisingly then, that the logged variable yields lower variance in MIC $-\rho^2$; *r* will be closer to MIC and the difference, MIC $-\rho^2$ will be smaller, lowering the variance of the statistic. Taking the logarithm has no effect on MIC, the values are identical across the log transformation. This is a powerful feature for exploration, we can defer understanding the nature of a relationship until we observe the relationship.

2) MAS*:* MAS measures asymmetry in MIC when viewed from the x and y perspectives. We found that only *LOC*, and the churn measures of *added* and *removed* yielded consistent measures of MAS $>> 0$. Many bivariate relationships of interest to ESE researchers are, at least nearly, monotonic in nature and, as we've seen, noise can impact MAS.

3) MEV *and* MCN*:* For obviously non-functional relationships, MEV will be measurably lower than MIC, see *e.g.* the circle in Fig. 2. We initially expected that MEV would be useful in identifying potentially nonfunctional relationships (from a bivariate perspective) in ESE data, and that some of these relationships could be viewed as superpositions of functions; however, in practice, MEV was seldom different than MIC suggesting the unlikely conclusion that ESE relationships are always functional.

To understand the difference more clearly we introduce a simple derived measure MMIC = MIC-MEV. MMIC was non-zero in only 16% of the raw variables over all studied releases. When the difference was larger than zero, it still tended to be quite small as can be seen in Fig. 6. In general, we found the MEV measure less useful than expected and the very small values difficult to interpret meaningfully.

We evaluated the complexity score MCN, comparing the distribution shown in Fig. 6 with the well known relationships from Fig. 2. There were only seven relationships where MCN < 3 all of which related either the number of *developers* or the number of *active developers* to the number of *defects* in both *Wicket* and *James*. Interestingly, the cases show completely different structures. In *James*, MIC is very small, .02 , with a very low MIC $-\rho^2$ indicating linearity. *Wicket* on the other hand exhibited extremely non-linear relationships, of low complexity, but with higher MIC scores. For both of these projects, both variables have very few distinct values. Adding even a small quantity of noise to *activedevs* results in an immediate jump in complexity around 5.0 and MIC to around 0.15. Based on how MIC is computed, the sample cell cardinality, along with the sample size, limits the potential number of cells. Indeed, the complexity is lower, but this is a property of the nature of the data, not the nature of the relationship. In our data, low MCN generally did not indicate a simple relationship, rather it indicated simple data.

> **Result 1:** *On ESE data* MIC *yields markedly lower values when compared directly to* r. *When more appropriately compared with r^2, however,* MIC *values have comparable magnitude.* MIC *has lower variance than either* r *or* rho *owing to lower sensitivity to ties and outliers.* MIC $-\rho^2$ *has wider variance owing to its direct relationship with* r. MEV *is, in most cases, indistinguishable from* MIC *rendering it of limited practical value in identifying non-functional relationships.* MCN *was highly unstable in the presence of noise.*

B. RQ2: Stability of the MINE Measures wrt Sample Size

For this experiment we selected stratified samples with respect to the defective file ratio, requiring the percentage of defective files to remain constant, across all releases of all projects. For each release we selected three sample sizes 50, 100, and 500 samples and required each release to have at least 110% of the sample size to run the experiment on that release. As can be seen in Fig. 4, only *James* could not meet this requirement for the 500 sample experiments. For each run of each release we gathered the MINE metrics MIC, MIC $-\rho^2$, and MEV as well as *r* and *r*ho. We ran each experiment 10 times for each release at each sample point. The aggregate results of the runs are shown in the box plots in Fig. 4.

A general trend across all of the measures is that the variance of each statistic decreases with sample size. With the exception of *Lucene*, however, the traditional correlation measures showed little change in the median correlation values . MIC on the other hand showed a consistent decrease in median value with increased sample size with a smaller change in variance. A Kruskal-Wallace non-parametric test of difference in medians is significant in all projects for MIC after Benjamini-Hochberg correction [18]. For both r^2 and *r*ho we cannot reject the null that the medians are different in any project. As we have observed, MIC is sensitive to noise. ESE data is noisy and a greater number of samples also means a greater number of data points that may have seemingly random relationships with other variables of interest. Thus, although we see the variance decrease yielding greater confidence in the true value of MIC, we also see its value decrease indicating a weaker relationship. A Kruskal Wallace test shows that the medians (of MIC) are different for all projects after correction. The non functionality score is particularly unstable for small sample sizes at which the non-functionality values are many orders of magnitude larger than the values observed on the entire dataset. The magnitude drops to close to zero for samples sizes of at least 100.

Result 2: *The* MINE *measures are sensitive to sample size but exhibit somewhat different behavior than traditional correlation measures owing to their sensitivity to noise. Some measures are particularly unstable below a sample size of* 100.

C. RQ3: Finding New Relationships with MIC

We consider the relationship between *added* lines and *defects*, as measured by *r*ho and MIC. The disparity between these values in *wicket 1.3.4*, where *r*ho = 0.86 and MIC = 0.23, is a specially nuanced and enlightening case. There are only 5 unique values of defects and 62 unique values of added lines. Specifically, over 95% of the nearly 2000 files have no defects linked to them. There are not only a very large number of cases of no defects, and no lines added, but also a non-trivial likelihood of added code being associated with defect repair. These two tendencies lead to a very high *r*ho. The MIC algorithm, on the other hand, takes a very different view of the data. $95\%s$ of this bivariate data set lie at origin and will remain in one cell no matter how a grid is superimposed over the data. When we look at the plot, the thousands of points at the origin weigh that point highly and result in low probabilities in other cells, but tell us little more about the relationship that we didn't know after the first point was placed there; specifically, the MIC algorithm observes that the non-origin points don't give much mutual information, and doesn't unduly weight the on-origin points. Interestingly enough, if we consider only the subset of points in *wicket 1.3.4* that have non-zero defects then *r*ho = 0.12 drops dramatically while MIC =0.16 is less impacted. Thus the large number of on-origin ties, which are noiseless with respect to the relationship, has a much higher impact on correlation than on mutual information. Thus, the practical implications are that the high *r*ho- low MIC disparity is giving us a warning to be careful. Closer examination reveals something *very subtle and nuanced*: on the one hand, the high *r*ho is telling us, very strongly, that when there are no added lines, there probably are no defects. On the other hand, the low MIC is telling us that there is probably little to be gained by preferentially focusing QC efforts on files with more lines added. The lesson here is that, in general *When analyzing multivariate project data, disparities between traditional correlations and* MIC *merit careful examination.*

Having few files with non-zero defects is not an uncommon scenario for ESE data and we considered what happens to correlation and MIC as the ratio of defective files increases. We plotted the values of both MIC and *r* against the proportion of defective files and found that the measures behave differently. Fig. 7 shows that while correlation (r) does not show any obvious relationship with the defective file ratio, MIC clearly does and consistently reports lower scores for datasets with fewer defective files.

Result 3: MIC *can identify non-trivial relationships that* r *cannot. Further, while class imbalance can lead to highly unpredictable correlation values,* MIC *does not demonstrate instability as class imbalance increases.*

VI. Conclusion

MIC is a useful data mining tool in general: it provides a new measure of association able to discover relationships among variables that may depend on each other in ways functionally very different from linear. It also decouples the measure of association from those of linearity, functionality, and monotonicity. And the fact that it is an equitable measure makes it particularly appealing as the baseline in comparisons of other measures. Thus, it is potentially very useful for the exploration of new data sets and for the generation of new hypothesis in old data sets that have not seen a more sophisticated analysis.

And while exploring a general data set with MIC would typically result in more hypotheses generated, there is no free lunch. MIC is sensitive to sample size and noise, and some of the associated MINE measures provide much less insight than hoped for on ESE data. It also has been shown to have lower statistical power than another recent correlation measure (distance correlation) in some cases, i.e. it is prone to more false negatives. For larger sample sizes this will be less of a problem, but a more thorough comparison between these two measures is certainly indicated. It is worth mentioning that the distance correlation is not equitable. On the other hand, MINE is expensive to compute. A sample set of 100 samples finished in 1.5 seconds compared to over 8 minutes for 20000 samples. Both DCOR and Kraskov's MI were considerably faster on our data.

In this paper we have illustrated the use of MIC on a variety of empirical metrics in software engineering. We have shown that it has more desirable distributional properties than other correlation measures. MIC also readily identified a number of nonlinear and non-functional relationships in our data, even among variables that have traditionally been modeled in linear models. It is an open question whether MINE's non-trivial findings (e.g. non-linearities, non-functionalities, and non-monotonicities) can be leveraged to build better predictive models, by, e.g. correcting for those findings. This is something we'll look at in great detail in the near future.

Practically, we conclude that a researcher in ESE would benefit from using the MINE measures in several ways: (1) for hypotheses generation, especially involving non-linear and non-functional models, MINE is a one-stop-shop for multiple analyses of large data sets; (2) in terms of better predictions, MIC is a more conservative measure and is less prone to eager overestimation of correlation in complex data sets; and (3) as an objective measure, MIC offers a neutral baseline to which other measures' performances can be compared.

Acknowledgments

All authors gratefully acknowledge support from the Air Force Office of Scientific Research, award FA955-11-1-0246, the National Science Foundation, grant numbers 0964703 and 0613949. Devanbu gratefully acknowledges support from IBM Research and Microsoft Research. We thank Earl Barr for fruitful discussions regarding some of our terminology. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

[1] D. Reshef, Y. Reshef, H. Finucane, S. Grossman, G. McVean, P. Turnbaugh, E. Lander, M. Mitzenmacher, and P. Sabeti, "Detecting novel associations in large data sets," *Science*, vol. 334, no. 6062, pp. 1518–1524, 2011.

[2] J. Rodgers and W. Nicewander, "Thirteen ways to look at the correlation coefficient," *American Statistician*, pp. 59–66, 1988.

[3] J. Cohen, *Applied multiple regression/correlation analysis for the behavioral sciences.* Lawrence Erlbaum, 2003.

[4] J. Carroll, "The nature of the data, or how to choose a correlation coefficient," *Psychometrika*, vol. 26, no. 4, pp. 347–372, 1961.

[5] R. Courtney and D. Gustafson, "Shotgun correlations in software measures," *Software Engineering Journal*, vol. 8, no. 1, pp. 5–13, 1993.

[6] L. Briand, K. Emam, and S. Morasca, "On the application of measurement theory in software engineering," *Empirical Software Engineering*, vol. 1, no. 1, pp. 61–88, 1996.

[7] E. Linfoot, "An informational measure of correlation," *Information and control*, vol. 1, no. 1, pp. 85–89, 1957.

[8] T. Speed, "A correlation for the 21st century," *Science*, vol. 334, no. 6062, pp. 1502–1503, 2011.

[9] G. Darbellay and I. Vajda, "Estimation of the information by an adaptive partitioning of the observation space," *Information Theory, IEEE Transactions on*, vol. 45, no. 4, pp. 1315–1321, 1999.

[10] R. Steuer, C. Daub, J. Selbig, and J. Kurths, "Measuring distances between variables by mutual information," *Innovations in Classification, Data Science, and Information Systems*, pp. 81–90, 2005.

[11] A. Kraskov, H. Stögbauer, and P. Grassberger, "Estimating mutual information," *Physical Review E*, vol. 69, no. 6, p. 066138, 2004.

[12] N. Simon and R. Tibshirani, "Comment on "detecting novel associations in large data sets" by reshef et. al, (science dec 2011)." *http://www-stat.stanford.edu/∼tibs/reshef/comment.pdf*, 2012.

[13] R Development Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2011, ISBN 3-900051-07-0. [Online]. Available: http://www.R-project.org

[14] M. D'Ambros, M. Lanza, and R. Robbes, "On the relationship between change coupling and software defects," in *Reverse Engineering, 2009. WCRE'09. 16th Working Conference on.* IEEE, 2009, pp. 135–144.

[15] R. Moser, W. Pedrycz, and G. Succi, "A comparative analysis of the efficiency of change metrics and static code attributes for defect prediction," in *Software Engineering, 2008. ICSE'08. ACM/IEEE 30th International Conference on.* IEEE, 2008, pp. 181–190.

[16] R. Purushothaman and D. Perry, "Toward understanding the rhetoric of small source code changes," *Software Engineering, IEEE Transactions on*, vol. 31, no. 6, pp. 511–526, 2005.

[17] D. A. Wheeler, "More than a gigabuck: Estimating GNU/Linux's size," Online Paper, 2001. [Online]. Available: http://dwheeler.com/sloc/redhat71-v1/redhat71sloc.html

[18] Y. Benjamini and Y. Hochberg, "Controlling the false discovery rate: a practical and powerful approach to multiple testing," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 57, no. 1, pp. 289–300, 1995.