
Proceedings Paper:
Foster, M. orcid.org/0000-0001-8233-9873, Wild, C., Hierons, R. et al. (1 more author) (2024) Causal test adequacy. In: 2024 IEEE Conference on Software Testing, Verification and Validation (ICST) Proceedings. 2024 IEEE Conference on Software Testing, Verification and Validation (ICST), 27-31 May 2024, Toronto, Canada. Institute of Electrical and Electronics Engineers (IEEE) , pp. 161-172. ISBN 9798350308198

https://doi.org/10.1109/ICST60714.2024.00023

Causal Test Adequacy

Michael Foster, Christopher Wild, Robert M. Hierons, and Neil Walkinshaw
The University of Sheffield, UK
Email: {m.foster, c.wild, r.hierons, n.walkinshaw}@sheffield.ac.uk

***Abstract*—Causal reasoning is becoming an increasingly popular technique for testing software. In this setting, the tester starts from a simple directed graph that captures their underlying understanding of causal relationships between relevant variables in the program, and this knowledge is then used to reason about causal input-output relationships that are observed during testing. One question that has not yet been addressed in this context is how to measure test adequacy: How do we know whether a causal relationship (or set of relationships) has been properly established by a test set? In this paper we present a metric inspired by Weyuker's notion of inference adequacy. For a given causal relationship, we estimate the causal effect from the test data. The basis of our adequacy metric is then an estimate of the convergence of this estimate, which we calculate using statistical bootstrapping. We evaluate our metric on tests for three diverse computational models. The results show a statistically significant correlation between our metric and a test suite's ability to detect mutants, and also that it is a good indicator of whether a sufficient number of system executions have been observed to trust the outcome of the test.**

***Index Terms*—software testing, causal inference, test adequacy**

I. Introduction

Causal reasoning and causal inference is becoming an increasingly popular topic of research in the field of software testing [1], [2], [3], [4]. The basic idea is to employ causal reasoning and statistical modelling to enable existing runtime data to be used as part of the testing process without biasing test outcomes.

This family of techniques has emerged in response to the increasing prevalence of stochastic and classically "untestable" [5] software, such as computational models, autonomous drivers, and cyberphysical systems. Testing these systems is fundamentally different from testing traditional software as tests typically cannot be framed as simple assertions of equality between expected and observed values. Due to the inherent stochasticity, tests are instead framed as statistical properties over multiple executions of the software. Thus, there is no longer a one to one correspondence between executions and tests.

While these techniques have been shown to be promising, they are still subject to the fundamental question of test adequacy [6]: How can we determine if we have collected enough test data?

For classical software, a popular solution is to monitor the coverage of test objectives such as source code statements [7], [8]. The relatively poor correlation between code coverage of a test set and its ability to expose faults [9], [10] makes this a questionable metric even in the traditional context. However, there are more fundamental reasons why coverage is inappropriate in the context of causal testing.

The main reason for this is the fact that tests are phrased as statistical properties over multiple runs of a system. This means that a test could fail because there is insufficient data to enable the correct conclusion to be drawn, even if the software actually satisfies the property being tested. In such cases, re-evaluating a failing test using more runs may enable it to pass, although this may not change the coverage of the test set.

In this paper, we present causal test adequacy, a test adequacy criterion which is well-suited to causal testing. Our approach is inspired by inference adequacy [11], a theoretical approach to assessing test adequacy, which is based on the ability to reverse engineer an accurate model of system behaviour from a test set. Our metric assesses test adequacy by estimating model convergence. That is, whether observing additional runs of the system would change the output of the inferred model. In the context of causal testing, this is a causal effect estimate. We do this by repeatedly retraining the model on samples of the test data and examining the distribution of model outputs. Our main contributions are as follows:

- A test adequacy metric based around causal reasoning.
- An openly available implementation [12].
- An empirical evaluation based around three case study systems. This shows that causal test adequacy can potentially indicate the fault finding capability of a test suite. Furthermore, given a failing test, causal test adequacy can indicate whether a test is failing because of a fault or simply due to a lack of data.

The remainder of this paper is structured as follows. Section II provides the necessary background for our technique in the context of a running example. Section III presents causal test adequacy. Section IV presents our research questions and empirical evaluation in the context of three subject systems. Section V discusses related work. Finally, Section VI concludes the paper and discussed potential future work.

II. Background

This section discusses the necessary background for this work. We first review existing software adequacy metrics. We then introduce causal testing in the context of a motivating example.

A. Software Test Adequacy

The aim of software testing is to discover faults in a program by executing it with concrete inputs, with the aim

This work was funded by the EPSRC CITCoM grant EP/T030526/1.

of discovering deviations between the program's specified and actual behaviour. A test adequacy criterion can determine whether a program has been tested "enough" [6] – i.e. whether a sufficient range of program inputs have been attempted so as to offer a high degree of confidence that there are no remaining faults lurking in the system. Traditional metrics tend to be based around program elements such as statements or branches. The aim is then to cover (i.e. execute) every unit at least once. However, there is a relatively poor correlation between code coverage of a test set and its ability to expose faults [9], [10].

In contrast to these approaches, inference adequacy [11] involves the use of test data to infer the behaviour of the program. A test set is classed as adequate when it enables the inference of a model which satisfies the specification and perfectly reproduces the behaviour of the program. Definition II.1 gives a formal definition.

Definition II.1. For a program P, specification S, test set T, and inferred model I_T, T is said to be *inference adequate* if

$$I_T \equiv P \qquad (1) \qquad\qquad I_T \equiv S \qquad (2)$$

Establishing the first equivalence $I_T \equiv P$ is challenging in practice, because it is essentially a model-based testing problem in and of itself. A typical approach (also proposed by [11]) is to create a further "reference" test set R (possibly by random generation [11]). Then, $I_T \equiv P \iff \forall r \in R.P(r) = I_T(r)$.

B. Causal Testing

Causal testing is an emerging technique for testing causal relationships between variables (typically inputs and outputs), especially in nondeterministic systems such as computational models [1], [2], cyberphysical systems [3], and autonomous driving [4]. The basic idea is to use a causal model to specify domain knowledge about the expected causal relationships between different variables. This typically takes the form of a directed acyclic graph (DAG) in which nodes represent variables, and an edge $X \to Y$ represents a direct causal effect of X on Y.

Despite their lightweight nature, causal DAGs underpin a powerful family of causal inference techniques [13]. Here, causal analysis is typically carried out in two parts: (1) *identification of the estimand* (isolating the causal effect by identifying variables which must be controlled for to block non-causal paths through the DAG), and (2) *estimation* (using statistical models that adjust for the identified confounding variables to estimate the causal effect using a *causal effect measure* such as the average treatment effect (ATE)). This estimate is also typically accompanied by 95% confidence intervals [14]. Causal testing then introduces a third step of checking whether the estimated effect is what we expect. For example, we can validate the presence of a causal effect by checking that the confidence intervals of the estimated ATE do not contain zero [15].

Definition II.2. Given a causal DAG $G = (V, W)$ which represents the causal relationships between relevant inputs and outputs of the system under test (SUT), a *causal test case* is a triple (X, Y, E), in which X and Y are nodes in V (representing SUT inputs and outputs, respectively), and E is the expected causal effect of X on Y. A *causal test suite* is a set of causal test cases.

C. Running Example: Testing Causality in a COVID Model

Covasim [16], [17] is an epidemiological agent-based model that has been used to inform COVID-19 policy decisions in several countries [18], [19], [20], [21]. Given this critical application, it is of paramount importance that this model is adequately tested.

Figure 1 shows a causal DAG for a simple scenario of Covasim investigating how *Population* and *Location* affect the number of *Infections*. Here, *Population* and *Location* are relevant inputs to the model. *Contacts* and *Susceptibility* are intermediate, internal variables, and *Infections* is an output of the model.

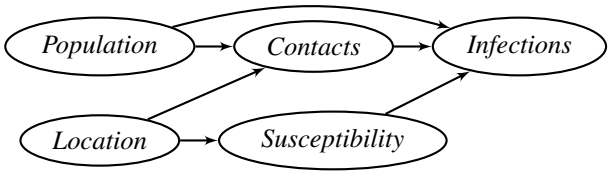


Fig. 1: Causal DAG of a simple run of Covasim.

The causal DAG in Figure 1 specifies the causal relationships we expect to be present in this setting. Indeed, it is possible to automatically transform any DAG into a suite of causal test cases that validate the specified relationships and independences [1]. However, because causal testing views test cases as statistical properties over multiple runs of the software, these test cases — the relationships we want to validate — exist in complete isolation from the test data which is used to evaluate them, with neither providing much testing value without the other.

This is a sharp contrast from traditional testing approaches, in which test *cases* and test *data* are almost synonymous, but gives rise to a very powerful feature of causal testing, which is that we do not require carefully controlled test executions, which can be time-consuming and costly to collect. Instead, we can reuse existing runtime logs or test data [1]. The mathematical underpinning of causal inference still allows us to draw causally valid conclusions from such data, even if the data is biased [22], because our causal DAG allows us to identify and adjust for this.

One causal test case we can derive from Figure 1 is the effect of *Contacts* on *Infections*. Since there is an edge connecting these two nodes, we expect there to be a nonzero causal effect. To isolate this causal effect, we need to adjust for *Population*, since this is a common cause of both. It therefore confounds the causal effect. We also need to adjust for *Location* since this also has causal pathways to both *Contacts* and *Infections* through *Susceptibility*, which we could adjust for directly instead of *Location*.

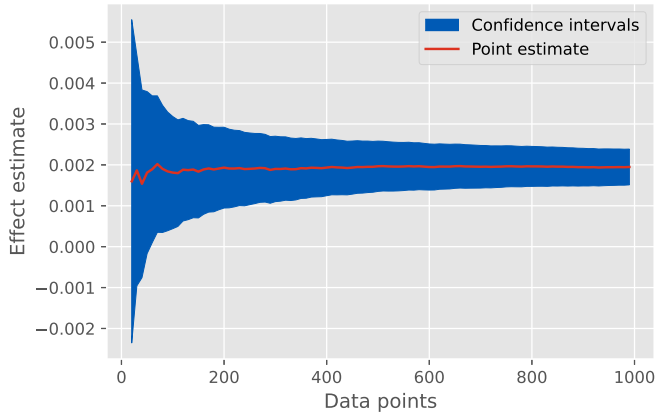


Fig. 2: Causal effect estimate vs. data set size

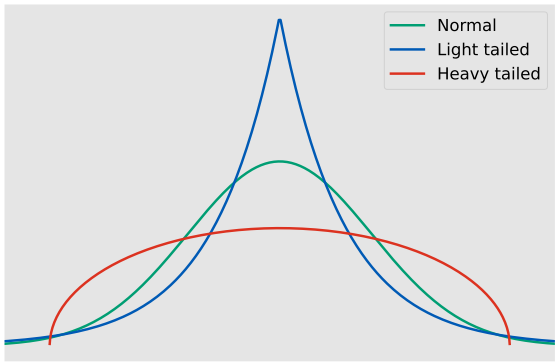


Fig. 3: Kurtosis of different distributions

D. Motivation

An overarching problem in this setting is that there is no published approach that can serve to establish the adequacy of a given set of test data for evaluating a given causal test case. To illustrate this, consider again the causal effect of *Contacts* on *Infections* in Covasim. Figure 2 shows how the causal effect estimate changes as we increase the amount of test data.

Initially, the confidence intervals are very wide, and even contain zero, indicating a possible lack of causal effect and so the test will fail. The confidence intervals then get gradually narrower, with the impact on the confidence intervals from adding test data gradually diminishing up to about 800 executions. After this, additional test data has very little effect, indicating that convergence has been reached.

Crucially, there is no standard technique in the literature for establishing whether this point of convergence has been reached, nor how many data points are required to get there. Intuitively, we could say that we have reached convergence when our confidence intervals become sufficiently narrow. However, this varies greatly between systems to so there is no universal notion of what "sufficiently narrow" means. A common approach is to simply use a predefined large number of data points [23], [24], but there is no way to know if this is sufficient, or is unreasonably high. In Section III, we will present an alternative criterion which can be used to estimate whether the point of convergence has been reached.

E. Statistical Techniques

The technique we present in Section III is based around two statistical techniques, which we define here.

1) Bootstrapping: Bootstrapping [25] is a common statistical approach used to assign measures of accuracy (such as confidence intervals) to estimated quantities (such as causal effects). The process works as follows: first, we repeatedly *resample* our dataset (with replacement) to generate many samples[1] which are the same size but with a few values duplicated and a few missing. We then use these samples to estimate the quantity of interest (e.g. the causal effect), and use the distribution of these estimates to estimate the accuracy measure we are interested in (e.g. the confidence intervals).

For example, to estimate the 95% confidence intervals of a causal effect, we could resample our dataset 200 times, estimate the causal effect using these samples, and estimate the confidence intervals by sorting these estimates into increasing order and taking the 5th and 195th values. By leaving out the top and bottom 5 values (i.e. 2.5% of our 200 samples), our intervals then cover 95% of our bootstrap estimates.

Bootstrapping is attractive because it can be used to estimate almost any statistical quantity (bias, variance, confidence intervals, hypothesis tests, etc.). Furthermore, it is non-parametric, so makes no assumptions on the underlying distribution of the dataset. Whatever this is, the distribution of bootstrapped estimates will converge on the true value [25], although the dataset must be representative of the underlying population. The only drawback is that we must repeatedly calculate our statistical estimate, which may be time consuming.

2) Kurtosis: Kurtosis, like variance, relates to the shape of a distribution. Where variance intuitively controls how "wide" a distribution is, kurtosis measures the "tailedness" of a distribution. That is, how much of the probability mass of a distribution is sitting in the tails, away from the centre. Light tailed (i.e. "pointy") distributions indicate a high degree of certainty, in that sampled values are likely to be close to one particular value. Conversely, heavy tailed (i.e. "flat") distributions are more uncertain and are closer to the uniform distribution. This is illustrated in Figure 3.

Formally, kurtosis is calculated as the fourth central moment divided by the square of the variance [26]. That is $E(X-\mu)^4/(E(X-\mu)^2)^2$ in which E is the expectation operator and μ is the mean. In this work, we use Fisher's definition, which subtracts three from this value, giving a more intuitive measurement. A perfect normal distribution then has a kurtosis of zero. Heavy tailed distributions have positive kurtoses, and light tailed distributions have negative kurtoses. There are no bounds in either direction, so a distribution could have an arbitrarily positive or negative kurtosis.

[1]The exact number of samples is a parameter of the technique. While more samples will give a more accurate estimate, there is no standard number.

III. Causal Test Adequacy

This section defines causal test adequacy, our model-based test adequacy criterion which indicates the adequacy of a given dataset for testing a particular causal relationship within a given causal DAG. The objective is to determine whether sufficient test data has been collected that the statistical model used to evaluate a causal test case has reached convergence. This will indicate whether we can trust a test outcome or whether we should collect more data.

A. Computing Causal Test Adequacy

Algorithm 1 shows how we estimate the adequacy of a dataset for a causal test case. Essentially, we do this by applying statistical bootstrapping; we repeatedly resample the data and use these samples to estimate the causal effect of interest. We then use the distribution of these effects as a measure of test adequacy. Our algorithm takes four inputs. The first is a causal test case as per Definition II.2 with a treatment X, an outcome Y, and an expected causal effect. The second input is a statistical model which can be used to estimate the causal effect of X on Y, accounting for any other variables in the DAG which may need to be adjusting for. For example, this may be a linear regressor or a machine learning model. The third argument is our set of test data. The final argument is a positive integer representing the number of bootstrap samples to take.

Algorithm 1 Causal data adequacy.

```
1:  fun ADEQUACY(causalTestCase, estimator, data, nSamples)
2:      (X, Y, expected) ← causalTestCase
3:      samples ← [RESAMPLE(data) | _ ∈ [1..nSamples]]
4:      estimates = []
5:      for sample ∈ samples do
6:          estimator.train(sample)
7:          estimate = estimator.estimate(X, Y)
8:          estimates.append(estimate)
9:      kurtosis ← KURTOSIS(estimates)
10:     outcomes ← [TESTORACLE(e, expected) | e ∈ estimates]
11:     passRate ← NUMPASSING(outcomes)
12:     return (passRate, kurtosis)
```

1) Resampling (line 3): The first step is to resample the dataset the specified number of times. As discussed in Section II, this involves randomly generating samples (with replacement) that are the same size as the original dataset. These samples will be approximately the same as our original dataset, but with a few missing values and a few duplicated values, leading to slight variations in the resulting estimates. These variations can then be used to estimate accuracy measures as they should reflect the underlying population [25].

2) Effect Estimation (lines 5-8): For each of the data samples we took in line 3, we train the provided statistical model to estimate the causal effect, adjusting for other variables in the DAG if necessary. Here, we expect to be estimating the adequacy of a causal test which has already been executed. Thus, the identification of the estimand will already have been performed. The provided statistical estimator should then be the same one used to execute the test originally. In this step, we are simply retraining the model using our resampled datasets and then estimating the corresponding causal effect.

3) Kurtosis (line 9): Having estimated the causal effect for each sample, a classical application of bootstrapping would be to estimate the confidence intervals as described in Section II. However, we cannot simply use the confidence intervals as a measure of test adequacy as they do not indicate whether our model has converged or whether observing more data would change the estimates. Instead, we use the kurtosis, which looks at the shape of the distribution.

Let us examine this in the context of Figures 2 and 3. The central limit theorem [27] tells us that the distribution of our effect estimates will tend towards the normal distribution as our dataset increases in size. As this happens, the kurtosis will tend towards zero, and the model will converge on a stable estimate. If we do not have enough test data, the distribution of the sampled causal effect estimates will *not* tend towards the normal distribution no matter how many times we resample it. Instead, we will tend towards either a heavy or light tailed distribution.

A heavy tailed distribution indicates that the model has not yet converged and its estimate is affected by the few values which are duplicated and omitted from the sample. Thus, we will observe a fairly uniform spread of effect estimates. As we observe more test data, individual values will have less of an effect and our model will converge on the true effect estimate. However, this does not mean the distribution will become arbitrarily light tailed as underlying nondermininsm in the system under test will always lead to some variation in the data. A light tailed distribution indicates that our dataset is so small that we have not fully observed the underlying nondeterminism of the system as the causal effect estimates from our samples are "too similar" to each other.

4) Test Outcomes (lines 10-11): Having calculated the causal effect estimate which corresponds to each sample, we can use this to determine the corresponding pass/fail test outcome. We call this the *bootstrap pass rate*. Ideally, each of the bootstrapped samples should give the same test outcome as for the original data set. However, due to the underlying stochasticity of the systems under test, we may encounter samples with different outcomes. If many of our bootstrap samples lead to a test outcome which is different from the true outcome, this again indicates insufficient test data. We will examine the relationship between kurtosis and bootstrap pass rate as part of our evaluation in Section IV.

B. Example Application

Having defined the abstract process of our causal test adequacy metric, let us now examine how it works in the context of our running Covasim example from Section II. As in Section II-C, we will test the presence of a causal effect of *Contacts* on *Infections* in the context of the DAG in Figure 1. For the purposes of this example, say that we have

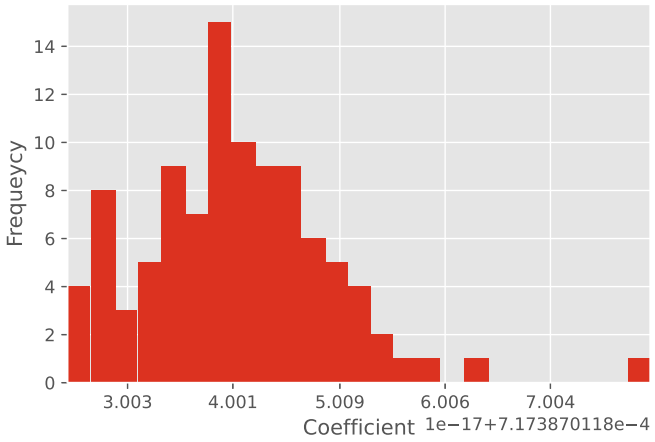


Fig. 4: The distribution of the 100 bootstrapped causal effect estimates.

run Covasim 50 times with different locations and population sizes. This data is available in our replication package [28]. When we execute our causal test case on this data, it fails. We want to calculate the causal test adequacy to work out whether there is a fault or whether we simply need to collect more data.

The first step is to resample the data with replacement (line 3). We will do this 100 times in this example. We then use each of these sampled datasets to train our statistical estimator, and then estimate the corresponding causal effect (lines 5-8). In this case, we used a linear regressor of the form $\mathit{Infections} = a \cdot \mathit{Contacts} + b \cdot \mathit{Population} + c \cdot \mathit{Susceptibility}$. Here, we include a term for our treatment *Contacts* and additional terms for (*Population*) and (*Susceptibility*) as the causal structure of the DAG in Figure 1 means we must adjust for these to properly isolate the causal effect of *Contacts* on *Infections*. We can then use the coefficient a as our causal effect estimate, as this represents the expected change in *Infections* if we increase *Contacts* by 1.

Figure 4 shows the distribution of our 100 causal effect estimates. While this does not give a smooth plot like in Figure 3, we can see that the distribution is somewhat heavy-tailed, and certainly not a bell-shaped curve. Indeed, we have a kurtosis of 2.55. We then call the test oracle on our effect estimates (line 10). In this case, we are looking for the simple presence of a causal effect. That is, the confidence intervals of each effect estimate should not contain zero. In this case, all 100 of our effect estimates lead to a failing test outcome.

On its own, the fact that both our original dataset and all 100 bootstrap samples lead to a failure might suggest a problem. However, the high kurtosis suggests that we may want to collect some additional runs of the system and rerun the test. Using a sample of 800 runs, which we can see from Figure 2 is about the point of convergence, gives a kurtosis of 0.003, corresponding to an almost perfect normal distribution. Furthermore, all 100 of the associated effect estimates, as well as the original test data, lead to a passing test outcome. Since there are no known faults in Covasim, this is the outcome we are expecting.

This shows that kurtosis can potentially be used to to judge whether we have observed enough data to be able to trust the outcome of a given causal test case, with kurtoses closer to zero indicating a more reliable outcome and kurtoses further from zero suggesting we should collect additional data values. In our evaluation in Section IV, we will investigate the extent to which this is the case.

IV. Evaluation

This section provides an evaluation of our coverage metric in the context of three *evaluative* case studies [29]. Our research questions are as follows:

1) (Fault finding) What is the relationship between causal test adequacy and fault finding?
   a) To what extent can causal test adequacy be used as a test adequacy *metric*?
   b) To what extent can causal test adequacy be used as a test adequacy *criterion*?
2) (Coverage) What is the relationship between causal test adequacy and traditional coverage-based metrics?
3) (Data) What is the relationship between causal test adequacy and the amount of test data?

RQ1 investigates the extent to which a high degree of causal test adequacy (i.e. a kurtosis close to zero and a high bootstrap pass rate) indicates that our system is fault-free. This is split into two subquestions. RQ1a investigates whether causal test adequacy can be used as a fine-grained measure of test adequacy, for example to guide a test generation tool. If this is the case, we would expect a strong correlation between kurtosis and fault finding ability. RQ1b investigates to what extent causal test adequacy can be used to classify a test set to be adequate or not. Here, we would expect tests with high fault finding capability to have a corrsponding kurtosis close to zero and tests with poor fault finding to have a kurtosis far from zero (either positive or negative).

RQ2 compares causal test adequacy with traditional code coverage to investigate whether code coverage alone can serve as a basis for judging whether we have enough test data. Given the complexity of the systems causal testing is designed for, we hypothesise that this will not be the case. Finally, RQ3 investigates how causal test adequacy changes with the amount of test data. Here, we expect kurtosis to tend towards zero as we observe increasingly many data points.

A. Subject Systems

We carry out our evaluation in the context of three subject systems. For each of these, we constructed a causal DAG to represent a realistic scenario involving some or all of the inputs and outputs of the system.

Poisson Line Tessellation. The Poisson line tessellation (PLT) model [30] uses a Poisson process to generate a series of lines that are positioned and oriented at random within a given area of interest, referred to as the *sampling*

window, to form a tessellation. The model takes three numerical input parameters: the width (W) and height (H) of the sampling window, and the intensity (I) of the process, with higher intensities leading to more lines. The intersections of these lines then form polygons. The model has four main outputs: the total numbers of lines (L_t) and polygons (P_t) within the sampling window, and the numbers of lines (L_u) and polygons (P_u) per unit area of the sampling window.

We selected this model because it has been the subject of prior research on causal testing [1]. In addition, Poisson processes are commonly used to model random processes for a range of important applications, including simulating road networks [31], [32] and modelling photon arrival in 3D imaging [33]. Here, we consider the same testing scenario considered in [1], which has the DAG shown in Figure 5.

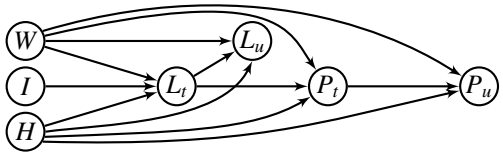


Fig. 5: Causal DAG of the PLT model.

Covasim. Covasim is the epidemiological agent-based model that was introduced as a motivating example in Section II. It is primarily used to simulate detailed COVID-19 scenarios to evaluate the impact of various interventions, such as vaccination and contact tracing [16], in specific demographics. These scenarios are configured via 64 input parameters and described by 56 time-series outputs. It has been used to inform important policy decisions across a range of countries, including the UK, US, and Australia [34], [18], [20], [35]. Here, we cover the testing scenario from the motivating example from Section II. Like the PLT model, Covasim has already been used in prior research on causal testing [1]. For our evaluation, we use the testing scenario from Section II, which has the DAG shown in Figure 1.

Spanish Flu Model. The Spanish Flu model [36] is an equation-based susceptible-infected-recovered (SIR) epidemiological model, which predicts how the numbers of individuals susceptible to a disease, infected with it, and recovered (or dead) evolve over the course of a pandemic. It has 14 parameters, 7 of which change over time. Because the model is equation based, there is no stochasticity, so it is deterministic for any given set of parameters. This makes the model very different to Covasim, but still challenging to test since the outcomes for any given set of parameters are difficult to predict, and the precise values of the expected outcomes (or even the parameters) for any given disease may not be known. We chose this particular model as it has previously been used in testing research [36], which we could use as a basis to draw our causal DAG, shown in Figure 6, in which I is the total number of infections and the other nodes represent various inputs of the model.

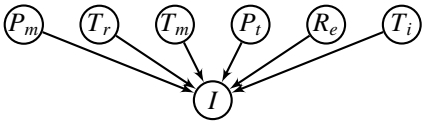


Fig. 6: Causal DAG of the Spanish Flu model.

These systems all present a significant testing challenge in that they are nondeterministic systems which implement complex behaviour that cannot be easily predicted from the input configuration. This puts them firmly in the category of systems that causal testing was designed for. Furthermore, they are all well used by existing testing literature and sufficiently well-documented and intuitive that we could draw causal graphs for our respective testing scenarios.

B. Methodology

Our evaluation consists of two studies[2]. The first study concerns RQ1 and examines the relationship between the causal adequacy of a test suite and its ability to detect faults in the context of the same three subject systems. The second study concerns RQs 2 and 3, and examines the relationship between our adequacy metric, the number of system executions, and code coverage in the context of the three subject systems discussed above. To facilitate our evaluation, we implemented Algorithm 1 as a module within the causal testing framework [1], which we use as a platform to carry out our causal testing. Our code for this can be found at [12].

Both of our studies required a suite of causal tests for each of our three systems. To generate our causal tests, we used the tool from [2] to systematically generate a causal test for each pair of nodes in the respective DAG. For pairs with an $X \rightarrow Y$ edge connecting them, we generated a causal test to validate either a positive or negative causal effect of treatment X on outcome Y. For unconnected pairs of nodes, we generated a causal test to validate the *absence* of such a causal effect.

Our two studies then involve evaluating the causal tests using different datasets. As discussed in Section II, this involves training a statistical model to estimate the causal effect. In all cases, we used a regression model and used the coefficient of the treatment variable as our causal effect measure. Since our goal is to evaluate our causal test adequacy metric rather than to precisely test each of our subject systems, we did not spend time precisely tailoring models for each subject system.

1) Study 1: Investigating the relationship between causal test adequacy and fault finding: To answer RQ1, we carried out mutation analysis using the python package `cosmic_ray` to generate mutants for each system. For each mutant, we collected data sets of varying sizes, and used these to evaluate

[2]Code and data for our replication package can be found at [28]. Our implementation of causal test adequacy forms part of the causal testing framework [12].

our causal tests. We collected datasets in increments of 10 between 20 and 100 runs, and then in increments of 100 between 100 and 1000 runs. For the Spanish flu model, we also collected datasets in increments of 1000 between 1000 and 10000 runs as even 1000 points was not enough to reach convergence.

The values of our input parameters were sampled uniformly from a range of realistic values for each system. The precise ranges can be seen in our replication package [28]. To mitigate for the effect of stochasticity, we used 30 data sets of each size.

We then evaluated our causal test suite using each data set from each mutant and calculated the mutation score associated with each data set. For each causal test, we calculated both the kurtosis and bootstrap pass rate, using 100 bootstrap samples. Since our systems are stochastic, it is possible for tests to fail on the original program, especially for small datasets. To prevent such tests from artificially inflating our mutation score, we only considered tests which passed on the original program and failed on the mutant. This is discussed in more detail in Section IV-D.

To answer RQ1a, we performed a Spearman ρ test to measure the correlation between kurtosis and mutation score across all three systems. This gives a value between -1 and 1, with -1 representing a perfect negative correlation, 1 representing a perfect positive correlation, and 0 representing no correlation. We used this test as it is non-parametric, i.e. it does not make the assumption that the underlying data is normally distributed. Here, we had to take an average kurtosis across all of the tests in the test suite for each data set since a mutation score only applies to test suites[3]. This essentially tells us the relationship between the average kurtosis of a test suite and its ability to detect faults. For kurtoses below zero, we expect a positive correlation such that increasing the kurtosis improves fault finding ability. For positive kurtoses, we expect a negative correlation such that a kurtosis of zero gives the highest fault finding capability.

To answer RQ1b, we performed another Spearman ρ test, this time measuring the correlation between the kurtosis and the number failing tests in the test suite. We measure this separately for tests which fail on the unmutated program (i.e. false failures) and on the mutatant (i.e. true failures). This tells us whether we can use kurtosis as a way of interpreting a failing test such that we can direct testing effort towards either collecting additional test data or investigating a potential fault.

2) Study 2: Examining the relationship between causal test adequacy, coverage, and the number of test executions: To answer RQs 2 and 3, we collected random data sets of different sizes, as per the previous study, for each SUT. We measured the statement coverage associated with each data set and then used it to evaluate the causal test suite, calculating the causal test adequacy for each causal test, again using a bootstrap size of 100. To mitigate for the stochasticity of the subject systems and data generation, we again collected 30 datasets of each size with different random seeds.

C. Results and Discussion

1) RQ1 (Fault finding): For kurtoses below zero, we have the expected positive correlation of 0.23 with p-value of 8.01e-7 (both to 3sf), so we can expect to discover more mutants as the kurtosis increases towards zero. For kurtoses above zero, we have correlation coefficient of -0.168 and a p-value of 1.33e-10 (both to 3 sf.), so we will discover less mutants as we continue to increase kurtosis beyond zero.

While these results are as expected, the correlations are weak, and it is possible to achieve a very low mutation score even with a kurtosis very close to zero. It is therefore worth noting that kurtosis, like any test adequacy metric, can only be taken as an *indication* of the quality of a test set rather than a direct measure. One contributing factor here may be our test oracles. Simply looking for positive or negative causal effects may not detect every mutant. For example, changing an addition operation to a multiplication would still maintain a positive relationship between variables, so the causal test would still pass and the mutant would not be discovered.

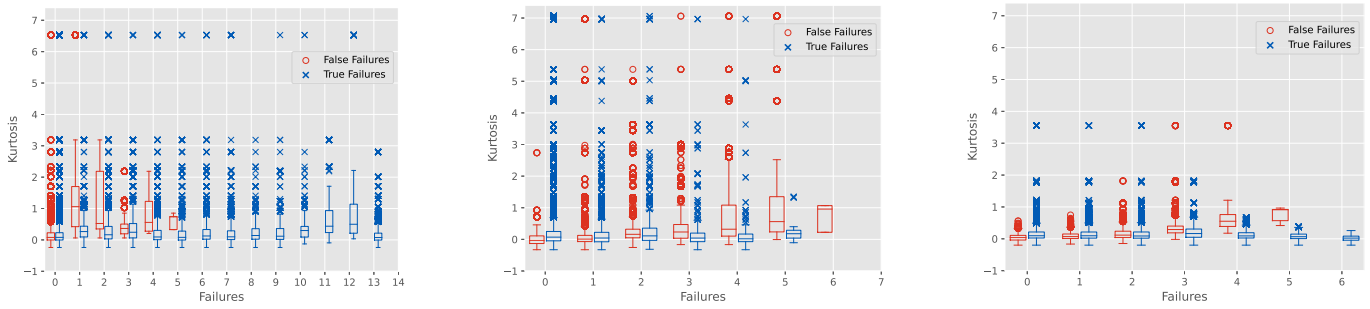
Another contributing factor is the fact that we removed tests which failed on the original program (which we call "false failures") when calculating the mutation score. Such tests cannot then be used to detect mutants. A detailed examination of our results suggests that this happens more for smaller data sets with larger kurtoses, but can still happen even with a kurtosis close to zero, especially for causal independence tests and when the causal effect between variables is weak. It is therefore useful to examine how kurtosis affects these false failures.

Figure 7 shows how the number of failing tests in each test suite varies with the (averaged) kurtosis. Here, a "false failure" is a test which has failed on the unmutated version of the program, so cannot have detected a fault. This can happen due to the stochastic nature of the systems under test and the fact that tests are therefore statistical properties over multiple runs of the system. A "true failure" is a test which has failed on the mutant, so has detected the fault. The plot shows two noteworthy results. Firstly, it shows a positive correlation between the kurtosis and the number of false failures. The Spearman ρ test gives a coefficient of 0.244 with a p-value of 0.0[4]. Secondly, it shows that there is negligible correlation between the kurtosis and the number of true failures (i.e. tests failing when there is a fault in the system), with the median kurtosis remaining around zero. Spearman ρ gives a coefficient of 0.049 with p-value=4.86e-226.

This result is interesting because it shows us how kurtosis can be used to help interpret a failing test and direct subsequent testing effort. In essence, if a test fails and has a high kurtosis, this indicates that the failure may not necessarily be genuine and that it is worthwhile collecting additional test runs to help

[3]Other aggregations such as the maximum, minimum, and mean absolute value led to similar results.

[4]While $p = 0$ is theoretically impossible, it indicates that the p-value is too small to be represented using floating point numbers.



(a) PLT (b) Covasim (c) Spanish Flu

Fig. 7: Mutation score vs causal test adequacy for each subject system.

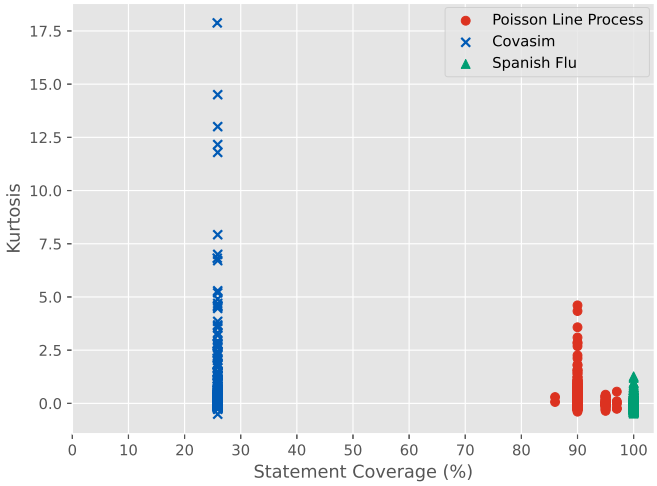


Fig. 8: Coverage vs kurtosis for each subject system.

increase the adequacy of the test set. Conversely, if the kurtosis is close to zero, collecting additional runs of the system may still change the test outcome, but it is more likely that the failure is genuine and that there is a problem with either the system or the test.

> There is a statistically significant correlation between kurtosis and mutation score. Kurtoses close to zero have better fault detection capability. There is also a statistically significant, and stronger correlation between kurtosis and the number of false failures in a test suite, such that the higher the kurtosis, the more likely it is that a failure is a result of insufficient data rather than a genuine fault.

2) RQ2 (Coverage): Figure 8 shows statement coverage plotted against causal test adequacy for the three systems. We again take the average causal adequacy of each test case in the respective causal test suites, since each test was executed with the same test data. A striking result here is that statement coverage remains almost constant, meaning there is clearly no relationship with causal test adequacy.

This is not necessarily as surprising as it initially seems because each of our testing scenarios, represented by the respective causal DAGs and the test input distribution, can only exercise a certain amount of program functionality. Once this has been achieved, further runs of the software will merely execute the same code with different data values, so will not affect statement coverage but may still significantly change causal test adequacy as additional data values may change the causal effect estimate.

This is especially true for Covasim, which is extremely stochastic, meaning the estimated causal effect can vary wildly. When we resample test datasets, especially small ones, we get a wide and uniform spread of causal effect estimates, leading to the broad spread of kurtoses we see in Figure 8.

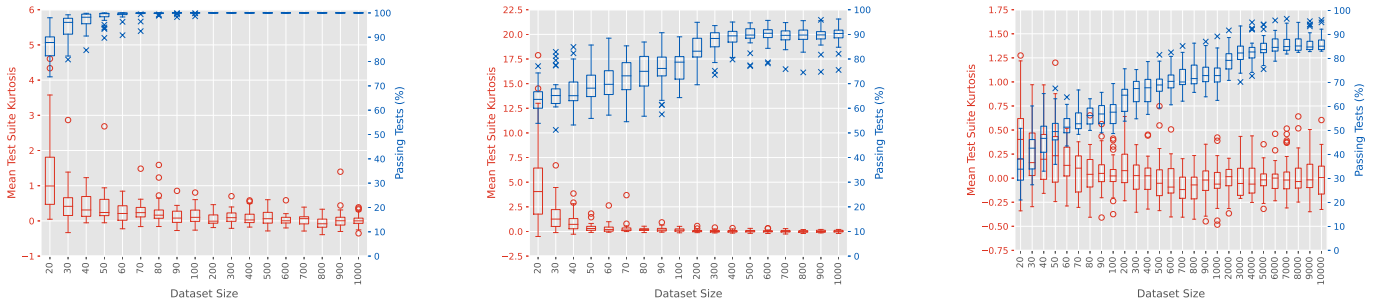
In Figure 8, negative kurtoses are fairly rare, only occurring in about a third of cases on average. This is because they only occur for samples with very similar values. Since our test inputs were generated uniformly, such samples are relatively uncommon, especially for larger amounts of of test data.

> There does not appear to be a relationship between statement coverage and causal test adequacy. This suggests that statement coverage is not a good indicator as to whether we have collected enough data for causal testing.

3) RQ3 (Data): Figure 9 shows how causal test adequacy varies with dataset size for each subject system. Again, causal adequacy is averaged for each causal test suite. Here, we can see that kurtosis tends towards zero as we increase the size of the dataset. This represents our model converging on a stable estimate, as in Figure 2. Once we reach this point, additional test executions do not significantly change causal adequacy.

The number of data points that are required to achieve this convergence is naturally dependent on the system. For the PLT model, this happens at around 200 data points. For our Covasim scenario, this happens at around 60 data points. The Spanish Flu model needs a larger number of data points, only converging at 5000 runs.

Figure 9 also shows the bootstrap pass rate for each subject system (again averaged for each test suite). That is, the number



(a) PLT (b) Covasim (c) Spanish Flu

Fig. 9: Dataset size vs causal test adequacy for each subject system.

of bootstrap samples for which the inferred model led to a passing outcome. Here, we are looking for all tests to pass since there are no (known) faults in the program. We see that as the kurtosis tends towards zero, the percentage of passing tests tends towards 100%, and that the two measures converge at a similar number of observed executions.

This result ties in nicely to Weyuker's inference adequacy [11], which deems a test suite adequate if it enables the inference of a model which matches the behaviour of both the specification and the implementation. Here, the bootstrap pass rate effectively gives us a relatively fine-grained way to assess whether the test data matches the specification, i.e. whether the expected causal effect is present.

> Kurtosis tends towards zero as we observe increasingly many test executions. At the same time, the bootstrap pass rate tends towards 100%. The two measures appear to converge after similar numbers of data points, after which additional executions no longer improve adequacy. However, the precise number of data points required varies greatly between systems.

D. Threats to Validity

1) Threats to internal validity: In study 1, our mutation scores ignore tests which failed on the original program. This likely reduces our overall mutation scores as it reduces the number of tests which can be used to detect a mutant, potentially affecting the correlation between mutation score and test adequacy. However, the alternative to this would be to include such tests, which would have falsely inflated our mutation scores, again potentially affecting the correlation between kurtosis and mutation score.

In both studies 2 and 3 our test inputs were generated uniformly. This gives rise to the risk that different distributions could have led to differently sized data sets with different coverage. We opted for the uniform distribution because this mitigates the risk of cherry-picked data values biasing results, and amounts to the tests that would be generated by conventional random testing [37].

2) Threats to external validity: In studies 2 and 3, our subject systems were published computational models written in Python. While the three systems have very different domains, causal structures and implementation styles, our results were consistent across all three systems. In our future work we will investigate other types of systems to ensure that the results generalise.

Our conclusions from RQ1 (relationship with fault finding ability) are based on the use of mutation analysis. For this we used the Cosmic Ray mutation engine. This gives rise to the threat that the mutants used here are not reflective of genuine faults. One focus of future work would be to replicate this study on curated sets of real faults, such as Defects4J [38].

V. Related Work

There is a plethora of software test adequacy metrics in the literature [39]. Many of these are related to code elements, with the general approach being to cover some sort of graph representation. While Weyuker has presented seven axioms which all test adequacy metrics should follow [40], several of these are specifically related to implementational elements of a program, so cannot be applied here.

In the context of testing scientific models, techniques such as sensitivity [41] and uncertainty analysis [42] are often employed, but these techniques do not solve the problem of determining how many test inputs are sufficient, and can require prohibitively many test executions [43]. While surrogate modelling [44], [45] can provide an economical way of approximating the output of the system, thereby enabling more test inputs to be run, the problem of judging how many test inputs to run still remains. Causal test adequacy addresses this problem by providing an estimate as to whether sufficiently many runs have been observed that additional runs are not likely to change the output of the inferred model.

In the field of health economics, providing evidence of model convergence is a key issue. This is typically addressed by collecting a predefined "large number" of data points such as 1000 or 10000 [23], [24]. One technique [46] suggests using the confidence intervals around the mean outcome, with the dataset being deemed to be sufficient once these confidence intervals no longer contain zero. However, as discussed in Section II, this does not indicate whether the

model has reached the point of convergence. Our causal test adequacy metric directly tackles this, and provides an estimate of whether convergence has been reached.

In this paper, and other works on causal testing [1], [2], it is assumed that causal DAGs are specified manually by a domain expert. While this is widely accepted in fields such as epidemiology and social sciences, there are two potential methods that could, in theory, (partially) automate this process. Firstly, causal DAGs can be generated using static analysis of source code [47], [48]. Secondly, the field of *causal discovery* (CD) [49] provides methods to infer causal structures from data by exploiting asymmetries that distinguish association from causation [50]. Furthermore, one work [51] places a bound on the number of samples needed for this. However, they only do so in the context of generalisation rather than causal discovery. Additionally, for all of these techniques, it must be noted that any faults in the source code would also be reflected in the DAG, meaning the correctness DAG would still need to be validated.

As with Weyuker's inference adequacy [11], causal test adequacy is based on the ability to infer a model from test data. This aspect of our work relates to a significant body of work on machine learning approaches for inferring models from test executions. Most of these techniques work in the context of regression testing, where the inferred model represents correct behaviour which can then be used to identify any faults arising in subsequent software versions. The inferred models include standard linear regression [52], state machines [53], and decision trees [54] amongst others. What distinguishes our work from these techniques is that, as with Weyuker's original work [11], we are specifically focussed on providing an estimate of the adequacy of the test set. However, where [11] seeks to infer an *accurate* behavioural model, causal test adequacy seeks to infer a *stable* one. This enables our technique to judge the adequacy of tests which assert that certain aspects of the program should *not* be related, in which cases the inference of an accurate model should not be possible, thereby limiting the applicability of inference adequacy.

VI. Conclusion

A critical question when testing software systems is "at what point can we stop testing?" [6]. For nondeterministic, hard to test software such as computational models, this question is particularly challenging to answer since there is not a one to one correspondence between tests and system executions. Instead, tests tend to be phrased as statistical properties over multiple executions, with the same set of executions potentially being used to evaluate multiple tests [1]. This, in addition to the relatively poor correlation with the ability to expose faults [9], [10], makes traditional coverage based metrics inappropriate here.

In this paper, we have presented a causality-driven test adequacy metric which indicates whether we have observed enough program executions to enable model convergence. That is, whether the observation of additional program executions would change our estimated causal effect. We do this by measuring the kurtosis of the causal effect estimates calculated using bootstrap samples of the test data, with a kurtosis of zero being optimal. We also measure and report the number of these bootstrapped estimates which lead to a passing test outcome.

Our evaluation suggests that, as we observe more executions of the software, the kurtosis converges on zero, and the bootstrap pass rate approaches 100%. However, there is no relationship with code coverage which, in our systems at least, remained more or less constant no matter how many runs are observed. There is a statistically significant correlation between kurtosis and mutation score, implying that test suites with a kurtosis closer to zero are better able to find faults. Furthermore, there is also a significant positive correlation between the kurtosis of a test suite and the number of tests that fail because of a lack of data, as opposed to the presence of an actual fault. Tests with a kurtosis closer to zero indicate that the failure is more likely to be genuine. This gives an indication as to whether testing effort is better spent investigating a potential fault or simply collecting additional runs of the software.

In future work, we plan to examine how causal test adequacy can be used to more efficiently generate datasets for causal testing. For example, the datasets we used in our evaluation were of a fixed size and sampled uniformly. Instead, we could pair our technique with some form of adaptive random testing [55] to generate datasets which give close to zero kurtosis for all causal tests in a test suite. We could also include a search for boundary values [56] to help ensure we sufficiently exercise all of a system's functionality.

Currently, causal test adequacy evaluates the suitability of a dataset to evaluate a single causal test. A desirable line of future work would be to investigate ways to aggregate this to provide the user with a single, meaningful value which covers the entire DAG (i.e. causal test suite). We believe this would improve the intuitive usability of causal test adequacy, especially if we could provide a percentage adequacy in line with common traditional metrics.

Furthermore, we intend to investigate the applications of this metric beyond causal, or indeed software, testing as it essentially presents a way of judging model convergence. For example, this seems to be an important problem in the field of health economics, as their current state of the art [46] admits to being "essentially arbitrary". With a more robust mathematical analysis, our causal test adequacy could potentially provide a more systematic solution.

References

[1] A. G. Clark, M. Foster, B. Prifling, N. Walkinshaw, R. M. Hierons, V. Schmidt, and R. D. Turner, "Testing causality in scientific modelling software," *ACM Trans. Softw. Eng. Methodol.*, 2023, just Accepted.

[2] A. G. Clark, M. Foster, N. Walkinshaw, and R. M. Hierons, "Metamorphic testing with causal graphs," in *2023 IEEE Conference on Software Testing, Verification and Validation (ICST)*. IEEE, 2023, pp. 153–164.

[3] C. M. Poskitt, Y. Chen, J. Sun, and Y. Jiang, "Finding causally different tests for an industrial control system," in *Proceedings of the 45th International Conference on Software Engineering*, ser. ICSE '23. IEEE Press, 2023, p. 2578–2590. [Online]. Available: https://doi.org/10.1109/ICSE48619.2023.00215

[4] L. Giamattei, R. Pietrantuono, and S. Russo, "Reasoning-based software testing," 2023. [Online]. Available: https://arxiv.org/abs/2303.01302

[5] E. J. Weyuker, "On testing non-testable programs," *The Computer Journal*, vol. 25, no. 4, pp. 465–470, 1982. [Online]. Available: https://doi.org/10.1093%2Fcomjnl%2F25.4.465

[6] P. G. Frankl and E. J. Weyuker, "An applicable family of data flow testing criteria," *IEEE Transactions on Software Engineering*, vol. 14, no. 10, pp. 1483–1498, 1988.

[7] A. Arcuri and L. Briand, "Adaptive random testing: An illusion of effectiveness?" in *Proceedings of the 2011 International Symposium on Software Testing and Analysis*, 2011, pp. 265–275.

[8] M. Ivanković, G. Petrović, R. Just, and G. Fraser, "Code coverage at google," in *Proceedings of the 2019 27th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2019, pp. 955–963.

[9] L. Inozemtseva and R. Holmes, "Coverage is not strongly correlated with test suite effectiveness," in *Proceedings of the 36th international conference on software engineering*, 2014, pp. 435–445.

[10] T. T. Chekam, M. Papadakis, Y. Le Traon, and M. Harman, "An empirical study on mutation, statement and branch coverage fault revelation that avoids the unreliable clean program assumption," in *2017 IEEE/ACM 39th International Conference on Software Engineering (ICSE)*. IEEE, 2017, pp. 597–608.

[11] E. J. Weyuker, "Assessing test data adequacy through program inference," *ACM Trans. Program. Lang. Syst. Transactions on Programming Languages and Systems*, vol. 5, no. 4, pp. 641–655, 1983.

[12] C. Wild, M. Foster, N. Walkinshaw, A. Clark, F. Allian, and R. Somers, "CITCOM Software Release," 2023. [Online]. Available: https://orda.shef.ac.uk/articles/software/CITCOM_Software_Release/24427516

[13] J. Pearl, *The book of why : the new science of cause and effect*. London, UK: Penguin Books, 2019.

[14] S. F. O'Brien and Q. L. Yi, "How do i interpret a confidence interval?" *Transfusion*, vol. 56, no. 7, pp. 1680–1683, 2016.

[15] M. A. Hernán and J. M. Robins, *Causal Inference: what if*. Boca Raton: Chapman & Hall/CRC, 2020.

[16] C. C. Kerr, R. M. Stuart, D. Mistry, R. G. Abeysuriya, K. Rosenfeld, G. R. Hart, R. C. Núñez, J. A. Cohen, P. Selvaraj, B. Hagedorn *et al.*, "Covasim: an agent-based model of COVID-19 dynamics and interventions," *PLOS Computational Biology*, vol. 17, no. 7, p. e1009149, 2021.

[17] Institute for Disease Modelling, "Covasim," https://github.com/InstituteforDiseaseModeling/covasim, 2022.

[18] C. C. Kerr, D. Mistry, R. M. Stuart, K. Rosenfeld, G. R. Hart, R. C. Núñez, J. A. Cohen, P. Selvaraj, R. G. Abeysuriya, M. Jastrzębski *et al.*, "Controlling COVID-19 via test-trace-quarantine," *Nature Communications*, vol. 12, no. 1, pp. 1–12, 2021.

[19] J. A. Cohen, D. Mistry, C. C. Kerr, and D. J. Klein, "Schools are not islands: Balancing covid-19 risk and educational benefits using structural and temporal countermeasures," *medRxiv*, 2020.

[20] J. Panovska-Griffiths, C. C. Kerr, R. M. Stuart, D. Mistry, D. J. Klein, R. M. Viner, and C. Bonell, "Determining the optimal strategy for reopening schools, the impact of test and trace interventions, and the risk of occurrence of a second COVID-19 epidemic wave in the uk: a modelling study," *The Lancet Child & Adolescent Health*, vol. 4, no. 11, pp. 817–827, 2020.

[21] N. Scott, A. Palmer, D. Delport, R. Abeysuriya, R. Stuart, C. C. Kerr, D. Mistry, D. J. Klein, R. Sacks-Davis, K. Heath *et al.*, "Modelling the impact of reducing control measures on the COVID-19 pandemic in a low transmission setting," *Med J Aust*, vol. 214, no. 2, pp. 79–83, 2020.

[22] E. Bareinboim and J. Pearl, "Causal inference and the data-fusion problem," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 113, no. 27, pp. 7345–7352, 2016. [Online]. Available: https://www.jstor.org/stable/26470690

[23] A. Hatswell, A. Bullement, M. Paulden, and M. Stevenson, "Probabilistic sensitivity analysis in health economic models; how many simulations should we run?" *Value in Health*, vol. 20, no. 9, p. A746, 2017. [Online]. Available: https://doi.org/10.1016%2Fj.jval.2017.08.2074

[24] A. H. Briggs and A. M. Gray, "Handling uncertainty when performing economic evaluation of healthcare interventions," *Health Technology Assess*, vol. 3, no. 2, pp. 1–134, 1999.

[25] C. Z. Mooney, R. D. Duval, and R. Duvall, *Bootstrapping: A nonparametric approach to statistical inference*. sage, 1993, no. 95.

[26] L. T. DeCarlo, "On the meaning and use of kurtosis," *Psychological Methods*, vol. 2, no. 3, pp. 292–307, 1997.

[27] H. Fischer, *A History of the Central Limit Theorem*. New York: Springer, 2011.

[28] M. Foster, C. Wild, R. Hierons, and N. Walkinshaw, "Causal Test Adequacy," 2023. [Online]. Available: https://orda.shef.ac.uk/articles/dataset/Causal_Test_Adequacy/24422104

[29] P. R. et al., "Empirical standards for software engineering research," 2021.

[30] R. Guderlei and J. Mayer, "Statistical metamorphic testing testing programs with random output by means of statistical hypothesis tests and metamorphic testing," in *Seventh International Conference on Quality Software (QSIC 2007)*. IEEE, 2007, pp. 404–409.

[31] V. V. Chetlur and H. S. Dhillon, "Coverage analysis of a vehicular network modeled as cox process driven by poisson line process," *IEEE Transactions on Wireless Communications*, vol. 17, no. 7, pp. 4401–4416, 2018.

[32] F. Morlot, "A population model based on a poisson line tessellation," in *2012 10th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt)*. IEEE, 2012, pp. 337–342.

[33] D. Shin, A. Kirmani, A. Colaço, and V. K. Goyal, "Parametric poisson process imaging," in *2013 IEEE Global Conference on Signal and Information Processing*, IEEE. IEEE, 2013, pp. 1053–1056.

[34] J. Panovska-Griffiths, C. C. Kerr, W. Waites, R. M. Stuart, D. Mistry, D. Foster, D. J. Klein, R. M. Viner, and C. Bonell, "The potential contribution of face coverings to the control of sars-cov-2 transmission in schools and broader society in the uk: a modelling study," 2020.

[35] R. M. Stuart, R. G. Abeysuriya, C. C. Kerr, D. Mistry, D. J. Klein, R. Gray, M. Hellard, and N. Scott, "The role of masks in reducing the risk of new waves of covid-19 in low transmission settings: a modeling study," 2020.

[36] L. L. Pullum and O. Ozmen, "Early results from metamorphic testing of epidemiological models," in *2012 ASE/IEEE International Conference on BioMedical Computing (BioMedCom)*. IEEE, 2012.

[37] R. Hamlet, "Random testing," 2002. [Online]. Available: https://doi.org/10.1002%2F0471028959.sof268

[38] R. Just, D. Jalali, and M. D. Ernst, "Defects4j: a database of existing faults to enable controlled testing studies for java programs," in *Proceedings of the 2014 International Symposium on Software Testing and Analysis*. New York: ACM, 2014.

[39] H. Zhu, P. A. V. Hall, and J. H. R. May, "Software unit test coverage and adequacy," *ACM Comput. Surv.*, vol. 29, no. 4, p. 366–427, 1997.

[40] E. J. Weyuker, "Axiomatizing software test data adequacy," *IIEEE Trans. Software Eng. Transactions on Software Engineering*, vol. SE-12, no. 12, pp. 1128–1138, 1986.

[41] J. E. Oakley and A. O'Hagan, "Probabilistic sensitivity analysis of complex models: a bayesian approach," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 66, no. 3, pp. 751–769, 2004.

[42] I. Farajpour and S. Atamturktur, "Error and uncertainty analysis of inexact and imprecise computer models," *Journal of Computing in Civil Engineering*, vol. 27, no. 4, pp. 407–418, 2013.

[43] S. Conti and A. O'Hagan, "Bayesian emulation of complex multi-output and dynamic computer models," *Journal of statistical planning and inference*, vol. 140, no. 3, pp. 640–651, 2010.

[44] E. T. Chang, M. Strong, and R. H. Clayton, "Bayesian sensitivity analysis of a cardiac cell model using a Gaussian process emulator," *PloS one*, vol. 10, no. 6, p. e0130252, 2015.

[45] I. Vernon, M. Goldstein, and R. Bower, "Galaxy Formation: Bayesian History Matching for the Observable Universe," *Statistical Science*, vol. 29, no. 1, pp. 81 – 90, 2014. [Online]. Available: https://doi.org/10.1214/12-STS412

[46] A. J. Hatswell, A. Bullement, A. Briggs, M. Paulden, and M. D. Stevenson, "Probabilistic sensitivity analysis in cost-effectiveness models: Determining model convergence in cohort models," *PharmacoEconomics*, vol. 36, no. 12, pp. 1421–1426, 2018. [Online]. Available: https://doi.org/10.1007%2Fs40273-018-0697-3

[47] A. Podgurski and Y. Küçük, "Counterfault: Value-based fault localization by modeling and predicting counterfactual outcomes," in *2020 IEEE International Conference on Software Maintenance and Evolution (ICSME)*. IEEE, 2020, pp. 382–393.

[48] S. Lee, D. Binkley, R. Feldt, N. Gold, and S. Yoo, "Causal program dependence analysis," 2021.

[49] D. Malinsky and D. Danks, "Causal discovery algorithms: A practical guide," *Philosophy Compass*, vol. 13, no. 1, p. e12470, 2018.

[50] C. Glymour, K. Zhang, and P. Spirtes, "Review of causal discovery methods based on graphical models," *Frontiers in genetics*, vol. 10, p. 524, 2019.
[51] T. Kyono, Y. Zhang, and M. van der Schaar, "Castle: Regularization via auxiliary causal graph discovery," 2020.
[52] A. Arrieta, J. Ayerdi, M. Illarramendi, A. Agirre, G. Sagardui, and M. Arratibel, "Using machine learning to build test oracles: an industrial case study on elevators dispatching algorithms," in *2021 IEEE/ACM International Conference on Automation of Software Test (AST)*. IEEE, 2021, pp. 30–39.
[53] N. Walkinshaw, R. Taylor, and J. Derrick, "Inferring extended finite state machine models from software executions," *Empirical Software Engineering*, vol. 21, pp. 811–853, 2016.
[54] L. C. Briand, Y. Labiche, Z. Bawar, and N. T. Spido, "Using machine learning to refine category-partition test specifications and test suites," *Information and Software Technology*, vol. 51, no. 11, pp. 1551–1564, 2009.
[55] T. Y. Chen, F.-C. Kuo, R. G. Merkel, and T. Tse, "Adaptive random testing: The ART of test case diversity," *Journal of Systems and Software*, vol. 83, no. 1, pp. 60–66, 2010. [Online]. Available: https://doi.org/10.1016%2Fj.jss.2009.02.022
[56] F. Dobslaw, R. Feldt, and F. de Oliveira Neto, "Automated black-box boundary value detection," 2022.