Fast and Powerful Conditional Randomization Testing via Distillation

Molei Liu*[1], Eugene Katsevich[2], Lucas Janson[3], and Aaditya Ramdas[4]

[1]Department of Biostatistics, Harvard Chan School of Public Health
[2]Statistics Department, Wharton School of the University of Pennsylvania
[3]Department of Statistics, Harvard University
[4]Departments of Statistics & Machine Learning, Carnegie Mellon University

Abstract

We consider the problem of conditional independence testing: given a response Y and covariates (X, Z), we test the null hypothesis that $Y \perp\!\!\!\perp X \mid Z$. The conditional randomization test (CRT) was recently proposed as a way to use distributional information about $X \mid Z$ to exactly (non-asymptotically) control Type-I error using any test statistic in any dimensionality without assuming anything about $Y \mid (X, Z)$. This flexibility in principle allows one to derive powerful test statistics from complex prediction algorithms while maintaining statistical validity. Yet the direct use of such advanced test statistics in the CRT is prohibitively computationally expensive, especially with multiple testing, due to the CRT's requirement to recompute the test statistic many times on resampled data. We propose the distilled CRT, a novel approach to using state-of-the-art machine learning algorithms in the CRT while drastically reducing the number of times those algorithms need to be run, thereby taking advantage of their power and the CRT's statistical guarantees without suffering the usual computational expense. In addition to distillation, we propose a number of other tricks like screening and recycling computations to further speed up the CRT without sacrificing its high power and exact validity. Indeed, we show in simulations that all our proposals combined lead to a test that has similar power to the most powerful existing CRT implementations but requires orders of magnitude less computation, making it a practical tool even for large data sets. We demonstrate these benefits on a breast cancer dataset by identifying biomarkers related to cancer stage.

Keywords: Conditional randomization test (CRT), model-X, conditional independence testing, high-dimensional inference, machine learning.

1 Introduction

In our increasingly data-driven world, it has become the norm in applications from genetics and health care to policy evaluation and e-commerce to seek to understand the relationship between a response variable of interest and a high-dimensional set of potential explanatory variables or covariates. While accurately estimating this entire relationship generally would require a nearly-infinite sample size, a less-intractable but still extremely useful question is to ask, for any given covariate, whether it actually contributes to the response variable's high-dimensional conditional

*This paper is a post-hoc merge of two parallel works by Liu & Janson (2020) and Katsevich & Ramdas (2020).

distribution. We address this problem by encoding a covariate's relevance as a conditional independence test, which requires no modeling assumptions to define. Denoting the response random variable by Y, a given covariate of interest by X, and a multidimensional set of further covariates by $Z = (Z_1, \dots, Z_p)$, the null hypothesis we seek to test is

$$H_0 : Y \perp\!\!\!\perp X \mid Z$$

against the alternative $H_1 : Y \not\perp\!\!\!\perp X \mid Z$. Testing this hypothesis for just a single covariate is sometimes all that is needed, such as in an observational study investigating whether a particular treatment (X) causes a change in a response (Y) after controlling for a set of measured confounding variables (Z). But in other applications no one covariate holds a priori precedence over another, and a researcher seeks any and all covariates that contribute to Y's conditional distribution. This variable selection objective can also be achieved by testing H_0 for each covariate in turn (with Z containing the other covariates) and plugging the resulting p-values into one of the many procedures from the extensive literature on multiple testing. In addition to the considerable statistical challenge of providing a valid and powerful test of H_0, it is of paramount importance to also ensure that test is computationally efficient, especially, as is often the case in modern applications, when either or both the sample size and dimension are large, and even more so when a variable selection objective requires the test to be run many times. Thus the goal of this paper is to present a test for conditional independence that is provably valid, empirically powerful, and computationally efficient, when the distribution of $X|Z$ is known or can be well approximated.

1.1 Background

Our work builds on the conditional randomization test (CRT) introduced in Candès et al. (2018). The CRT is a general framework for conditional independence testing that can use any test statistic one chooses and exactly (non-asymptotically) control the Type-I error regardless of the data dimensionality. The CRT's guarantees assume nothing whatsoever about $Y \mid (X, Z)$, but instead assume $X \mid Z$ is known. This so-called "model-X" framework — in contrast to the canonical approach of assuming a strong model for $Y \mid (X, Z)$ — is perhaps easiest to justify when a wealth of unlabeled data (pairs (X_i, Z_i) without corresponding Y_i) is available, but has also been found to be quite robust even when $X \mid Z$ is estimated using only the labeled data.

In order to define the CRT, we first need notation for our data. For $i \in \{1, \dots, n\}$, let $(Y_i, X_i, Z_i) \in \mathbb{R}^{p+2}$ be i.i.d. copies of (Y, X, Z), and denote the column vector of Y_i's by $\boldsymbol{y} \in \mathbb{R}^n$, the column vector of X_i's by $\boldsymbol{x} \in \mathbb{R}^n$, and the matrix whose rows are the Z_i's by $\boldsymbol{Z} \in \mathbb{R}^{n \times p}$. The CRT is given by Algorithm 1, and its Type-I error guarantee follows below.

Algorithm 1 The conditional randomization test (CRT).

```
Input: The distribution of $\boldsymbol{x} \mid \boldsymbol{Z}$, data $(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{Z})$, test statistic function $T$, and number of randomizations $M$.
For $m = 1, 2, ..., M$: Sample $\boldsymbol{x}^{(m)}$ from the distribution of $\boldsymbol{x} \mid \boldsymbol{Z}$, conditionally independently of $\boldsymbol{x}$ and $\boldsymbol{y}$.
Output: CRT $p$-value $\frac{1}{M+1}\left(1 + \sum_{m=1}^{M} \mathbf{1}_{\left\{T(\boldsymbol{y},\boldsymbol{x}^{(m)},\boldsymbol{Z}) \geq T(\boldsymbol{y},\boldsymbol{x},\boldsymbol{Z})\right\}}\right)$.
```

Theorem 1 (Candès et al. (2018))**.** *The CRT p-value $p(\boldsymbol{X}, \boldsymbol{y})$ satisfies*

$$\mathbb{P}_{H_0}(p(\boldsymbol{X}, \boldsymbol{y}) \leq \alpha) \leq \alpha \text{ for all } \alpha \in [0, 1].$$

For many common models of $X \mid Z$, the conditionally-independent sampling of $\boldsymbol{x}^{(m)}$ is straightforward. And even in more complex models it is still often easy to sample $\boldsymbol{x}^{(m)}$ conditionally-exchangeably with $\boldsymbol{x}$ and conditionally-independently of $\boldsymbol{y}$ (for instance by conditioning on an inferred latent variable), which is sufficient for Theorem 1 to hold. Because Theorem 1 only relies on the exchangeability of the vectors $\boldsymbol{x}, \boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(M)}$ under H_0, it is entirely agnostic to the choice of test statistic T. This enables some very powerful choices, such as T's derived from modern machine learning algorithms, from Bayesian inference (though neither the prior nor model for $Y \mid (X, Z)$ need be well-specified), or from highly domain-specific knowledge or intuition. Unfortunately the most powerful statistics are often particularly expensive to compute, and as can be seen from Algorithm 1, T must be applied $M+1$ times in order to compute a single p-value. When testing all the covariates at once, this computational problem is compounded as not only does each test require $M+1$ applications of T, but M must be roughly of order p to ensure the p-values are sufficiently high-resolution to make any discoveries with multiple testing procedures such as Benjamini–Hochberg (Benjamini & Hochberg, 1995).

1.2 Our contribution

We resolve this computational challenge in Section 2 by introducing a technique we call distillation that can still leverage any high-dimensional modeling or supervised learning algorithm, but presents dramatic computational savings by only requiring the expensive high-dimensional computation to be performed once, instead of $M+1$ times. We call our proposed method the distilled CRT, or dCRT, and show how to further improve its computation in multiple testing settings in Section 3.

We demonstrate in simulations in Section 4 that there is little difference in power between the dCRT and its more expensive CRT counterpart (in this paper we will refer to the CRT implementation as originally proposed without distillation or HRT speedup as the original CRT, or oCRT), and what small differences exist can be explained by factors that are separate from distillation. Meanwhile, our proposals save orders of magnitude in computation over the oCRT even for medium-scale problems (the savings only increase for larger data). We also show in simulations that the dCRT is comparably powerful to other state-of-the-art conditional independence tests, and is also robust to misspecification in the distribution of X.

In Section 5, we apply the dCRT to a breast cancer dataset and discover more clinically-informative somatic mutations than competing methods, and we cite independent scientific work corroborating each of the discoveries we make. Finally, we close with a discussion in Section 6.

The dCRT inherits several attractive properties of the CRT: it can be derandomized to an arbitrary extent through computation (increasing M) and yields finite-sample valid p-values for all variables that can be used for downstream multiple testing analyses with a variety of error metrics, including not only the false discovery rate but also the family-wise error rate and others.

1.3 Related work

Our work builds upon the CRT framework of Candès et al. (2018), with the goal of making it computationally tractable without sacrificing power. Our work is perhaps most similar in its goal to the HRT of Tansey et al. (2018), which uses data splitting to enable the use of complex modeling in the CRT with far less computation by doing all the complex modeling on the first part of the data and testing on the second part. A domain-specific version of the HRT is applied by Bates et al. (2020) to genetic trio studies by (using causal terminology) learning a model on observational data and using it within the CRT on randomized experimental data; the power of a similar "hybrid" CRT approach is studied in Katsevich & Ramdas (2020b). We show in Section 4 that data splitting comes with a substantial power loss compared to the dCRT and oCRT. Tansey et al. (2018) addresses this with cross-fitting, but in doing so loses the guarantee on Type-I error control of the CRT (and dCRT). Other works have extended the CRT (Berrett et al., 2020; Bellot & van der Schaar, 2019) in ways that do not address its computational intractability. For the variable selection problem, model-X knockoffs (Candès et al., 2018) can simultaneously test conditional independence for each covariate, yielding a false discovery rate-controlling rejection set. Model-X knockoffs is inherently a *multiple* testing method, with power to detect groups of non-null variables without quantifying their individual significances. On the other hand, the dCRT is a *single* testing method which can be paired with multiple testing procedures if desired. We elaborate further on the comparison between dCRT and model-X knockoffs in the discussion.

We note a pair of methods, double machine learning (Chernozhukov et al., 2018) and the generalized covariance measure (Shah & Peters, 2018), that both test conditional independence under assumptions that nearly (but not quite, due to moment conditions on Y) subsume ours, and whose test statistic resembles and can even be identical to certain special cases of the dCRT. However, their statistics only resemble a special case of the dCRT—the dCRT framework includes many other statistics which deviate substantially from double machine learning/generalized covariance measure and can be more powerful in certain settings. Furthermore, the cutoffs for their test statistics are both based on asymptotic normality, while the dCRT is non-asymptotically exact regardless of the distribution of its test statistic (see Appendix D.5).

1.4 Notation

Let $I = (i_i, i_2, \dots, i_k) \subseteq \{1, \dots, n\}$ and $J = (j_1, j_2, \dots, j_\ell) \subseteq \{1, 2, \dots, p\}$ be subsets of samples and variables, respectively, and consider a matrix $\boldsymbol{A} = (\boldsymbol{a}_1, \boldsymbol{a}_2, \dots, \boldsymbol{a}_p) \in \mathbb{R}^{n \times p}$ with $\boldsymbol{a}_j = (A_{1j}, A_{2j}, \dots, A_{nj})^\intercal$. We denote by $A_{I,J}$ the sub-matrix of A with rows in I and columns in J. We use the subscripts j, $-J'$, and $\bullet$ as shorthand for $J = \{j\}, \{1, \dots, p\} \setminus J'$, and $\{1, \dots, p\}$, respectively, and the same for the first index. For example, $A_{\bullet,-j}$ represents the matrix A with the jth column removed. For any two vectors $\boldsymbol{a}_j$ and $\boldsymbol{a}_\ell$, let $\boldsymbol{a}_j \odot \boldsymbol{a}_\ell = (A_{1j}A_{1\ell}, A_{2j}A_{2\ell}, \dots, A_{nj}A_{n\ell})^\intercal$ denote their elementwise product, and for $L = \{\ell_1, \ell_2, \dots, \ell_k\}$ let $\boldsymbol{a}_j \odot \boldsymbol{A}_L = (\boldsymbol{a}_j \odot \boldsymbol{a}_{\ell_1}, \dots, \boldsymbol{a}_j \odot \boldsymbol{a}_{\ell_k})$; these will be used when fitting interaction effects.

2 The distilled conditional randomization test

2.1 Main idea

It is natural to derive CRT test statistics from machine learning methods with high predictive and estimation accuracy. Indeed the original paper proposing the CRT (Candès et al., 2018) used as test statistic $T_{\text{oCRT}}(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{Z}) := |\hat{\beta}_x^{\mathsf{lasso}}|$, the absolute value of the fitted coefficient on $\boldsymbol{x}$ from the lasso (Tibshirani, 1996) of $\boldsymbol{y}$ on $(\boldsymbol{x}, \boldsymbol{Z})$ with penalty parameter chosen by cross-validation. Although powerful and computationally much faster than many other machine learning algorithms, it is still expensive to repeatedly run the lasso on large data sets hundreds or more times just to compute a single CRT p-value, and many times more than that in multiple-testing scenarios when a CRT p-value for each covariate is needed.

Consider now the following alternative test statistic which captures the essence of our proposal. First fit a cross-validated lasso of $\boldsymbol{y}$ on only $\boldsymbol{Z}$ to obtain the p-dimensional coefficient vector $\hat{\beta}_z^{\mathsf{loco}}$. Then fit a least-squares regression of the residual $(\boldsymbol{y} - \boldsymbol{Z}\hat{\beta}_z^{\mathsf{loco}})$ on $\boldsymbol{x}$ to obtain the scalar coefficient $\hat{\beta}_x^{\mathsf{loco}}$ and take its absolute value $T_{\text{dCRT}}(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{Z}) := |\hat{\beta}_x^{\mathsf{loco}}|$ as the test statistic. Here, the superscription "loco" represents "leave-one-covariate-out" regression as $\boldsymbol{x}$ is left out when regressing $\boldsymbol{y}$ solely on $\boldsymbol{Z}$.

We introduce this notation to distinguish the leave-one-covariate-out construction from the oCRT lasso statistics when needed, although in the remaining paper, we will just use $(\widehat{\beta}_x, \widehat{\beta}_z)$ to represent $(\widehat{\beta}_x^{\mathsf{loco}}, \widehat{\beta}_z^{\mathsf{loco}})$ when there is no need to distinguish them from $(\widehat{\beta}_x^{\mathsf{lasso}}, \widehat{\beta}_z^{\mathsf{lasso}})$. It may seem as though little has changed from the preceding paragraph—we would expect T_{oCRT} and T_{dCRT} to have similar statistical properties and require nearly the same computation. Although the statistical properties of T_{oCRT} and T_{dCRT} are indeed very similar and they do require nearly the same time to compute once, they require dramatically different computation within the CRT. The key difference is that the expensive $(p+1)$-dimensional lasso fit in T_{oCRT} must be recomputed for each resample of $\boldsymbol{x}$, while the expensive p-dimensional lasso fit in T_{dCRT} must only be computed once, since that lasso does not depend on $\boldsymbol{x}$ and hence is identical for all its resamples. In the CRT, neither $\boldsymbol{y}$ nor $\boldsymbol{Z}$ change during the resampling procedure, and we take advantage of this by applying our expensive computation to only $\boldsymbol{y}$ and $\boldsymbol{Z}$ so it only has to be done once. All that is required for each resample's computation of T_{dCRT} is a univariate regression, whose computational expense is much lower than a p-dimensional lasso.

We can generalize this idea far beyond the lasso or linear regressions. The core proposal is to distill all the high-dimensional information in $\boldsymbol{Z}$ about $\boldsymbol{y}$ into a low-dimensional representation, without looking at $\boldsymbol{x}$. Then the test statistic estimates a relationship between $\boldsymbol{x}$ and the leftover information in $\boldsymbol{y}$ by only looking at $\boldsymbol{x}$, $\boldsymbol{y}$, and the distilled (low-dimensional) function of $\boldsymbol{Z}$. Thus all the computation on high-dimensional data, namely the distillation, only needs to be performed once, while the computation that is repeatedly applied to the resampled data is low-dimensional and hence relatively fast.

2.2 Formal presentation of dCRT

We now formalize the idea from the previous subsection in Algorithm 2.

Algorithm 2 The distilled conditional randomization test (dCRT).

```
Input: The distribution of $\boldsymbol{x} \mid \boldsymbol{Z}$, data $(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{Z})$, $\boldsymbol{y}$-distillation-fitting function $\mathcal{D}_y$, $\boldsymbol{x}$-distillation function $d_x$, test statistic function $T$, and number of randomizations $M$.
 Distill $\boldsymbol{Z}$'s information about $\boldsymbol{y}$ into $\boldsymbol{d}_y = \mathcal{D}_y(\boldsymbol{y}, \boldsymbol{Z})$ and about $\boldsymbol{x}$ into $\boldsymbol{d}_x = d_x(\boldsymbol{Z})$.
 For $m = 1, 2, ..., M$: Sample $\boldsymbol{x}^{(m)}$ from the distribution of $\boldsymbol{x} \mid \boldsymbol{Z}$, conditionally independently of $\boldsymbol{x}$ and $\boldsymbol{y}$.
 Output: dCRT $p$-value $\frac{1}{M+1}\left(1 + \sum_{m=1}^{M} \mathbf{1}_{\left\{T(\boldsymbol{y}, \boldsymbol{x}^{(m)}, \boldsymbol{d}_y, \boldsymbol{d}_x) \geq T(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{d}_y, \boldsymbol{d}_x)\right\}}\right)$.
```

The key difference from the more general CRT in Algorithm 1 is that the test statistic function T in Algorithm 2 only sees information about the high-dimensional $\boldsymbol{Z}$ through its $\boldsymbol{y}$- and $\boldsymbol{x}$-distillations $\boldsymbol{d}_y$ and $\boldsymbol{d}_x$, which are both computed just once in the first line of the algorithm. $\mathcal{D}_y$ and d_x should be chosen such that the distillation step produces $\boldsymbol{d}_y$ and $\boldsymbol{d}_x$ with dimension much less than p, so that T's inputs are low-dimensional. Then since T is the only repeatedly-applied function and its computation does not suffer from the high-dimensionality of the original data, the dCRT's computation will be dominated by the single application of $\mathcal{D}_y$. For instance, in the dCRT example in Section 2.1, $\boldsymbol{d}_x$ is not used and $\mathcal{D}_y$ fits a lasso of $\boldsymbol{y}$ on $\boldsymbol{Z}$ and returns $\boldsymbol{d}_y = \boldsymbol{Z}\hat{\beta}_z$, while $T(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{d}_y) = |(\boldsymbol{y} - \boldsymbol{d}_y)^\top \boldsymbol{x}|/\|\boldsymbol{x}\|^2$ requires negligible computation by comparison.

We emphasize that $\mathcal{D}_y$ can really be any regression algorithm and Theorem 1 still holds, since for any choice of $\mathcal{D}_y$ the dCRT is still a special case of the CRT. Thus it can take advantage of the predictive power of state-of-the-art machine learning algorithms, precise knowledge in the form of a Bayesian prior, or even imprecise domain expertise or intuition applied by trying many different regressions of $\boldsymbol{y}$ on $\boldsymbol{Z}$ and choosing whichever "feels" best (as long as $\boldsymbol{x}$ is not factored into that decision). In the sequel we provide some suggestions and default choices.

2.3 The d$_0$CRT: fast, powerful, and intuitive

The most computationally-efficient and intuitive class of dCRT procedures has both $\boldsymbol{y}$- and $\boldsymbol{x}$-distillations reduce $\boldsymbol{Z}$ to an output with a single column. We label this subclass of dCRT procedures as d$_0$CRT because it represents the choice to maximally-distill each row of $\boldsymbol{Z}$ down to a single scalar. Assuming T's computation generally increases with the dimension of its inputs, the d$_0$CRT also represents a particularly computationally-efficient class of dCRTs.

A natural approach to constructing a d$_0$CRT, especially when Y is continuous, is to have distillation take the form of conditional mean functions. That is, let $d_x(\boldsymbol{Z}) = \mathbb{E}[\boldsymbol{x} \mid \boldsymbol{Z}]$ and have $\mathcal{D}_y$ fit an estimate of the analogous regression function for $\boldsymbol{y}$, i.e., $\mathcal{D}_y(\boldsymbol{y}, \boldsymbol{Z}) \approx \mathbb{E}[\boldsymbol{y} \mid \boldsymbol{Z}]$. Then T can be chosen as an empirical measure of dependence between the residuals $\boldsymbol{y} - \boldsymbol{d}_y$ and $\boldsymbol{x} - \boldsymbol{d}_x$, such as the square of the fitted coefficient when regressing the former on the latter. This approach is also easy to understand and implement since it just requires choosing $\mathcal{D}_y$ and T, with $\mathcal{D}_y$ just performing a (possibly nonparametric) regression while T can be thought of as computing a test statistic for testing the independence between two scalar random variables from a paired sample of size n: $(\boldsymbol{y} - \boldsymbol{d}_y, \boldsymbol{x} - \boldsymbol{d}_x)$. As both regression and bivariate independence testing are highly-studied topics, users can easily draw from their statistical training, domain expertise, and a rich literature in order to design an appropriate d$_0$CRT for their particular problem. The following is a generic example we found to be computationally efficient and powerful in our simulations.

Example 1 (lasso-based d$_0$CRT). $\boldsymbol{d}_y = \boldsymbol{Z}\hat{\beta}_z$ *is the fitted predictions from a cross-validated lasso of* $\boldsymbol{y}$ *on* $\boldsymbol{Z}$, $\boldsymbol{d}_x = \mathbb{E}\left[\boldsymbol{x} \mid \boldsymbol{Z}\right]$, *and* $T(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{d}_y, \boldsymbol{d}_x) = |\hat{\beta}_x| := \frac{|(\boldsymbol{y}-\boldsymbol{d}_y)^\top(\boldsymbol{x}-\boldsymbol{d}_x)|}{\|\boldsymbol{x}-\boldsymbol{d}_x\|^2}$.

More generally, the d$_0$CRT's distillation need not be couched in terms of finding conditional means. For instance, an appealing analogue of Example 1 for binary Y might fit $\hat{\beta}_z$ by a cross-validated L_1-penalized logistic regression of $\boldsymbol{y}$ on $\boldsymbol{Z}$ and otherwise leave $\mathcal{D}_y$ and $\boldsymbol{d}_x$ unchanged (note $\boldsymbol{Z}\hat{\beta}_z$ no longer approximates $\mathbb{E}[\boldsymbol{y} \mid \boldsymbol{Z}]$), and take $T(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{d}_y, \boldsymbol{d}_x)$ to be absolute value of the fitted coefficient from a logistic regression of $\boldsymbol{y}$ on $\boldsymbol{x} - \boldsymbol{d}_x$ with offset $\boldsymbol{d}_y$.

2.4 The d$_\text{I}$CRT: accounting for interactions

Of the three functions applied in Algorithm 2, only T takes both $\boldsymbol{y}$ and $\boldsymbol{x}$ as arguments and hence the choice of T is how a user can encode the kinds of non-null relationships between Y and X that are deemed plausible. But because T only sees $\boldsymbol{Z}$ through $\boldsymbol{d}_y$ and $\boldsymbol{d}_x$, any plausible models for Y must be expressed using only $\boldsymbol{x}$, $\boldsymbol{d}_y$, and $\boldsymbol{d}_x$. This means that the d$_0$CRT has almost no capacity to model even first-order interactions between X and Z. For instance, suppose $p = 3$ and $Z_j \overset{i.i.d.}{\sim} \mathcal{N}(0,1)$, $X \sim Z_1 + \mathcal{N}(0,1)$, and $Y \sim Z_2 + XZ_3 + \mathcal{N}(0,1)$. Then the best possible distillations of $\boldsymbol{x}$ and $\boldsymbol{y}$ are $\boldsymbol{d}_x = \boldsymbol{Z}_1$ and $\boldsymbol{d}_y = \boldsymbol{Z}_2 + \boldsymbol{Z}_1 \odot \boldsymbol{Z}_3$, making it impossible for T to encode the true conditional mean of $\boldsymbol{y}$, namely, $\boldsymbol{Z}_2 + \boldsymbol{x} \odot \boldsymbol{Z}_3$, from just $\boldsymbol{x}$, $\boldsymbol{d}_x$, and $\boldsymbol{d}_y$.

To address this limitation of the d$_0$CRT, one can simply increase the dimension of $\boldsymbol{d}_y$ and $\boldsymbol{d}_x$ to explicitly include possible columns of $\boldsymbol{Z}$ with which $\boldsymbol{x}$ might be expected to interact. But of course increasing the dimension of $\boldsymbol{d}_y$ and $\boldsymbol{d}_x$ tends to come at a computational cost, since their low-dimensionality is exactly what makes the dCRT fast in the first place. Thus one needs some sort of prior, domain knowledge, or heuristic for choosing based on either the pair $(\boldsymbol{y}, \boldsymbol{Z})$ or $(\boldsymbol{x}, \boldsymbol{Z})$ (but not based on $(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{Z})$ together) a small subset of columns of $\boldsymbol{Z}$ that $\boldsymbol{x}$ might plausibly interact with. One option is to split the data into two independent parts and use one part in an unconstrained way to select columns of $\boldsymbol{Z}$ that are likely to interact with $\boldsymbol{x}$, and then to leverage these selections in a dCRT run only on the other part. We propose here an alternative that avoids sample splitting, based on the common statistical practice of only allowing for interactions between variables with strong main effects. This practice of enforcing hierarchy in interactions has a long history in applied and theoretical statistics under many different names (Nelder, 1977; Cox, 1984; Peixoto, 1987; Hamada & Wu, 1992; Chipman, 1996; Bien et al., 2013).

Our proposed method for incorporating interactions, which we call the d$_{\text{I}}$CRT, is to have $\mathcal{D}_y$ still distill $\boldsymbol{Z}$ into one column to best-capture the relationship between $\boldsymbol{y}$ and $\boldsymbol{Z}$, but then to additionally return (as further columns of $\boldsymbol{d}_y$) a limited subset of columns of $\boldsymbol{Z}$ whose contributions to that fitted relationship are strongest. Then T can be chosen as a test statistic that allows $\boldsymbol{x}$ to interact with those columns of $\boldsymbol{Z}$ contained in $\boldsymbol{d}_y$, while still prioritizing the main effect of $\boldsymbol{x}$. As a generic example we found to be powerful to detect hierarchical interactions without losing much power in the absence of interactions, consider the following.

Example 2 (lasso-based d$_{\text{I}}$CRT)**.** $\boldsymbol{d}_y = (\boldsymbol{Z}\hat{\beta}_z, \boldsymbol{Z}_{\bullet,\text{top}(k)}) := (\boldsymbol{d}_{y,1}, \boldsymbol{d}_{y,\text{-}1})$ *is the fitted predictions from a cross-validated lasso of* $\boldsymbol{y}$ *on* $\boldsymbol{Z}$ *concatenated with the columns of* $\boldsymbol{Z}$ *corresponding to the* k *largest entries of* $|\hat{\beta}_z|$, $\boldsymbol{d}_x = \mathbb{E}\left[\boldsymbol{x} \mid \boldsymbol{Z}\right]$, *and* $T(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{d}_y, \boldsymbol{d}_x) = \hat{\beta}_{x,1}^2 + \frac{1}{k}\sum_{j=2}^{k+1}\hat{\beta}_{x,j}^2$, *where* $\hat{\beta}_x \in \mathbb{R}^{k+1}$ *are the fitted coefficients from a least-squares fit of* $(\boldsymbol{y} - \boldsymbol{d}_{y,1})$ *on* $(\boldsymbol{x} - \boldsymbol{d}_x)$ *and* $(\boldsymbol{x} - \boldsymbol{d}_x) \odot \boldsymbol{d}_{y,\text{-}1}$.

The normalization by $1/k$ of $\sum_{j=2}^{k+1}\hat{\beta}_{x,j}^2$ encodes our hierarchical prioritization of the main effect $\hat{\beta}_{x,1}$ over the interaction effects. For small k we still expect the computation to be dominated by $\mathcal{D}_y$, but it also represents a statistical trade-off in how widely to search for interactions; we found the performance to be quite stable to k in our simulations, but set as a default $k = \lceil 2\log(p) \rceil$. Note that k could also be chosen after looking at $(\boldsymbol{y}, \boldsymbol{Z})$, and more generally, one can construct many different types of d$_{\text{I}}$CRT. For instance, one can adapt Example 2 to binary Y in an analogous way as was done for Example 1 by replacing linear regressions with logistic regressions and using $\boldsymbol{d}_{y,1}$ as an offset in T. Or one could have $\mathcal{D}_y$ and/or T use the predictions and default variable importance measures from a random forest. We explore some of these options in simulations in Section 4.

2.5 Running the dCRT without resampling

Distillation provides massive computational savings within the CRT by only requiring a single evaluation of the by-far-most-expensive function $\mathcal{D}_y$. But it still requires $M + 1$ evaluations of T, which can sometimes still contribute nontrivially to the computation time, and requires the user to choose the tuning parameter M which trades off computation and statistical power. It turns out that in certain cases the simplicity of T in the dCRT can be leveraged to remove the resampling of $\boldsymbol{x}^{(m)}$ entirely and compute an exact p-value directly from the single function evaluation $T(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{d}_y, \boldsymbol{d}_x)$.

For intuition, suppose $X \mid Z \sim \mathcal{N}(Z^\top\gamma, \sigma^2)$, and consider the d$_0$CRT with T as in Example 1,

$$T(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{d}_y, \boldsymbol{d}_x) = \frac{|(\boldsymbol{y} - \boldsymbol{d}_y)^\top(\boldsymbol{x} - \boldsymbol{d}_x)|}{\|\boldsymbol{x} - \boldsymbol{d}_x\|^2}.$$

Then since the (d)CRT conditions on $\boldsymbol{y}$ and $\boldsymbol{Z}$ (and hence also $\boldsymbol{d}_y$ and $\boldsymbol{d}_x = \boldsymbol{Z}\beta$),

$$(\boldsymbol{y} - \boldsymbol{d}_y)^\top(\boldsymbol{x} - \boldsymbol{d}_x) \sim \mathcal{N}\left(0, \sigma^2\|\boldsymbol{y} - \boldsymbol{d}_y\|^2\right). \tag{1}$$

The denominator of T makes things a bit more complicated, but the nature of the statistic does not change much if we replace the denominator by its expectation or, equivalently (since multiplying

T by a fixed constant has no effect on its resulting p-value), simply replace it by $T'(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{d}_y, \boldsymbol{d}_x) = |(\boldsymbol{y} - \boldsymbol{d}_y)^\top(\boldsymbol{x} - \boldsymbol{d}_x)|$. We then get immediately that the exact p-value — i.e., the p-value that would result from taking the limit as $M \to \infty$ — can be computed as $2\left(1 - \Phi\left(\frac{T'(\boldsymbol{y},\boldsymbol{x},\boldsymbol{d}_y,\boldsymbol{d}_x)}{\sigma\|\boldsymbol{y}-\boldsymbol{d}_y\|}\right)\right)$ without ever resampling $\boldsymbol{x}^{(m)}$ or recomputing T', where Φ is the standard normal cumulative distribution function.

The same principle can be applied to non-Gaussian X: since the distribution of $(\boldsymbol{x} - \boldsymbol{d}_x) \mid \boldsymbol{Z}$ is known and the rows are independent, $(\boldsymbol{x} - \boldsymbol{d}_x)$ can be element-wise transformed via scalar monotone functions to be i.i.d. $\mathcal{N}(0, 1)$ given $\boldsymbol{Z}$. For conditionally-continuously-distributed $(\boldsymbol{x} - \boldsymbol{d}_x)$, this can be done via the probability inverse transform, while for distributions with atoms the atoms need to be carefully randomized (though just once); see Appendix A for details.

As long as $(\boldsymbol{x} - \boldsymbol{d}_x)$ is independent Gaussian or transformed to be, the same principle can also be applied to some more complex T functions. For instance, in Example 2, we can again replace the random "denominator" (in this case the matrix inverse in the least-squares formula for $\hat{\beta}_x$) with its conditional expectation given $\boldsymbol{Z}$, and end up with a quadratic form in Gaussian random variables. Efficient algorithms for computing the quantiles of a quadratic form in Gaussian random variables exist (Duchesne & De Micheaux, 2010) and can be applied to again compute the exact dCRT p-value without any resampling; see Appendix A for details.

3 Variable selection and multiple testing via the dCRT

Conditional independence testing is often done in the context of a variable selection problem. Given p covariates $X_1, \dots, X_p$ and a response Y, the goal is to discover the covariates X_j that are conditionally associated with the response, i.e., $Y \not\perp\!\!\!\perp X_j \mid X_{-j}$. For a given j, we arrive at the problem formulation from the previous two sections by setting $X = X_j$ and $Z = X_{-j}$. This change of notation highlights the fact that the effects of all variables are of interest, rather than that of one special variable. Given a design matrix $\boldsymbol{X} \in \mathbb{R}^{n\times p}$ and a response vector $\boldsymbol{y}$, we propose to approach the variable selection problem by applying the dCRT to $(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{Z}) = (\boldsymbol{y}, \boldsymbol{X}_{\bullet,j}, \boldsymbol{X}_{\bullet,-j})$ for each covariate j, followed by a multiple testing procedure on the resulting p-values. Two common error rates to control are the family-wise error rate and the false discovery rate. The former can be easily achieved based on the Bonferroni correction, which works under arbitrary p-value dependence. The latter is usually done via the Benjamini–Hochberg procedure. Even though the p-values are technically not positively dependent in the sense required for mathematical false discovery rate control (Benjamini & Yekutieli, 2001), the Benjamini–Hochberg procedure is known to be very robust to dependent p-values in all but adversarially-constructed settings, as confirmed in our simulations.

Regardless of error rate, the straightforward application of the dCRT to the variable selection problem requires computing $\mathcal{D}_y$ a total of p times, once for each variable. Note that these are entirely parallel computations, so for certain problem dimensionalities and parallel computing resources, this is entirely feasible. However, in large-scale variable selection applications such as genome-wide association studies, there may be too many covariates for the direct application of dCRT to each. In the following subsections we present two computational shortcuts that make variable selection via the dCRT feasible for large-scale applications.

3.1 Data-dependent screening of variables

A natural acceleration of the dCRT for variable selection is to first use the data to identify a preliminary subset $\mathcal{S} \subseteq \{1, \dots, p\}$ of promising covariates via a screening function $S : (\boldsymbol{X}, \boldsymbol{y}) \mapsto \mathcal{S}$.

We can then compute (d)CRT p-values $p_j(\boldsymbol{X}, \boldsymbol{y})$ (via Algorithms 1 or 2) for only $j \in \mathcal{S}$ while setting the p-values for all the other covariates to 1, yielding the screened p-values

$$p'_j(\boldsymbol{X}, \boldsymbol{y}) = \begin{cases} p_j(\boldsymbol{X}, \boldsymbol{y}) & \text{if } j \in S(\boldsymbol{X}, \boldsymbol{y}); \\ 1 & \text{if } j \notin S(\boldsymbol{X}, \boldsymbol{y}). \end{cases} \tag{2}$$

For instance, $\mathcal{S}$ could be the active set of a cross-validated lasso fit of $\boldsymbol{y}$ on all the covariates.

In general, a screening step like this applied before the (d)CRT breaks the exchangeability between the original and resampled test statistics which Theorem 1 relies on to guarantee p-value validity. Despite this failure of exchangeability, the screening can only inflate a p-value and thus does not affect its validity.

Theorem 2. *Let j be a null variable. For any screening rule S, the screened p-value $p'_j(\boldsymbol{X}, \boldsymbol{y})$ obtained from equation* (2) *is stochastically larger than uniform.*

Proof. By equation (2), for any $u \in [0, 1]$, $\mathbb{P}(p'_j(\boldsymbol{X}, \boldsymbol{y}) \le u) \le \mathbb{P}(p_j(\boldsymbol{X}, \boldsymbol{y}) \le u) \le u$. □

Thus, with the small computational overhead of a single well-chosen screening function, we can expect to dramatically cut the computation time of using the (d)CRT for variable selection. Indeed we found in our simulations that simple screenings substantially decreased computation time without affecting the power.

3.2 Recycling computation for L$_1$-regularized M-estimators

In some cases, we may want to compute p-values for all variables under consideration, even if only a small fraction of these are statistically significant. For instance, these may be needed for downstream analysis tasks like calibration assessment or meta-analysis. In such cases, we must look beyond the screening approach. In this section, we present a way of recycling computation for L_1-regularized M-estimators including the lasso (recall Examples 1 and 2). This reduces the number of $\mathcal{D}_y$ computations from p to $|\mathcal{A}|$, where $\mathcal{A}$ is the active set of the lasso on $(\boldsymbol{X}, \boldsymbol{y})$.

Let $\mathcal{D}_y$ be the cross-validated lasso with strictly convex and differentiable loss function ℓ. Variable selection via the dCRT based on this distillation function requires computing

$$\widehat{\beta}(\boldsymbol{X}_{\bullet,-j}, \boldsymbol{y}; \lambda) := \underset{\beta \in \mathbb{R}^{p-1}}{\arg\min} \sum_{i=1}^{n} \ell(Y_i, X_{i,-j}\beta) + \lambda \|\beta\|_1 \tag{3}$$

for each $j = 1, \dots, p$, along a grid of regularization parameters. There is redundancy among these p lasso problems; they all differ from the the full lasso problem on $(\boldsymbol{X}, \boldsymbol{y})$ by just one variable. We may therefore expect that we can save computation by somehow recycling computation across these lasso problems. The next lemma suggests a means to this end:

Lemma 1. *Suppose the columns of $\boldsymbol{X}$ are in general position and that the loss ℓ is differentiable and strictly convex. Then, for any $\lambda > 0$,*

$$\widehat{\beta}_j(\boldsymbol{X}, \boldsymbol{y}; \lambda) = 0 \quad \Longrightarrow \quad \widehat{\beta}(\boldsymbol{X}_{\bullet,-j}, \boldsymbol{y}; \lambda) = \widehat{\beta}_{-j}(\boldsymbol{X}, \boldsymbol{y}; \lambda). \tag{4}$$

In words, Lemma 1 states that removing an inactive variable from the lasso does not change the fitted coefficient vector. This has important computational implications (potentially even outside the scope of this paper)—it suggests that we can avoid refitting the lasso (3) for most variables j, instead recycling the lasso fit on the full design matrix. Of course, the parameter λ is usually tuned via cross-validation, which introduces extra complications. However, we claim that if λ is chosen in an appropriate data-dependent way, then an analogous result will still hold.

To make this precise, consider a grid of regularization parameters

$$\lambda(1) > \lambda(2) > \cdots > \lambda(G) > 0 \tag{5}$$

and a corresponding set of cross-validation errors $\mathcal{E}_1, \ldots, \mathcal{E}_G$. Define a rule $\widehat{g}$ to select the penalty parameter λ based on cross-validation errors $\mathcal{E}_1, \ldots, \mathcal{E}_G$ to be *sequential* if these values are traversed in this order, and at some stopping time $\widetilde{g}$, the algorithm terminates and chooses $\lambda(\widehat{g})$ for some $\widehat{g} \leq \widetilde{g}$. For example, for any integer $\Delta \geq 1$, the following rule is sequential:

$$\widehat{g} \equiv \min\{g : \mathcal{E}_g \leq \min(\mathcal{E}_{g+1}, \ldots, \mathcal{E}_{g+\Delta})\},$$

which is the first time along the regularization path that the cross-validation error is smaller than the following Δ steps (the first 'local minimum' on the cross-validation path, and the 'sparsest' of all such local minima). In this case the stopping time is $\widetilde{g} = \widehat{g} + \Delta$. The lasso with any sequential rule $\widehat{g}$ has the property (4).

Theorem 3. *Fix a grid of regularization parameters* (5)*. Consider applying L_1-regularized regression with loss ℓ on the whole data $(\boldsymbol{X}, \boldsymbol{y})$, with λ selected by K-fold cross-validation and a sequential stopping rule $\widehat{g}$. Let $\widehat{g}(\boldsymbol{X}, \boldsymbol{y})$ and $\widetilde{g}(\boldsymbol{X}, \boldsymbol{y})$ be the resulting grid point and stopping time, respectively. Letting $\{1, \ldots, n\} = D_1 \cup \cdots \cup D_K$ denote the split of the data into non-overlapping folds, define the active set*

$$\begin{aligned}\mathcal{A} = \{j \in \{1, \ldots, p\} : \;& \widehat{\beta}_j(\boldsymbol{X}, \boldsymbol{y}; \lambda(\widehat{g}(\boldsymbol{X}, \boldsymbol{y}))) \neq 0 \textit{ or} \\ & \widehat{\beta}_j(\boldsymbol{X}_{\text{-}D_k,\bullet}, \boldsymbol{y}_{\text{-}D_k}; \lambda(g)) \neq 0 \textit{ for some } k, g \leq \widetilde{g}(\boldsymbol{X}, \boldsymbol{y})\}.\end{aligned} \tag{6}$$

If the loss ℓ is differentiable and strictly convex, and the columns of $\boldsymbol{X}$ and $\boldsymbol{X}_{\text{-}D_k,\bullet}$ are in general position for each k, then excluding non-active variables j does not alter the fitted coefficients: for each $j \notin \mathcal{A}$,

$$\widehat{g}(\boldsymbol{X}_{\bullet,\text{-}j}, \boldsymbol{y}) = \widehat{g}(\boldsymbol{X}, \boldsymbol{y}) \textit{ and } \widehat{\beta}(\boldsymbol{X}_{\bullet,\text{-}j}, \boldsymbol{y}; \lambda(\widehat{g}(\boldsymbol{X}_{\bullet,\text{-}j}, \boldsymbol{y}))) = \widehat{\beta}_{\text{-}j}(\boldsymbol{X}, \boldsymbol{y}; \lambda(\widehat{g}(\boldsymbol{X}, \boldsymbol{y}))). \tag{7}$$

Theorem 3 states that for each variable j *not* in the active set, we need not re-run the lasso holding out variable j; we can instead fit the full lasso once and then read off the coefficient vector. This computational shortcut, summarized in Algorithm 4, reduces the number of lasso applications required by the dCRT from p to $|\mathcal{A}|$. Depending on the sparsity of the problem, this reduction can save several orders of magnitude of computation. It is known that at most, the lasso solution has $\min(p, n)$ nonzero entries (Tibshirani, 2013), though often it is much sparser.

4 Statistical performance of the dCRT

4.1 Implications of distillation for power

Our motivation for proposing the dCRT is computational; using distilled test statistics accelerates the CRT by orders of magnitude compared to the originally-proposed lasso coefficient test statistic. In this section, we discuss the statistical implications of this computational acceleration. While distillation is a flexible framework that can encompass a variety of test statistics, for concreteness in this section we narrow our focus to the d$_0$CRT. Our goal is to carefully compare the d$_0$CRT to its "undistilled" counterpart, the oCRT based on the absolute lasso coefficient. Our main conclusion is that, perhaps surprisingly, distillation does not have much effect on the the power of the CRT. We present the main reasoning behind this conclusion here and defer the details to Appendix C.

To emphasize the exclusion of $\boldsymbol{x}$ from the lasso regression, let $\widehat{\beta}_z^{\mathsf{loco}}$ be the fitted coefficients in the lasso regression of $\boldsymbol{y}$ on $\boldsymbol{Z}$ and let $\widehat{\beta}_x^{\mathsf{loco}} = (\boldsymbol{x} - \boldsymbol{d}_x)^\top(\boldsymbol{y} - \boldsymbol{Z}\widehat{\beta}_z^{\mathsf{loco}})/\|\boldsymbol{x} - \boldsymbol{d}_x\|^2$ as in the definition of the d$_0$CRT. By contrast, let $(\widehat{\beta}_x^{\mathsf{lasso}}, \widehat{\beta}_z^{\mathsf{lasso}})$ denote the fitted coefficients in the lasso regression of $\boldsymbol{y}$ on $\boldsymbol{x}$ and $\boldsymbol{Z}$, so the oCRT is based on the test statistic $|\widehat{\beta}_x^{\mathsf{lasso}}|$. Let us also assume in this section, as we did in Section 2.5, that $X \mid Z \sim N(Z^\top\gamma, s^2)$. Finally, suppose (for intuition) that $Y \mid X, Z$ follows a Gaussian linear model with coefficients β_x and β_z.

The obvious difference between oCRT and dCRT is that the latter is based on a lasso regression excluding the variable of interest while the former is based on a lasso regression on all variables. Thus, $\widehat{\beta}_z^{\mathsf{loco}} \neq \widehat{\beta}_z^{\mathsf{lasso}}$ and so of course $\widehat{\beta}_x^{\mathsf{loco}} \neq \widehat{\beta}_x^{\mathsf{lasso}}$. However, there are two additional differences that must be accounted for in order to understand the relationship between the two methods. First of all, $\widehat{\beta}_x^{\mathsf{lasso}}$ has a nonzero probability of being equal to zero, while $\widehat{\beta}_x^{\mathsf{loco}}$ is almost surely nonzero. Secondly, $\widehat{\beta}_x^{\mathsf{lasso}}$ does not necessarily have a null distribution centered on zero, whereas $\widehat{\beta}_x^{\mathsf{loco}}$ does (recall equation (1)).

In Appendix C, we examine the impacts of these two properties of the oCRT. We find that the sparsity that $\widehat{\beta}_x^{\mathsf{lasso}}$ inherits from the lasso can only hurt the power of the oCRT and propose a simple alternative based on removing the soft threshold operator. Furthermore, we show that, depending on the distribution of $X \mid Z$ and on the locations and signs of the nonzero elements of (β_x, β_z), the null distribution of $\widehat{\beta}_x^{\mathsf{lasso}}$ can either be centered at the origin, or left or right of the origin. By using the absolute value of the potentially off-center test statistic $\widehat{\beta}_x^{\mathsf{lasso}}$, the oCRT gains or loses power to the extent that the null distribution is shifted to the right or left, respectively. This motivates us to propose a centered and non-soft-thresholded version of the oCRT test statistic.

Using numerical simulations, we found essentially no difference between the performance of the dCRT and the centered, non-soft-thresholded version of the oCRT. In other words, after accounting for the aforementioned two differences, the distillation step has little or no impact on the power of the CRT. This conclusion may seem surprising, since on first glance leaving $\boldsymbol{x}$ out appears to cause some of the signal (namely the contribution of $\boldsymbol{x}$ to $\boldsymbol{y}$ that can instead be explained by $\boldsymbol{Z}$) to be regressed out. One may expect this effect to decrease the power of the dCRT. However, this is not the case because it is precisely the component of $\boldsymbol{x}$ that *cannot* be explained by $\boldsymbol{Z}$ that carries signal. Therefore, dropping $\boldsymbol{x}$ and regressing $\boldsymbol{Z}$ out of $\boldsymbol{y}$ first does not have much effect on the power of the dCRT. This intuition would be precise if $\widehat{\beta}_z^{\mathsf{loco}}$ were obtained from (unpenalized) linear regression of y on $\boldsymbol{Z}$. Indeed, it is a well-known property of linear regression that the coefficient of $\boldsymbol{x}$ can be obtained by first regressing $\boldsymbol{Z}$ out of $\boldsymbol{x}$ and $\boldsymbol{y}$, and then regressing the residual of $\boldsymbol{y}$ onto that of $\boldsymbol{x}$.

4.2 Numerical comparisons of power, speed, robustness, and stability

Going beyond the numerical simulations in the previous section, we designed an extensive simulation suite to systematically assess the power and other operating characteristics of dCRT and compare it to that of several alternative methods. Preferring to compare the dCRT to existing methods, we chose to benchmark it against the originally proposed oCRT instead of the modified version considered above. We must keep in mind, however, that this choice also complicates the comparison for the aforementioned reasons. Furthermore, we suspect that the centering and soft-thresholding issues may impact the performance of knockoffs as well. Unlike for the CRT, however, it is harder to pull these aspects apart for knockoffs. The soft thresholding affects both the one-bit p-values and the ordering of the variables, so removing it may not result in a uniform improvement like it did for oCRT. Regarding centering, it is not obvious how to recenter the knockoffs null distribution because knockoffs does not really use a null distribution. We leave the study of these phenomena for knockoffs to future work, and in the meantime compare dCRT to published implementations of the latter.

In the interest of space we defer the details of our simulations to the appendix and present here a detailed summary of the takeaways of those simulations, directly linking each takeaway to the figure and section of the appendix with the corresponding simulation(s) supporting it. The main focus of our simulations is examining the performance of the dCRT through the d_0CRT and d_ICRT given by Examples 1 and 2, respectively. Except where explicitly stated otherwise, we apply them in a resampling-free manner per Section 2.5 and, when simulating a variable selection task, with screening (using the cross-validated lasso for selection) per Section 3.1. For variable selection simulations, we take each of the p-value methods (oCRT, dCRT, HRT) and apply the Benjamini–Hochberg procedure when targeting false discovery rate control and the Bonferroni correction when targeting family-wise error rate control. Source code for running the dCRT and reproducing our results can be found along with example scripts for illustration at `https://github.com/moleibobliu/Distillation-CRT`.

We compared the dCRT to the oCRT in a broader set of simulations than those referenced in Section 4.1, including linear and logistic regression models and d_ICRT as well as d_0CRT. We chose the smaller problem size of $n = p = 300$ to accommodate the computational burden of the oCRT. We found that distillation dramatically reduces CRT computation; both the d_0CRT and d_ICRT conferred a computational savings of approximately 500 times over the oCRT (Table 2). The relative powers of dCRT and oCRT (Figures 5, 6) were consistent with what we found in Section 4.1, with dCRT sometimes more powerful and sometimes less powerful than the oCRT. The oCRT was more powerful when signals were equally spaced while the dCRT was more powerful when signals were adjacent to each other. We suspect these differences to be caused mainly by the discussed soft thresholding and centering issues. See Appendix D.2 for details.

The dCRT is more powerful than the HRT: In both the aforementioned $n = p = 300$ simulations and a larger simulation with $n = p = 800$, the dCRT computation times were mostly within an order of magnitude of the HRT (Tables 2 and 3). But across settings that included a range of n up to 1400, a range of p up to 3200, a range of signal magnitudes, a range of sparsities, a range of covariance structures for X, and a range of models for $Y \mid X$, both dCRT methods had consistently and substantially higher power than the HRT (up to about 50 percentage points higher); see Figures 5, 6, 7, 8, 9, 10, 11. See Appendices D.2 and D.3 for details.

When controlling false discovery rate, the relative performance of dCRT and knockoffs varies across simulation settings, similar to the relative performance of the dCRT and oCRT. The dCRT methods tend to have higher power than knockoffs when signal variables are adjacent and lower power than knockoffs when the signal variables are equally spaced. The power comparison between dCRT and knockoffs is a subtle one, and we leave its further investigation for future work. In very sparse settings, dCRT still has power, while knockoffs does not (due to its reliance on the Selective SeqStep+ procedure (Barber & Candès, 2015)). In such regimes, the family-wise error rate may be more appropriate, and the dCRT can be used to control this error rate as well. See Figures 5, 6, 7, 8, 9, 10, 11. The dCRT is more computationally expensive than knockoffs, but usually within an order of magnitude (Tables 2 and 3). Finally, the dCRT has substantially less algorithmic variability than knockoffs, as measured by the expected Jaccard similarity between two rejection sets obtained by re-running the methods with different seeds (Figure 12). See Appendices D.2, D.3, and D.4 for details.

The d_ICRT is stable to the choice of k and has slightly less power than the d_0CRT in additive models but can have substantially higher power in the presence of interactions: In a simulation with an additive model, the power of the d_ICRT was identical as k ranged from 2–22 (the default value of $k = 2\log(p)$ would have been 13), while in a model with five true interactions, the power only varied from about 50% to about 40% over the same range of k (Figure 15). Throughout all our simulations in additive models we found the $\mathrm{d_0CRT}$ to be slightly but consistently more powerful than the $\mathrm{d_ICRT}$ (e.g., Figures 5, 6, 7, 8, 9, 10, 11, 17, 19), but in the presence of interactions obeying the hierarchy principle discussed in Section 2.4, we found that the $\mathrm{d_ICRT}$ could be quite a bit more powerful (up to about 25 percentage points) than the $\mathrm{d_0CRT}$ (Figure 14). See Appendices D.6 and D.7 for details.

The dCRT can leverage nonparametric machine learning algorithms for substantial power gains in highly-nonlinear models: In a simulation in which X's relationship with Y was highly-nonlinear and interacted with five Z_j's, our default (lasso-based) $\mathrm{d_ICRT}$ had somewhat higher power than $\mathrm{d_0CRT}$ (as much as about 20 percentage points), but a different, random-forest-based $\mathrm{d_ICRT}$ had far higher power than the lasso-based $\mathrm{d_ICRT}$ (as much as about 50 percentage points) (Figure 16). See Appendix D.8 for details.

The dCRT is quite robust to misspecification of X's distribution: When the distribution of $X \mid Z$ is Poisson even with a very small mean parameter (making it highly discrete and heavily skewed) but approximated by a Gaussian with matching mean and variance, both the $\mathrm{d_0CRT}$ and $\mathrm{d_ICRT}$ maintain Type-I error control and high power (Figure 17). Furthermore, when the covariates are jointly Gaussian and the $X \mid Z$ distributions are estimated in-sample using any of three standard methods detailed in Appendix D.10, the Type-I error of both dCRT methods always remains close to the nominal level (Figure 18). See Appendices D.9 and D.10 for details.

The resampling-free versions of the dCRT are faster and just as powerful as the non-resampling-free dCRT except when $X \mid Z$ is highly discrete: The resampling-free modification sped up the $\mathrm{d_0CRT}$ by 2.5 times in an $n = p = 800$ simulation and sped up the $\mathrm{d_ICRT}$ by 11 times in an $n = p = 800$ simulation, even after applying screening (Table 4). When $X \mid Z$ is Gaussian, changing the form of the test statistics of the $\mathrm{d_0CRT}$ and $\mathrm{d_ICRT}$ as proposed in paragraphs 2 and 4, respectively, of Section 2.5 had a negligible effect on their power (Figure 20). When $X \mid Z$ is non-Gaussian and must be transformed to Gaussian as described in paragraph 3 of Section 2.5, we found essentially no power loss for the resampling-free $\mathrm{d_0CRT}$ and $\mathrm{d_ICRT}$ relative to their non-resampling-free counterparts when $X \mid Z$ was Gamma-distributed (with shape = 3 and rate = 0.5, so that skew > 1 and excess kurtosis = 2), while there was substantial power loss (up to about 40 percentage points) when $X \mid Z$ was binary and hence required substantial exogenous randomization to be transformed to Gaussian, though the resampling-free dCRTs were still more powerful (up to about 10 percent) than the HRT (Figure 21). See Appendices D.11 and D.12.

Screening makes the dCRT faster without affecting its power: In a simulation with $n = p = 800$, screening reduced the computation time by a factor of about 5 for both $\mathrm{d_0CRT}$ and $\mathrm{d_ICRT}$ (Table 5) without perceptibly hurting power (Figure 22). See Appendix D.13 for details.

5 Identifying biomarkers for breast cancer

As a final demonstration of the effectiveness of the dCRT, we apply it to the data set from Curtis et al. (2012), consisting of $n = 1,396$ staged oestrogen-receptor-positive cases of breast cancer, each with expression level (mRNA) and copy number aberration (CNA) measured for $p = 164$ genes, which were studied in Pereira et al. (2016). Our goal is to find genes on which cancer stage depends, conditional on the remaining genes and all CNAs, while controlling either the false discovery rate or family-wise error rate at level 0.1. Discovering such biomarkers for cancer can reveal new pathways and mechanisms for cancer progression; see Shen et al. (2019) for a recent application of model-X knockoffs to the same end.

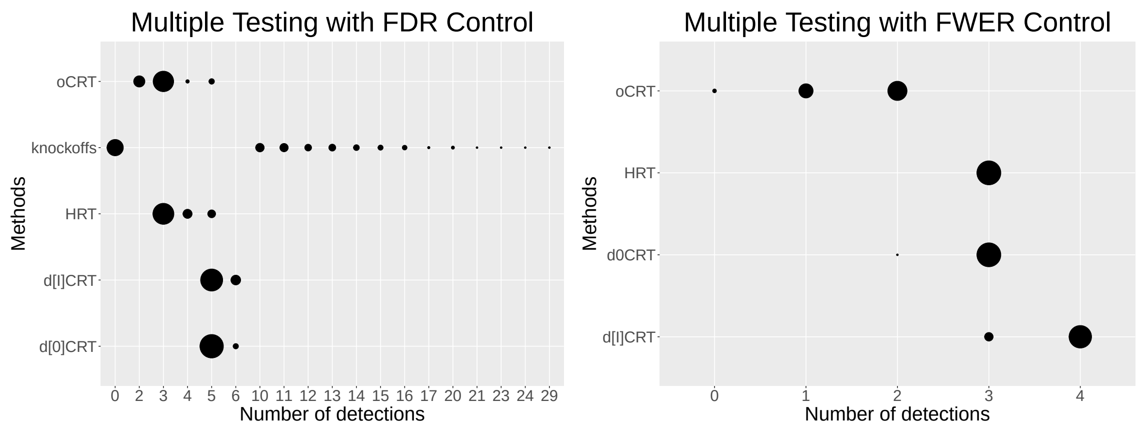


Figure 1: Summary of the numbers of discoveries over 300 repetitions, with false discovery rate and family-wise error rate control, in the breast cancer application. Area of each black point is proportional to the frequency that the corresponding method makes this number of discoveries in the 300 repetitions. The dCRT approaches are more powerful than the competitors oCRT and HRT. The knockoffs have no discoveries in around 45% of the experiments.

After log-transforming the gene expressions, we adjust to them using the CNA data with linear model as in Solvang et al. (2011); Lahti et al. (2012); Leday et al. (2013) and modeled the processed gene expressions jointly as multivariate Gaussian similar to Shen et al. (2019). We applied the $\mathrm{d_0CRT}$, the $\mathrm{d_ICRT}$, the oCRT, the HRT, and model-X knockoffs and compared the results. See Appendix E for details of the data pre-processing, covariate-modeling, and method implementations. Each method is run 300 times. Table 1 contains average runtimes (in R) for all methods, showing that the dCRTs are quite fast compared to the oCRT. In particular, the oCRT takes over 7 hours to run while the dCRTs take under a minute.

Figure 1 presents the distribution of the numbers of discoveries among the 300 repetitions for all the methods. Methods including dCRT, oCRT and HRT have stable outputs about the number of detected genes. In terms of false discovery rate control, $\mathrm{d_0CRT}$ and $\mathrm{d_ICRT}$ detect exactly 5 genes in more than 80% repetitions and $\geq$5 genes at all times. oCRT and HRT detect exactly 3 genes in more than 70% repetitions and always have fewer discoveries than dCRT. While the knockoffs have 0 discoveries in about 45% repetitions but $\geq$10 discoveries in the remaining times, which implies that knockoffs fail to produce stable output. Knockoffs' instablility and lack of power is due to the sparsity of discoverable genes. In terms of family-wise error rate control, $\mathrm{d_0CRT}$ and HRT have 3 discoveries in most runs, $\mathrm{d_ICRT}$ has 4 discoveries and oCRT has 2.

Average computation times (minutes)

$\mathrm{d_0CRT}$	$\mathrm{d_ICRT}$	oCRT	knockoffs	HRT
0.8	0.8	443.3	0.3	3.1

Table 1: Average computation times (300 repetitions in R) in the breast cancer application. Note that our use of the resampling-free version of the dCRT makes it faster than the HRT in this case.

When used to control the false discovery rate, it turns out that all five genes discovered by the dCRT — *FBXW7*, *MAP3K13*, *HRAS*, *GPS2*, and *RUNX1*; see Appendix 6 for their corresponding average p-values and frequencies of being detected — have been linked in independent research to cancer, suggesting the dCRT makes promising discoveries. In particular, *FBXW7* encodes a member of the F-box protein family and its mutations are detected in ovarian and breast cancer cell lines (Liu et al., 2019; Kirzinger et al., 2019); *MAP3K13* belongs to the serine/threonine protein kinase family acting as a regulator for cancer (Han et al., 2016); *HRAS* belongs to the RAS oncogene family which is related to the transforming of genes of mammalian sarcoma retroviruses, and defects in this gene have been implicated in a variety of cancers (Geyer et al., 2018); over-expression of *GPS2* in mammalian cells may suppress signals mediated by RAS/MAPK and interfere with JNK activity, all of which are cancer-related (Jarmalavicius et al., 2010; Huang et al., 2016); *RUNX1* has been found to activate certain signaling pathways that promote tumor metastasis (Li et al., 2019).

6 Discussion

The HRT provided the first indication that a variant of the CRT could be computationally tractable, albeit at the cost of statistical performance. In this paper, we demonstrate that leaving out *variables* instead of *samples* creates a procedure that is not quite as fast (though still a tiny fraction of the oCRT's computational cost) but much more powerful. This brings the dCRT into the realm of fast and powerful model-X methods, where knockoffs is currently the methodology of choice. Knockoffs and dCRT have complementary strengths, which we discuss briefly below.

Model-X knockoffs addresses the variable selection problem, targeting false discovery rate control. It is very computationally efficient, requiring just one high-dimensional model fit. Furthermore, our simulations confirm that knockoffs is quite powerful in several settings. These advantages have led to the successful application of knockoffs to genome-wide association studies (Sesia et al., 2019, 2020). By comparison, the dCRT still requires several high-dimensional model fits and is therefore more computationally costly. On the other hand, dCRT computation benefits from being embarrassingly parallelizable, so modern parallel computing resources can greatly reduce its runtime. As far as power goes, the relative performance of the two methods varies with simulation setting (see Section 4 and Appendix D); neither procedure uniformly dominates the other (when controlling the false discovery rate).

Aside from these considerations, the dCRT provides a few important advantages over knockoffs. The first is that, unlike knockoffs, the dCRT provides p-values (arbitrarily fine-grained and essentially exact) for each conditional independence hypothesis. In addition to providing an interpretable measure of significance, this decoupling of statistical significance quantification from downstream analyses such as multiple testing brings great versatility. Indeed, dCRT p-values can be used for single hypothesis testing, multiple hypothesis testing with a variety of error rates, and any number of other tasks that take p-values as input. While the knockoffs framework has gradually been extended to handle analysis tasks beyond false discovery rate control — e.g. k-family-wise error rate control by Janson & Su (2016), and simultaneous false discovery probability control by Katsevich & Ramdas (2020a) — such extensions require custom solutions and some are currently out of reach (such as single testing or family-wise error rate control). Another advantage of the dCRT is that it has little or no variability across runs. On the other hand, knockoffs is a randomized procedure; this randomization can lead to variability in the performance of the procedure on a given data set; see Appendix D.4 and Figure 4 of Sesia et al. (2019).

The dCRT is therefore a useful addition to the model-X methodology toolbox. Much work still remains to refine this new tool for better power and even faster computation. Indeed, many degrees of freedom in the construction of the dCRT test statistic remain to be explored. For example, should the statistic be based on the fitted coefficient of a variable or on the loss function? What is the best way to test groups of variables? The recent theoretical exploration of the CRT (Katsevich &

Ramdas, 2020b) may help guide the search for powerful test statistics. Another open question is whether there are efficient resampling-free dCRT variants for highly discrete covariates. Finally, the dependence structure of (d)CRT p-values is an important subject for further exploration. We may not always be able to plug-and-play (d)CRT p-values in multiple testing procedures, since their dependency structure is currently unknown. In a related development, Bates et al. (2020) recently proposed a clever method of generating independent HRT p-values for groups of linearly-structured covariates.

Despite these open questions, our initial demonstrations of the dCRT on simulated and real data are quite promising. We are therefore optimistic about the prospects of the dCRT for fruitful practical applications, and look forward to continued improvements in the computational and statistical efficiency of model-X methodology.

Acknowledgements

We would like to thank Siyuan Ma, Wenshuo Wang, Dae Woong Ham, Lu Zhang, Shuangning Li and Emmanuel Candès for helpful discussions and feedback on the paper. We would also like to thank two referees and an associate editor for constructive feedback that helped improve our paper. This work used the Extreme Science and Engineering Discovery Environment (XSEDE) (Towns et al., 2014), supported by National Science Foundation grant ACI-1548562. Specifically, it used the Bridges system (Nystrom et al., 2015), which is supported by NSF award ACI-1445606, at the Pittsburgh Supercomputing Center (PSC).

References

Barber, R. F. & Candès, E. J. (2015). Controlling the false discovery rate via knockoffs. *The Annals of Statistics* **43**, 2055–2085.

Bates, S., Sesia, M., Sabatti, C. & Candès, E. (2020). Causal inference in genetic trio studies. *Proceedings of the National Academy of Sciences* **117**, 24117–24126.

Bellot, A. & van der Schaar, M. (2019). Conditional independence testing using generative adversarial networks. In *Advances in Neural Information Processing Systems*.

Benjamini, Y. & Hochberg, Y. (1995). Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society: Series B* **57**, 289–300.

Benjamini, Y. & Yekutieli, D. (2001). The control of the false discovery rate in multiple testing under dependency. *The Annals of Statistics* **29**, 1165–1188.

Berrett, T. B., Wang, Y., Barber, R. F. & Samworth, R. J. (2020). The conditional permutation test for independence while controlling for confounders. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **82**, 175–197.

Bien, J., Taylor, J. & Tibshirani, R. (2013). A lasso for hierarchical interactions. *The Annals of Statistics* **41**, 1111.

Candès, E., Fan, Y., Janson, L. & Lv, J. (2018). Panning for gold: model-X knockoffs for high dimensional controlled variable selection. *Journal of the Royal Statistical Society: Series B* **80**, 551–577.

Chernozhukov, V., Chetverikov, D., Demirer, M., Duflo, E., Hansen, C., Newey, W. & Robins, J. (2018). Double/debiased machine learning for treatment and structural parameters. *The Econometrics Journal* **21**, C1–C68.

Chipman, H. (1996). Bayesian variable selection with related predictors. *Canadian Journal of Statistics* **24**, 17–36.

Cox, D. R. (1984). Interaction. *International Statistical Review* , 1–24.

Curtis, C., Shah, S. P., Chin, S.-F., Turashvili, G., Rueda, O. M., Dunning, M. J., Speed, D., Lynch, A. G., Samarajiwa, S., Yuan, Y. et al. (2012). The genomic and transcriptomic architecture of 2,000 breast tumours reveals novel subgroups. *Nature* **486**, 346–352.

Davies, R. B. (1980). Algorithm as 155: The distribution of a linear combination of χ^2 random variables. *Journal of the Royal Statistical Society: Series C* **29**, 323–333.

Duchesne, P. & De Micheaux, P. L. (2010). Computing the distribution of quadratic forms: Further comparisons between the Liu–Tang–Zhang approximation and exact methods. *Computational Statistics & Data Analysis* **54**, 858–862.

Friedman, J., Hastie, T. & Tibshirani, R. (2008). Sparse inverse covariance estimation with the graphical lasso. *Biostatistics* **9**, 432–441.

Geyer, F. C., Li, A., Papanastasiou, A. D., Smith, A., Selenica, P., Burke, K. A., Edelweiss, M., Wen, H. C., Piscuoglio, S. & Schultheis, A. M. (2018). Recurrent hotspot mutations in HRAS-Q61 and PI3K-AKT pathway genes as drivers of breast adenomyoepitheliomas. *Nature Communications* **9**, 1–16.

Hamada, M. & Wu, C. J. (1992). Analysis of designed experiments with complex aliasing. *Journal of Quality Technology* **24**, 130–137.

Han, H., Chen, Y., Cheng, L., Prochownik, E. V. & Li, Y. (2016). microRNA-206 impairs c-Myc-driven cancer in a synthetic lethal manner by directly inhibiting MAP3K13. *Oncotarget* **7**, 16409.

Huang, J., Ma, S. & Zhang, C.-H. (2008). Adaptive lasso for sparse high-dimensional regression models. *Statistica Sinica* **18**, 1603–1618.

Huang, X., Xiao, F., Wang, S., Yin, R., Lu, C., Li, Q., Liu, N., Wang, L. & Li, P. (2016). G protein pathway suppressor 2 (GPS2) acts as a tumor suppressor in liposarcoma. *Tumor Biology* **37**, 13333–13343.

Imhof, J. P. (1961). Computing the distribution of quadratic forms in normal variables. *Biometrika* **48**, 419–426.

Janson, L. & Su, W. (2016). Familywise error rate control via knockoffs. *Electronic Journal of Statistics* **10**, 960–975.

Jarmalavicius, S., Trefzer, U. & Walden, P. (2010). Differential arginine methylation of the G-protein pathway suppressor GPS-2 recognized by tumor-specific T-cells in melanoma. *The FASEB Journal* **24**, 937–946.

Katsevich, E. & Ramdas, A. (2020a). Simultaneous high-probability bounds on the false discovery proportion in structured, regression, and online settings. *The Annals of Statistics, to appear* .

Katsevich, E. & Ramdas, A. (2020b). A theoretical treatment of conditional independence testing under model-X. *arXiv preprint arXiv:2005.05506* .

Kirzinger, M. W., Vizeacoumar, F. S., Haave, B., Gonzalez Lopez, C., Bonham, K., Kusalik, A. & Vizeacoumar, F. J. (2019). Humanized yeast genetic interaction mapping predicts synthetic lethal interactions of FBXW7 in breast cancer. *BMC medical genomics* **12**, 112.

Lahti, L., Schäfer, M., Klein, H. U., Bicciato, S. & Dugas, M. (2012). Cancer gene prioritization by integrative analysis of mRNA expression and dna copy number data: a comparative review. *Briefings in Bioinformatics* **14**, 27–35.

Leday, G. G., van der Vaart, A. W., van Wieringen, W. N. & van de Wiel, M. A. (2013). Modeling association between DNA copy number and gene expression with constrained piecewise linear regression splines. *The Annals of Applied Statistics* **7**, 823–845.

Ledoit, O. & Wolf, M. (2004). A well-conditioned estimator for large-dimensional covariance matrices. *Journal of multivariate analysis* **88**, 365–411.

Li, Q., Lai, Q., He, C., Fang, Y., Yan, Q., Zhang, Y., Wang, X., Gu, C., Wang, Y., Ye, L., Han, L., Lin, X., Chen, J., Cai, J., Li, A. & Liu, S. (2019). RUNX1 promotes tumour metastasis by activating the Wnt/β-catenin signalling pathway and EMT in colorectal cancer. *Journal of Experimental & Clinical Cancer Research* **38**, 334.

Liu, F., Zou, Y., Wang, F., Yang, B., Zhang, Z., Luo, Y., Liang, M., Zhou, J. & Huang, O. (2019). FBXW7 mutations promote cell proliferation, migration, and invasion in cervical cancer. *Genetic testing and molecular biomarkers* **23**, 409–417.

Liu, H., Tang, Y. & Zhang, H. H. (2009). A new chi-square approximation to the distribution of non-negative definite quadratic forms in non-central normal variables. *Computational Statistics & Data Analysis* **53**, 853–856.

Nelder, J. (1977). A reformulation of linear models. *Journal of the Royal Statistical Society: Series A* **140**, 48–63.

Nystrom, N. A., Levine, M. J., Roskies, R. Z. & Scott, J. R. (2015). Bridges: A Uniquely Flexible HPC Resource for New Communities and Data Analytics. In *Proceedings of the 2015 XSEDE Conference: Scientific Advancements Enabled by Enhanced Cyberinfrastructure*, XSEDE '15. New York, NY, USA: ACM.

Peixoto, J. L. (1987). Hierarchical variable selection in polynomial regression models. *The American Statistician* **41**, 311–313.

Pereira, B., Chin, S.-F., Rueda, O. M., Vollan, H.-K. M., Provenzano, E., Bardwell, H. A., Pugh, M., Jones, L., Russell, R., Sammut, S.-J., Tsui, D. W. Y., Liu, B., Dawson, S.-J., Abraham, J., Northen, H., Peden, J. F., Mukherjee, A., Turashvili, G., Green, A. R., McKinney, S., Oloumi, A., Shah, S., Rosenfeld, N., Murphy, L., Bentley, D. R., Ellis, I. O., Purushotham, A., Pinder, S. E., Børresen-Dale, A.-L., Earl, H. M., Pharoah, P. D., Ross, M. T., Aparicio, S. & Caldas, C. (2016). The somatic mutation profiles of 2,433 breast cancers refine their genomic and transcriptomic landscapes. *Nature communications* **7**, 11479.

Sesia, M., Katsevich, E., Bates, S., Candès, E. & Sabatti, C. (2020). Multi-resolution localization of causal variants across the genome. *Nature Communications* **11**, 1093.

Sesia, M., Sabatti, C. & Candès, E. J. (2019). Gene hunting with hidden Markov model knockoffs. *Biometrika* **106**, 1–18.

Shah, R. D. & Peters, J. (2018). The hardness of conditional independence testing and the generalised covariance measure. *The Annals of Statistics, to appear* .

Shen, A., Fu, H., He, K. & Jiang, H. (2019). False discovery rate control in cancer biomarker selection using knockoffs. *Cancers* **11**, 744.

Solvang, H. K., Lingjærde, O. C., Frigessi, A., Børresen Dale, A. L. & Kristensen, V. N. (2011). Linear and non-linear dependencies between copy number aberrations and mRNA expression reveal distinct molecular pathways in breast cancer. *BMC bioinformatics* **12**, 197.

Tansey, W., Veitch, V., Zhang, H., Rabadan, R. & Blei, D. M. (2018). The holdout randomization test: Principled and easy black box feature selection. *arXiv preprint arXiv:1811.00645* .

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B* **58**, 267–288.

Tibshirani, R. J. (2013). The lasso problem and uniqueness. *Electronic Journal of Statistics* **7**, 1456–1490.

Towns, J., Cockerill, T., Dahan, M., Foster, I., Gaither, K., Grimshaw, A., Hazlewood, V., Lathrop, S., Lifka, D., Peterson, G. D., Roskies, R., Scott, J. & Wilkins-Diehr, N. (2014). XSEDE: Accelerating Scientific Discovery. *Computing in Science & Engineering* **16**, 62–74.

Zou, H. (2006). The adaptive lasso and its oracle properties. *Journal of the American Statistical Association* **101**, 1418–1429.

Zou, H. & Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B* **67**, 301–320.

Appendix

A Resampling-free distilled CRT

A.1 Resampling-free lasso-based $\mathrm{d_I}$CRT

In this section, we describe the resampling-free version of the lasso-based $\mathrm{d_I}$CRT of Example 2 for gaussian X, in analog to the resampling-free $\mathrm{d_0}$CRT detailed in Section 2.5. We follow the notation of Example 2 and for any $\boldsymbol{a} = (a_1, \ldots, a_n)^\intercal \in \mathbb{R}^n$, let $\mathrm{diag}(\boldsymbol{a})$ denote the diagonal matrix with its (i,i)-th entry being a_i for $i = 1, 2, \ldots, n$. Then in Example 2,

$$\begin{aligned}\hat{\beta}_x =& \left[(\mathbf{1}, \boldsymbol{Z}_{\bullet,\mathrm{top}(k)})^\intercal \mathrm{diag}^2(\boldsymbol{\epsilon}_x)(\mathbf{1}, \boldsymbol{Z}_{\bullet,\mathrm{top}(k)})\right]^{-1} (\mathbf{1}, \boldsymbol{Z}_{\bullet,\mathrm{top}(k)})^\intercal \mathrm{diag}(\boldsymbol{\epsilon}_x)(\boldsymbol{y} - \boldsymbol{d}_y)\\ =& \hat{\mathbb{H}}^{-1}(\mathbf{1}, \boldsymbol{Z}_{\bullet,\mathrm{top}(k)})^\intercal \mathrm{diag}(\boldsymbol{y} - \boldsymbol{d}_y)\boldsymbol{\epsilon}_x,\end{aligned}$$

where $\hat{\mathbb{H}} = (\mathbf{1}, \boldsymbol{Z}_{\bullet,\mathrm{top}(k)})^\intercal \mathrm{diag}^2(\boldsymbol{\epsilon}_x)(\mathbf{1}, \boldsymbol{Z}_{\bullet,\mathrm{top}(k)})$ and $\boldsymbol{\epsilon}_x = \boldsymbol{x} - \boldsymbol{d}_x$. And the test statistics

$$T(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{d}_y, \boldsymbol{d}_x) = \hat{\beta}_{x,1}^2 + \frac{1}{k}\sum_{j=2}^{k+1} \hat{\beta}_{x,j}^2 = \|\hat{\mathbb{H}}^{-1}\tilde{\boldsymbol{Z}}_{\bullet,\mathrm{top}(k)}\boldsymbol{\epsilon}_x\|_2^2$$

where $\tilde{\boldsymbol{Z}}_{\bullet,\mathrm{top}(k)} = (\mathbf{1}, k^{-1/2}\boldsymbol{Z}_{\bullet,\mathrm{top}(k)})^\intercal \mathrm{diag}(\boldsymbol{y} - \boldsymbol{d}_y)$. In analog to the resampling-free $\mathrm{d_0}$CRT introduced in Section 2.5, we replace $\hat{\mathbb{H}}$ with its conditional expectation given $(\boldsymbol{y}, \boldsymbol{Z})$, i.e., $\mathbb{H} = \sigma_x^2(\mathbf{1}, \boldsymbol{Z}_{\bullet,\mathrm{top}(k)})^\intercal(\mathbf{1}, \boldsymbol{Z}_{\bullet,\mathrm{top}(k)})$ with σ_x^2 being the conditional variance of X given Z. Then the test statistics of the resampling-free version of $\mathrm{d_I}$CRT can be constructed as $\|\mathbb{H}^{-1}\tilde{\boldsymbol{Z}}_{\bullet,\mathrm{top}(k)}\boldsymbol{\epsilon}_x\|_2^2$. Conditional on $(\boldsymbol{y}, \boldsymbol{Z})$, it is a quadratic form of the gaussian vector $\boldsymbol{\epsilon}_x$ under the null. Accurate and efficient computational methods have been proposed to handle such problems (see, e.g., Imhof (1961); Davies (1980); Liu et al. (2009)). We use the method proposed by Imhof (1961) and realized by R package CompQuadForm (Duchesne & De Micheaux, 2010) to compute the p-value of $\|\mathbb{H}^{-1}\tilde{\boldsymbol{Z}}_{\bullet,\mathrm{top}(k)}\boldsymbol{\epsilon}_x\|_2^2$.

A.2 Resampling-free dCRT with non-Gaussian X

Let $\Phi(\cdot)$ denote the cumulative distribution function of the standard normal distribution and denote by $\sigma_i^2 = \mathrm{Var}(X_i \mid Z_{i\cdot})$. In Algorithm 3, we describe how to transform non-Gaussian X to be Gaussian with the same conditional variance, so that the resampling-free dCRT (for certain statistics) can be applied. Lemma A1 establishes the properties that make $\boldsymbol{u}$ a good Gaussian stand-in for $\boldsymbol{x} - \boldsymbol{d}_x$,

Algorithm 3 Gaussian transformation.

If X is continuous with conditional cumulative distribution function $F(x \mid Z)$,, let $U_i = \sigma_i\Phi^{-1}(F(X_i \mid Z_i))$ for $i = 1, 2, \ldots, n$.

If X is discrete and supported on $\mathcal{X} = \{a_k : k \in K\}$ where $K \subseteq \mathbb{N}$ is some set of indices, $a_{k_1} < a_{k_2}$ for all $k_1 < k_2$ and $\mathbb{P}(X_i = a_k \mid Z_i) \neq 0$ for all $k \in K$: for $X_i = a_k$, draw V_i uniformly from $\left[\mathbb{P}(X_i < a_k \mid Z_i), \mathbb{P}(X_i \leq a_k \mid Z_i)\right]$ and $U_i = \sigma_i\Phi^{-1}(V_i)$.

Output: $\boldsymbol{u} = (U_1, U_2, \ldots, U_n)^\intercal$.

so that it can be used in a test statistic T in the same way as $\boldsymbol{x} - \boldsymbol{d}_x$ while being amenable to the resampling-free speedup.

Lemma A1. *The U_i ouput by Algorithm 3 are (i) monotonically increasing in X_i, (ii) distributed as $\mathcal{N}(0,\sigma_i^2)$ given Z_i, and (iii) independent from Z_i.*

Proof. For (i), when X_i is continuous, we note that both $\Phi(x)$ and $F(x \mid Z_i)$ are increasing and this implies U_i is unique and monotonically increasing with X_i. When X_i is discrete, noting that the range of V_i does not intersect as X_i takes different values and the range of V_i is increasing with X_i, we again have that U_i is monotonically increasing with X_i.

For (ii), when X_i is continuous, let $V_i = F(X_i \mid Z_i)$ and when X_i is discrete, let V_i be defined as in Algorithm 3. Since V_i is uniformly distributed on $[0,1]$ conditional on Z_i, we have that for any $u \in \mathbb{R}$,

$$\begin{aligned}\mathbb{P}(U_i \le u \mid Z_i) &= \mathbb{P}(\Phi(U_i/\sigma_i) \le \Phi(u/\sigma_i) \mid Z_i) \\ &= \mathbb{P}\left(V_i \le \Phi(u/\sigma_i) \mid Z_i\right) = \Phi(u/\sigma_i),\end{aligned}$$

which indicates that $\mathbb{P}(U_i/\sigma_i \le v \mid Z_i) = \Phi(v)$ and $U_i \sim \mathcal{N}(0,\sigma_i^2)$. Also, we have $\mathbb{P}(U_i/\sigma_i \le v) = \Phi(v) = \mathbb{P}(U_i/\sigma_i \le v \mid Z_i)$ for all $v \in \mathbb{R}$, which indicates that $U_i/\sigma_i \perp\!\!\!\perp Z_i$. □

B Recycling computation for lasso-based distillation

Here, prove Lemma 1 and Theorem 3 from Section 3.2. These lead to Algorithm 4 below.

Algorithm 4 dCRT for L_1-regularized M-estimators.

```
Input: $\boldsymbol{X}, \boldsymbol{y}$, sequence $\lambda(1) > \cdots > \lambda(G) > 0$, loss $\ell$, sequential rule $\widehat{g}$.
 Fit a cross-validated lasso on $(\boldsymbol{X}, \boldsymbol{y})$ to obtain $\widehat{\beta}(\boldsymbol{X}, \boldsymbol{y}; \lambda(\widehat{g}(\boldsymbol{X}, \boldsymbol{y})))$, and record the active set $\mathcal{A}$ as defined in (6).
 For $j \in \mathcal{A}$: Refit the lasso on $(\boldsymbol{X}_{\bullet,-j}, \boldsymbol{y})$ to obtain $\widehat{\beta}_{-j} \equiv \widehat{\beta}(\boldsymbol{X}_{\bullet,-j}, \boldsymbol{y}; \lambda(\widehat{g}(\boldsymbol{X}_{\bullet,-j}, \boldsymbol{y})))$.
 For $j \notin \mathcal{A}$: Set $\widehat{\beta}_{-j} = \widehat{\beta}_{-j}(\boldsymbol{X}, \boldsymbol{y}; \lambda(\widehat{g}(\boldsymbol{X}, \boldsymbol{y})))$.
 For each $j$, let $d_{y,j} = \boldsymbol{X}_{\bullet,-j}\widehat{\beta}_{-j}$.
 Output: Distillations $d_{y,j}$ for each variable $j$.
```

of Lemma 1. Since $\widehat{\beta}(\boldsymbol{X}, \boldsymbol{y}; \lambda)$ is a minimizer of the convex objective

$$\sum_{i=1}^{n} \ell(\boldsymbol{y}_i, \boldsymbol{X}_{i,\bullet}\beta) + \lambda\|\beta\|_1, \tag{8}$$

0 must belong to its subgradient at this point. This means that there exists an $\widehat{s} \in \mathbb{R}^p$ such that

$$\sum_{i=1}^{n} \boldsymbol{X}_{i,j'}\dot{\ell}(\boldsymbol{y}_i, \boldsymbol{X}_{i,\bullet}\widehat{\beta}(\boldsymbol{X}, \boldsymbol{y}; \lambda)) + \lambda \cdot \widehat{s}_{j'} = 0 \quad \text{for all } j' = 1, \ldots, p, \tag{9}$$

where $\widehat{s}_{j'} = \operatorname{sign}(\widehat{\beta}_{j'}(\boldsymbol{X}, \boldsymbol{y}; \lambda))$ if $\widehat{\beta}_{j'}(\boldsymbol{X}, \boldsymbol{y}; \lambda) \neq 0$ and $\widehat{s}_{j'} \in [-1,1]$ otherwise. If $\widehat{\beta}_j(\boldsymbol{X}, \boldsymbol{y}; \lambda) = 0$, then we have

$$\boldsymbol{X}_{i,-j}\widehat{\beta}_{-j}(\boldsymbol{X}, \boldsymbol{y}; \lambda) = \boldsymbol{X}_{i,\bullet}\widehat{\beta}(\boldsymbol{X}, \boldsymbol{y}; \lambda) \text{ for every } i = 1, \ldots, n,$$

which together with equation (9) implies that

$$\text{for all } j' \neq j, \quad \sum_{i=1}^{n} \boldsymbol{X}_{i,j'}\dot{\ell}(\boldsymbol{y}_i, \boldsymbol{X}_{i,-j}\widehat{\beta}_{-j}(\boldsymbol{X}, \boldsymbol{y}; \lambda)) + \lambda \cdot s_{j'} = 0.$$

Therefore, $\widehat{\beta}_{-j}(\boldsymbol{X}, \boldsymbol{y}; \lambda)$ satisfies the first-order optimality condition in the convex problem (3), so it must be a minimizer. Given the assumed general position of the columns of $\boldsymbol{X}$ and the assumptions on ℓ, the minimizer of the problem (3) is unique (Tibshirani, 2013). Therefore, $\widehat{\beta}(\boldsymbol{X}_{\bullet,-j}, \boldsymbol{y}; \lambda) = \widehat{\beta}_{-j}(\boldsymbol{X}, \boldsymbol{y}; \lambda)$, as desired. □

of Theorem 3. Fix $j \notin \mathcal{A}$. For this variable, $\widehat{\beta}_j(\boldsymbol{X}_{-D_k,\bullet}, \boldsymbol{y}_{-D_k}; \lambda(g)) = 0$ for each fold k and for all $g \leq \widetilde{g}(\boldsymbol{X}, \boldsymbol{y})$. By Lemma 1 applied to $(\boldsymbol{X}_{-D_k,\bullet}, \boldsymbol{y}_{-D_k})$, it follows that

$$\widehat{\beta}(\boldsymbol{X}_{-D_k,-j}, \boldsymbol{y}_{-D_k}; \lambda(g)) = \widehat{\beta}_{-j}(\boldsymbol{X}_{-D_k,\bullet}, \boldsymbol{y}_{-D_k}; \lambda(g)) \quad \text{for each fold } k \text{ and all } g \leq \widetilde{g}(\boldsymbol{X}, \boldsymbol{y}).$$

Therefore, the lasso cross-validation errors for $(\boldsymbol{X}, \boldsymbol{y})$ and $(\boldsymbol{X}_{\bullet,-j}, \boldsymbol{y})$ coincide for each $g \leq \widetilde{g}(\boldsymbol{X}, \boldsymbol{y})$:

$$\mathcal{E}_{-j,g} = \sum_{i=1}^{n} \ell(\boldsymbol{y}_i, \boldsymbol{X}_{i,-j}\widehat{\beta}(\boldsymbol{X}_{-D_k,-j}, \boldsymbol{y}_{-D_k}; \lambda(g))) = \sum_{i=1}^{n} \ell(\boldsymbol{y}_i, \boldsymbol{X}_{i,\bullet}\widehat{\beta}(\boldsymbol{X}_{-D_k,\bullet}, \boldsymbol{y}_{-D_k}; \lambda(g))) = \mathcal{E}_g.$$

Because the rule to choose λ is sequential, we conclude that $\widetilde{g}(\boldsymbol{X}_{\bullet,-j}, \boldsymbol{y}) = \widetilde{g}(\boldsymbol{X}, \boldsymbol{y})$ and also $\widehat{g}(\boldsymbol{X}_{\bullet,-j}, \boldsymbol{y}) = \widehat{g}(\boldsymbol{X}, \boldsymbol{y})$. The conclusion (7) now follows from another application of Lemma 1, this time with the full data $(\boldsymbol{X}, \boldsymbol{y})$ and the regularization parameter $\lambda(\widehat{g}(\boldsymbol{X}_{\bullet,-j}, \boldsymbol{y})) = \lambda(\widehat{g}(\boldsymbol{X}, \boldsymbol{y}))$. □

C Comparing dCRT to oCRT

C.1 Setup

To draw a distinction with the oCRT, let us denote by

$$(\widehat{\beta}_x^{\text{lasso}}, \widehat{\beta}_z^{\text{lasso}}) = \underset{\beta_x, \beta_z}{\arg\min}\ \frac{1}{2}\|y - x\beta_x - Z\beta_z\|^2 + \lambda|\beta_x| + \lambda\|\beta_z\|_1$$

the solution to the full lasso problem, while denoting by

$$\widehat{\beta}_z^{\text{loco}} = \underset{\beta_z}{\arg\min}\ \frac{1}{2}\|y - Z\beta_z\|^2 + \lambda\|\beta_z\|_1; \quad \widehat{\beta}_x^{\text{loco}} = \frac{(x - d_x)^\top(y - Z\widehat{\beta}_z^{\text{loco}})}{\|x - d_x\|^2}$$

the solution to the distilled lasso problem. In this notation, we have

$$T_{\text{dCRT}} = |\widehat{\beta}_x^{\text{loco}}| \quad \text{and} \quad T_{\text{oCRT}} = |\widehat{\beta}_x^{\text{lasso}}|. \tag{10}$$

To contrast $\widehat{\beta}_x^{\text{lasso}}$ to $\widehat{\beta}_x^{\text{loco}}$, it is helpful to rewrite the former in a way that parallels the definition of the latter. To this end, note that the KKT conditions for the full lasso problem state that

$$x^\top(y - x\widehat{\beta}_x^{\text{lasso}} - Z\widehat{\beta}_z^{\text{lasso}}) = \lambda \cdot \text{sign}(\widehat{\beta}_x^{\text{lasso}}), \tag{11}$$

where $\text{sign}(\widehat{\beta}_x^{\text{lasso}})$ is defined as any number in $[-1, 1]$ when $\widehat{\beta}_x^{\text{lasso}} = 0$. Rearranging yields

$$\widehat{\beta}_x^{\text{lasso}} = S_{\lambda/\|x\|^2}\left(\frac{x^\top(y - Z\widehat{\beta}_z^{\text{lasso}})}{\|x\|^2}\right), \tag{12}$$

where S_λ is the soft-threshold operator. Using this identity, let us rewrite definition (10) as follows:

$$T_{\text{dCRT}} = \left|\frac{(x - d_x)^\top(y - Z\widehat{\beta}_z^{\text{loco}})}{\|x - d_x\|^2}\right|; \quad T_{\text{oCRT}} = \left|S_{\lambda/\|x\|^2}\left(\frac{x^\top(y - Z\widehat{\beta}_z^{\text{lasso}})}{\|x\|^2}\right)\right|. \tag{13}$$

Having written these test statistics side by side in this way, we observe three differences:

1. T_{oCRT} includes a soft thresholding operation while T_{dCRT} does not.
2. T_{dCRT} uses $x - d_x$ where T_{oCRT} uses x.
3. T_{oCRT} is based on the full lasso coefficients $\widehat{\beta}_z^{\text{lasso}}$, while T_{dCRT} is based on the lasso coefficients $\widehat{\beta}_z^{\text{loco}}$ resulting from holding out the variable x.

We may think of the third difference as being the main one between oCRT and dCRT, but the first and second differences must also be accounted for. In fact, we claim that the first and second differences actually account for the majority of the difference in power between oCRT and dCRT. In the remainder of this section, we examine each of these differences and their implications one at a time.

We illustrate these differences in the context of the following simple numerical simulation setup. Consider $n = p = 800$, with the rows of the design matrix distributed as $N(0, \Sigma)$ for $\Sigma_{ij} = \rho^{|i-j|}$ and the response generated from the Gaussian linear model with $s = 50$ nonzero coefficients. The nonzero coefficients have the same magnitude but alternating signs. We let x represent one of the non-null variables and Z represent all other variables. We consider $\rho = 0$ or $\rho = 0.5$, and we consider the non-null variables either being adjacent to each other or equally spaced.

C.2 The effect of soft thresholding on the oCRT

Note that the soft-thresholding operation in the oCRT can cause the test statistic to be exactly equal to zero sometimes, leading to a p-value of one. We argue that this can only decrease the power of the oCRT.

Let $\lambda' = \lambda/\|x\|^2$ be the value of λ selected by cross-validation and scaled by $\|x\|^2$ and let $W = \frac{x^\top(y - Z\widehat{\beta}_z^{\text{lasso}})}{\|x\|^2}$. Furthermore, let $\widetilde{\lambda}'$ and $\widetilde{W}$ be the corresponding quantities obtained from a resampled dataset $(\widetilde{x}, y, Z)$. Then, we find that

$$\begin{aligned} p_{\text{oCRT}} &= \mathbb{P}\left[|S_{\widetilde{\lambda}'}(\widetilde{W})| \geq |S_{\lambda'}(W)| \ \mid \ x, y, Z\right] \\ &\approx \mathbb{P}\left[|S_{\lambda'}(\widetilde{W})| \geq |S_{\lambda'}(W)| \ \mid \ x, y, Z\right] \\ &\geq \mathbb{P}\left[|\widetilde{W}| \geq |W| \ \mid \ x, y, Z\right]. \end{aligned}$$

The second line rests on the approximation $\widetilde{\lambda}' \approx \lambda'$; i.e. that resampling does not change the scaled penalty parameter by too much. This is a plausible approximation whose rigorous study we defer to future work. The inequality in the third line follows from the observation that for any $w_1, w_2 \in \mathbb{R}$ and $\lambda \geq 0$,

$$|w_1| \geq |w_2| \quad \Longrightarrow \quad |S_\lambda(w_1)| \geq |S_\lambda(w_2)|.$$

In other words, removing the soft threshold from the oCRT test statistic can only decrease the oCRT p-value. We illustrate this in Figure 2, considering the situation when $\rho = 0.5$. We observed that the oCRT version without soft threshold uniformly dominates the original across all our simulations.

C.3 The effect of distilling x

For the purposes of this subsection, we ignore the normalizations in the denominators of T_{oCRT} and T_{dCRT} in equation (13) and remove the soft threshold from T_{oCRT}, leaving us with

$$T_{\text{dCRT}} = |(x - d_x)^\top(y - Z\widehat{\beta}_z^{\text{loco}})| \quad \text{and} \quad T_{\text{oCRT}} = |x^\top(y - Z\widehat{\beta}_z^{\text{lasso}})|. \tag{14}$$

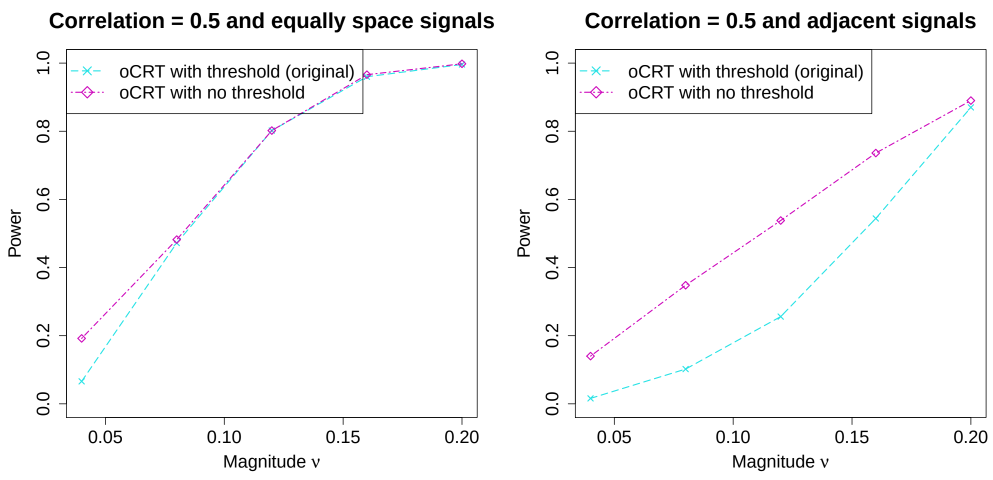


Figure 2: We compare the power of the oCRT with no soft threshold with that of oCRT with soft threshold (used in Candès et al. (2018)) in the simulation with the auto-regressive correlation coefficient $\rho = 0.5$, equally spaced or adjacent signals, and the magnitude of the signals varying, as described in Section 4.1.

Now, recall from Section 2.5 that under the null we have

$$(x - d_x)^\top (y - d_y) \sim N(0, \sigma^2 \|y - d_y\|^2).$$

In particular, the null distribution of $(x - d_x)^\top (y - d_y)$ is centered on the origin. This means that taking absolute values results in a two-tailed test, as usual. However, we observe that the quantity $x^\top (y - Z\widehat{\beta}_z^{\text{lasso}})$ in the oCRT is not necessarily centered on the origin under the null hypothesis. Therefore, taking an absolute value of this quantity to define the oCRT may not result in a usual two-tailed test. This can lead to unintended consequences, as we demonstrate next in the context of our simulation example.

In Figure 3, we compare the dCRT and oCRT obtained from definition (14) in three scenarios: $\rho = 0$ and equally spaced signals, $\rho = 0.5$ and equally spaced signals, and $\rho = 0.5$ and adjacent signals. Across the top row are the powers of these two methods to reject the null hypothesis at level 0.05, and we see that in the first scenario the powers are about equal, in the second scenario the power of the oCRT is larger, and in the third scenario the power of the dCRT is larger. What does this have to do with centering? In the bottom row, we show the test statistics and resampling-based null distributions (prior to taking the absolute value) for a typical run. We observe that the oCRT test statistic and null distribution look very similar to that of the dCRT, up to a horizontal shift (a shift right in the second scenario and a shift left in the third scenario). This horizontal shift would not make any difference for one-sided tests, but in this case we are conducting two-sided tests by applying the absolute value. Therefore, a right-shifted test statistic and null distribution has the effect of decreasing the p-value by reducing the contribution of the left tail. On the other hand, a left-shifted test statistic and null distribution can drastically increase the p-value by exaggerating the contribution of the left tail. The differences in power observed in the top row of Figure 3 seem to reflect these differences in centering.

In sum, this centering issue for the oCRT can either increase or decrease its power, but it is unclear on any given dataset which direction the effect will be. This lack of centering may therefore be undesirable, even if it does in some cases boost power. To remedy this, one can construct a

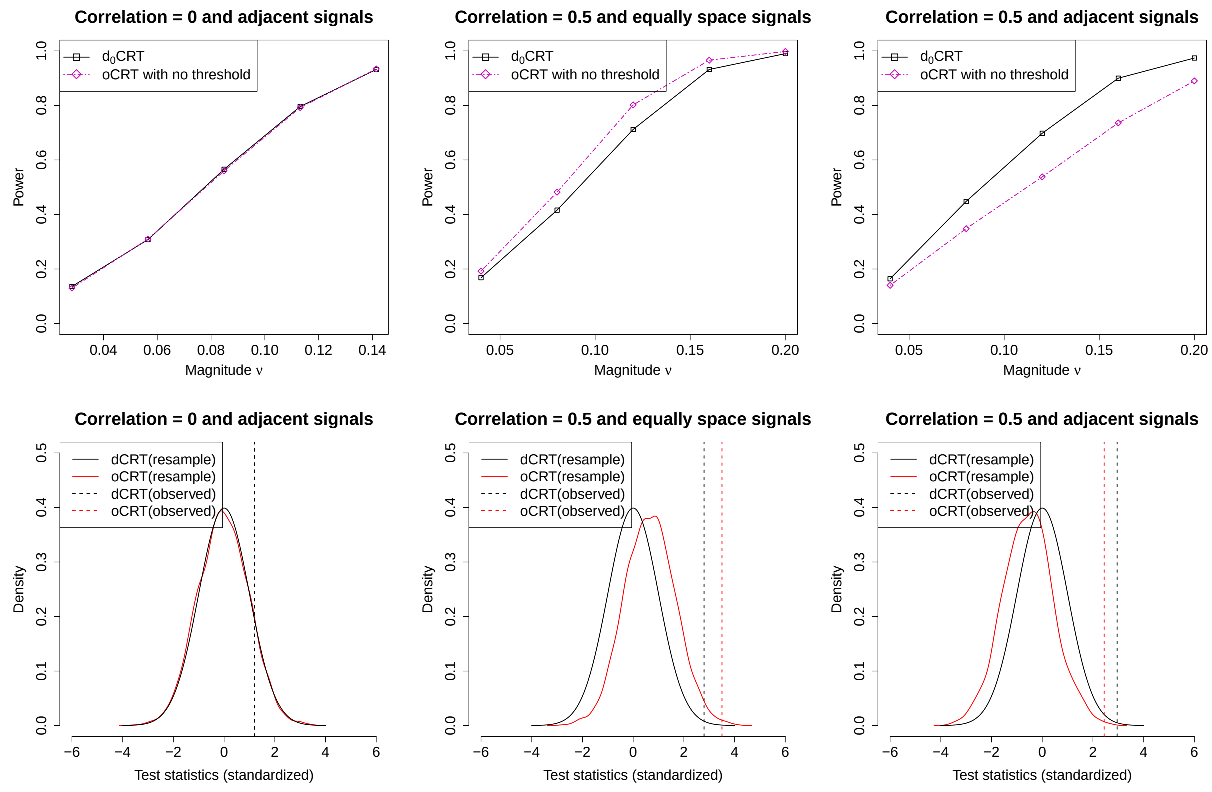


Figure 3: On the top panel, we compare the power of d_0CRT with that of oCRT with no soft threshold, under the different settings described in Section 4.1 including the correlation coefficient $\rho = 0$, $\rho = 0.5$ and equally spaced signals, and $\rho = 0.5$ and adjacent signals, with the magnitude of the signals varying. On the bottom panel, we randomly pick one repetition under each of the three settings, smooth and plot the empirical densities of the resampled d_0CRT and oCRT test statistics, with their observed test statistics indicated by the vertical lines.

bona fide two-tailed p-value:

$$p_{\text{oCRT}} = 2 \cdot \min \left(\mathbb{P}\left[\widetilde{T}_{\text{oCRT}} \geq T_{\text{oCRT}}\right], \mathbb{P}\left[-\widetilde{T}_{\text{oCRT}} \geq -T_{\text{oCRT}}\right] \right). \tag{15}$$

C.4 Comparing the dCRT to the oCRT

We therefore arrive at three versions of the oCRT: original, original but without soft-threshold, and original but without soft-threshold and with centering. In Figure 4, we consider how these three oCRT variants compare to the dCRT.

The most important observation in this figure is that the dCRT performs almost identically to the centered and non-soft-thresholded oCRT. Recalling the three discrepancies between the oCRT and dCRT from the beginning of this section, only the third—the effect on the lasso coefficients of leaving one covariate out—remains. We conclude that the effect of leaving x out of the lasso on the power of the CRT is minimal. In conclusion, the computational acceleration facilitated by distillation comes at little or no cost in power, after properly accounting for the effects of soft thresholding and centering.

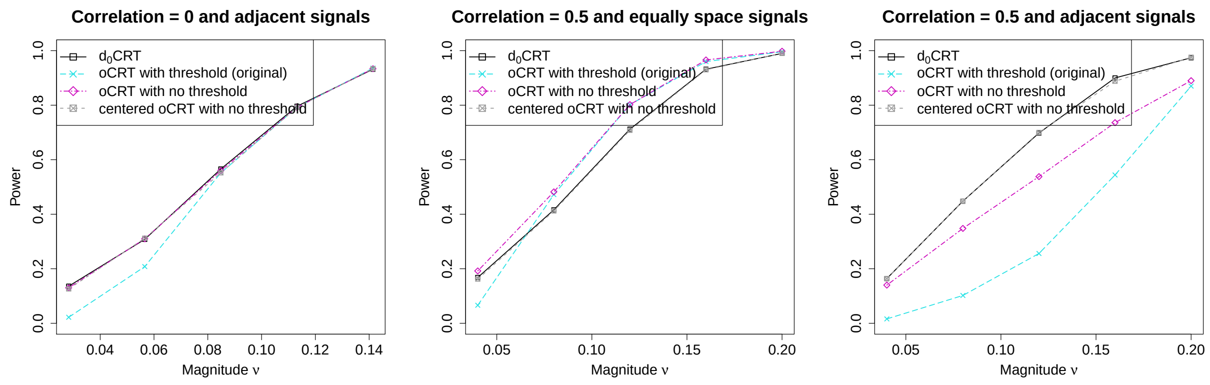


Figure 4: We compare the power of d_0CRT, the oCRT with soft threshold, oCRT without soft threshold, and oCRT without soft threshold and with centering, under the different settings described in Section 4.1 including the correlation coefficient $\rho = 0$, $\rho = 0.5$ and equally spaced signals, and $\rho = 0.5$ and adjacent signals, with the magnitude of the signals varying.

D Simulation results

In this section, we present the details of the simulations summarized in Section 4. Source code for conducting dCRT and other benchmark methods in our simulation studies can be found at `https://github.com/moleibobliu/Distillation-CRT`.

D.1 Method implementation details

We describe here the implementation choices and tuning parameters used for the main methods employed in our simulations; these descriptions apply everywhere to the simulated methods unless specifically stated otherwise. For many of our methods we use the lasso, which is implemented in the R package glmnet with family="binomial" if Y is binary and family="gaussian" otherwise, and penalty parameter selected by 10-fold cross-validation.

The d_0CRT and d_ICRT are the resampling-free versions of Examples 1 and 2, respectively. Note that the resampling-free dCRT may not be the most powerful choice for binary responses; we could have used a resampling-based univariate logistic regression statistic instead, but choose not to for computational purposes.

The dimension k in Examples 2 is set as $k = 2\log(p)$. When we combine the dCRT methods with screening from Section 3.1, the screening is done by running the 10-fold-cross-validated lasso and keeping only the covariates with nonzero fitted coefficients.

The other methods we include are HRT and knockoffs (the last only in simulations targeting false discovery rate control). We implement the HRT of Algorithm 1 in Tansey et al. (2018) with linear model fitted by the lasso, empirical risk function set to logistic loss for binary Y and sum of squared error otherwise, and a data split of 50%-50%. Due to data-splitting, the fitted lasso of the HRT is independent of the data used for hypothesis testing. Thus, the Benjamini–Hochberg and Bonferroni correction procedures can be used on the p-values of the variables selected by the lasso, instead of on the full set of variables. This reduction of the multiplicity burden improves the power of the HRT, and we apply this screening step in all simulations and data analysis. In multiple testing simulations, Benjamini–Hochberg is applied to the p-values of all methods except knockoffs. As we set $p \leq 800$ and false discovery rate level $\alpha = 0.1$ for multiple testing, we set of the number of resamples $M = 50,000$ for the CRT approaches (oCRT, HRT, non-resampling-free dCRTs). This choice was made to ensure these methods' powers are not affected by M, since $M/5 = 10,000 > p/\alpha$, the smallest possible Benjamini–Hochberg cutoff in our simulations. The only exception is in Appendix D.3 where p reached as high as $3,200$, and there we choose $M = 200,000$ to ensure $M/5 > p/\alpha$. For single hypothesis testing simulations where the significance level was 0.05, we set $M = 2,000$ to ensure that we still have $M > 1/0.05$. For knockoffs, we use the lasso coefficient difference statistics as defined in the (3.6) of Candès et al. (2018), the "knockoffs+" threshold, and the SDP knockoff construction when $p < 1000$ and the equicorrelated construction otherwise. We note that for both autocorrelated and equicorrelated variables, SDP and equicorrelated knockoffs are quite similar.

D.2 Moderate size data simulation

We first compare the dCRT with the oCRT procedure in Candès et al. (2018). We generate Gaussian covariates as AR(1), with autocorrelation coefficient 0.5. The true model for Y is chosen as either a Gaussian linear model with unit residual variance or a logistic regression model, and in either case the coefficient vector was set to have s nonzero entries of equal magnitude ν and random signs (each independently having equal probability of being positive or negative). Two types of structures of the coefficient support are considered separately: adjacent support with the first s coefficients being non-zero, and equally-spaced support with the non-zero coefficients indexed by $\{1, [p/s] + 1, 2[p/s] + 1, \ldots, (s-1)[p/s] + 1\}$. We pursue two multiple testing goals of selecting non-null variables while controlling the false discovery rate or family-wise error rate, both at level $\alpha = 0.1$.

In addition to the methods described in Appendix D.1, we implement the oCRT with three different test statistics: the fitted coefficients of a linear or logistic lasso regression, elastic net (Zou & Hastie, 2005) regression (penalty $\lambda(\|\boldsymbol{\beta}\|_1 + \|\boldsymbol{\beta}\|_2^2/2)$), and adaptive lasso regression (Zou, 2006; Huang et al., 2008), each tuned with 10-fold cross-validation. Due to the high computational burden of these oCRTs, we focus on moderate size data with $n = 300$, $p = 300$ and the sparsity level $s = 30$ and vary ν to observe a range of powers.

The resulting average power in the linear and logistic settings is plotted against the signal magnitude ν in Figure 5 for false discovery rate control and Figure 6 for family-wise error rate control, and the false discovery rate plots are presented in Appendix D.14. All methods control the false discovery rate. For both false discovery rate and family-wise error rate control, the $\mathrm{d_0}$CRT and $\mathrm{d_I}$CRT significantly outperform the HRT for both types of support, perform better than knockoffs and all the oCRT methods for adjacent support but worse than them for equally spaced support. The $\mathrm{d_I}$CRT has slightly less power than the $\mathrm{d_0}$CRT due to its allowance of interaction effects, since the true model is exactly additive.

To study and compare the computational efficiency of the methods, we present in Table 2 the average computation time of the methods, with all algorithms implemented in R. Compared with the oCRT procedures, the dCRTs drastically reduce the computation time and are thus much more user-friendly. Knockoffs and the HRT use less time than dCRT because they only fit a high dimensional regression once.

D.3 Large size data simulation

In this section, we conduct simulation studies of a scale beyond the oCRT's computational feasibility, and hence focus on the remaining methods whose computation stays manageable. As a baseline, we set $n = p = 800$, again use AR(1) covariates with autocorrelation 0.5, generate Y from Gaussian linear model with unit residual variance, and use a coefficient vector with $s = 50$ nonzero

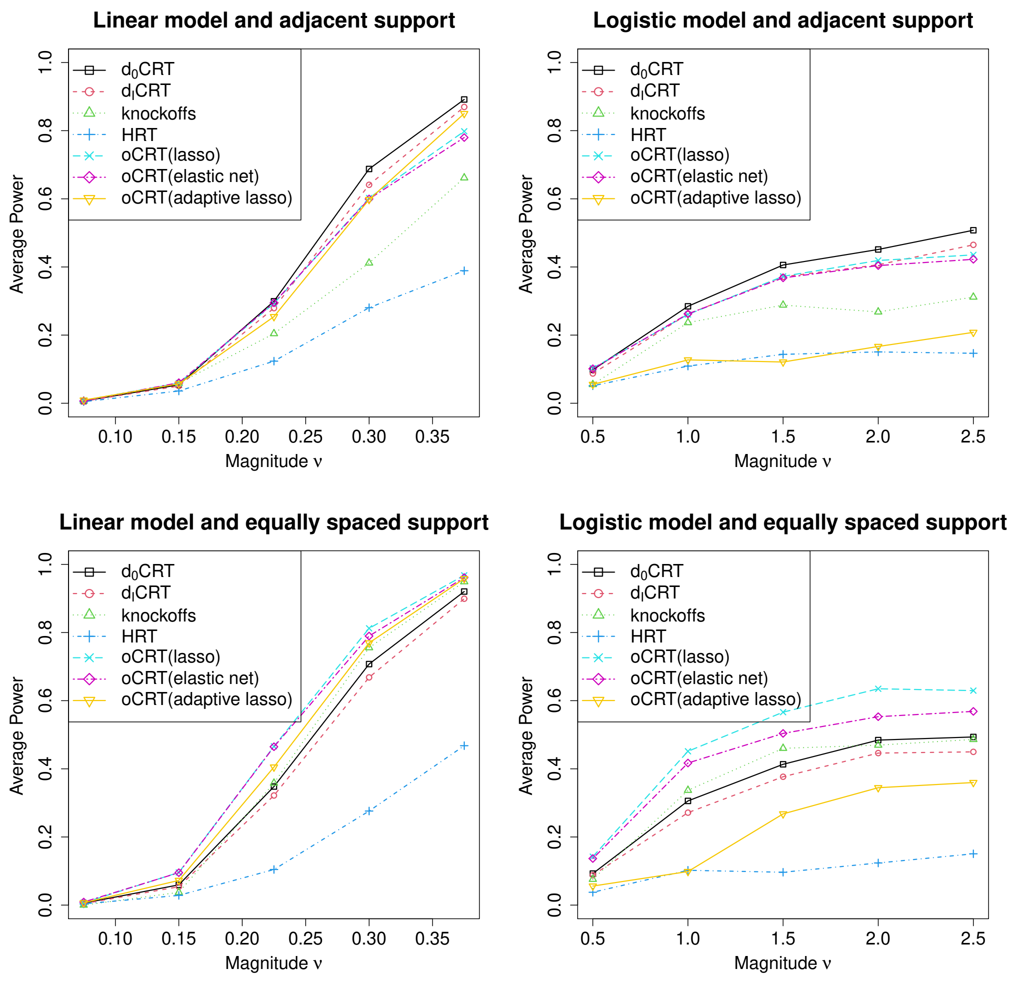


Figure 5: Average powers of false discovery rate control of the $n = p = 300$ simulation of Appendix D.2. All standard errors are below 0.01 and are hence excluded. The dCRT approaches are the most powerful for adjacent support but slightly worse than oCRT methods and knockoffs for equally-spaced support.

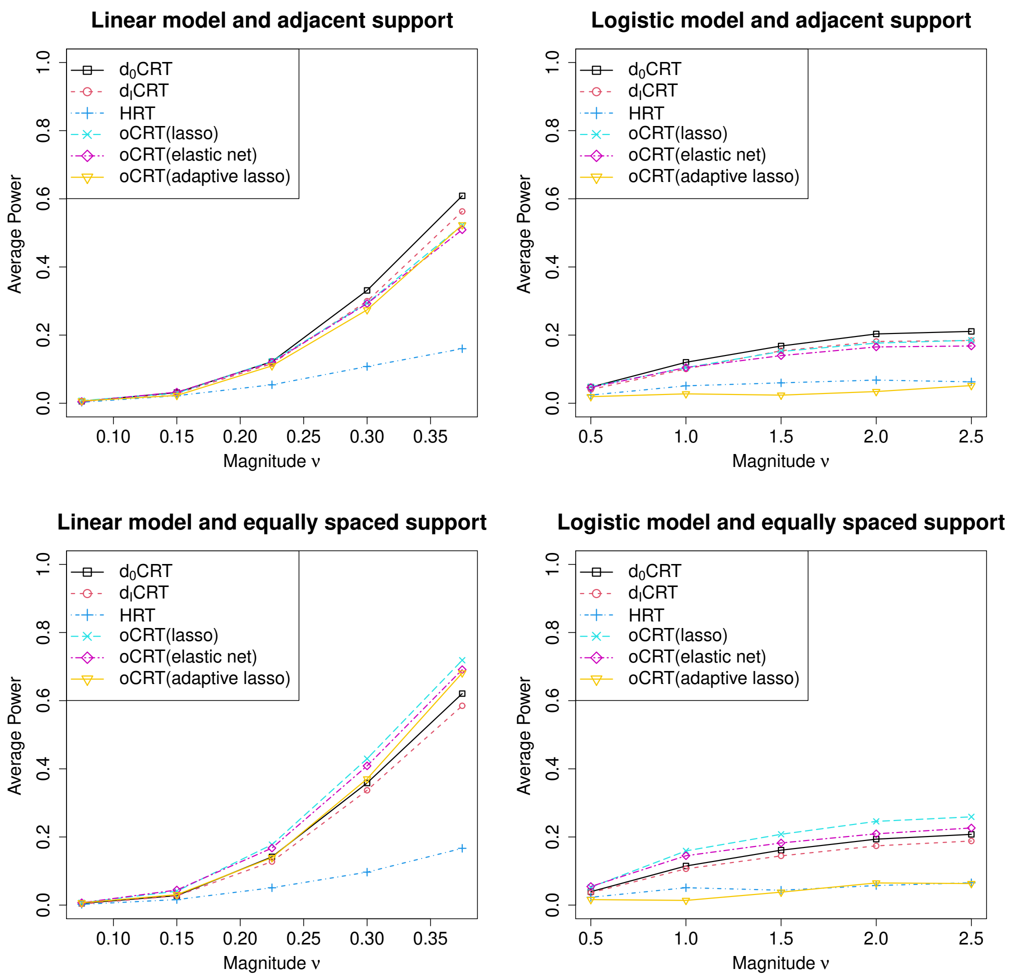


Figure 6: Average powers of family-wise error rate control of the $n = p = 300$ simulation of Appendix D.2. All standard errors are below 0.01 and are hence excluded. The dCRT approaches are the most powerful for adjacent support but slightly worse than oCRT methods for equally-spaced support.

Average computation times (minutes)

	d_0CRT	d_ICRT	knockoff	HRT	oCRT (L)	oCRT (EN)	oCRT (AL)
Linear	0.6	0.6	0.2	0.3	355.9	425.5	378.9
Logistic	1.7	1.8	0.1	0.5	309.0	461.0	391.8

Table 2: Average computation times (in minutes) of the $n = p = 300$ simulations of Appendix D.2. dCRT is much more efficient than the oCRT, and slightly more expensive than knockoffs and HRT. We used abbreviations for Lasso (L), Elastic Net (EN) and Adaptive Lasso (AL).

entries of equal magnitude $\nu = 0.175$ (chosen to make the power around 0.5) and random signs (each independently having equal probability of being positive or negative). Again, the adjacent and equally-spaced supports are studied separately and we pursue controlled variable selection with nominal false discovery rate or family-wise error rate $\alpha = 0.1$.

Each of the four average power plots in Figure 7 varies one the parameters (ν, n, p, or s) from this baseline simulation setup, with the ranges given by the x-axes. The two dCRTs have similar performance, both of them outperform the HRT in all cases for both false discovery rate and family-wise error rate control. They perform better than knockoffs for adjacent support but worse than knockoffs for equally-spaced support under most cases. When the sparsity level s is below 10, the power of knockoffs drops to 0 because of the effect mentioned in Section 4.

We present the average computation times when $n = 800$, $p = 800$ and $s = 50$ in Table 3. Knockoffs and HRT still run faster than the dCRT methods since they only fit high dimensional model once in the whole procedure.

Average computation times (minutes)

d_0CRT	d_ICRT	knockoffs	HRT
9.8	10.2	1.8	6.2

Table 3: Average computation times (in minutes) of the $n = p = 800$ simulations of Appendix D.3. As expected, knockoffs is the fastest, while dCRT is slightly slower than the HRT.

We now study varying the covariate and response models from this same baseline simulation. First we generate Gaussian covariates with covariance structure set as AR(1) with autocorrelations 0.25, 0.5, and 0.75, and equicorrelated with correlations 0.15, 0.3, and 0.45. Second we generate Y from three additional models given by:

(i) **Poisson model:** Y is generated from a Poisson generalized linear model with the same coefficient vector as the baseline model.

(ii) **Logistic model:** Y is generated from a logistic regression with the same coefficient vector as the baseline model except $\nu = 0.5$.

(iii) **Polynomial model:** Y is generated from a Gaussian model with conditional mean given by a polynomial that starts from the baseline model with $\nu = 0.105$ and takes each covariate with a nonzero coefficient and adds a term equal to 0.3 times its cube.

The signal magnitudes of each setting are chosen to make the powers of the main methods close to 0.5, for convenience of comparison. Note that under (i) and (iii), we still fit linear lasso, though the model is wrong. The resulting average powers of both these simulations are plotted in Figure 11. The d_0CRT and d_ICRT have significantly higher power than HRT in all settings. The comparison

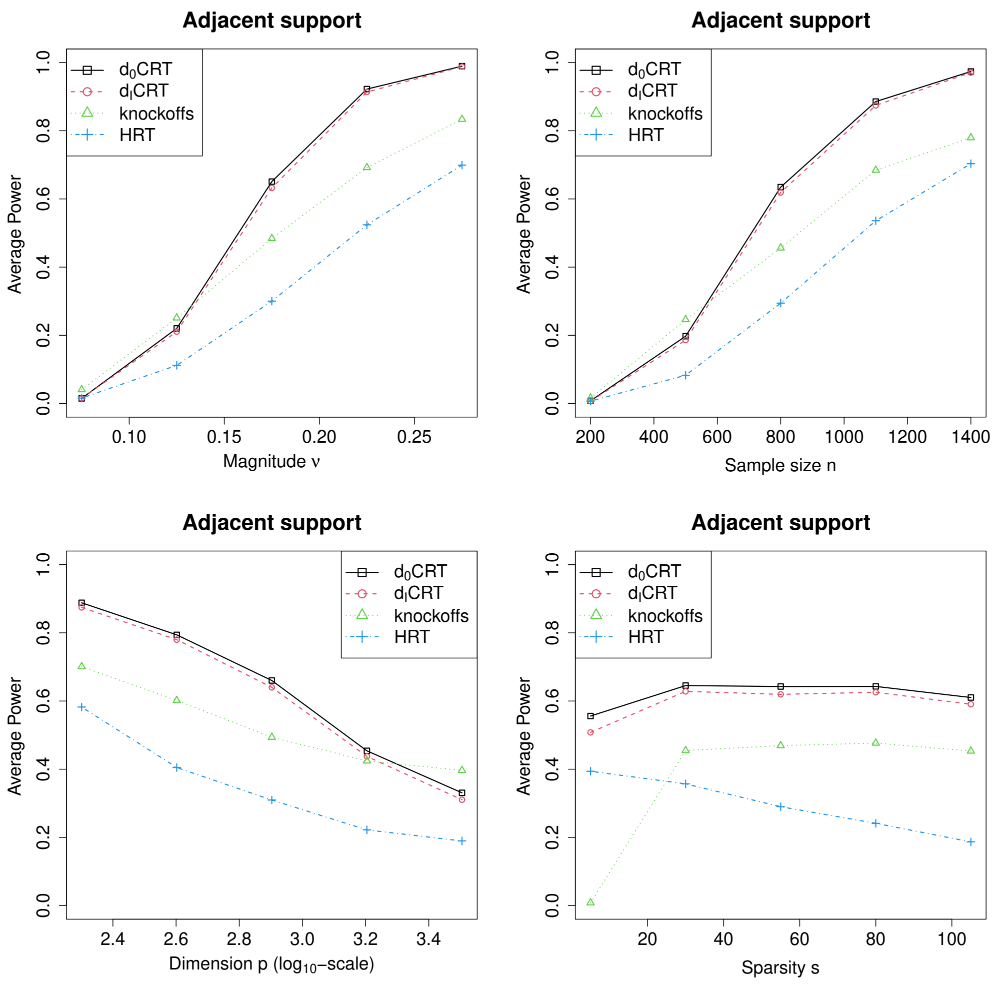


Figure 7: Average powers of false discovery rate control of the large scale simulations of Appendix D.3 that vary the coefficient magnitude, sample size, dimension, and coefficient sparsity with adjacent support. All standard errors are below 0.01. Our dCRT approaches are typically the most powerful.

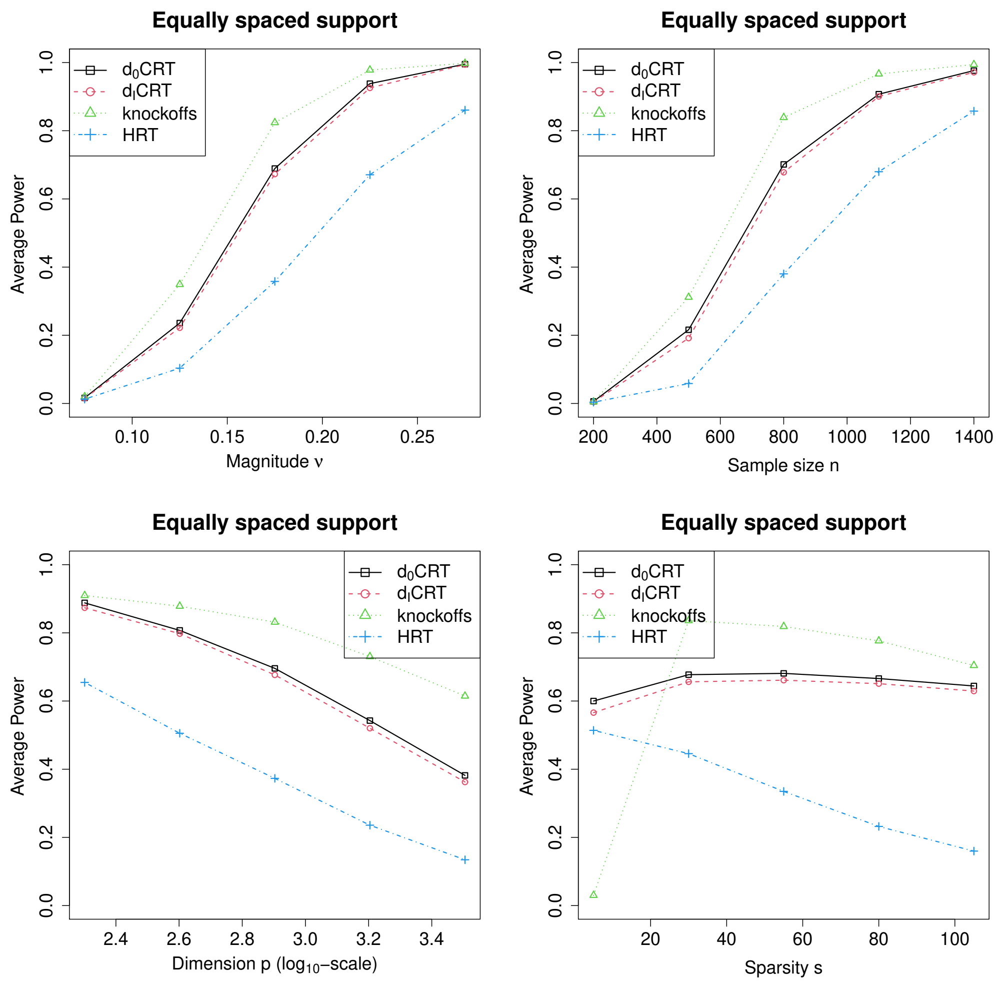


Figure 8: Average powers of false discovery rate control of the large scale simulations of Appendix D.3 that vary the coefficient magnitude, sample size, dimension, and coefficient sparsity with equally spaced support. All standard errors are below 0.01. The dCRTs have lower power than knockoffs in most cases.

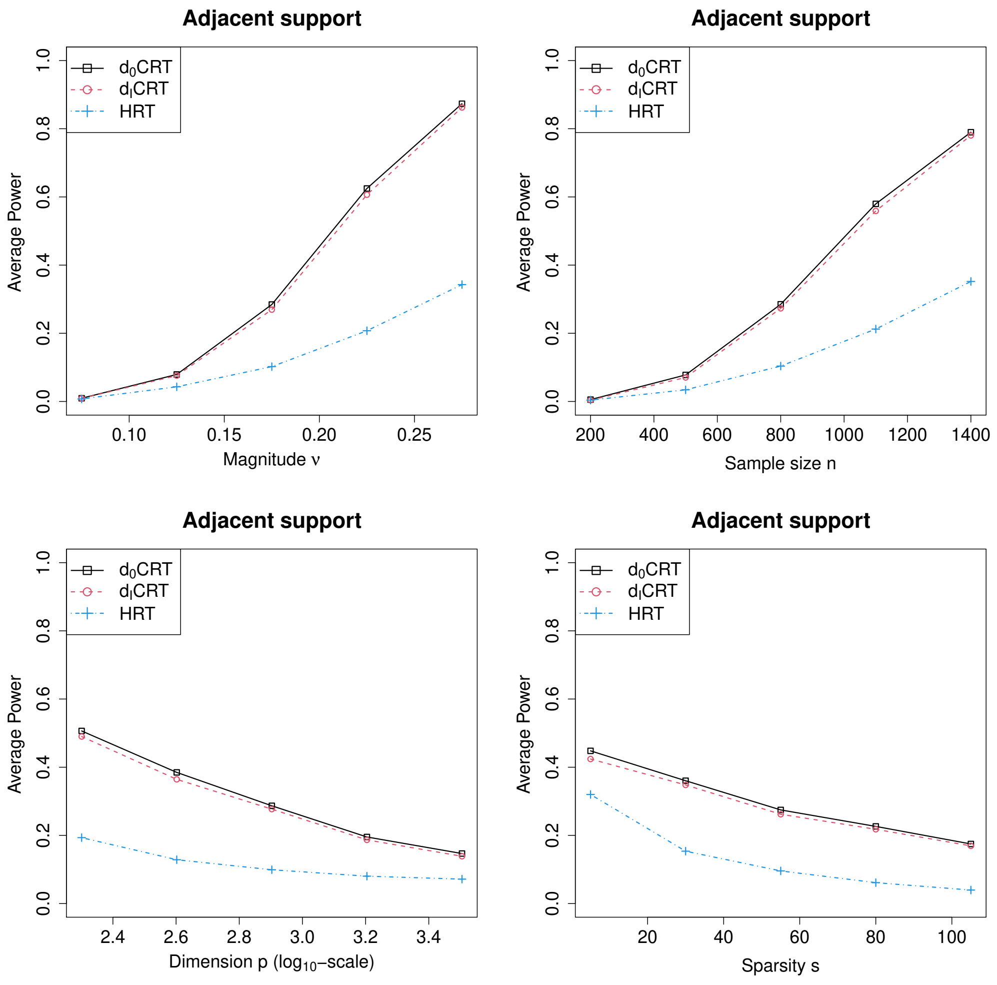


Figure 9: Average powers of family-wise error rate control of the large scale simulations of Appendix D.3 that vary the coefficient magnitude, sample size, dimension, and coefficient sparsity with adjacent support. All standard errors are below 0.01. The dCRTs are more powerful than the HRT.

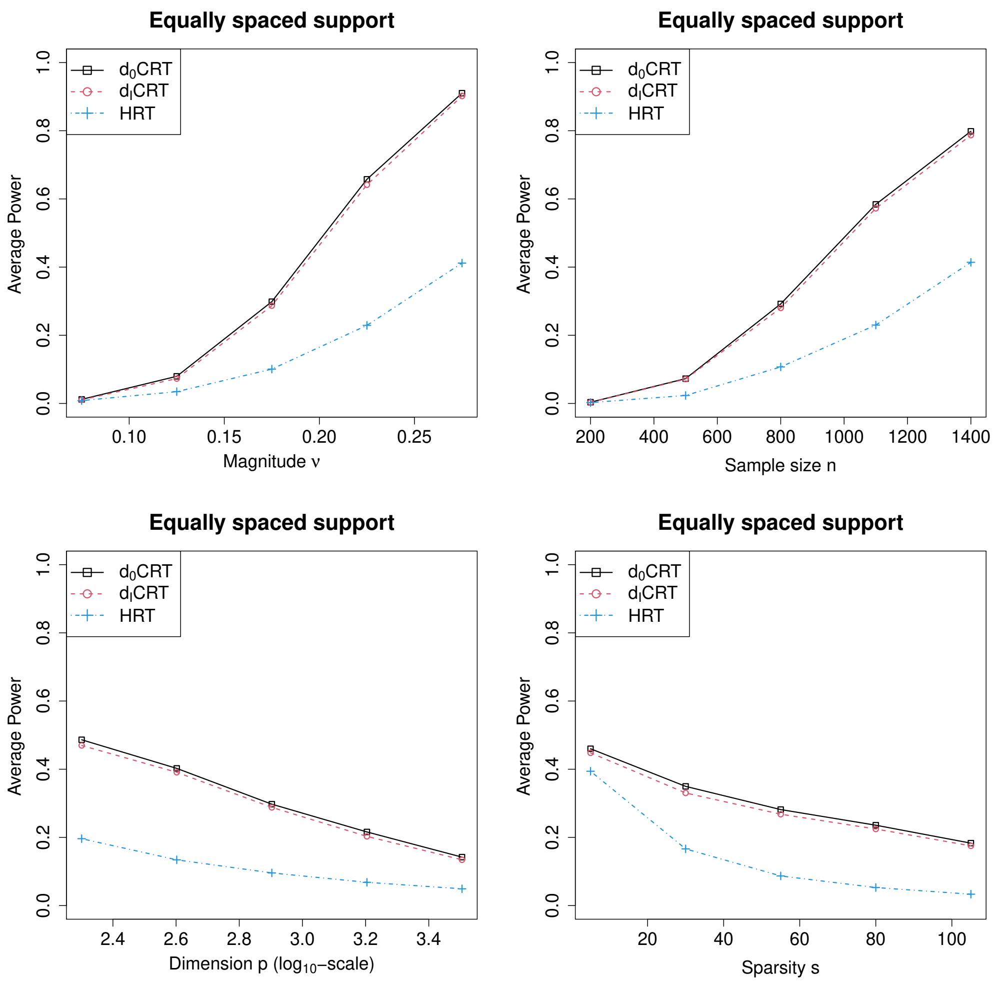


Figure 10: Average powers of family-wise error rate control of the large scale simulations of Appendix D.3 that vary the coefficient magnitude, sample size, dimension, and coefficient sparsity with equally spaced support. All standard errors are below 0.01. The dCRTs are more powerful than the HRT.

to knockoffs is more complicated. Knockoffs generally performs worse under adjacent support and under spaced support with equicorrelated design, suggesting that knockoffs is sensitive to correlations among signal variables. On the other hand, dCRT performs worse under highly autocorrelated designs. These subtle differences are intriguing and we leave their further investigation to future work. Again, all the methods control false discovery rate properly with the desired level 0.1 in the numerical studies corresponding to Figures 7 and 11, and we present the false discovery rate plot in Appendix D.14.

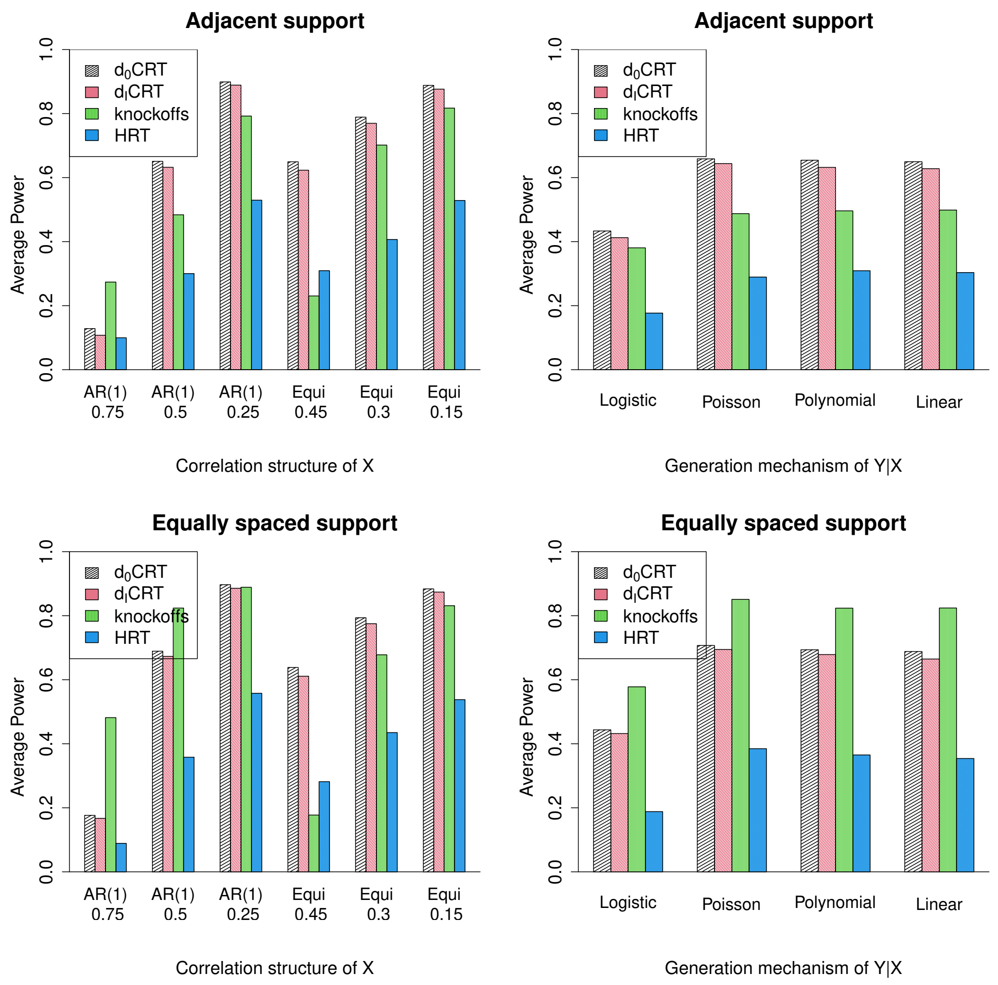


Figure 11: Average powers of the large scale simulations of Appendix D.3 that vary the covariate and response models; all standard errors are below 0.01. The dCRTs have higher power than knockoffs in most scenarios with adjacent support or equicorrelated design, lower power than knockoffs for equally-spaced support and autocorrelated design, and consistently higher power than HRT.

D.4 Assessing algorithmic variability

In addition to assessing power, we also assessed *algorithmic variability* of the methods compared. We define algorithmic variability as the degree to which the rejection set changes when rerunning a method on the same data with different random seeds. We quantified this notion using the following definition. Given data $\boldsymbol{X}, \boldsymbol{y}$, suppose $\mathcal{R}^{(1)} = \mathcal{R}^{(1)}(\boldsymbol{X}, \boldsymbol{y})$ and $\mathcal{R}^{(2)} = \mathcal{R}^{(2)}(\boldsymbol{X}, \boldsymbol{y})$ are the resulting rejection sets based on two random seeds. Their similarity can be measured via the *Jaccard Index*, defined

$$J(\mathcal{R}^{(1)}, \mathcal{R}^{(2)}) \equiv \frac{|\mathcal{R}^{(1)} \cap \mathcal{R}^{(2)}|}{|\mathcal{R}^{(1)} \cup \mathcal{R}^{(2)}|}.$$

This quantity is between 0 and 1, with higher Jaccard Indices representing greater similarity. We define the *stability* of the rejection set as the expectation of this quantity:

$$\text{stability}(\boldsymbol{X}, \boldsymbol{y}) \equiv \mathbb{E}\left[J\left(\mathcal{R}^{(1)}(\boldsymbol{X}, \boldsymbol{y}), \mathcal{R}^{(2)}(\boldsymbol{X}, \boldsymbol{y})\right) \middle| \boldsymbol{X}, \boldsymbol{y}\right]. \tag{16}$$

We assessed this measure of stability in the context of the large data size simulation from Section D.3. We reran each method for each realization of the data with 50 different seeds to obtain 50 different rejection sets $R^{(r)}(\boldsymbol{X}, \boldsymbol{y})$ for $1 \leq r \leq 50$. Then, we approximated the stability via

$$\widehat{\text{stability}}(\boldsymbol{X}, \boldsymbol{y}) \equiv \frac{1}{\binom{50}{2}} \sum_{1 \leq r_1 < r_2 \leq 50} J\left(\mathcal{R}^{(r_1)}(\boldsymbol{X}, \boldsymbol{y}), \mathcal{R}^{(r_2)}(\boldsymbol{X}, \boldsymbol{y})\right).$$

Figure 12 compares the average stability of each method for different values of the signal strength. We see that the dCRT yields the most stable rejection sets (with stability at least 0.9 for all parameter values tested), followed by the HRT, followed by knockoffs. The increase in algorithmic variability for HRT and knockoffs is greater for intermediate values of the signal strength, as one would expect.

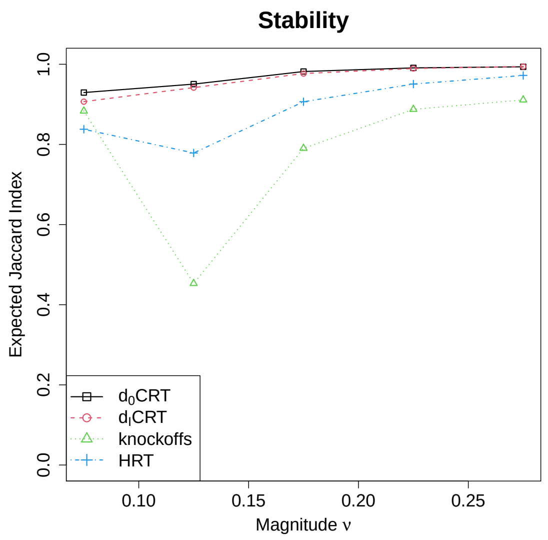


Figure 12: Expected similarity—measured using the Jaccard Index—between the rejection sets resulting from two independent repetitions of each method (see definition (16)). We follow the large scale simulation setting in Appendix D.3 with equally spaced signals and the coefficient magnitude varying.

D.5 Comparing dCRT with double machine learning and generalized covariance measure

As mentioned in the introduction, the test statistic of the resampling-free $\mathrm{d_0CRT}$ is quite similar to that of double machine learning (Chernozhukov et al., 2018) and generalized covariance measure

(Shah & Peters, 2018) when X is Gaussian. However, we remind the reader that both double machine learning and generalized covariance measure rely on asymptotic normality to calibrate their tests, requiring large samples and well-behaved tails for validity. By comparison, CRT-based methods including the dCRT and HRT are valid in finite samples for any data distribution.

We demonstrate this difference by setting $n = 30$, $p = 100$ and drawing each covariate independently from a Laplace distribution with mean 0 and variance 2/9. We generate Y from a linear model with the residual ϵ also drawn from the Laplace distribution of mean 0 and variance 1/2. Our target is again multiple testing with false discovery rate level 0.1. We compare HRT, double machine learning, generalized covariance measure and dCRT. When implementing double machine learning and generalized covariance measure, we use our assumed exact model-X knowledge to construct the exact partial residual for each covariate, and use the lasso on $(\boldsymbol{y}, \boldsymbol{Z})$ to obtain the partial residuals for Y. For double machine learning, we use 8-fold cross-fitting. The resulting false discovery rates are presented in Figure 13. Both double machine learning and generalized covariance measure have false discovery rate level substantially above the nominal 0.1 under all magnitudes, while HRT and dCRT still control the false discovery rate below 0.1 (as guaranteed by Theorem 1).

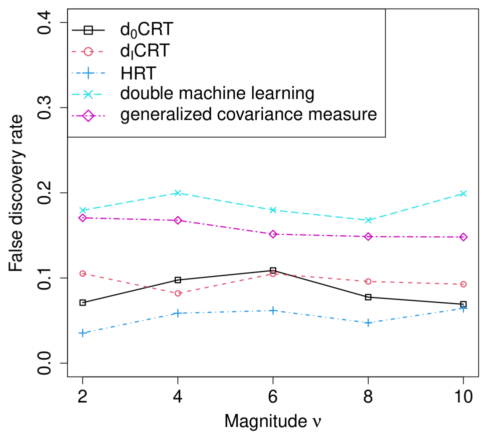
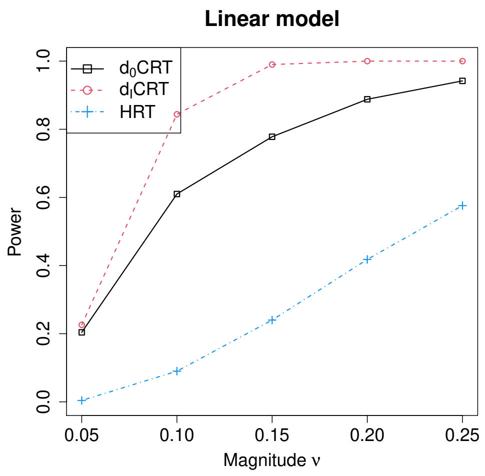


Figure 13: False discovery rates of the simulation in Appendix D.5 comparing computationally efficient CRT methods to double machine learning and generalized covariance measure. All standard errors are below 0.01. double machine learning and generalized covariance measure do not control the false discovery rate at the target level 0.1, but the other methods do.

D.6 Power improvement of the $\mathrm{d_I}$CRT in the presence of interactions

All previous simulations have shown similar, if slightly worse, performance for the $\mathrm{d_I}$CRT compared to the $\mathrm{d_0}$CRT. This is because the models have all been additive (technically a logistic regression model is not additive, but the logistic-regression-derived statistics used by both dCRT methods fit to the logistic-transformed Y, which does follow an additive model). To demonstrate the benefits of the $\mathrm{d_I}$CRT to characterize more complex effects, we conduct here a non-additive simulation with first-order interactions that obey the hierarchy principle described in Section 2.4. We take $n = p = 800$ and generate $(X, Z^\intercal)^\intercal$ from AR(1) with autocorrelation 0.5. Letting $\mu(X, Z) = \nu(X + \sum_{k=1}^{5} Z_{j_k} + 1.5X \sum_{k=1}^{5} Z_{j_k})$ with $j_1, \dots, j_5$ randomly picked from $\{1, 2, \dots, 799\}$, we generate Y either from a Gaussian model with conditional mean given by $\mu(X, Z)$ or from a Bernoulli model with $\log(\mathbb{P}(Y = 1 \mid X, Z)/\mathbb{P}(Y = 0 \mid X, Z)) = \mu(X, Z)$. The target is to test the single hypothesis $Y \perp\!\!\!\perp X \mid Z$ at level 0.05 (hence knockoffs does not apply). Figure 14 shows the powers of the $\mathrm{d_0CRT}$, $\mathrm{d_ICRT}$, and HRT. As is expected, $\mathrm{d_ICRT}$ has substantially higher power than $\mathrm{d_0CRT}$.



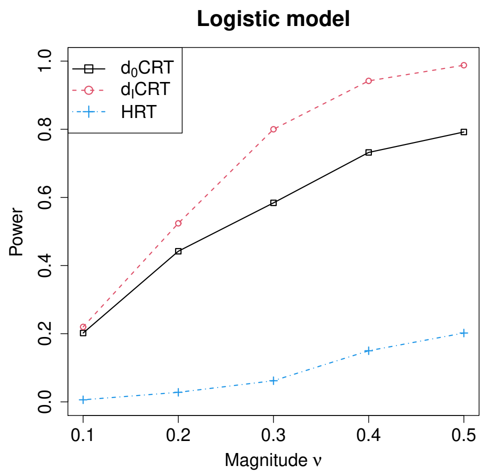


Figure 14: Powers of the simulations in Appendix D.6 comparing methods in the presence of interactions; standard errors are below 0.03. The dCRT is much more powerful than the HRT.

D.7 Stability of the $\mathrm{d_I}$CRT to the choice of k

In this section, we study the sensitivity of $\mathrm{d_ICRT}$ to the choice of k defined in Example 2. We simulate the $\mathrm{d_ICRT}$ in the baseline setting (linear model) of Section D.3 and the linear interaction model setting of Section D.6 for varying choices of k (in both settings, the default $k = 2\log(p) \approx 13$). The results in Figure 15 show that the choice of k has nearly no impact on the power of the $\mathrm{d_ICRT}$ in the linear model setting. In the interaction setting, the power of the $\mathrm{d_ICRT}$ decreases with k for $k > 5$ since there are only 5 true interactions in the model, but the trend is quite gradual and the $\mathrm{d_ICRT}$'s power stays above that of $\mathrm{d_0CRT}$ through $k = 22$.

D.8 A random-forest-based $\mathrm{d_I}$CRT

Examples 1 and 2 are inherently rooted in generalized linear models, and we expect them to perform well in situations where a generalized linear model captures much of the interesting dependence between Y and X. But there is nothing limiting the dCRT's application to such settings, and in this section we demonstrate the power of a random-forest-based $\mathrm{d_ICRT}$ in a setting that is far from a generalized linear model.

Example 3 (Random-forest-based $\mathrm{d_ICRT}$)**.** *Let $\boldsymbol{d}_{y,1}$ be the fitted predictions from a random forest fitting $\boldsymbol{y}$ to $\boldsymbol{Z}$, let $\boldsymbol{d}_{y,-1}$ be the columns of $\boldsymbol{Z}$ corresponding to the k largest values of the default variable importance measure in the R package randomForest, and let $T(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{d}_y, \boldsymbol{d}_x)$ fit a random forest of $\boldsymbol{y}$ on $\boldsymbol{x} - \boldsymbol{d}_x$ and $\boldsymbol{d}_y$ and return the default variable importance measure for $\boldsymbol{x} - \boldsymbol{d}_x$.*

We take $n = p = 800$ and $(X, Z^\intercal)^\intercal$ as following an AR(1) model with autocorrelation 0.5. We choose a conditional model for Y in which the magnitude of the effect of X on Y is heterogeneous and varies with Z: $\mu(X, Z) = \nu[0.5X^2 + \sin(0.5\pi X)](0.3 + \sum_{k=1}^{5} Z_{j_k})$, and Y is standard normal noise added to $\mu(X, Z)$. We simulate tests of $Y \perp\!\!\!\perp X \mid Z$ at significance 0.05 and plot the results in

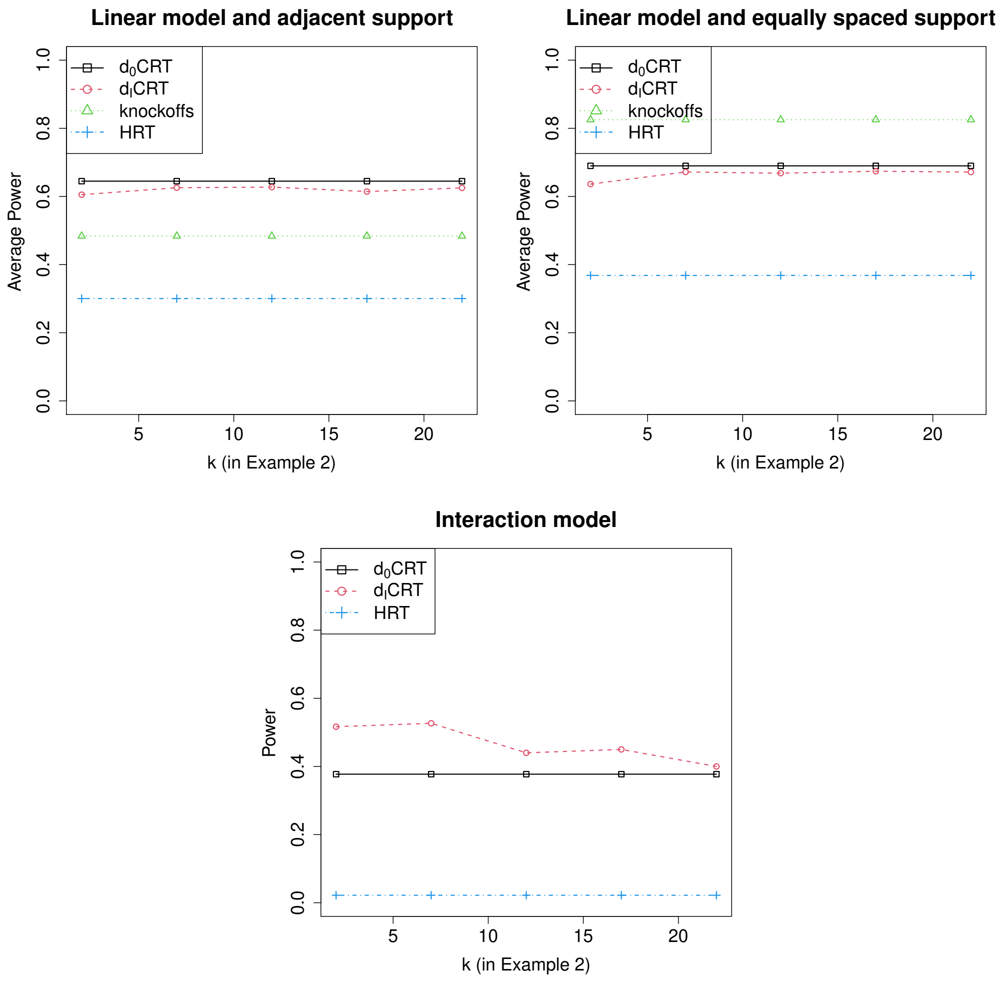


Figure 15: Powers of the simulation in Appendix D.7 evaluating the stability of the d_ICRT to the choice of k; all standard errors are below 0.03. The d_ICRT is fairly stable to the choice of k.

Figure 16. The random-forest-based $\mathrm{d_I CRT}$ is denoted by "$\mathrm{d_I CRT}$ (forest)" and uses 100 trees for distillation and 30 trees for computation of T. The additional function-approximation flexibility of random forests imparts a substantial gain in power compared to $\mathrm{d_0 CRT}$, $\mathrm{d_I CRT}$, HRT, which are all implemented based on generalized linear models.

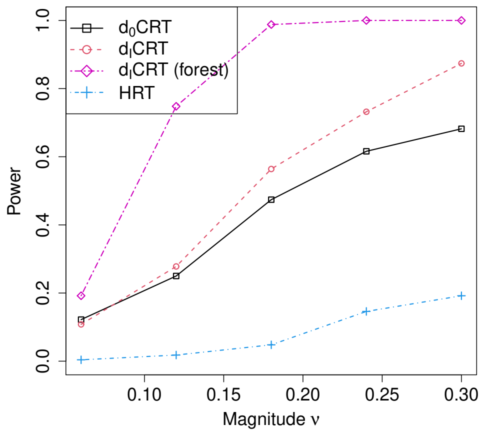


Figure 16: Powers of the simulations in Appendix D.8 demonstrating a random-forest-based $\mathrm{d_I CRT}$ on a complex ground truth model; all standard errors are below 0.03. The flexibility of the random forest variant makes it much more powerful than the generalized linear model-based $\mathrm{d_0 CRT}$ and $\mathrm{d_I CRT}$.

D.9 Robustness: Known first and second moments

We designed numerical experiments to study the robustness of the dCRTs, i.e., whether the methods still control Type-I error and have power when the $X \mid Z$ distribution is misspecified. We first consider the case when one has no knowledge of the conditional distribution of $X \mid Z$ except its first two moments, and simply treats $X \mid Z$ as conditionally Gaussian with matching moments. We let $n = p = 800$ and generate $Z = (Z_1, Z_2, \dots, Z_{799})^\intercal$ from a Gaussian AR(1) model with autocorrelation 0.5 and sample X as conditionally Poisson:

$$X = 0.15 \sum_{j=1}^{50} \varphi_j Z_j + \delta, \quad \text{where} \quad \delta = \frac{O - r}{\sqrt{r}} \quad \text{with} \quad O \mid Z \sim \mathrm{Poi}(r),$$

where each φ_j is independently and uniformly drawn from $\{-1, 1\}$ and $\mathrm{Poi}(r)$ represents the Poisson distribution with mean r. When r is small, $(O-r)/\sqrt{r}$ is quite skewed with its tail behaviour highly different from Gaussian while as r becomes larger, $(O-r)/\sqrt{r}$ converges to a $\mathcal{N}(0,1)$. We run the dCRT as if $\delta \sim \mathcal{N}(0,1)$, and hence r measures the degree of misspecification (lower r corresponds to more misspecification). For Y, we use linear or logistic model linked with $\nu X + 0.15 \sum_{j=1}^{50} \psi_j Z_j$.

Our target is to test for $Y \perp\!\!\!\perp X \mid Z$ with level 0.05. To study the performance in Type-I error control we set $\nu = 0$, while to study the power we let $\nu = 0.1$ for linear model and $\nu = 0.2$ for logistic model. We compare $\mathrm{d_0 CRT}$, $\mathrm{d_I CRT}$, and HRT, with the same specification as the previous section except that $X - \mathbb{E}[X|Z]$ is approximated as $\mathcal{N}(0,1)$ when modelling X. The resulting Type-I error and power versus $\log_2(r)$ are plotted in Figure 17. Even when r is as small as 0.5, the Type-I error of the dCRTs remain below their nominal level and their powers are relatively similar to the nearly-well-specified setting of $r = 64$.

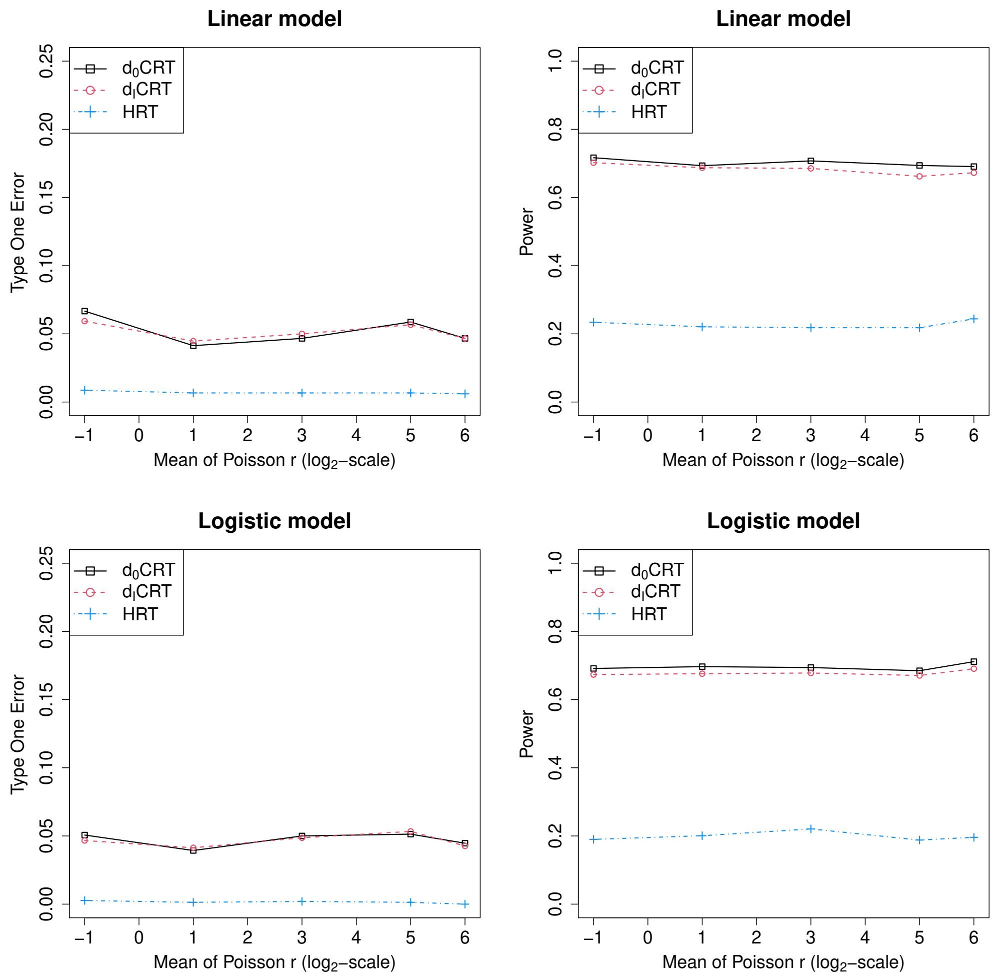


Figure 17: Type-I error rates and powers of the simulations in Appendix D.9 measuring robustness to misspecification in terms of the parameter r. All standard errors are below 0.03. All methods control Type-I error, and the dCRTs are more powerful than HRT.

D.10 In-sample-estimated moments

Next, we study the case when one knows a model family for $X \mid Z$ but needs to estimate its parameters in-sample. Again, we set $n = 800$, $p = 800$, $s = 50$, generate the covariates from a Gaussian AR(1) distribution with autocorrelation 0.5 and generate Y from a linear model with magnitude $\nu = 0.175$ or logistic model with magnitude $\nu = 0.5$, which again makes the power roughly 0.5. Again, we study adjacent and equally spaced supported signals separately. Then, as part of our dCRT procedures, we use the $n = 800$ samples to estimate the conditional distribution parameters of the covariates. For this purpose, we consider three commonly used approaches as the options:

(i) **The Ledoit–Wolf estimator:** We follow Ledoit & Wolf (2004) to obtain an estimate of the covariance matrix of the covariates which is the optimal (in terms of mean square error) linear shrinkage of the sample empirical covariance to the identity matrix. Letting the resulting estimator be $\widehat{\Sigma}$, we actually use a rescaled version given by $D\widehat{\Sigma}D$, where $D = \text{diag}\{d_1, d_2, \cdots, d_p\}$ is a $p \times p$ diagonal matrix and d_j is the ratio of the estimated conditional variance by inverting $\widehat{\Sigma}$ and that estimated using the mean squared residuals. Here, the multiplier D serves to de-bias the estimated conditional variances of the covariates, and this was important since this determined the conditional variance of the resampled covariates.

(ii) **Graphical lasso:** We implement the graphical lasso (Friedman et al., 2008) tuned by cross-validation to estimate the precision matrix and invert it to estimate the covariance matrix of the covariates. Again, we rescale that estimate in the same way as in (i) using D.

(iii) **Nodewise lasso:** For each covariate X, we fit the lasso tuned by cross-validation to model its conditional mean given Z, and use the resulting mean squared regression residuals to estimate the conditional variance of $X \mid Z$. Knockoffs is not included in this case since this nodewide lasso, applied to each covariate in turn, does not in general provide a coherent covariance matrix, and a covariance matrix is required to generate Gaussian knockoffs.

The goal of the simulation is controlled variable selection with false discovery rate level 0.1. The resulting false discovery rate and average power under different models and the above estimation strategies of $X \mid Z$ are presented in Figure 18 and Figure 19, respectively. When the conditional distribution parameters are estimated by graphical lasso or nodewise lasso, false discovery rates of all the methods are well controlled by the nominal level under all the model and signal space settings. Also, their powers are close to the ideal model-X case presented in the right panel of Figure 11. With Ledoit–Wolf estimation, the false discovery rates of the two dCRT methods increase only very slightly above nominal.

D.11 Measuring the effect of the resampling-free modification: Gaussian covariates

Section 2.5 proposes a resampling-free version of d_0CRT and d_ICRT requiring a small modification to their test statistics; we show here this modification does not affect their powers. Under the baseline setting of Section D.3 and the setting with Gaussian covariates and interactions in Section D.6, we compare the resampling-free dCRTs with their non-resampling-free versions in terms of average power. Figure 20 shows the resampling-free modification makes essentially no difference to their powers.

We also include computation times for the baseline setting of Section D.3 in Table 4, showing that the resampling-free versions of the dCRTs confer a substantial computational savings.

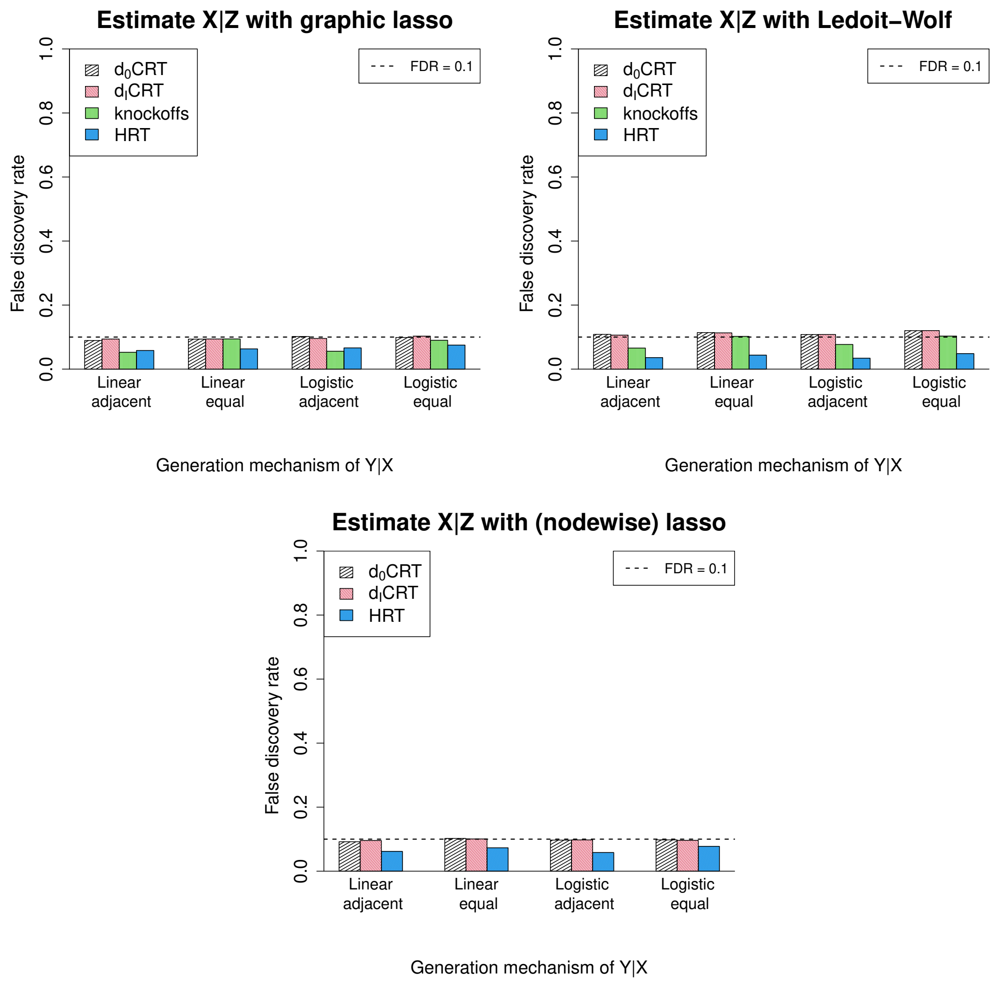


Figure 18: False discovery rates of the simulations in Appendix D.10 measuring robustness to in-sample estimation of the covariate covariance matrix obtained via three common approaches, with the model for Y varying. All standard errors are below 0.01. The false discovery rates of the dCRT remain close to the nominal level 0.1 in all settings.

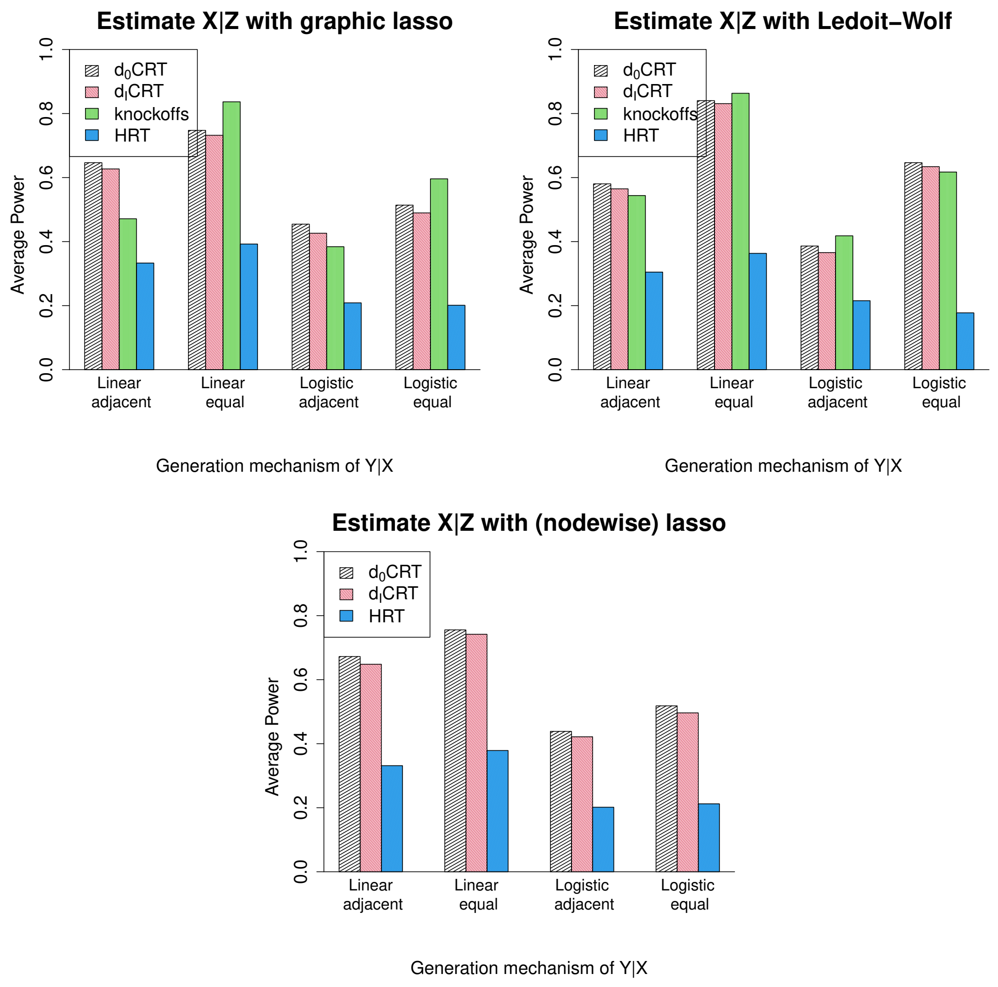


Figure 19: Average power of the simulations in Appendix D.10 measuring robustness to in-sample estimation of the covariate covariance matrix obtained via three common approaches, with the model for Y varying. All standard errors are below 0.01.

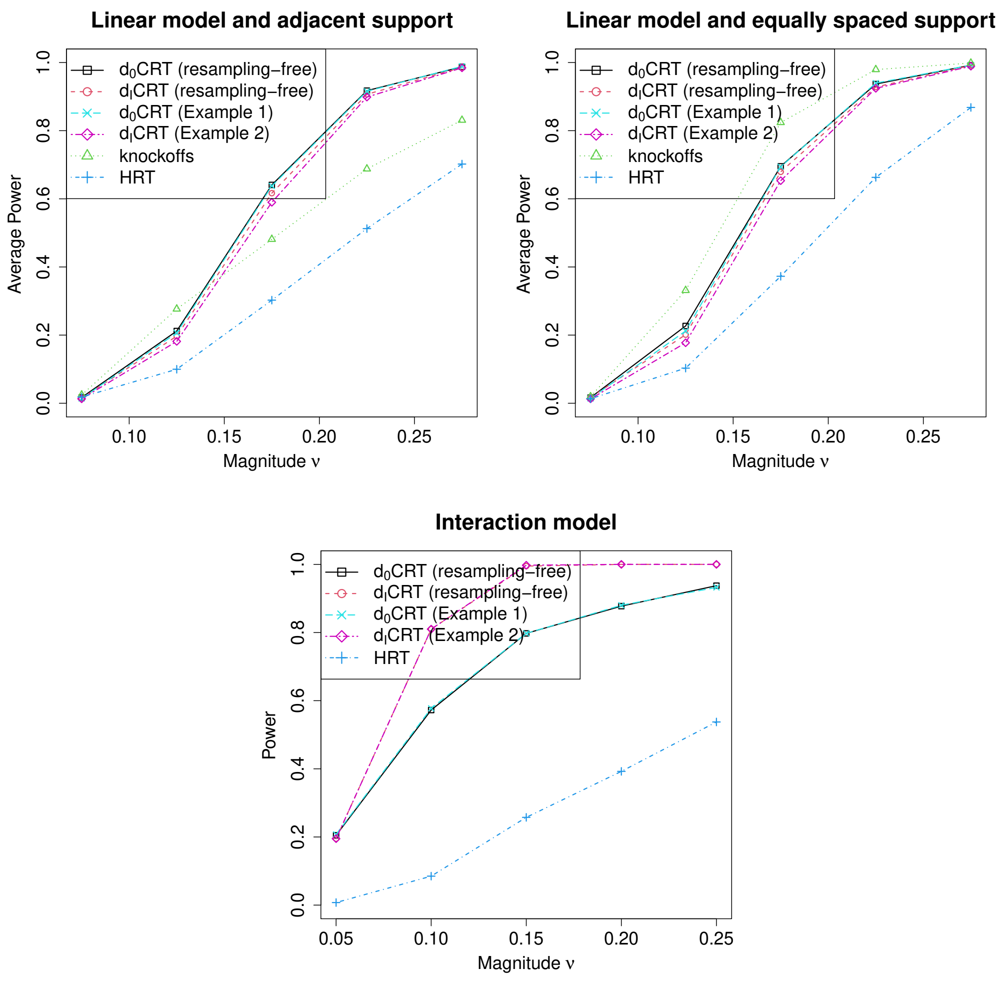


Figure 20: Powers of the simulation in Appendix D.11 measuring the effect of the resampling-free modification to the d_0CRT and d_ICRT test statistics; all standard errors are below 0.03. The resampling-free modifications of the dCRT have essentially the same power as the resampled versions.

Average computation times (minutes)

d_0CRT (resampling-free)	d_0CRT (Example 1)	d_ICRT (resampling-free)	d_ICRT (Example 2)	knockoff	HRT
9.8	26.1	10.2	120.7	1.8	6.2

Table 4: Average computation times of the linear model simulations of Appendix D.11. The resampling-free modifications of the dCRT lead to considerable runtime savings.

D.12 Measuring the effect of the resampling-free modification: Non-Gaussian covariates

As we introduced in Section 2.5 and detailed in Appendix A, when $X \mid Z$ is non-Gaussian, it must be transformed to Gaussian in order to apply the resampling-free speedup; we examine here the effect this transformation has on power. We generate covariates i.i.d. from two different distributions: (i) Gamma with shape 3 and rate 0.5 and (ii) Bernoulli with mean 0.5. We took $n = p = 800$, $s = 50$ and Y generated from linear (in the untransformed covariates) model and performed multiple testing for variable selection at false discovery rate level 0.1. Our main goal is to compare the d_0CRT of Example 1 and the d_ICRT of Example 2 with their respective resampling-free counterparts, though we also run the HRT and knockoffs. The resulting average powers versus signal strength ν are shown in Figure 21. For Gamma X, the Gaussian transformation comes with almost no loss in power while for Bernoulli X, the resampling-free dCRTs lose substantial power but still outperform the HRT. This is due to the highly non-Gaussian nature of a Bernoulli(0.5) distribution and the need for substantial exogenous randomness to be added to X to make it Gaussian. Knockoffs performs competitively with the dCRT methods in both simulations, and we attribute this to the covariate independence which allows very high-quality knockoffs to be used.

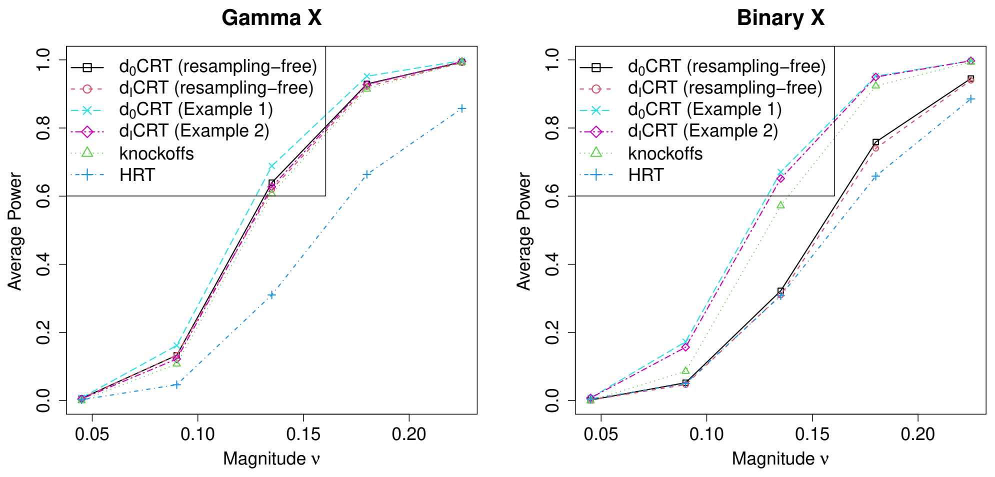


Figure 21: Powers of the simulation in Appendix D.12 measuring the effect of the Gaussian transformation in the resampling-free dCRTs. All standard errors are below 0.01. The dCRT resampling-based approaches have the most power in non-Gaussian settings as well, but the resampling-free modifications have lower power in the binary case.

D.13 Impact of screening on computation efficiency and power

Here we demonstrate the effect of the screening modification introduced in Section 3.1 on computation time and power. We again simulate the baseline setting in Section D.3 and compare the power and computation time of the dCRT methods with screening with the dCRT procedures without using screening. In Table 5 we present computation times demonstrating that screening can substantial improve the computational efficiency of dCRT. And the corresponding average powers are shown in Figure 22, demonstrating that screening has nearly no impact on the power of the d_0CRT or d_ICRT.

Average computation times (minutes)

d_0CRT (screening)	d_0CRT (full)	d_ICRT (screening)	d_ICRT (full)	knockoff	HRT
9.8	46.1	10.2	47.1	1.8	6.2

Table 5: Average computation times (in minutes) of the simulations in Appendix D.13. Screening leads to large computational savings at nearly no cost to the statistical power.

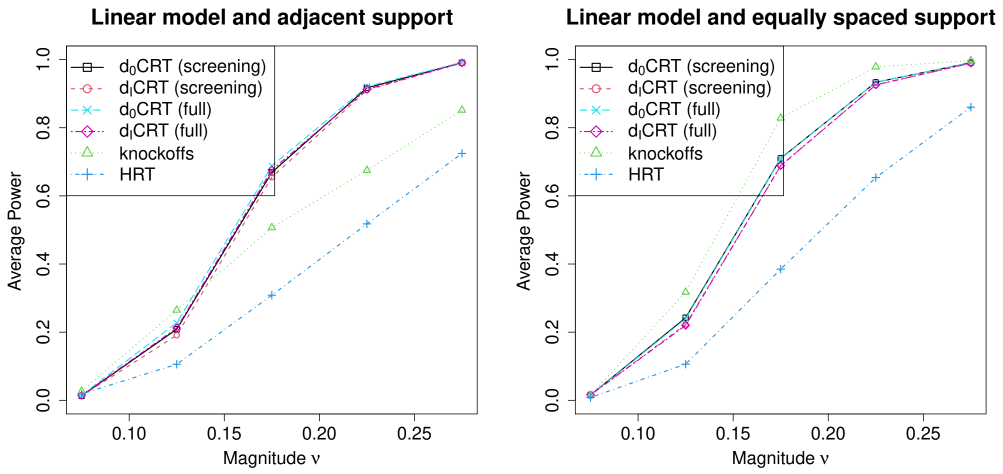


Figure 22: Average powers of the simulation in Appendix D.13 measuring the effect of the screening modification. All standard errors are below 0.01. The screened dCRTs have identical power to the dCRTs without screening, as expected.

D.14 Additional false discovery rate results

We compile false discovery rate results of our simulations with well-specified covariate distributions here. The false discovery rate is guaranteed to be controlled by knockoffs and the p-values of the CRT procedures including the dCRTs are guaranteed to be valid, but they do not satisfy the conditions for the Benjamini–Hochberg procedure to control the false discovery rate. In practice, they do control FDR, as Figures 23–26 show. This adds further support to a widely acknowledged empirical observation that the BH procedure rarely (if ever) violates FDR control in practice, outside of truly adversarial simulation setups with specially designed dependence structures.

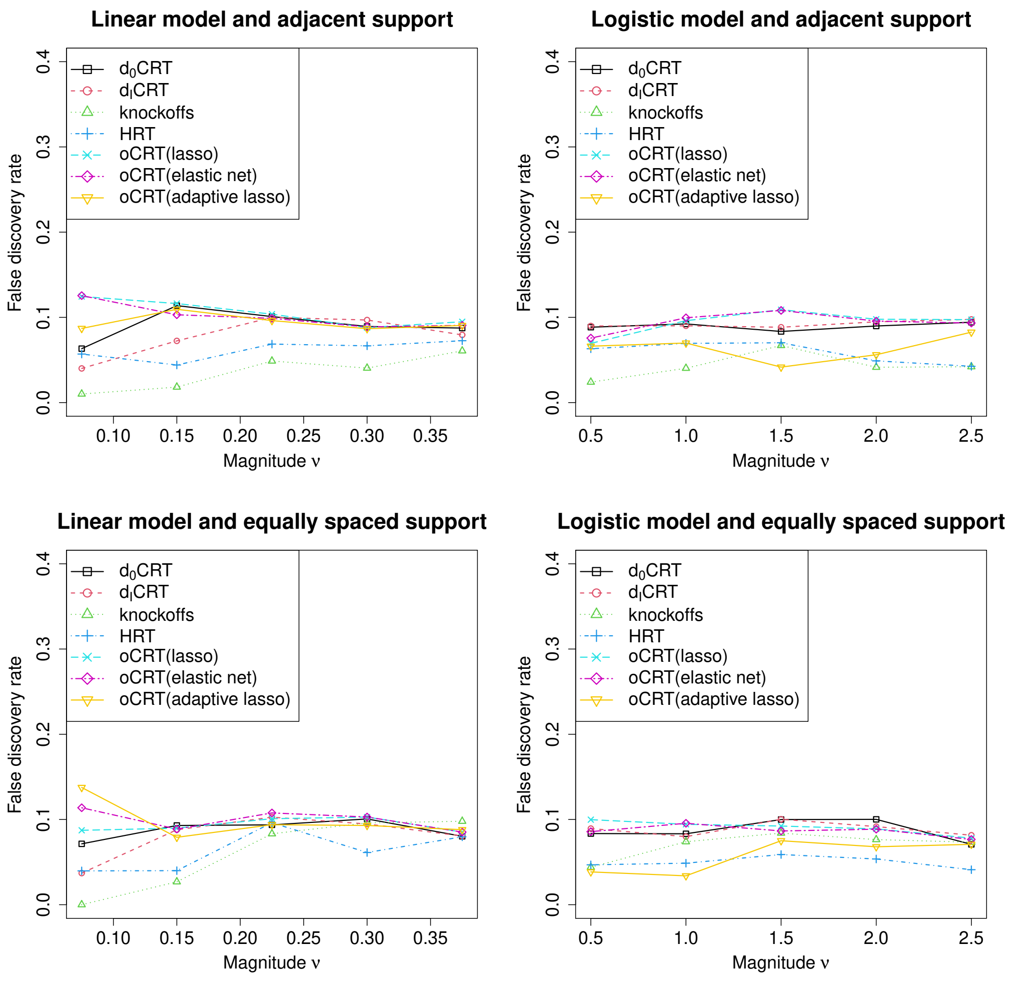


Figure 23: False discovery rates of the $n = p = 300$ simulation of Appendix D.2; standard errors are below 0.01. All methods control the false discovery rate at the target level 0.1, as desired.

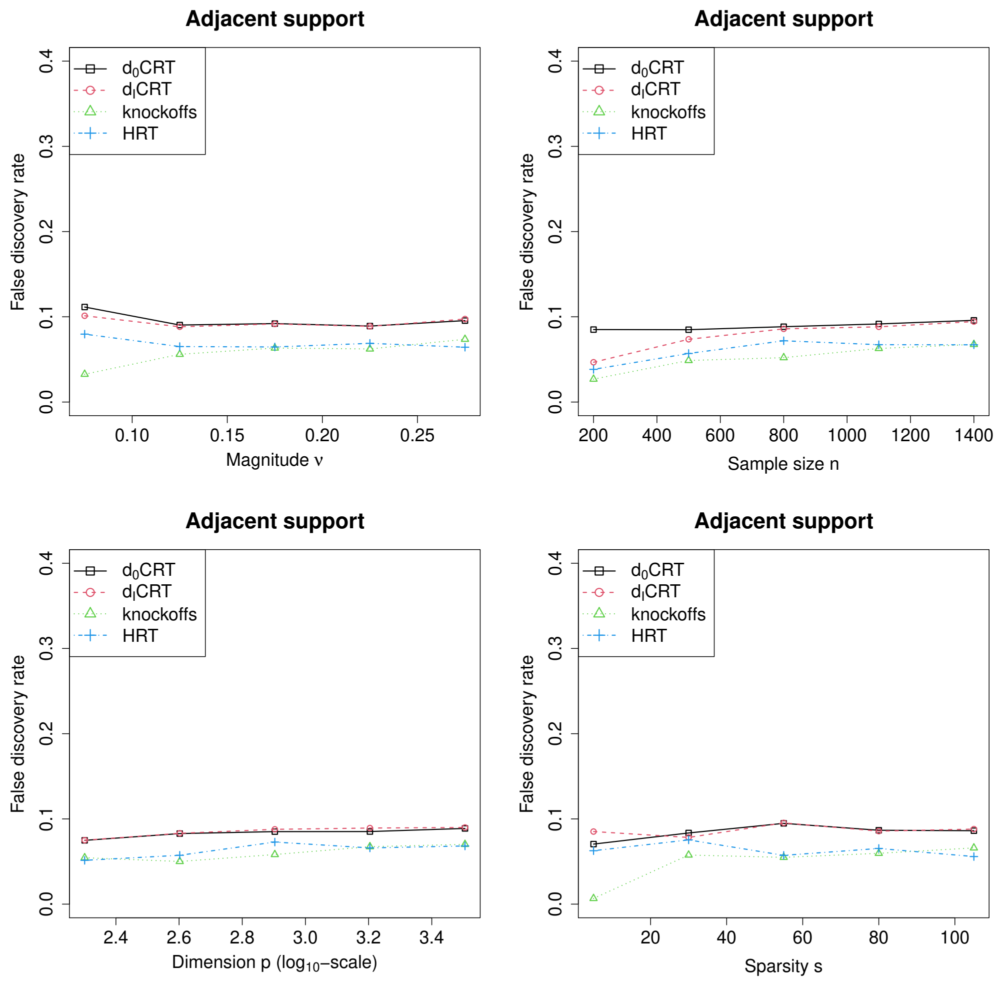


Figure 24: False discovery rates of the large scale simulations of Appendix D.3 that vary the coefficient magnitude, sample size, dimension, and coefficient sparsity with adjacent support. All standard errors are below 0.01; all methods control false discovery rate across all settings.

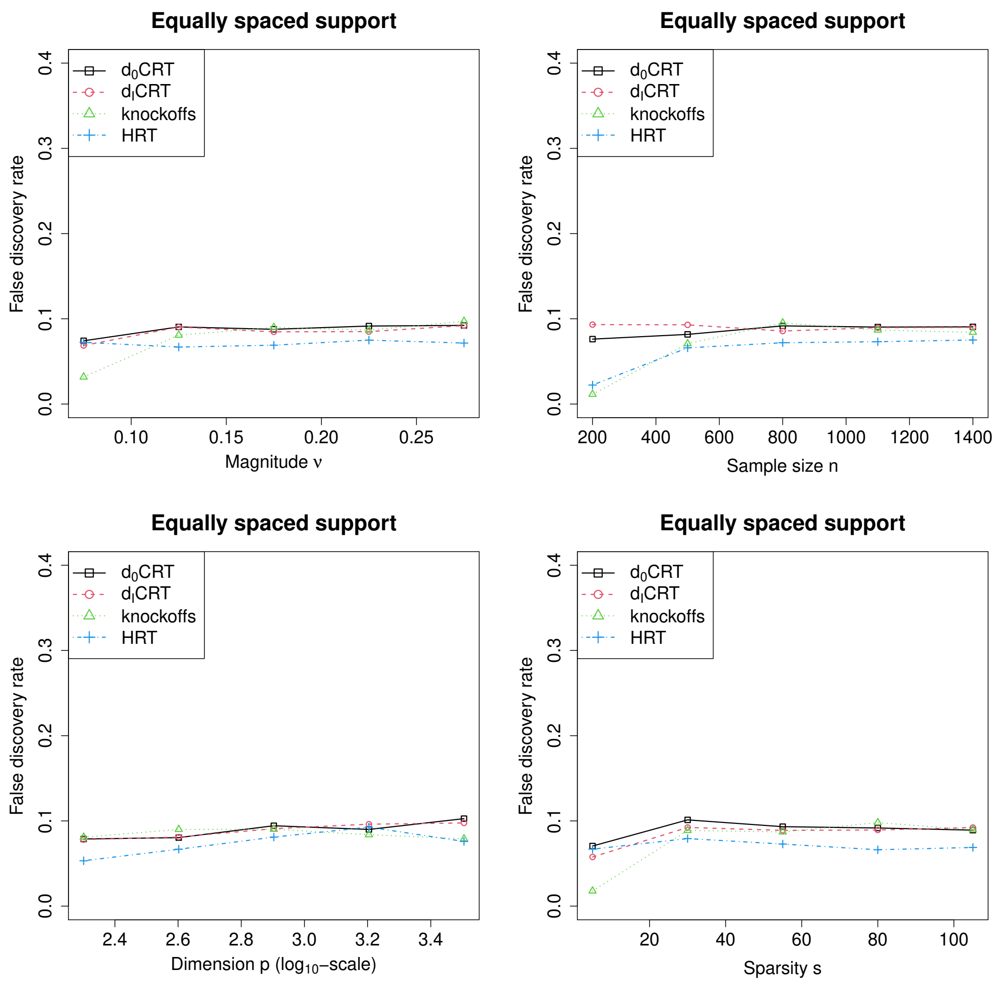


Figure 25: False discovery rates of the large scale simulations of Appendix D.3 that vary the coefficient magnitude, sample size, dimension, and coefficient sparsity with equally spaced support. All standard errors are below 0.01; all methods control the false discovery rate in all settings.

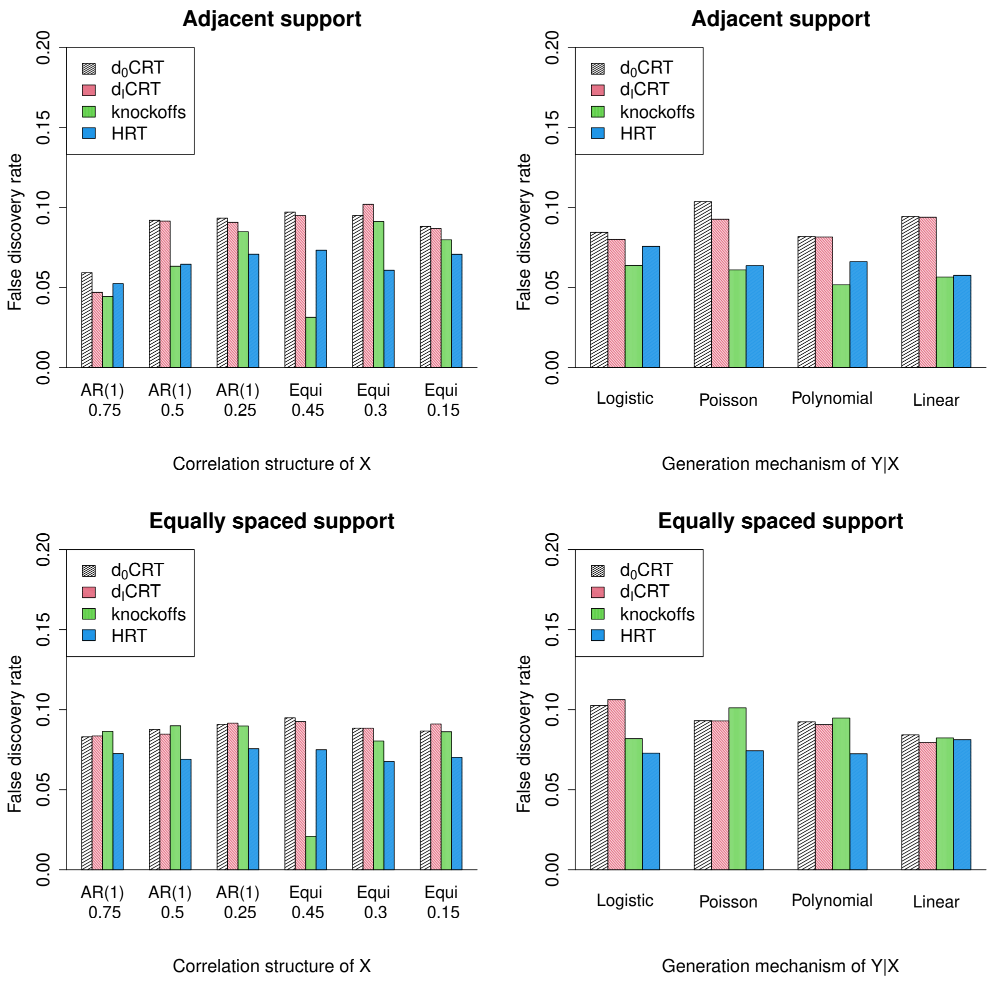


Figure 26: False discovery rates of the large scale simulations of Appendix D.3 that vary the covariate and response models; all standard errors are below 0.01. All methods control false discovery rate in all settings.

E Breast cancer data analysis

In this section, we present the details of our analysis of the breast cancer data set in Section 5; our code for pre-processing and analyzing the data is available at `https://github.com/moleibobliu/Distillation-CRT`. The list of $p = 164$ candidate genes is obtained as the set of genes among those measured by Curtis et al. (2012) that are the most frequently mutated according to Pereira et al. (2016) (see Supplementary Data 1 at `https://www.nature.com/articles/ncomms11479#Sec32`). The CNA, gene expression and clinical data itself is from cBioPortal and can be downloaded from `https://www.cbioportal.org/study/summary?id=brca_metabric`. The raw cancer stage used in our response variable is from the column labeled $TUMOR\_STAGE$ in their table for clinical data. It consists of three categories, 1, 2 and 3 that represent the progression stage of breast cancer. Since there were relatively few observations in category 1, we merge categories 1 and 2 together, resulting in a binary response. And the samples for analysis were chosen as all the patients with ER+ given in the clinical table.

Now we introduce the procedures for modeling the covariates. To model the expression levels of each gene G_j conditional on its corresponding CNA level C_j, we follow the methods proposed and discussed in Solvang et al. (2011); Lahti et al. (2012); Leday et al. (2013) to fit a piecewise linear regression of each G_j on each C_j. Denoting the fitted residuals as $\widetilde{G}_j$ and $\widetilde{\mathbf{G}} = (\widetilde{G}_1, \ldots, \widetilde{G}_p)$ we then model $\widetilde{\mathbf{G}}$ as mean-zero multivariate Gaussian (similar to Shen et al. (2019)) and estimate its precision matrix via a similar procedure as in Section D.10. That is, we remove the mean of each $\widetilde{G}_j$, fit graphical lasso tuned with cross-validation to estimate the precision matrix, and finally take the inverted precision matrix estimate and multiply it by a diagonal matrix to match the conditional variance of each $\widetilde{G}_j$ with the mean square of its residuals.

As in the simulations, the $\mathrm{d_0}$CRT and $\mathrm{d_I}$CRT we use are the resampling-free logistic regression versions of Examples 1 and 2 along with screening with the logistic lasso. Again, we do not use a logistic regression test statistic to allow for the computational gains of the resampling-free modification. We also implement knockoffs, the HRT, and the oCRT as in the simulations section with analogous logistic lasso statistics. The number of resamples for the HRT and the oCRT is set as $M = 25,000$, again satisfying $M/5 > p/\alpha$ as the false discovery rate or family-wise error rate level α is set as 0.1.

We summarize the discovered genes and their average p-values (over the 300 repetitions) estimated by each method in Table 6, their frequencies of being detected in terms of false discovery rate control in Table 7, and their frequencies of being detected in terms of family-wise error rate control in Table 8.

Gene	$\mathrm{d_0}$CRT	$\mathrm{d_I}$CRT	oCRT(lasso)	HRT
FBXW7	1.1×10^{-3}	5.8×10^{-4}	2.0×10^{-3}	2.1×10^{-2}
GPS2	2.5×10^{-4}	2.2×10^{-4}	3.5×10^{-4}	1.0×10^{-2}
HRAS	1.6×10^{-3}	1.7×10^{-3}	3.0×10^{-3}	6.1×10^{-4}
MAP3K13	1.1×10^{-4}	8.0×10^{-5}	3.3×10^{-3}	1
NRAS	6.1×10^{-3}	9.5×10^{-3}	6.5×10^{-3}	1.4×10^{-3}
RUNX1	2.4×10^{-4}	2.0×10^{-4}	5.8×10^{-4}	2.9×10^{-4}

Table 6: Significant genes and their corresponding average p-values over 300 repetitions.

Gene	d_0CRT	d_ICRT	oCRT(lasso)	HRT	knockoffs
FBXW7	1	1	0.70	0	0.54
GPS2	1	1	1	0.24	0.53
HRAS	1	1	0.14	1	0.54
MAP3K13	1	1	0.06	0	0.13
NRAS	0	0	0	1	0.54
RUNX1	1	1	1	1	0.54

Table 7: Frequency of being detected over the 300 repetitions, with false discovery rate control at the level 0.1.

Gene	d_0CRT	d_ICRT	oCRT(lasso)	HRT
FBXW7	0	0.88	0	0
GPS2	1	1	0.97	0
HRAS	0	0	0	1
MAP3K13	1	1	0	0
NRAS	0	0	0	1
RUNX1	1	1	0.64	1

Table 8: Frequency of being detected over the 300 repetitions, with family-wise error rate control at the level 0.1.